Équipe d'ingénierie de l'Internet (IETF)
Request for Comments : 9260
RFC rendues obsolètes : 4460, 4960, 6096, 7053, 8540
Catégorie : Sur la voie de la normalisation
ISSN : 2070-1721

R. Stewart, Netflix, Inc.
M. Tüxen, Münster Univ. of Appl. Sciences
K. Nielsen, Kamstrup A/S

juin 2022
Traduction Claude Brière de L'Isle

Protocole de transmission de contrôle de flux

Résumé
Le présent document décrit le protocole de transmission de contrôle de flux (SCTP, *Stream Control Transmission Protocol*) et rend obsolète la RFC 4960. Il incorpore la spécification du registre des fanions de tronçons de la RFC 6096 et la spécification du bit I des tronçons DATA de la RFC 7053. Donc, les RFC 6096 et 7053 sont aussi rendues obsolètes par ce document. De plus, les RFC 4460 et 8540, qui décrivent les errata pour SCTP, sont rendues obsolètes par ce document.

SCTP a été à l'origine conçu pour le transport des messages de signalisation du réseau téléphonique public commuté (RTPC) sur les réseaux IP. Il convient aussi pour d'autres applications, par exemple, WebRTC.

SCTP est un protocole de transport fiable qui fonctionne par dessus un réseau de paquets sans connexion comme IP. Il offre les services suivants à ses utilisateurs :
- transfert non dupliqué sans erreur avec accusé de réception des données d'utilisateur,
- fragmentation des données pour se conformer à la taille de MTU de chemin découverte,
- livraison en séquence des messages d'utilisateur au sein de flux multiples, avec une option pour la livraison dans l'ordre d'arrivée des messages d'utilisateur individuels,
- groupement facultatif de messages multiples d'utilisateur en un seul paquet SCTP, et
- tolérance aux fautes au niveau réseau par la prise en charge du multi rattachements à l'une et/ou l'autre extrémité d'une association.

La conception de SCTP inclut le comportement approprié d'évitement d'encombrement et la résistance aux attaques d'inondation et d'usurpation d'identité.

Statut de ce mémoire
Ceci est un document de l'Internet sur la voie de la normalisation.

Le présent document a été produit par l'équipe d'ingénierie de l'Internet (IETF). Il représente le consensus de la communauté de l'IETF. Il a subi une révision publique et sa publication a été approuvée par le groupe de pilotage de l'ingénierie de l'Internet (IESG). Tous les documents approuvés par l'IESG ne sont pas candidats à devenir une norme de l'Internet ; voir la Section 2 de la RFC5741.

Les informations sur le statut actuel du présent document, tout errata, et comment fournir des réactions sur lui peuvent être obtenues à http://www.rfc-editor.org/info/rfc9260.

Table des Matières

1. Introduction

Cette section explique les raisons qui sous tendent le développement du protocole de transmission de contrôle de flux (SCTP, *Stream Control Transmission Protocol*) les services qu'il offre, et les concepts de base nécessaires pour comprendre la description détaillée du protocole.

Le présent document rend obsolète la [RFC4960]. En plus de cela, il incorpore les spécifications du registre des fanions de tronçons de la [RFC6096] et la spécification du bit I des tronçons DATA de la [RFC7053]. Donc, la [RFC6096] et la [RFC7053] sont aussi rendues obsolètes par le présent document.

1.1 Motivation

TCP [RFC0793] a accompli un immense service comme principal moyen de transfert fiable de données dans les réseaux IP. Cependant, un nombre croissant d'applications récentes ont trouvé TCP trop limitant, et ont incorporé leur propre protocole fiable de transfert de données par dessus UDP [RFC0768]. Les limitations que les utilisateurs ont souhaité dépasser sont les suivantes :

- TCP fournit à la fois le transfert fiable de données et la livraison dans le strict ordre de transmission des données. Certaines applications ont besoin du transfert fiable sans maintenance de séquence, tandis que d'autres seraient satisfaites avec un ordre partiel des données. Dans ces deux cas, le blocage de tête de ligne offert par TCP cause des retards inutiles.
- Le mode flux par nature de TCP est souvent un inconvénient. Les applications ajoutent leur propre marquage d'enregistrement pour délimiter leurs messages, et font un usage explicite de la facilité de pousser pour s'assurer qu'un message complet est transféré dans un délai raisonnable.
- La portée limitée des prises TCP complique la tâche de fournir des capacités de transfert de données à forte disponibilité qui utilisent des hôtes multi rattachements.
- TCP est relativement vulnérable aux attaques de déni de service, comme les attaques de SYN.

Le transport de la signalisation du RTPC à travers le réseau IP est une application pour laquelle toutes ces limitations de TCP s'appliquent. Bien que cette application ait directement motivé le développement de SCTP, d'autres applications peuvent trouver que SCTP satisfait bien leurs exigences.Un exemple en est l'utilisation des canaux de données dans l'infrastructure WebRTC.

1.2 Vue architecturale de SCTP

SCTP est vu comme une couche entre l'application d'utilisateur SCTP ("utilisateur SCTP" en bref) et un service de réseau de paquets sans connexion comme IP. La suite de ce document suppose que SCTP fonctionne par dessus IP. Le service de base offert par SCTP est le transfert fiable de messages d'utilisateur entre des utilisateurs SCTP homologues. Il effectue le service dans le contexte d'une association entre deux points d'extrémité SCTP. La Section 11 du présent document décrit l'API qui existe à la frontière entre SCTP et les couches supérieures à SCTP.

SCTP est par nature en mode connexion, mais l'association SCTP est un concept plus large que celui de la connexion TCP. SCTP donne le moyen à chaque point d'extrémité SCTP (paragraphe 1.3) de fournir à l'autre point d'extrémité (durant le démarrage de l'association) une liste d'adresses de transport (c'est-à-dire, plusieurs adresses IP combinées à un accès SCTP) à travers lesquelles ce point d'extrémité peut être atteint et à partir desquelles il va générer des paquets SCTP. L'association

embrasse les transferts sur toutes les combinaisons possibles de source/destination qui peuvent être générées à partir des listes de chaque point d'extrémité.

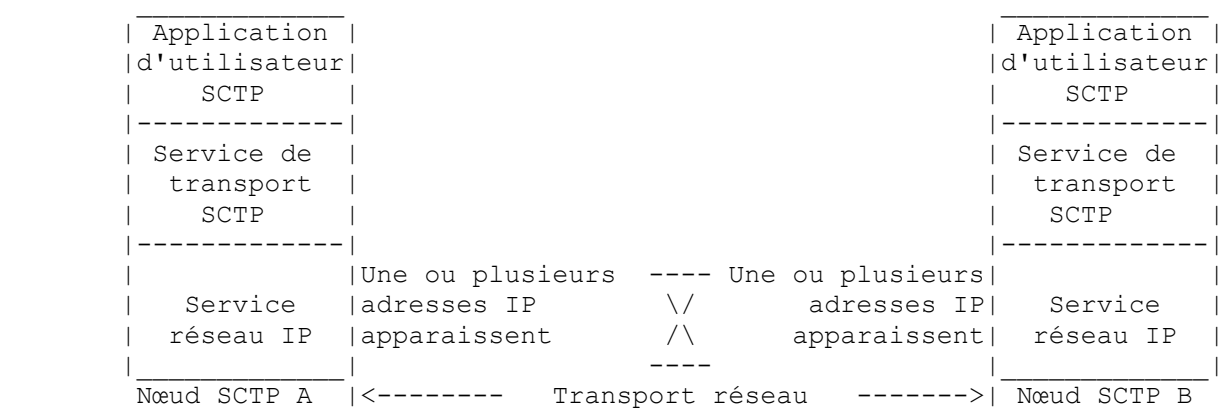


Figure 1 : Association SCTP

En plus d'encapsuler les paquets SCTP dans IPv4 ou IPv6, il est aussi possible d'encapsuler les paquets SCTP dans UDP comme spécifié dans la [RFC6951] ou de les encapsuler dans DTLS comme spécifié dans la [RFC8261].

1.3 Termes clés

Certains termes du langage utilisé pour décrire SCTP ont été introduits dans les paragraphes qui précèdent. Ce paragraphe donne une liste consolidée des termes clés et leur définition.

adresse de destination inactive : adresse qui n'a pas eu de message d'utilisateur depuis un certain temps, normalement l'intervalle HEARTBEAT *(battement de cœur)* ou plus.

adresse de transport : elle est normalement définie par une adresse de couche réseau, un protocole de couche transport, et un numéro d'accès de couche transport. Dans le cas de SCTP sur IP, une adresse de transport est définie par la combinaison d'une adresse IP et d'un numéro d'accès SCTP (où SCTP est le protocole de transport).

adresse de transport de destination active : adresse de transport sur un point d'extrémité d'homologue qu'un point d'extrémité transmetteur considère disponible pour recevoir des messages d'utilisateur.

adresse de transport de destination inactive : adresse qui est considérée comme inactive à cause d'erreurs, et indisponible pour transporter des messages d'utilisateur.

application d'utilisateur SCTP (ou utilisateur SCTP) : entité logique d'application de couche supérieure qui utilise les services de SCTP, aussi appelée le protocole de couche supérieure (ULP, *Upper-Layer Protocol*).

association SCTP : relations de protocole entre points d'extrémité SCTP, composée de deux points d'extrémité SCTP et des informations d'état de protocole incluant des étiquettes de vérification et l'ensemble actuellement actif de numéros de séquence de transmission (TSN, *Transmission Sequence Number*) etc. Une association peut être identifiée de façon univoque par les adresses de transport utilisées par les points d'extrémité dans l'association. Deux points d'extrémité SCTP NE DOIVENT PAS avoir plus d'une association SCTP entre eux à tout moment.

bloc de contrôle de transmission (TCB, *Transmission Control Block*) : structure de données interne créée par un point d'extrémité SCTP pour chacune de ses associations SCTP existantes avec les autres points d'extrémité SCTP. Le TCB contient toutes les informations d'état et de fonctionnement pour que le point d'extrémité maintienne et gère l'association correspondante.

chemin : route prise par les paquets SCTP envoyés d'un point d'extrémité SCTP à une adresse de transport de destination spécifique de son point d'extrémité SCTP homologue. Envoyer à des adresses de transport de destination différentes ne garantit pas nécessairement d'obtenir des chemins séparés. Dans la présente spécification, un chemin est identifié par l'adresse de transport de destination, car l'acheminement est supposé être stable. Cela inclut, en particulier, l'adresse de source choisie lors de l'envoi de paquets à l'adresse de destination.

chemin principal : le chemin principal est l'adresse de destination et l'adresse de source qui seront mises par défaut dans un paquet sortant pour le point d'extrémité homologue. La définition inclut l'adresse de source car une mise en œuvre PEUT souhaiter spécifier les deux adresses de destination et de source pour mieux contrôler le chemin de retour pris par les tronçons de réponse et sur quelle interface le paquet est transmis quand l'envoyeur des données est multi rattaché.

données en instance (ou données en vol) : taille totale des tronçons DATA associés à des TSN en instance. Un tronçon DATA retransmis est compté une fois dans les données en instance. Un tronçon DATA qui est classé comme perdu mais qui n'a pas encore été retransmis n'est pas dans les données en instance.

étiquettes de lien *(Tie-Tag)* : deux nombres aléatoires de 32 bits qui ensemble font un nom occasionnel de 64 bits. Ces étiquettes sont utilisées dans un mouchard d'état et un TCB afin qu'une association qui redémarre puisse être reliée à l'association originale au sein du point d'extrémité qui n'a pas redémarré et pas encore révélé les vraies étiquettes de vérification d'une association existante.

étiquette de vérification : entier non signé de 32 bits qui est généré au hasard. L'étiquette de vérification fournit une clé qui permet à un receveur de vérifier que le paquet SCTP appartient à l'association en cours et n'est pas un paquet vieux ou périmé d'une association précédente.

fenêtre d'encombrement (cwnd, *Congestion Window*) : variable SCTP qui limite les données en instance, en nombre d'octets, qu'un envoyeur peut envoyer à une adresse de transport de destination particulière avant de recevoir un accusé de réception.

fenêtre de réception (rwnd, *Receiver Window*) : variable SCTP qu'un envoyeur de données utilise pour mémoriser la fenêtre de receveur la plus récemment calculée de son homologue, en nombre d'octets. Cela donne à l'envoyeur une indication de l'espace disponible dans la mémoire tampon d'entrée du receveur.

flux : canal logique unidirectionnel établi d'un point d'extrémité SCTP associé à un autre, au sein duquel tous les messages d'utilisateur sont livrés en séquence, sauf ceux soumis au service de livraison non ordonnée.

Note : la relation entre les numéros de flux dans les directions opposées est strictement dépendante de la façon dont l'application les utilise. Il est de la responsabilité de l'utilisateur SCTP de créer et gérer ces corrélations si c'est ce qui est désiré.

groupement de tronçons : opération de multiplexage facultative, par laquelle plus d'un tronçon peut être porté dans le même paquet SCTP.

groupement de messages d'utilisateur : opération de multiplexage facultative, par laquelle plus d'un message d'utilisateur peut être porté dans le même paquet SCTP. Chaque message d'utilisateur occupe son propre tronçon DATA.

message (ou message d'utilisateur) : données soumises à SCTP par le protocole de couche supérieure (ULP, *Upper-Layer Protocol*).

message déclassé : les messages déclassés sont "déclassés" par rapport à tout autre message ; cela inclut les autres messages déclassés ainsi que les messages en ordre. Un message déclassé peut être livré avant ou après des messages ordonnés envoyés sur le même flux.

message d'utilisateur : unité de données livrée à travers l'interface entre SCTP et son utilisateur.

message ordonné : message d'utilisateur qui est livré dans l'ordre par rapport à tous les messages d'utilisateur précédents envoyés au sein du flux sur lequel le message a été envoyé.

mouchard d'état *(State Cookie)* : conteneur de toutes les informations nécessaires pour établir une association.

numéro de séquence de flux : numéro de séquence de 16 bits utilisé en interne par SCTP pour assurer la livraison en séquence des messages d'utilisateur dans un certain flux. Un numéro de séquence de flux est attaché à chaque message d'utilisateur.

numéro de séquence de transmission (TSN, *Transmission Sequence Number*) : numéro de séquence de 32 bits utilisé en interne par SCTP. Un TSN est attaché à chaque tronçon contenant des données d'utilisateur pour permettre au point d'extrémité SCTP receveur d'en accuser réception et détecter les livraisons dupliquées.

ordre des octets du réseau : octet de poids fort en premier, autrement dit, gros boutien.

paquet SCTP (ou paquet) : unité de livraison des données à travers l'interface entre SCTP et le réseau de paquets sans connexion (par exemple, IP). Un paquet SCTP inclut l'en-tête SCTP commun, d'éventuels tronçons de contrôle SCTP, et des données d'utilisateur encapsulées au sein des tronçons DATA SCTP.

paquet sorti de nulle part (OOTB, *"Out of the Blue"*) : paquet correctement formé, pour lequel le receveur ne peut pas identifier l'association à laquelle il appartient. Voir le paragraphe 8.4.

point d'accusé de réception de TSN cumulatif : numéro de séquence de transmission (TSN, *Transmission Sequence Number*) du dernier tronçon DATA acquitté via le champ Accusé de réception de TSN cumulatif d'un tronçon SACK.

point d'extrémité SCTP : envoyeur/receveur logique des paquets SCTP. Sur un hôte multi rattachements, un point d'extrémité SCTP est représenté à ses homologues comme une combinaison d'un ensemble d'adresses de transport de destination éligibles auxquelles les paquets SCTP peuvent être envoyés et d'un ensemble d'adresses de transport de source éligibles desquelles les paquets SCTP peuvent être reçus. Toutes les adresses de transport utilisées par un point d'extrémité SCTP DOIVENT utiliser le même numéro d'accès, mais peuvent utiliser plusieurs adresses IP. Une adresse de transport utilisée par un point d'extrémité SCTP NE DOIT PAS être utilisée par un autre point d'extrémité SCTP. En d'autres termes, une adresse de transport est unique pour un point d'extrémité SCTP.

seuil de démarrage lent (ssthresh, *Slow-Start Threshold*) : variable SCTP. C'est le seuil que le point d'extrémité va utiliser pour déterminer si il effectue le démarrage lent ou l'évitement d'encombrement sur une adresse de transport de destination particulière. ssthresh est en nombre d'octets.

taille en vol *(flightsize)* : nombre d'octets de données en instance pour une adresse de transport de destination particulière à un moment donné quelconque.

taille maximum de tronçon DATA d'association (AMDCS, *Association Maximum DATA Chunk Size*) : plus petite taille maximum de tronçon DATA du chemin (PMDCS, *Path Maximum DATA Chunk Size*) de toutes les adresses de destination.

taille maximum de tronçon DATA du chemin (PMDCS, *Path Maximum DATA Chunk Size*) : taille maximum (incluant l'en-tête de tronçon DATA) d'un tronçon DATA qui tient dans un paquet SCTP n'excédant pas la PMTU d'une adresse de destination particulière.

tronçon *(tronçon)* : unité d'information au sein d'un paquet SCTP, consistant en un en-tête de tronçon et un contenu spécifique du tronçon.

tronçon de contrôle : tronçon non utilisé pour transmettre des données d'utilisateur, c'est-à-dire, chaque tronçon qui n'est pas un tronçon DATA.

TSN en instance à un point d'extrémité SCTP : TSN (et le tronçon DATA associé) qui a été envoyé par le point d'extrémité mais pour lequel il n'a pas encore été reçu d'accusé de réception.

unité maximum de transmission de chemin (PMTU, *Path Maximum Transmission Unit*) : taille maximum (incluant l'en-tête SCTP commun et tous les tronçons incluant leur bourrage) d'un paquet SCTP qui peut être envoyé à une adresse de destination particulière sans utiliser de fragmentation de niveau IP.

1.4 Abréviations

MAC *(Message Authentication Code)* [RFC2104] = code d'authentification de message
RTO *(Retransmission Timeout)* = fin de temporisation de retransmission
RTT *(Round-Trip Time)* = délai d'aller retour
RTTVAR *(Round-Trip Time Variation)* = variation du délai d'aller retour
SCTP *(Stream Control Transmission Protocol)* = protocole de transmission de contrôle de flux
SRTT *(Smoothed RTT)* = délai d'aller retour lissé
TCB (*Transmission Control Block)* = bloc de contrôle de transmission
TLV = Type-Longueur-Valeur (format de codage)
TSN (*Transmission Sequence Number)* = numéro de séquence de transmission

ULP *(Upper-Layer Protocol)* = protocole de couche supérieure

1.5 Vue fonctionnelle de SCTP

Le service de transport SCTP peut être décomposé en un certain nombre de fonctions. Celles-ci sont décrites à la Figure 2 et expliquées dans la suite de cette section.

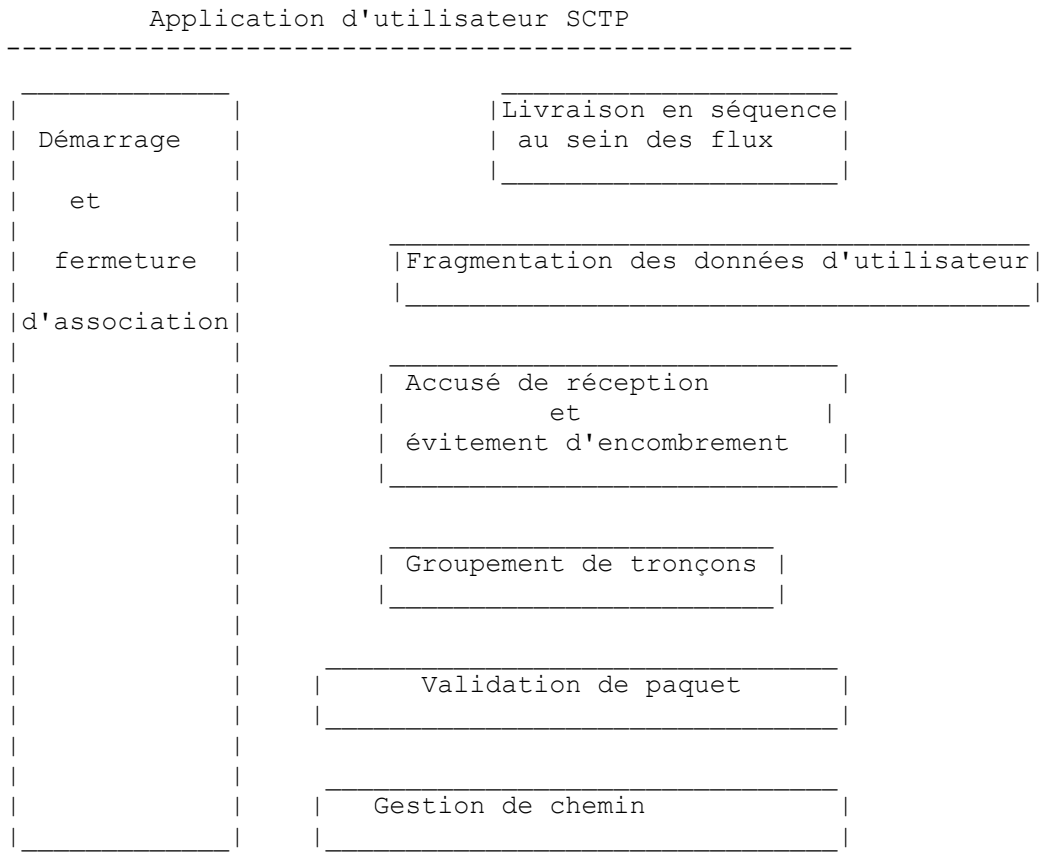


Figure 2 : Vue fonctionnelle du service de transport SCTP

1.5.1 Démarrage et suppression d'association

Une association est initiée par une demande de la part de l'utilisateur SCTP (voir la description de la primitive ASSOCIATE (ou SEND) à la Section 11).

Un mécanisme de mouchard, similaire à celui décrit par Karn et Simpson dans la [RFC2522], est employé durant l'initialisation pour assurer la protection contre les attaques de synchronisation. Le mécanisme de mouchard utilise une prise de contact en quatre phases dont les deux dernières peuvent porter des données d'utilisateur pour un établissement rapide. La séquence de démarrage est décrite à la Section 5 de ce document.

SCTP fournit une clôture en douceur (c'est-à-dire, la fermeture) d'une association active sur demande de l'utilisateur SCTP. Voir la description de la primitive SHUTDOWN à la Section 11. SCTP permet aussi une clôture sans douceur (c'est-à-dire, une interruption) soit à la demande de l'utilisateur (primitive ABORT) soit par suite d'une condition d'erreur détectée au sein de la couche SCTP. La Section 9 décrit les deux procédures de fermeture, en douceur et sans douceur.

SCTP ne prend pas en charge d'état semi ouvert (comme TCP) dans lequel un côté peut continuer d'envoyer des données tandis que l'autre côté est fermé. Quand l'un ou l'autre des points d'extrémité effectue une fermeture, l'association chez chaque homologue va arrêter d'accepter de nouvelles données provenant de son utilisateur et seulement livrer les données dans la file d'attente au moment de la clôture en douceur (voir la Section 9).

1.5.2 Livraison en séquence dans les flux

Le terme "flux" est utilisé dans SCTP pour se référer à une séquence de messages d'utilisateur qui sont livrés au protocole de couche supérieure dans l'ordre par rapport aux autres messages au sein du même flux. Ceci diffère de son usage dans TCP, où il se réfère à une séquence d'octets (dans le présent document, un octet est supposé être de 8 bits).

L'utilisateur SCTP peut spécifier au moment du démarrage de l'association le nombre de flux qui vont être pris en charge par l'association. Ce nombre est négocié avec l'extrémité distante (voir au paragraphe 5.1.1). Les messages d'utilisateur sont associés aux numéros de flux (primitives SEND, RECEIVE, Section 11). En interne, SCTP alloue un numéro de séquence de flux à chaque message qui lui est passé par l'utilisateur SCTP. Du côté receveur, SCTP s'assure que les messages sont livrés à l'utilisateur SCTP en séquence au sein d'un flux donné. Cependant, alors qu'un flux peut être bloqué en attendant le prochain message d'utilisateur dans la séquence, la livraison provenant d'autres flux peut se faire.

SCTP fournit un mécanisme pour outrepasser le service de livraison en séquence. Les messages d'utilisateur envoyés en utilisant ce mécanisme sont livrés à l'utilisateur SCTP aussitôt qu'ils sont reçus.

1.5.3 Fragmentation des données d'utilisateur

Quand nécessaire, SCTP fragmente les messages d'utilisateur pour assurer que la taille du paquet SCTP passé à la couche inférieure n'excède pas la PMTU. Une fois qu'un message d'utilisateur a été fragmenté, cette fragmentation ne peut plus être changée. À réception, les fragments sont rassemblés en messages complets avant d'être passés à l'utilisateur SCTP.

1.5.4 Accusé de réception et évitement d'encombrement

SCTP alloue un numéro de séquence de transmission (TSN, *Transmission Sequence Number*) à chaque fragment de données d'utilisateur ou message non fragmenté. Le TSN est indépendant de tout numéro de séquence de flux alloué au niveau du flux. L'extrémité receveuse accuse réception de tous les TSN reçus, même si il y a des trous dans la séquence. Si un fragment de données d'utilisateur ou un message non fragmenté a besoin d'être retransmis, le TSN qui lui est alloué est utilisé. De cette façon, une livraison fiable est tenue fonctionnellement séparée de la livraison de flux en séquence.

La fonction d'accusé de réception et d'évitement d'encombrement est chargée de la retransmission des paquets quand un accusé de réception n'a pas été reçu en temps utile. La retransmission de paquets est conditionnée par des procédures d'évitement d'encombrement similaires à celles utilisées pour TCP. Voir aux Sections 6 et 7 une description détaillée des procédures de protocole associées à cette fonction.

1.5.5 Groupement de tronçons

Comme décrit à la Section 3, le paquet SCTP tel que livré à la couche inférieue consiste en un en-tête commun suivi par un ou plusieurs tronçons. Chaque tronçon contient des données d'utilisateur ou des informations de contrôle SCTP. Une mise en œuvre SCTP qui prend en charge le groupement du côté envoyeur pourrait retarder l'envoi des messages d'utilisateur afin de permettre le groupement de tronçons DATA correspondants.

L'utilisateur SCTP a la faculté de demander qu'une mise en œuvre SCTP ne retarde pas l'envoi d'un message d'utilisateur juste pour cela. Cependant, même si l'utilisateur SCTP a choisi cette option, la mise en œuvre SCTP pourrait retarder l'envoi pour d'autres raisons (par exemple, à cause du contrôle d'encombrement ou du contrôle de flux) et pourrait aussi grouper plusieurs tronçons DATA, si c'est possible.

1.5.6 Validation de paquet

Un champ obligatoire Étiquette de vérification et un champ de 32 bits Somme de contrôle (voir à l'Appendice A la description de la somme de contrôle de contrôle de redondance cyclique de 32 bits (CRC32c, *32-bit Cyclic Redundancy Check*) sont inclus dans l'en-tête commun SCTP. La valeur de l'étiquette de vérification est choisie par chaque extrémité de l'association durant le démarrage de l'association. Les paquets reçus sans la valeur d'étiquette de vérification attendue sont éliminés, comme protection contre les attaques aveugles de déguisement et contre les paquets SCTP périmés provenant d'une association précédente. La somme de contrôle CRC32c est établie par l'envoyeur de chaque paquet SCTP pour fournir une protection supplémentaire contre la corruption des données dans le réseau. Le receveur d'un paquet SCTP avec une somme de contrôle CRC32c invalide élimine en silence le paquet.

1.5.7 Gestion de chemin

L'utilisateur SCTP envoyeur est capable de manipuler l'ensemble d'adresses de transport utilisées comme destinations pour

les paquets SCTP au moyen des primitives décrites à la Section 11. La fonction de gestion de chemin surveille l'accessibilité au moyen des battements de cœur quand d'autre trafic de paquets est inadéquat pour fournir cette information et informe l'utilisateur SCTP quand l'accessibilité de toute adresse de transport d'extrémité distante change. La fonction de gestion de chemin choisit l'adresse de transport de destination pour chaque paquet SCTP sortant sur la base des instructions de l'utilisateur SCTP et de l'état d'accessibilité actuellement perçu de l'ensemble de destinations éligible. La fonction de gestion de chemin est aussi chargée de faire rapport de l'ensemble éligible d'adresses de transport locales au point d'extrémité homologue durant l'établissement de l'association, et de faire rapport des adresses de transport retournées du point d'extrémité homologue à l'utilisateur SCTP.

Au démarrage de l'association, un chemin principal est défini pour chaque point d'extrémité SCTP, et est utilisé pour l'envoi normal des paquets SCTP.

À l'extrémité receveuse, la gestion de chemin est chargée de vérifier l'existence d'une association SCTP valide à laquelle appartient le paquet SCTP entrant avant de le passer à la suite du traitement.

Note : la gestion de chemin et la validation de paquet sont faites en même temps, de sorte que bien que décrites séparément ci-dessus, en réalité, elles ne peuvent pas être effectuées comme des éléments séparés.

1.6 Arithmétique des numéros de série

Il est essentiel de se rappeler que l'espace réel de numéros de séquence de transmission est fini, bien que très grand. Cet espace va de 0 à 2**32 - 1. Comme cet espace est fini, toutes les arithmétiques traitant des numéros de séquence de transmission DOIVENT être effectuées modulo 2**32. Cette arithmétique non signée préserve les relations des numéros de séquence lorsque ils reviennent de 2**32 - 1 à 0. Il y a quelques subtilités dans l'arithmétique de calcul modulo, donc un grand soin devrait être apporté à la programmation de la comparaison de telles valeurs. Quand on se réfère aux TSN, le symbole "≤" signifie "inférieur ou égal"(modulo 2**32).

Les comparaisons et l'arithmétique sur les TSN dans le présent document DEVRAIENT utiliser l'arithmétique des numéros de série telle que définie dans la [RFC1982] où SERIAL_BITS = 32.

Un point d'extrémité NE DEVRAIT PAS transmettre un tronçon a DATA avec un TSN de plus de 2**31 - 1 au dessus du TSN de début de sa fenêtre d'envoi en cours. Le faire causerait des problèmes pour comparer les TSN.

Les TSN reviennent à zéro quand ils atteignent 2**32 - 1. C'est-à-dire que le TSN suivant qu'un tronçon DATA DOIT utiliser après la transmission de TSN = 2*32 - 1 est TSN = 0.

Toute arithmétique faite sur des numéros de séquence de flux DEVRAIT utiliser l'arithmétique des numéros de série telle que définie dans la [RFC1982] où SERIAL_BITS = 16. Toute autre arithmétique et comparaison dans le présent document utilisera l'arithmétique normale.

1.7 Changements par rapport à la RFC 4960

SCTP a été à l'origine défini dans la [RFC4960], que le présent document rend obsolète. Les lecteurs intéressés par les détails des divers changements qu'incorpore le présent document peuvent consulter la [RFC8540].

En plus de cela et de quelques changements rédactionnels, les changements suivants ont été incorporés à ce document :

* Mise à jour des références.
* Amélioration du langage relatif aux niveaux d'exigence.
* Permettre à la primitive ASSOCIATE de prendre plusieurs adresses distantes ; se référer aussi à la spécification d'API de prise.
* Se référer à la spécification de découverte de la MTU de chemin de couche de mise en paquet (PLPMTUD, *Packetization Layer Path MTU Discovery*) pour la découverte de la MTU de chemin.
* Passer la description du traitement de ICMP de l'Appendice au corps du texte.
* Suppression de l'Appendice décrivant le traitement de la notification explicite d'encombrement (ECN, *Explicit Congestion Notification*).
* Description plus précise du traitement de la taille de paquet en introduisant la PMTU, PMDCS, et AMDCS.
* Ajout de la définition du tronçon de contrôle.
* Amélioration de la description du traitement des tronçons INIT et INIT ACK avec des paramètres obligatoires invalides.

* Permettre d'utiliser L > 1 pour un comptage d'octets approprié (ABC, *Appropriate Byte Counting*) durant le démarrage lent.
* Décrire explicitement la réinitialisation du contrôleur d'encombrement sur des changements de chemin.
* Améliorer la terminologie pour rendre clair que la présente spécification ne décrit pas une architecture de maillage complet.
* Améliorer la description de la génération de numéro de séquence (numéro de séquence de transmission et numéro de séquence de flux).
* Améliorer la description du reniement.
* Ne plus exiger le changement de l'accusé de réception de TSN cumulatif pour augmenter la fenêtre d'encombrement. Cela améliore la cohérence avec le traitement de l'évitement d'encombrement.
* Améliorer la description du mouchard d'état.
* Corriger l'API pour restituer les messages en cas de défaillance d'association.

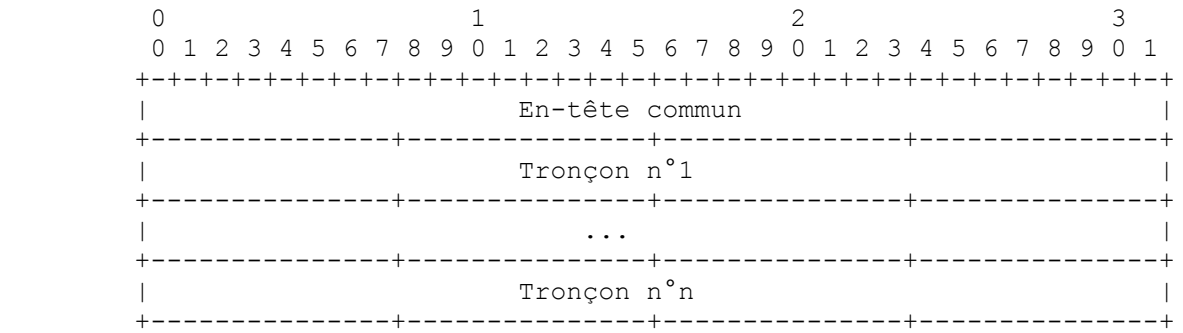
2. Conventions

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDÉ", "NON RECOMMANDÉ", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119], [RFC8174] quand, et seulement quand ils apparaissent tout en majuscules, comme montré ci-dessus.

3. Format de paquet SCTP

Un paquet SCTP est composé d'un en-tête commun et de tronçons. Un tronçon contient des informations de contrôle ou des données d'utilisateur.

Le format du paquet SCTP est montré ci-dessous :

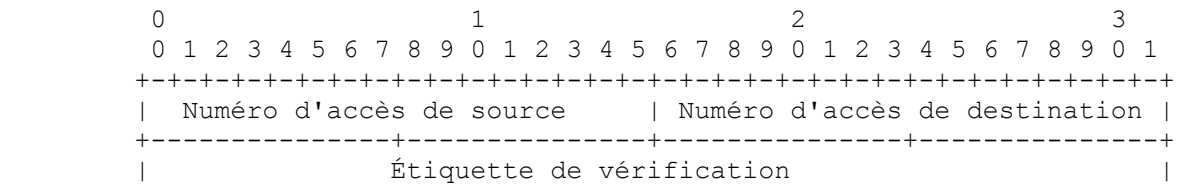


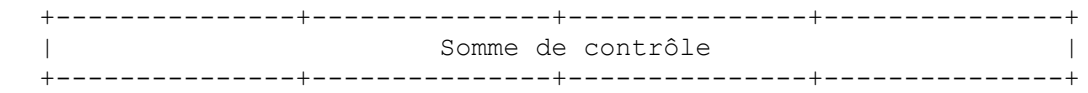
Les tronçons INIT, INIT ACK, et SHUTDOWN COMPLETE NE DOIVENT PAS être groupés avec d'autres tronçons dans un paquet SCTP. Tous les autres tronçons PEUVENT être groupés pour former un paquet SCTP qui n'excède pas la PMTU. Voir au paragraphe 6.10 les détails du groupage de tronçons.

Si un message de données d'utilisateur ne tient pas dans un paquet SCTP, il peut être fragmenté en plusieurs tronçons en utilisant la procédure définie au paragraphe 6.9.

Tous les champs entiers dans un paquet SCTP DOIVENT être transmis dans l'ordre des octets du réseau, sauf mention contraire.

3.1 Description des champs d'en-tête communs SCTP





Numéro d'accès de source : 16 bits (entier non signé). C'est le numéro d'accès de l'envoyeur SCTP. Il peut être utilisé par le receveur en combinaison avec l'adresse IP de source, l'accès de destination SCTP, et éventuellement l'adresse IP de destination pour identifier l'association à laquelle appartient ce paquet. Le numéro d'accès 0 NE DOIT PAS être utilisé.

Numéro d'accès de destination : 16 bits (entier non signé). C'est le numéro d'accès SCTP auquel ce paquet est destiné. L'hôte receveur va utiliser ce numéro d'accès pour démultiplexer le paquet SCTP au point d'extrémité/application receveur correct. Le numéro d'accès 0 NE DOIT PAS être utilisé.

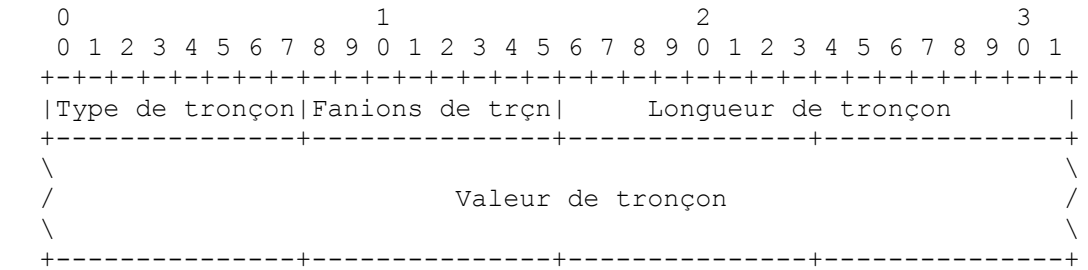
Étiquette de vérification : 32 bits (entier non signé). Le receveur de ce paquet utilise l'étiquette de vérification pour valider l'envoyeur du paquet SCTP. À l'émission, la valeur de cette étiquette de vérification DOIT être réglée à la valeur de l'étiquette Initier reçue du point d'extrémité homologue durant l'initialisation de l'association, avec les exceptions suivantes :

- un paquet contenant un tronçon INIT DOIT avoir une étiquette de vérification de zéro ;
- un paquet contenant un tronçon SHUTDOWN COMPLETE avec le bit T établi DOIT avoir l'étiquette de vérification copiée du paquet avec le tronçon SHUTDOWN ACK ;
- un paquet contenant un tronçon ABORT PEUT avoir l'étiquette de vérification copiée du paquet qui a causé l'envoi du ABORT. Voir les détails aux paragraphes 8.4 et 8.5.

Somme de contrôle : 32 bits (entier non signé). Ce champ contient la somme de contrôle de ce paquet SCTP. Son calcul est discuté au paragraphe 6.8. SCTP utilise l'algorithme CRC32c décrit à l'Appendice A pour le calcul de la somme de contrôle.

3.2 Description des champs d'en-tête de tronçon

La figure ci-dessous illustre le format des champs pour les tronçons à transmettre dans le paquet SCTP. Chaque tronçon est formaté avec un champ Type de tronçon, un champ Fanions de tronçon, un champ Longueur de tronçon, et un champ Valeur de tronçon.



Type de tronçon : 8 bits (entier non signé). Ce champ identifie le type des informations contenues dans le champ Valeur de tronçon. Il prend une valeur de 0 à 254. La valeur 255 est réservée pour une utilisation future comme champ d'extension.

Les valeurs de types de tronçon sont définies comme suit :

Valeur de l'identifiant	Type de tronçon
0	Données de charge utile (DATA)
1	Initiation (INIT)
2	Accusé de réception d'initiation (INIT ACK)
3	Accusé de réception sélectif (SACK)
4	Demande de battement de cœur (HEARTBEAT)
5	Accusé de réception de battement de cœur (HEARTBEAT ACK)
6	Interrompre (ABORT)
7	Fermer (SHUTDOWN)
8	Accusé de réception de fermeture (SHUTDOWN ACK)
9	Erreur de fonctionnement (ERROR)
10	Mouchard d'état (COOKIE ECHO)

11	Accusé de réception de mouchard (COOKIE ACK)
12	Réservé pour écho de notification explicite d'encombrement (ECNE)
13	Réservé pour fenêtre d'encombrement réduite (CWR)
14	Fermeture achevée (SHUTDOWN COMPLETE)
15 à 62	- disponible
63	- réservé pour des extensions de tronçons définies par l'IETF
64 à 126	- disponible
127	- réservé pour des extensions de tronçons définies par l'IETF
128 à 190	- disponible
191	- réservé pour des extensions de tronçons définies par l'IETF
192 à 254	- disponible
255	- réservé pour des extensions de tronçons définies par l'IETF

Tableau 1 : Types de tronçons

Note : les types de tronçons ECNE et CWR sont réservés pour une utilisation future de la notification explicite d'encombrement (ECN).

Les types de tronçon sont codés de façon telle que les 2 bits de poids fort spécifient l'action à entreprendre si le point d'extrémité qui les traite ne reconnaît pas le type de tronçon.

00 - Arrêter le traitement de ce paquet SCTP et l'éliminer, ne pas traiter d'autres tronçons dans celui-là.
01 - Arrêter le traitement de ce paquet SCTP et l'éliminer, ne pas traiter d'autres tronçons dans celui-là, et faire rapport du tronçon non reconnu dans un 'Type de tronçon non reconnu'.
10 - Sauter ce tronçon et continuer le traitement.
11 - Sauter ce tronçon et continuer le traitement, mais en faire rapport dans un tronçon ERROR en utilisant la cause d'erreur 'Type de tronçon non reconnu'.

Tableau 2 : Traitement des tronçons inconnus

Fanions de tronçon: 8 bits. L'usage de ces bits dépend du type de tronçon donné dans le champ Type de tronçon. Sauf mention contraire, ils sont réglés à 0 à l'émission et ignorés à réception.

Longueur de tronçon : 16 bits (entier non signé). Cette valeur représente la taille du tronçon en octets, incluant les champs Type de tronçon, Fanions de tronçon, Longueur de tronçon, et Valeur de tronçon. Donc, si le champ Valeur de tronçon est de longueur zéro, le champ Longueur va être réglé à 4. Le champ Longueur de tronçon ne comporte aucun bourrage de tronçon. Cependant, il inclut bien tout bourrage des paramètres de longueur variable autres que le dernier paramètre dans le tronçon.

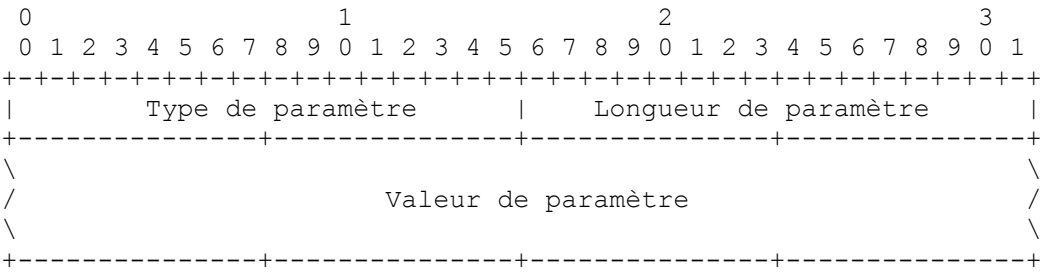
Note : Une mise en œuvre robuste devrait accepter le tronçon, que le bourrage final soit ou non inclus dans la longueur de tronçon.

Valeur de tronçon : longueur variable. Le champ Valeur de tronçon contient les informations réelles à transférer dans le tronçon. L'usage et le format de ce champ dépendent du type de tronçon. La longueur totale d'un tronçon (incluant les champs Type, Longueur, et Valeur) DOIT être un multiple de 4 octets. Si la longueur du tronçon n'est pas un multiple de 4 octets, l'envoyeur DOIT bourrer le tronçon avec des octets de zéros, et ce bourrage n'est pas inclus dans le champ Longueur de tronçon. L'envoyeur NE DOIT PAS bourrer avec plus de 3 octets. Le receveur DOIT ignorer les octets de bourrage.

Les tronçons définis dans SCTP sont décrits en détail au paragraphe 3.3. Les lignes directrices pour les extensions de tronçon définies par l'IETF se trouvent au au paragraphe 15.1 de ce document.

3.2.1 Format de paramètre facultatif/de longueur variable

Les valeurs de tronçon des tronçons de contrôle SCTP consistent en un en-tête spécifique du type de tronçon des champs exigés, suivi par zéro, un ou plusieurs paramètres. Les paramètres facultatifs et de longueur variable contenus dans un tronçon sont définis dans un format Type-Longueur-Valeur comme montré ci-dessous.



Type de paramètre de tronçon : 16 bits (entier non signé). Le champ Type est un identifiant de 16 bits du type de paramètre. Il prend une valeur de 0 à 65534. La valeur de 65535 est réservée pour des extensions définies par l'IETF. Les valeurs autres que celles définies dans des descriptions de tronçon SCTP spécifiques sont réservées à l'usage de l'IETF.

Longueur de paramètre : 16 bits (entier non signé). Le champ Longueur de paramètre contient la taille du paramètre en octets, incluant les champs Type de paramètre, Longueur de paramètre, et Valeur de paramètre. Donc, un paramètre avec un champ Valeur de paramètre de longueur zéro va avoir un champ Longueur de 4. La longueur de paramètre n'inclut aucun octet de bourrage.

Valeur de paramètre : longueur variable. Le champ Valeur de paramètre contient les informations réelles à transférer dans le paramètre.

La longueur totale d'un paramètre (incluant les champs Type, Longueur, et Valeur de paramètre) DOIT être un multiple de 4 octets. Si la longueur du paramètre n'est pas un multiple de 4 octets, l'envoyeur bourre le paramètre à la fin (c'est-à-dire, après le champ Valeur de paramètre) avec des octets de zéros. La longueur du bourrage n'est pas incluse dans le champ Longueur de paramètre. Un envoyeur NE DOIT PAS bourrer avec plus de 3 octets. Le receveur DOIT ignorer les octets de bourrage.

Les types de paramètre sont codés de telle façon que les 2 bits de plus fort poids spécifient l'action qui doit être entreprise si le point d'extrémité traitant ne reconnaît pas le type de paramètre.

00 - Arrêter le traitement de ce paramètre, ne pas traiter d'autres paramètres dans ce tronçon.
01 - Arrêter le traitement de ce paramètre, ne pas traiter d'autres paramètres dans ce tronçon, et faire rapport du paramètre non reconnu dans un "Type de paramètre non reconnu", comme décrit au paragraphe 3.2.2..
10 - Sauter ce paramètre et continuer le traitement.
11 - Sauter ce paramètre et continuer le traitement mais faire rapport du paramètre non reconnu dans un "Paramètre non reconnu", comme décrit au paragraphe 3.2.2.

Tableau 3 : Traitemet des paramètres inconnus

Noter que dans les quatre cas, quand un tronçon INIT ou INIT ACK est reçu, un tronçon INIT ou COOKIE ECHO est respectivement envoyé en réponse. Dans les cas 00 ou 01, le traitement des paramètres après le paramètre inconnu est annulé, mais aucun traitement déjà effectué n'est annulé.

Les paramètres SCTP actuels sont définis dans les paragraphes spécifiques de tronçon SCTP. Les règles pour les extensions de paramètre définies par l'IETF sont définies au paragraphe 15.2. Les types de paramètres DOIVENT être uniques sur tous les tronçons. Par exemple, le type de paramètre "5" est utilisé pour représenter une adresse IPv4 (voir au paragraphe 3.3.2.1). La valeur "5" est alors réservée à travers tous les tronçons pour représenter une adresse IPv4 et NE DOIT PAS être réutilisée avec une signification différente dans tout autre tronçon.

3.2.2 Rapport des paramètres non reconnus

Si le receveur d'un tronçon INIT détecte des paramètres non reconnus et doit en faire rapport conformément au paragraphe 3.2.1, il DOIT mettre le ou les paramètres "Paramètre non reconnu" dans le tronçon INIT ACK envoyé en réponse au tronçon INIT. Noter que si le receveur du tronçon INIT n'est pas en train d'établir une association (par exemple, du fait du manque de ressources) une cause d'erreur "Paramètre non reconnu" ne sera pas incluse dans un tronçon ABORT envoyé à l'envoyeur du tronçon INIT.

Si le receveur de tout autre tronçon (par exemple,INIT ACK) détecte des paramètres non reconnus et doit en faire rapport conformément au paragraphe 3.2.1, il DEVRAIT grouper le tronçon ERROR contenant la cause d'erreur "Paramètres non reconnus" avec le tronçon envoyé en réponse au tronçon (par exemple, COOKIE ECHO). Si le receveur d'un tronçon INIT

ACK ne peut pas grouper le tronçon COOKIE ECHO avec le tronçon ERROR, le tronçon ERROR PEUT être envoyé séparément mais pas avant que le COOKIE ACK ait été reçu.

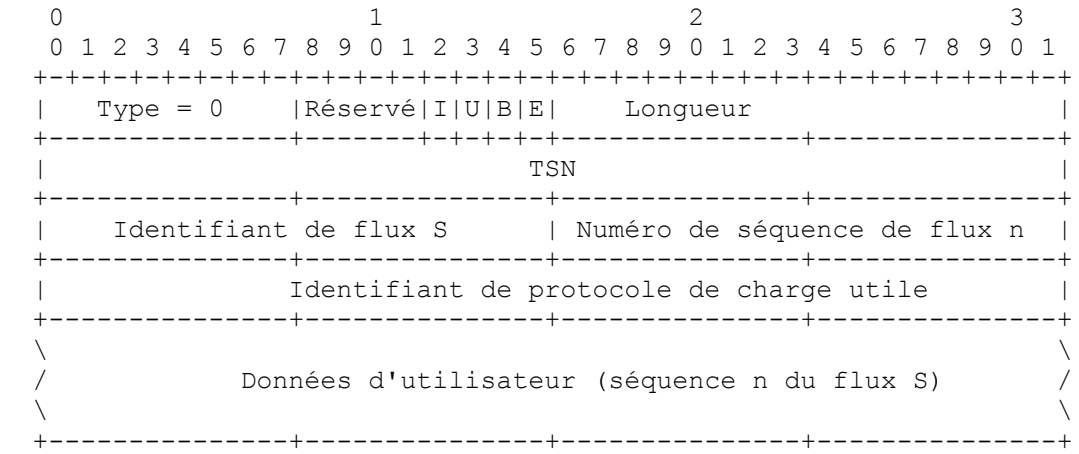
Note : Chaque fois qu'un COOKIE ECHO est envoyé dans un paquet, il DOIT être le premier tronçon.

3.3 Définition des tronçons SCTP

Cette section définit le format des différents types de tronçons SCTP.

3.3.1 Données de charge utile (DATA) (0)

Le format suivant DOIT être utilisé pour le tronçon DATA :



Réservé : 4 bits. Tous réglés à 0 à l'émission et ignorés par le receveur.

Bit I : 1 bit. Le bit (I)mmédiat PEUT être établi par l'envoyeur chaque fois que l'envoyeur d'un tronçon DATA peut bénéficier du renvoi sans délai du tronçon SACK correspondant. Voir la Section 4 de la [RFC7053] pour la discussion des avantages.

Bit U : 1 bit. C'est le bit Non ordonné ; réglé à 1, il indique que c'est un tronçon de données non ordonnées, et qu'il n'y a pas de numéro de séquence de flux alloué à ce tronçon DATA. Donc, le receveur DOIT ignorer le champ Numéro de séquence de flux. Après le réassemblage (si nécessaire) les tronçons DATA non ordonnés DOIVENT être envoyés à la couche supérieure par le receveur sans tenter de les réordonner. Si un message d'utilisateur non ordonné est fragmenté, chaque fragment du message DOIT avoir son bit U réglé à 1.

Bit B : 1 bit. Bit de début de fragment, si il est établi, il indique le premier fragment d'un message d'utilisateur.

Bit E : 1 bit. Bit de fin de fragment, si il est établi, il indique le dernier fragment d'un message d'utilisateur.

Longueur : 16 bits (entier non signé). Ce champ indique la longueur du tronçon DATA en octets depuis le début du champ Type jusqu'à la fin du champ Données d'utilisateur en excluant tout bourrage. Un tronçon DATA avec un octet de données d'utilisateur va avoir un champ Longueur réglé à 17 (indiquant 17 octets). Un tronçon DATA avec un champ Données d'utilisateur de longueur L va avoir le champ Longueur réglé à (16 + L) (indiquant 16 + L octets) où L DOIT être supérieur à 0

TSN : 32 bits (entier non signé). Cette valeur représente le TSN pour ce tronçon DATA. La gamme valide de TSN est de 0 à 4 294 967 295 (2**32 - 1). Le TSN revient à 0 après avoir atteint 4 294 967 295.

Identifiant de flux S : 16 bits (entier non signé). Identifie le flux auquel appartiennent les données d'utilisateur suivantes.

Numéro de séquence de flux n : 16 bits (entier non signé). Cette valeur représente le numéro de séquence de flux des données d'utilisateur suivantes au sein du flux S. La gamme valide est de 0 à 65 535. Quand un message d'utilisateur est fragmenté par SCTP pour le transport, le même numéro de séquence de flux DOIT être porté dans chacun des fragments du message.

Identifiant de protocole de charge utile : 32 bits (entier non signé). Cette valeur représente un identifiant de protocole spécifié par l'application (ou la couche supérieure). Cette valeur est passée à SCTP par sa couche supérieure et envoyée à son homologue. Cet identifiant n'est pas utilisé par SCTP mais peut être utilisé par certaines entités de réseau, ainsi que par l'application de l'homologue, pour identifier le type d'informations portées dans ce tronçon DATA. Ce champ DOIT être envoyé même dans les tronçons DATA fragmentés (pour s'assurer qu'il est disponible aux agents dans le réseau). Noter que ce champ n'est pas touché par une mise en œuvre de SCTP ; la couche supérieure est chargée pour l'hôte de toutes conversions de l'ordre des octets du réseau de ce champ. La valeur 0 indique qu'aucun identifiant d'application n'est spécifié par la couche supérieure pour ces données de charge utile.

Données d'utilisateur : longueur variable. Ce sont les données d'utilisateur de la charge utile. La mise en œuvre DOIT bourrer la fin des données jusqu'à une limite de 4 octets avec des octets de zéros. Aucun bourrage NE DOIT être inclus dans le champ Longueur. Un envoyeur NE DOIT jamais ajouter plus de 3 octets de bourrage.

Un message d'utilisateur non fragmenté DOIT avoir les deux bits B et E établis à 1. Régler les deux bits B et E à 0 indique un fragment intermédiaire d'un message d'utilisateur multi fragments, comme indiqué dans le tableau suivant :

B E	Description
1 0	Première pièce d'un message d'utilisateur fragmenté
0 0	Pièce intermédiaire d'un message d'utilisateur fragmenté
0 1	Dernière pièce d'un message d'utilisateur fragmenté
1 1	Message non fragmenté

Tableau 4 : Fanions de description de fragment

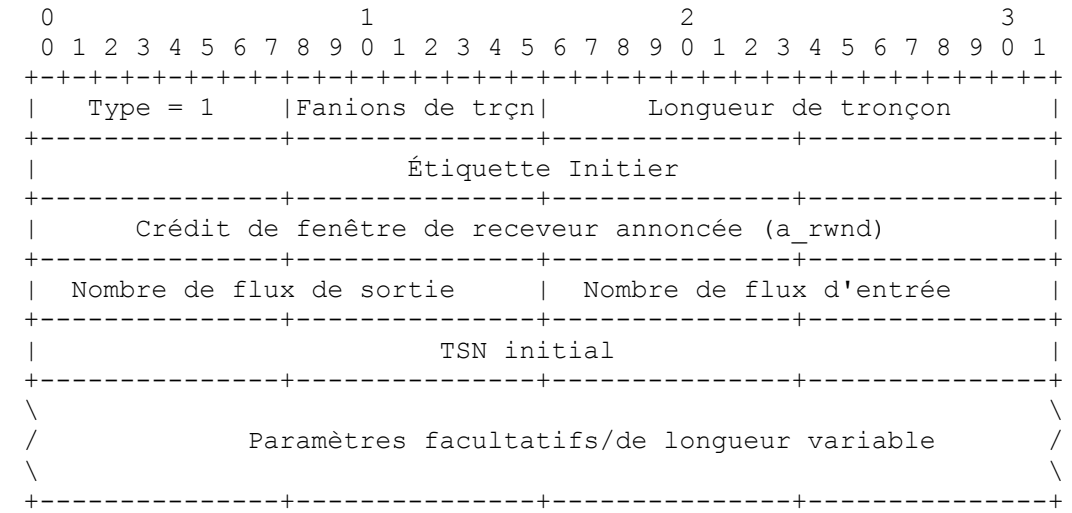
Quand un message d'utilisateur est fragmenté en plusieurs tronçons, les TSN sont utilisés par le receveur pour réassembler le message. Cela signifie que les TSN pour chaque fragment d'un message d'utilisateur fragmenté DOIVENT être strictement en séquence.

Les TSN des tronçons DATA envoyés DEVRAIENT être strictement en séquence.

Note : L'extension décrite dans la [RFC8260] peut être utilisée pour atténuer le blocage de tête de ligne lors du transfert de grands messages d'utilisateur.

3.3.2 Initialisation (INIT) (1)

Ce tronçon est utilisé pour initier une association SCTP entre deux points d'extrémité. Le format du tronçon INIT est montré ci-dessous:



Le tronçon INIT contient les paramètres suivants. Sauf mention contraire, chaque paramètre DOIT seulement être inclus une fois dans le tronçon INIT.

Paramèteres de longueur fixe	État
Étiquette Initier	Obligatoire
Crédit de fenêtre de receveur annoncée	Obligatoire
Nombre de flux de sortie	Obligatoire
Nombre de flux d'entrée	Obligatoire
TSN initial	Obligatoire

Tableau 5 : Paramètres de longueur fixe des tronçons INIT

Paramètres de longueur variable	État	Valeur de type
Adresse IPv4 (Note 1)	Facultatif	5
Adresse IPv6 (Note 1)	Facultatif	6
Préservation de Mouchard	Facultatif	9
Réservé pour capacité ECN (Note 2)	Facultatif	32768 (0x8000)
Adresse de nom d'hôte (Note 3)	Facultatif	11
Types d'adresse pris en charge (Note 4)	Facultatif	12

Tableau 6 : Paramètres de longueur variable des tronçons INIT

Note 1 : Les tronçons INIT peuvent contenir plusieurs adresses qui peuvent être IPv4 et/ou IPv6 dans toute combinaison.

Note 2 : Le champ Capacité ECN est réservé pour de futures utilisations de la notification explicite d'encombrement.

Note 3 : Un tronçon INIT NE DOIT PAS contenir plus d'un paramètre Adresse de nom d'hôte. Le receveur d'un tronçon INIT contenant un paramètre Adresse de nom d'hôte DOIT envoyer un tronçon ABORT et PEUT inclure une cause d'erreur "Adresse non résolvable".

Note 4 : Ce paramètre, quand il est présent, spécifie tous les types d'adresses que le point d'extrémité envoyeur peut supporter. L'absence de ce paramètre indique que le point d'extrémité envoyeur peut accepter tout type d'adresse.

Si un tronçon INIT est reçu avec tous les paramètres obligatoires qui sont spécifiés pour le tronçon INIT, le receveur DEVRAIT alors traiter le tronçon INIT et renvoyer un INIT ACK. Le receveur du tronçon INIT PEUT plus tard grouper un tronçon ERROR avec le tronçon COOKIE ACK. Cependant, des mises en œuvre restrictives PEUVENT renvoyer un tronçon ABORT en réponse au tronçon INIT.

Le champ Fanions de tronçon dans les tronçons INIT est réservé, et tous les bits dans ce champ DEVRAIENT être réglés à 0 par l'envoyeur et ignorés par le receveur.

Étiquette Initier : 32 bits (entier non signé). Le receveur du tronçon INIT (l'extrémité qui répond) enregistre la valeur du paramètre Étiquette Initier. Cette valeur DOIT être placée dans le champ Étiquette de vérification de chaque paquet SCTP que le receveur du tronçon INIT transmet dans cette association. Il est permis à l'étiquette Initier d'avoir toute valeur sauf 0. Voir au paragraphe 5.3.1 des détails sur le choix de la valeur de l'étiquette. Si la valeur de l'étiquette Initier dans un tronçon INIT reçu se trouve être 0, le receveur DOIT éliminer le paquet en silence.

Crédit de fenêtre de receveur annoncée (a_rwnd) : 32 bits (entier non signé). Cette valeur représente l'espace de mémoire tampon dédié, en nombre d'octets, que l'envoyeur du tronçon INIT a réservé en association avec cette fenêtre. Le crédit de fenêtre de receveur annoncée NE DOIT PAS être inférieur à 1500. Un receveur d'un tronçonn INIT avec une valeur de a_rwnd réglée à une valeur inférieure à 1500 DOIT éliminer le paquet, DEVRAIT envoyer un paquet en réponse contenant un tronçon ABORT et en utilisant l'étiquette Initier comme étiquette de vérification, et NE DOIT PAS changer l'état d'une association existante. Durant la vie de l'association, cet espace de mémoire tampon NE DEVRAIT PAS être diminué (c'est-à-dire que de la mémoire tampon dédiée soit retirée à cette association) ; cependant, un point d'extrémité PEUT changer la valeur d'un a_rwnd qu'il envoie dans les tronçons SACK.

Nombre de flux sortants (OS, *Outbound Stream*) : 16 bits (entier non signé). Définit le nombre de flux sortants que l'envoyeur de ce tronçon INIT souhaite créer dans cette association. La valeur de 0 NE DOIT PAS être utilisée. Un receveur d'un INIT dont la valeur d'OS est réglée à 0 DOIT éliminer le paquet, DEVRAIT envoyer un paquet en réponse contenant un tronçon ABORT et en utilisant l'étiquette Initier comme étiquette de vérification, et NE DOIT PAS changer l'état d'une association existante.

Nombre de flux entrants (MIS, *Inbound Stream*) : 16 bits (entier non signé). Définit le nombre maximum de flux que l'envoyeur de ce tronçon INIT permet à l'extrémité homologue de créer dans cette association. La valeur 0 NE DOIT

PAS être utilisée.

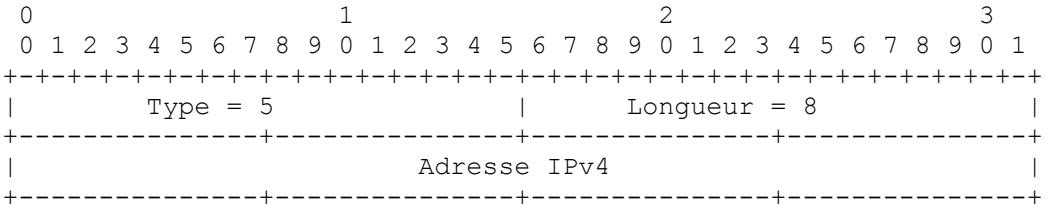
Note : Il n'y a pas de négociation du nombre réel de flux mais les deux points d'extrémité vont plutôt utiliser le min(demandé, offert). Voir les détails au paragraphe 5.1.1. Un receveur d'un tronçon INIT avec la valeur de MIS réglée à 0 DOIT éliminer le paquet, DEVRAIT envoyer en réponse un paquet contenant un tronçon ABORT et en utilisant l'étiquette Initier comme étiquette de vérification, et NE DOIT PAS changer l'état d'une association existante.

TSN initial (I-TSN) : 32 bits (entier non signé). Définit le TSN initial que l'envoyeur du tronçon INIT va utiliser. La gamme valide est de 0 à 4 294 967 295 et le TSN initial DEVRAIT être réglé à une valeur aléatoire dans cette gamme. Les méthodes décrites dans la [RFC4086] peuvent être utilisées pour rendre aléatoire le TSN initial.

3.3.2.1 Paramètres facultatifs/de longueur variable dans les tronçons INIT

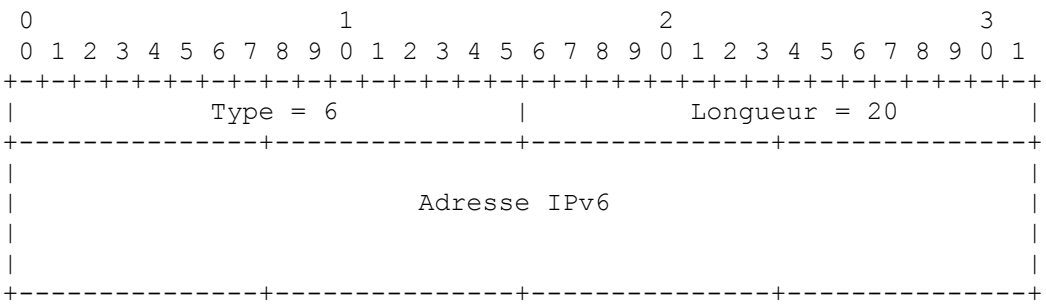
Les paramètres suivants ont le format de Type-Longueur-Valeur défini au paragraphe 3.2.1. Tous les champs Type-Longueur-Valeur DOIVENT venir après les champs de longueur fixe. (Les champs de longueur fixe sont définis à la section précédente.

3.3.2.1.1 Adresse IPv4 (5)



Adresse IPv4 : 32 bits (entier non signé). Contient une adresse IPv4 du point d'extrémité envoyeur. Elle est codée en binaire.

3.3.2.1.2 Adresse IPv6 (6)



Adresse IPv6 : 128 bits (entier non signé). Contient une adresse IPv6 [RFC8200] du point d'extrémité envoyeur. Elle est codée en binaire.

Un envoyeur NE DOIT PAS utiliser une adresse IPv6 transposée en IPv4 [RFC4291], mais DEVRAIT plutôt utiliser un paramètre Adresse IPv4 pour une adresse IPv4. Combinée au numéro d'accès de source dans l'en-tête commun SCTP, la valeur passée dans un paramètre Adresse IPv4 ou IPv6 indique une adresse de transport que l'envoyeur du tronçon INIT va prendre en charge pour l'association à initier. C'est-à-dire, durant la vie de cette association, cette adresse IP peut apparaître dans le champ Adresse de source d'un datagramme IP envoyé de l'envoyeur du tronçon INIT, et peut être utilisée comme adresse de destination d'un datagramme IP envoyé du receveur du tronçon INIT.

Plus d'un paramètre Adresse IP peut être inclus dans un tronçon INIT quand l'envoyeur INIT est multi rattachements. De plus, un point d'extrémité multi rattachements peut avoir accès à différents types de réseaux ; donc, plus d'un type d'adresse peut être présent dans un tronçon INIT, c'est-à-dire, des adresses IPv4 et IPv6 sont permises dans le même tronçon INIT.

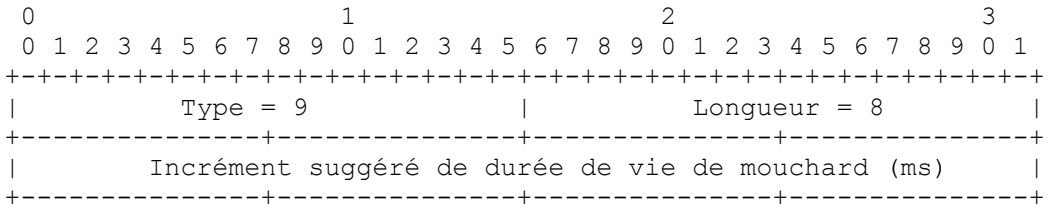
Si le tronçon INIT contient au moins un paramètre Adresse IP, l'adresse de source du IP datagramme IP qui contient le tronçon INIT et toutes adresses supplémentaires fournies dans le INIT peuvent alors être utilisées comme destinations par le point d'extrémité qui reçoit le tronçon INIT. Si le tronçon INIT ne contient aucun paramètre Adresse IP, le point

d'extrémité recevant le tronçon INIT DOIT utiliser l'adresse de source associée au datagramme IP reçu comme seule adresse de destination pour l'association.

Noter que n'utiliser aucun paramètre Adresse IP dans le tronçon INIT et INIT ACK est une façon pour rendre plus probable le fonctionnement d'une association à travers un boîtier de traduction d'adresse réseau (NAT, *Network Address Translation*).

3.3.2.1.3 Préservation du mouchard (9)

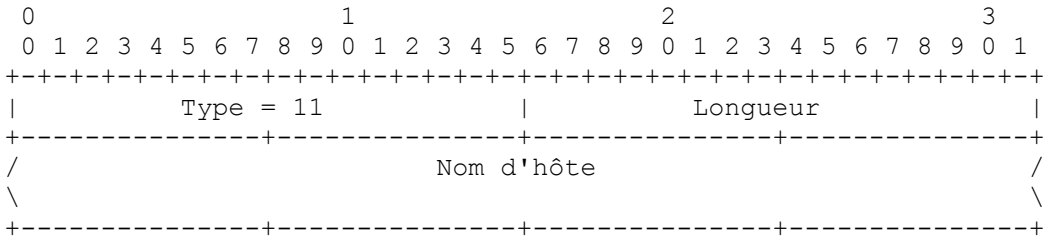
L'envoyeur de l'INIT utilise ce paramètre pour suggérer au receveur du tronçon INIT une durée de vie plus longue du mouchard d'état.



Incrément suggéré de durée de vie de mouchard : 32 bits (entier non signé). Ce paramètre indique au receveur de combien d'incréments en millisecondes l'envoyeur souhaite que le receveur ajoute à sa durée de vie par défaut du mouchard. Ce paramètre facultatif PEUT être ajouté au tronçon INIT par l'envoyeur quand il tente à nouveau d'établir une association à un homologue avec qui une précédente tentative d'établissement d'association a échoué à cause d'une erreur de fonctionnement de mouchard périmé. Le receveur PEUT choisir d'ignorer l'augmentation de durée de vie de mocuhard suggérée pour ses propres raisons de sécurité.

3.3.2.1.4 Adresse de nom d'hôte (11)

L'envoyeur d'un tronçon INIT ou INIT ACK NE DOIT PAS inclure ce paramètre. L'usage du paramètre Adresse de nom d'hôte est déconseillé. Le receveur d'un tronçon INIT ou INIT ACK contenant un paramètre Adresse de nom d'hôte DOIT envoyer un tronçon ABORT et PEUT inclure une cause d'erreur "Adresse non résolvable".

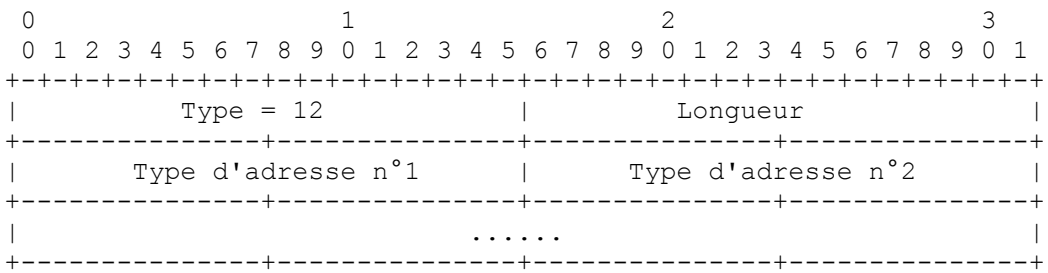


Nom d'hôte : longueur variable. Ce champ contient un nom d'hôte dans "syntaxe de nom d'hôte" selon le paragraphe 2.1 de la [RFC1123]. La méthode pour résoudre le nom d'hôte sort du domaine d'application de SCTP.

Au moins une terminaison nulle est incluse dans la chaîne Nom d'hôte et DOIT être incluse dans la longueur.

3.3.2.1.5 Types d'adresse pris en charge (12)

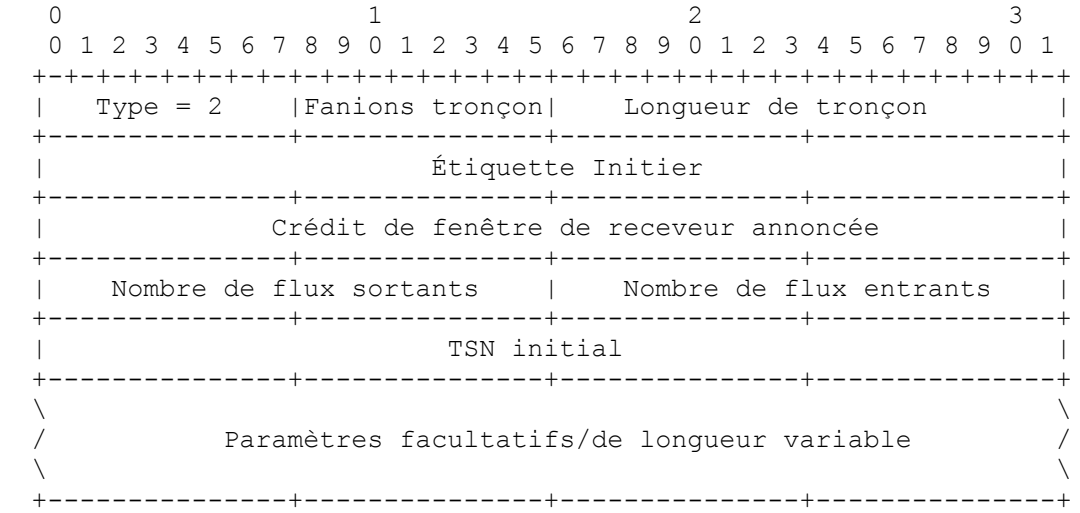
L'envoyeur du tronçon INIT utilise ce paramètre pour faire la liste de tous les types d'adresse qu'il peut prendre en charge.



Type d'adresse : 16 bits (entier non signé). Ceci est rempli avec la valeur de type de la TLV Adresse correspondante (par exemple, 5 pour indiquer IPv4, et 6 pour indiquer IPv6). La valeur indiquant le paramètre Adresse de nom d'hôte NE DOIT PAS être utilisée lors de l'envoi de ce paramètre et DOIT être ignorée à la réception de ce paramètre.

3.3.3 Accusé de réception d'initialisation (INIT ACK) (2)

Le tronçon INIT ACK est utilisé pour accuser réception de l'initialisation d'une association SCTP. Le format du tronçon INIT ACK est montré ci-dessous :



La partie paramètre du INIT ACK est formatée de façon similaire à celle du tronçon INIT. Les paramètres suivants sont spécifiés pour le tronçon INIT ACK :

Paramètres de longueur fixe	**État**
Étiquette Initier	Obligatoire
Crédit de fenêtre de réception annoncée	Obligatoire
Nombre de flux sortants	Obligatoire
Nombre de flux entrants	Obligatoire
TSN initial	Obligatoire

Table 7 : Paramètres de longueur fixe des tronçons INIT ACK

Il utilise deux paramètres de longueur variable supplémentaires : Mouchard d'état et Paramètre non reconnu.

Paramètres de longueur variable	**État**	**Valeur de type**
Mouchard d'état	Obligatoire	7
Adresse IPv4 (Note 1)	Facultatif	5
Adresse IPv6 (Note 1)	Facultatif	6
Paramètre non reconnu	Facultatif	8
Réservé pour capacité ECN (Note 2)	Facultatif	32768 (0x8000)
Adresse de nom d'hôte (Note 3)	Facultatif	11

Tableau 8 : Paramètres de longueur variable des tronçons INIT ACK

Note 1 : Les tronçons INIT ACK peuvent contenir un nombre quelconque de paramètres Adresse IP qui peuvent être IPv4 et/ou IPv6 dans toutes combinaisons.

Note 2 : Le champ Capacité ECN est réservé pour une future utilisation de la Notification explicite d'encombrement.

Note 3 : Un tronçon INIT ACK NE DOIT PAS contenir le paramètre Adresse de nom d'hôte. Le receveur d'un tronçon INIT ACK contenant un paramètre Adresse de nom d'hôte DOIT envoyer un tronçon ABORT et PEUT inclure une cause d'erreur "Adresse non résolvable".

Le Champ Fanions de tronçons dans les tronçons INIT ACK est réservé, et tous les bits qu'il contient DEVRAIENT être

réglés à 0 par l'envoyeur et ignorés par le receveur.

Étiquette Initier : 32 bits (entier non signé). Le receveur du tronçon INIT ACK enregistre la valeur du paramètre Étiquette Initialiser. Cette valeur DOIT être placée dans le champ Étiquette de vérification de chaque paquet SCTP que le receveur du tronçon INIT ACK transmet dans cette association. L'étiquette Initier NE DOIT PAS prendre la valeur 0. Voir au paragraphe 5.3.1 les détails du choix de la valeur de l'étiquette Initier. Si un point d'extrémité dans l'état COOKIE-WAIT reçoit un tronçon INIT ACK avec l'étiquette Initier réglée à 0, il DOIT détruire le TCB et DEVRAIT envoyer un tronçon ABORT avec le bit T établi. Si un tel tronçon INIT ACK est reçu dant tout état autre que CLOSED ou COOKIE-WAIT, il DEVRAIT être éliminé en silence (voir le paragraphe 5.2.3).

Crédit de fenêtre de receveur annoncée (a_rwnd) : 32 bits (entier non signé). Cette valeur représente l'espace de mémoire tampon dédié, en nombre d'octets, que l'envoyeur du tronçon INIT ACK a réservé dans l'association à cette fenêtre. Le crédit de fenêtre de receveur annoncée NE DOIT PAS être inférieur à 1500. Un receveur d'un tronçon INIT ACK avec la valeur de a_rwnd réglée à une valeur inférieure à 1500 DOIT éliminer le paquet, DEVRAIT envoyer un paquet en réponse contenant un tronçon ABORT et en utilisant l'étiquette Initier comme étiquette de vérification, et NE DOIT PAS changer l'état d'une association existante. Durant la vie de l'association, cet espace de mémoire tampon NE DEVRAIT PAS être diminué (c'est-à-dire que des mémoires tampon dédiées soient retirées ce cette association) ; cependant, un point d'extrémité PEUT changer la valeur d'un a_rwnd qu'il envoie dans les tronçons SACK.

Nombre de flux sortants (OS) : 16 bits (entier non signé). Définit le nombre de flux sortants que l'envoyeur de ce tronçon INIT ACK souhaite créer dans cette association. La valeur de 0 NE DOIT PAS être utilisée, et la valeur NE DOIT PAS être supérieure à la valeur de MIS envoyée dans le tronçon INIT. Si un point d'extrémité dans l'état COOKIE-WAIT reçoit un tronçon INIT ACK avec la valeur d'OS réglée à 0, il DOIT détruire le TCB et DEVRAIT envoyer un tronçon ABORT. Si un tel tronçon INIT ACK est reçu dans tout état autre que CLOSED ou COOKIE-WAIT, il DEVRAIT être éliminé en silence (voir le paragraphe 5.2.3).

Nombre de flux entrants (MIS) : 16 bits (entier non signé). Définit le nombre maximum de flux que l'envoyeur de ce tronçon INIT ACK permet à l'extrémité homologue de créer dans cette association. La valeur 0 NE DOIT PAS être utilisée.

Note : Il n'y a pas de négociation du nombre réel de flux mais les deux points d'extrémité vont plutôt utiliser le min(demandé, offert). Voir les détails au paragraphe 5.1.1.

Si un point d'extrémité dans l'état COOKIE-WAIT reçoit un tronçon INIT ACK avec la valeur de MIS réglée à 0, il DOIT détruire le TCB et DEVRAIT envoyer un tronçon ABORT. Si un tel tronçon INIT ACK est reçu dans tout état autre que CLOSED ou COOKIE-WAIT, il DEVRAIT être éliminé en silence (voir le paragraphe 5.2.3).

TSN initial (I-TSN) : 32 bits (entier non signé). Définit le TSN initial que l'envoyeur du tronçon INIT ACK va utiliser initialement. La gamme valide est de 0 à 4 294 967 295 et le TSN initial DEVRAIT être réglé à une valeur aléatoire dans cette gamme. Les méthodes décrites dans la [RFC4086] peuvent être utilisées pour le choix du TSN initial aléatoire.

Note de mise en œuvre : Une mise en œuvre DOIT être prête à recevoir un tronçon INIT ACK assez grand (de plus de 1500 octets) du fait de la taille variable du mouchard d'état et de la liste variable d'adresses. Par exemple si celui qui répond au tronçon 'INIT a 1000 adresses IPv4 qu'il souhaite envoyer, il aura besoin d'au moins 8 000 octets pour coder cela dans le tronçon INIT ACK.

Si un tronçon INIT ACK est reçu avec tous les paramètres obligatoires qui sont spécifiés pour le tronçon INIT ACK, le receveur DEVRAIT alors traiter le tronçon INIT ACK et renvoyer un tronçon COOKIE ECHO. Le receveur du tronçon INIT ACK PEUT grouper un tronçon ERROR avec le tronçon COOKIE ECHO. Cependant, des mises en œuvre restrictives PEUVENT renvoyer un tronçon ABORT en réponse au tronçon INIT ACK.

En combinaison avec l'accès de source porté dans l'en-tête commun SCTP, chaque paramètre Adresse IP dans le tronçon INIT ACK indique au receveur du tronçon INIT ACK une adresse de transport valide prise en charge par l'envoyeur du tronçon INIT ACK pour la durée de vie de l'association initialisée.

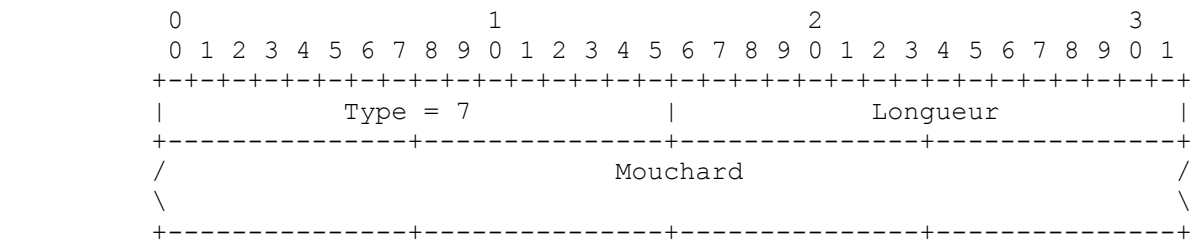
Si le tronçon INIT ACK contient au moins un paramètre Adresse IP, l'adresse de source du datagramme IP contenant le INIT ACK et toutes adresses supplémentaires fournies dans le tronçon INIT ACK PEUVENT alors être utilisées comme destinations par le receveur du tronçon INIT ACK. Si le tronçon INIT ACK ne contient aucun paramètre Adresse IP, le receveur du tronçon INIT ACK DOIT utiliser l'adresse de source associée au datagramme IP reçu comme seule adresse de destination pour l'association.

Les paramètres Mouchard d'état et Paramètres non reconnus utilisent le format de Type-Longueur-Valeur défini au paragraphe 3.2.1 et décrit ci-dessous. Les autres champs sont définis de la même façon que leur contrepartie dans le tronçon INIT.

3.3.3.1 Paramètres facultatifs ou de longueur variable dans les tronçons INIT ACK

Les paramètres Mouchard d'état et Paramètre non reconnu utilisent le format de Type-Longueur-Valeur, défini au paragraphe 3.2.1, et sont décrits ci-dessous. Le paramètre Adresse IPv4 est décrit au paragraphe 3.3.2.1.1, et le paramètre Adresse IPv6 est décrit au paragraphe 3.3.2.1.2. Le paramètre Adresse de nom d'hôte est décrit au paragraphe 3.3.2.1.4 et NE DOIT PAS être inclus dans un tronçon INIT ACK. Tous les champs de Type-Longueur-Valeur DOIVENT être placés après les champs de longueur fixe. (Les champs de longueur fixe sont définis au paragraphe précédent.)

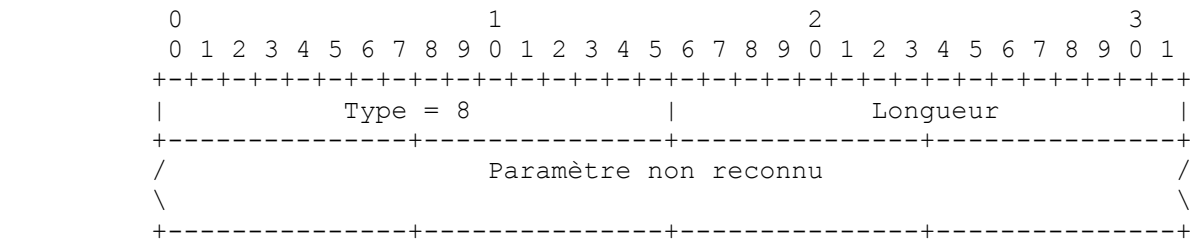
3.3.3.1.1 Mouchard d'état (7)



Mouchard : longueur variable. Cette valeur de paramètre DOIT contenir toutes les informations d'état et de paramètre nécessaires pour que l'envoyeur de ce tronçon INIT ACK crée l'association, ainsi qu'un code d'authentification de message (MAC, *Message Authentication Code*). Voir au paragraphe 5.1.3 les détails de la définition du mouchard d'état.

3.3.3.1.2 Paramètre non reconnu (8)

Ce paramètre est retourné à l'origine du tronçon INIT quand le tronçon INIT contient un paramètre non reconnu dont le type indique qu'il DEVRAIT être rapporté à l'envoyeur.



Paramètre non reconnu : longueur variable. Le champ Valeur de paramètre va contenir un paramètre non reconnu copié du tronçon INIT complet avec les champs Type, Longueur, et Valeur de paramètre.

3.3.4 Accusé de réception sélectif (SACK) (3)

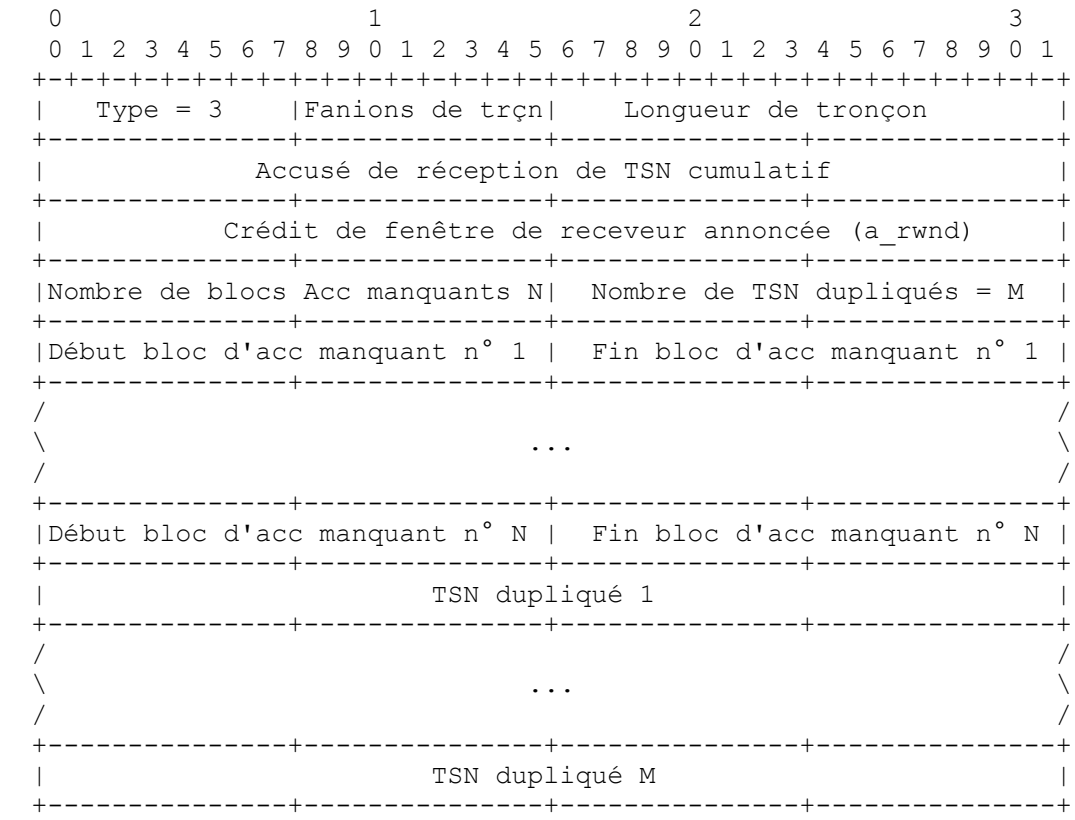
Ce tronçon est envoyé au point d'extrémité homologue pour accuser réception des tronçons DATA reçus et informer le point d'extrémité homologue de trous dans les sous séquences de tronçons DATA reçus comme représentés par leurs TSN.

Le tronçon SACK DOIT contenir les champs Accusé de réception de TSN cumulatif, Crédit de fenêtre de réception annoncé (a_rwnd), Nombre de blocs d'accusé de réception manquants, et Nombre de TSN dupliqués.

Par définition, la valeur du paramètre Accusé de réception de TSN cumulatif est le dernier TSN reçu avant une cassure dans la séquence des TSN reçus ; la prochaine valeur de TSN suivant celle là n'a pas encore été reçue au point d'extrémité qui envoie le tronçon SACK. Ce paramètre accuse donc réception de tous les TSN inférieurs ou égaux à cette valeur. Le traitement d'un a_rwnd par le receveur du SACK est discuté en détails au paragraphe 6.2.1.

Le tronçon SACK contient aussi zéro, un ou plusieurs blocs d'accusé de réception manquants. Chaque bloc d'accusé de réception manquant accuse réception d'une sous séquence des TSN reçus à la suite d'une coupure dans la séquence des TSN

reçus. Les blocs d'accusé de réception manquants DEVRAIENT être isolés. Cela signifie que le TSN juste avant chaque bloc d'accusé de réception manquant et le TSN juste après chaque bloc d'accusé de réception manquant n'ont pas été reçus. Par définition, tous les TSN acquittés par les blocs d'accusé de réception manquants sont supérieurs à la valeur de l'accusé de réception de TSN cumulatif.



Fanions de tronçon : 8 bits. Réglé tout à 0 à l'émission et ignoré à réception.

Accusé de réception de TSN cumulatif : 32 bits (entier non signé). Le plus grand TSN, tel que tous les TSN qui lui sont inférieurs ou égaux ont été reçus et que le suivant n'a pas été reçu. Dans le cas où aucun tronçon DATA n'a été reçu, cette valeur est réglée au TSN initial de l'homologue moins un.

Crédit de fenêtre de receveur annoncée (a_rwnd) : 32 bits (entier non signé). Ce champ indique en octets l'espace mis à jour de mémoire tampon de réception de l'envoyeur de ce tronçon SACK ; voir les détails au paragraphe 6.2.1.

Nombre de blocs d'accusé de réception manquants : 16 bits (entier non signé). Indique le nombre de blocs d'accusé de réception manquants inclus dans ce tronçon SACK.

Nombre de TSN dupliqués : 16 bits. Ce champ contient le nombre de TSN dupliqués que le point d'extrémité a reçus. Chaque TSN dupliqué est mentionné à la suite de la liste des blocs d'accusé de réception manquants.

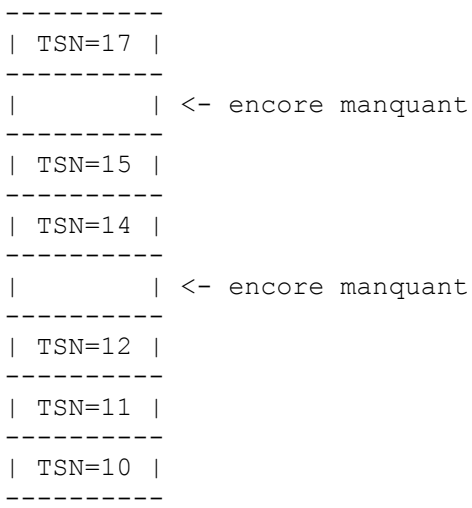
Blocs d'accusé de réception manquants : Ces champs contiennent les blocs d'accusé de réception manquants. Ils sont répétés pour chaque bloc d'accusé de réception manquant jusqu"au nombre de blocs d'accusé de réception manquants défini dans le champ Nombre de blocs d'accusé de réception manquants. Tous les tronçons DATA avec des TSN supérieurs ou égaux à (Accusé de réception de TSN cumulatif + Début de bloc d'accusé de réception manquant) et inférieurs ou égaux à (Accusé de réception de TSN cumulatif + Fin de bloc d'accusé de réception manquant) de chaque bloc d'accusé de réception manquant sont supposés avoir été reçus correctement.

Début de bloc d'accusé de réception manquant : 16 bits (entier non signé). Indique le décalage de début de TSN pour ce bloc d'accusé de réception manquant. Pour calculer le numéro de TSN réel, l'accusé de réception de TSN cumulatif est ajouté à ce décalage. Ce TSN calculé identifie le plus bas TSN reçu dans ce bloc d'accusé de réception manquant.

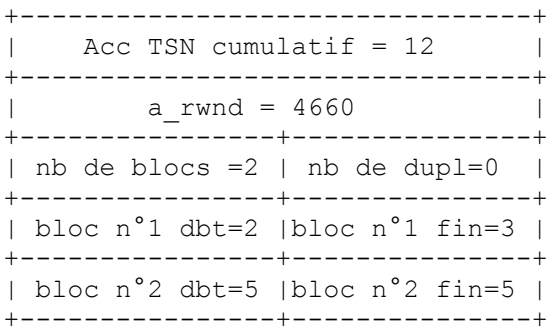
Fin de bloc d'accusé de réception manquant : 16 bits (entier non signé). Indique le décalage de fin de TSN pour ce bloc d'accusé de réception manquant. Pour calculer le numéro de TSN réel, l'accusé de réception de TSN cumulatif est ajouté

à ce décalage. Ce TSN calculé identifie le TSN du dernier tronçon DATA reçu dans ce bloc d'accusé de réception manquant.

Par exemple, supposons que le receveur ait les tronçons DATA suivants qui viennent d'arriver au moment où il décide d'envoyer un accusé de réception sélectif,



ensuite la partie paramètre du tronçon SACK DOIT être construite comme suit (en supposant que le nouveau a_rwnd est réglé à 4660 par l'envoyeur) :



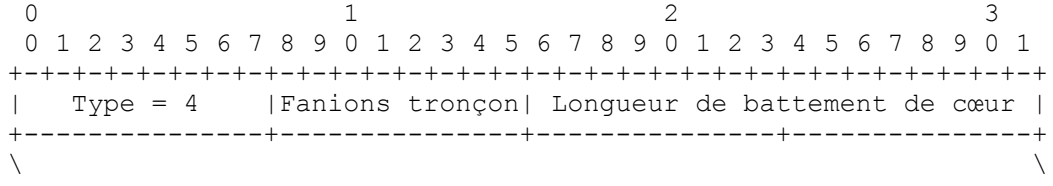
TSN dupliqué : 32 bits (entier non signé). Indique le nombre de fois qu'un TSN a été reçu dupliqué depuis l'envoi du dernier tronçon SACK. Chaque fois qu'un receveur obtient un TSN dupliqué (avant d'envoyer le tronçon SACK) il l'ajoute à la liste des dupliqués. Le compte des dupliqués est réinitialisé à zéro après chaque envoi de tronçon SACK.

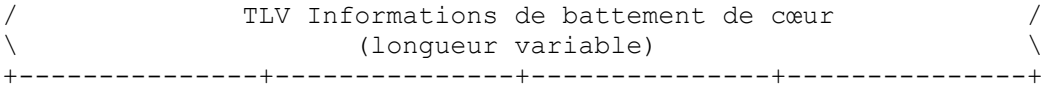
Par exemple, si un receveur obtenait le TSN 19 trois fois, il mentionnerait deux fois 19 dans le tronçon SACK sortant. Après l'envoi du tronçon SACK, si il a reçu encore un TSN 19 de plus, il va mentionner 19 comme dupliqué une fois dans le prochain tronçon SACK sortant.

3.3.5 Demande de battement de cœur (HEARTBEAT) (4)

Un point d'extrémité DEVRAIT envoyer un tronçon HEARTBEAT (HB) à son point d'extrémité homologue pour sonder l'accessibilité d'une adresse de transport de destination particulière définie dans la présente association.

Le champ Paramètre contient les informations de battement de cœur, qui sont une structure de données opaques de longueur variable comprise seulement par l'envoyeur.





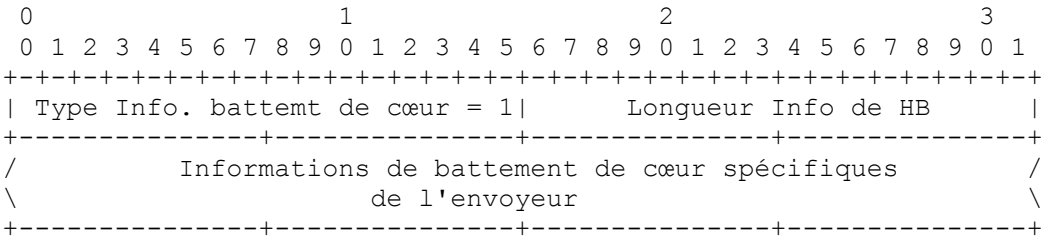
Fanions de tronçon : 8 bits. Réglé à zéro à l'émission et ignoré à réception.

Longueur de battement de cœur : 16 bits (entier non signé). Réglé à la taille du tronçon en octets, incluant l'en-tête de tronçon et le champ Informations de battement de cœur.

Informations de battement de cœur : longueur variable. Défini comme un paramètre de longueur variable en utilisant le format décrit au paragraphe 3.2.1, c'est-à-dire :

Paramètres variables	**État**	**Valeur de type**
Informations de battement de cœur	Obligatoire	1

Tableau 9 : Paramètres de longueur variable des tronçons Battement de cœur

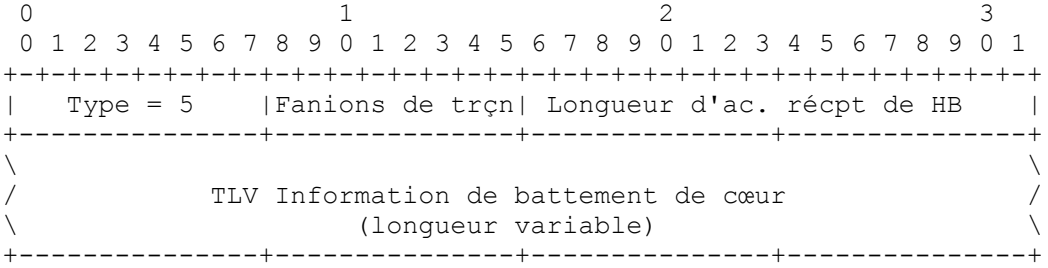


Le champ Informations de battement de cœur spécifiques de l'envoyeur DEVRAIT inclure des informations sur l'heure en cours chez l'envoyeur quand ce tronçon HEARTBEAT est envoyé et sur l'adresse de transport de destination à laquelle ce tronçon HEARTBEAT est envoyé (voir au paragraphe 8.3). Cette information est simplement reflétée par le receveur dans le tronçon HEARTBEAT ACK (voir au paragraphe 3.3.6). Noter aussi que le tronçon HEARTBEAT est à la fois pour vérifier l'accessibilité et pour la vérification du chemin (voir au paragraphe 5.4). Quand un tronçon HEARTBEAT est utilisé pour la vérification de chemin, il DOIT contenir un nom occasionnel aléatoire de 64 bits ou plus (la RFC4086] fournit des lignes directrices pour cela).

3.3.6 Accusé de réception de battement de cœur (HEARTBEAT ACK) (5)

Un point d'extrémité DOIT envoyer ce tronçon à son point d'extrémité homologue en réponse à un tronçon HEARTBEAT (voir au paragraphe 8.3). Un paquet contenant le tronçon HEARTBEAT ACK est toujours envoyé à l'adresse IP de source du datagramme IP contenant le tronçon HEARTBEAT auquel cet accusé de réception répond.

Le champ de paramètre contient une structure de données opaques de longueur variable.



Fanions de tronçon : 8 bits. Réglé à zéro à l'émission et ignoré à réception.

Longueur d'accusé de réception de battement de cœur : 16 bits (entier non signé). Réglé à la taille du tronçon en octets, incluant l'en-tête de tronçon et le champ Informations de battement de cœur.

Informations de battement de cœur : longueur variable. Ce champ DOIT contenir le paramètre Informations de battement de cœur (comme défini au paragraphe 3.3.5) de la demande de battement de cœur à laquelle répond cet accusé de réception de battement de cœur.

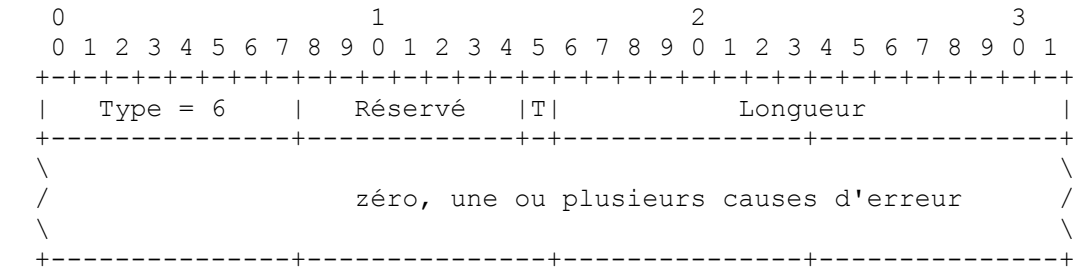
Paramètres variables	État	Valeur de type
Informations de battement de cœur	Obligatoire	1

Tableau 10 : Paramètres de longueur variable des tronçons Battement de cœur

3.3.7 Interruption d'association (ABORT) (6)

Le tronçon ABORT est envoyé à l'homologue d'une association pour clore l'association. Le tronçon ABORT PEUT contenir des causes d'erreur pour informer le receveur de la raison de l'interruption. Les tronçons DATA NE DOIVENT PAS être regroupés avec des tronçons ABORT. Les tronçons de contrôle (sauf INIT, INIT ACK, et SHUTDOWN COMPLETE) PEUVENT être regroupés avec un tronçon ABORT, mais ils DOIVENT être placés avant le tronçon ABORT dans le paquet SCTP ; autrement, ils vont être ignorés par le receveur.

Si un point d'extrémité reçoit un tronçon ABORT avec une erreur de format ou si aucun TCB n'est trouvé, il DOIT l'éliminer en silence. De plus, dans toutes les circonstances, un point d'extrémité qui reçoit un tronçon ABORT NE DOIT PAS répondre à ce tronçon ABORT par l'envoi d'un autre tronçon ABORT de sa part.



Fanions de tronçon : 8 bits

Réservé : 7 bits. Réglé à zéro à l'émission et ignoré à réception.

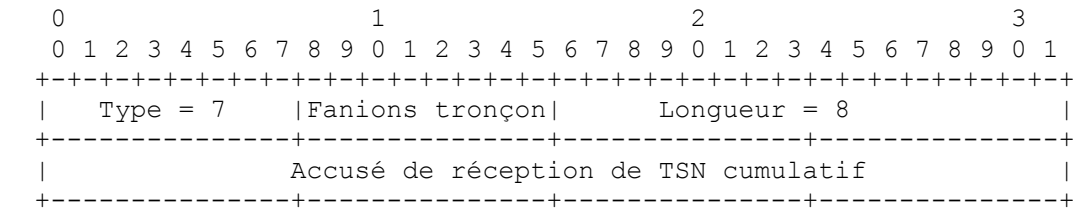
Bit T : 1 bit. Le bit T est réglé à 0 si l'envoyeur a rempli l'étiquette de vérification attendue par l'homologue. Si l'étiquette de vérification est reflétée, le bit T DOIT être réglé à 1. Refléter signifie que l'étiquette de vérification envoyé est la même que celle reçue.

Longueur : 16 bits (entier non signé). Réglé à la taille du tronçon en octets, incluant l'en-tête de tronçon et tous les champs Cause d'erreur présents. Voir au paragraphe 3.3.10 les définitions de cause d'erreur.

Note : Des règles particulières s'appliquent à ce tronçon pour la vérification ; voir les détails au paragraphe 8.5.1.

3.3.8 Fermeture d'association (SHUTDOWN) (7)

Un point d'extrémité dans une association DOIT utiliser ce tronçon pour initier une fermeture en douceur de l'association avec son homologue. Ce tronçon a le format suivant.



Fanions de tronçon : 8 bits. Réglé à 0 à l'émission et ignoré à réception.

Longueur : 16 bits (entier non signé). Indique la longueur du paramètre. Réglé à 8.

Accusé de réception de TSN cumulatif : 32 bits (entier non signé). Le plus grand TSN, tel que tous les TSN qui lui sont inférieurs ou égaux ont été reçus et que le prochain n'a pas été reçu.

Note : Comme le tronçon SHUTDOWN ne contient pas de bloc d'accusé de réception manquant, il ne peut pas être utilisé

pour accuser réception des TSN reçus déclassés. Dans un tronçon SACK, l'absence de bloc d'accusé de réception manquant précédemment inclus indique que le receveur des données a renoncé aux tronçons DATA associés.

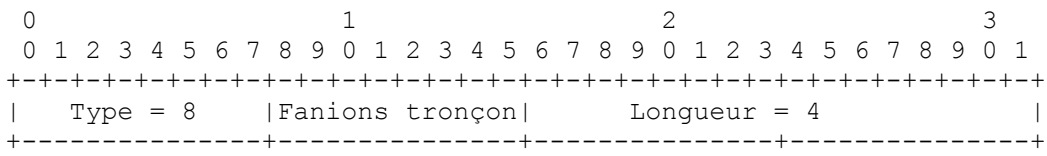
Comme le tronçon SHUTDOWN ne contient pas de bloc d'accusé de réception manquant, le receveur du tronçon SHUTDOWN NE DOIT PAS interpréter l'absence de bloc d'accusé de réception manquant comme une renonciation. (Voir au paragraphe 6.2 des informations sur la renonciation.)

L'envoyeur du tronçon SHUTDOWN PEUT le grouper avec un tronçon SACK pour indiquer des trous dans les TSN reçus.

3.3.9 Accusé de réception de fermeture (SHUTDOWN ACK) (8)

Ce tronçon DOIT être utilisé pour accuser réception du tronçon SHUTDOWN à l'achèvement du processus de fermeture ; voir les détails au paragraphe 9.2.

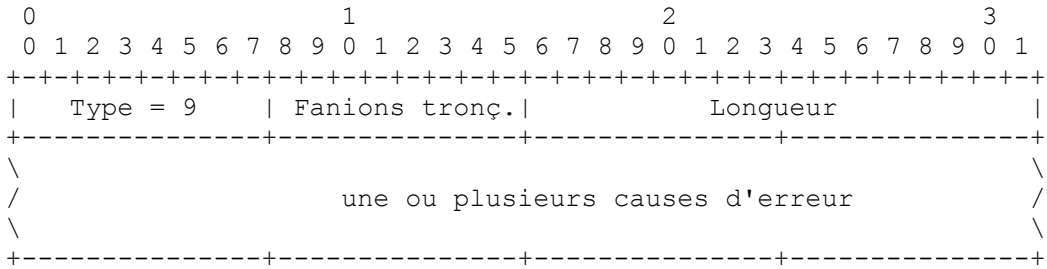
Le tronçon SHUTDOWN ACK n'a pas de paramètre.



Fanions tronçon : 8 bits. Réglé à 0 à l'émission et ignoré à réception.

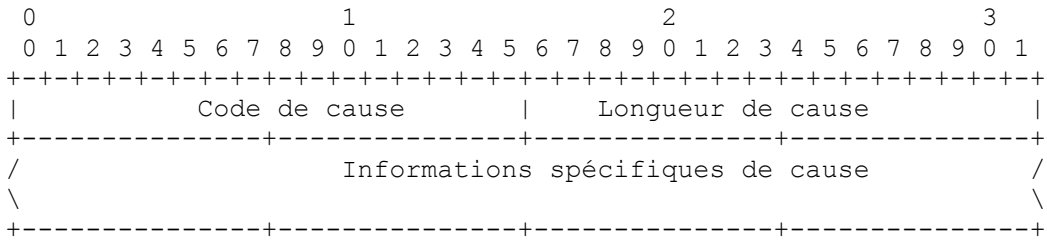
3.3.10 Erreur de fonctionnement (ERROR) (9)

Un point d'extrémité envoie ce tronçon à son point d'extrémité homologue pour lui notifier certaines conditions d'erreur. Il contient une ou plusieurs causes d'erreur. Une erreur de fonctionnement n'est pas considérée comme fatale en et par elle-même, mais la cause d'erreur correspondante PEUT être utilisée avec un tronçon ABORT pour rapporter une condition fatale. Un tronçon ERREUR a le format suivant :



Fanions tronçon : 8 bits. Réglé à 0 à l'émission et ignoré à réception.

Longueur : 16 bits (entier non signé). Réglé à la taille du tronçon en octets, incluant l'en-tête de tronçon et tous les champs Cause d'erreur présents. Les causes d'erreur sont définies comme des paramètres de longueur variable utilisant le format décrit au paragraphe 3.2.1, c'est-à-dire :



Code de cause : 16 bits (entier non signé). Définit le type de condition d'erreur rapportée.

Valeur du code de cause	Code de cause
1	Identifiant de flux invalide
2	Paramètre obligatoire manquant

3	Erreur de mouchard périmé
4	Plus de ressources
5	Adresse non résolvable
6	Type de tronçon non reconnu
7	Paramètre obligatoire invalide
8	Paramètres non reconnus
9	Pas de données d'utilisateur
10	Mouchard reçu pendant la fermeture
11	Redémarrage d'une association avec de nouvelles adresses
12	Interruption à l'initiative de l'utilisateur
13	Violation du protocole

Table 11 : Codes de cause

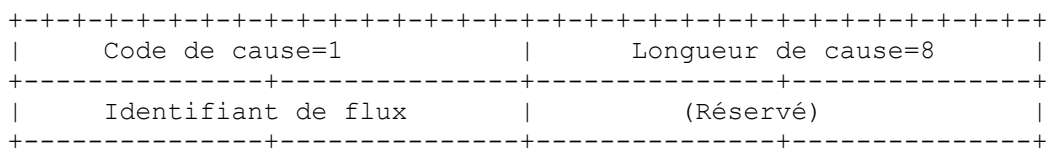
Longueur de cause : 16 bits (entier non signé). Réglé à la taille du paramètre en octets, incluant les champs Code de cause, Longueur de cause, et Informations spécifiques de cause.

Informations spécifiques de cause : longueur variable. Ce champ porte les détails de la condition d'erreur.

Les paragraphes 3.3.10.1 à 3.3.10.13 définissent les causes d'erreur pour SCTP. Les lignes directrices de l'IETF pour définir de nouvelles valeurs de cause d'erreur sont discutées au paragraphe 14.3.

3.3.10.1 Identifiant de flux invalide (1)

Indique que le point d'extrémité a reçu un tronçon DATA envoyé à un flux non existant.

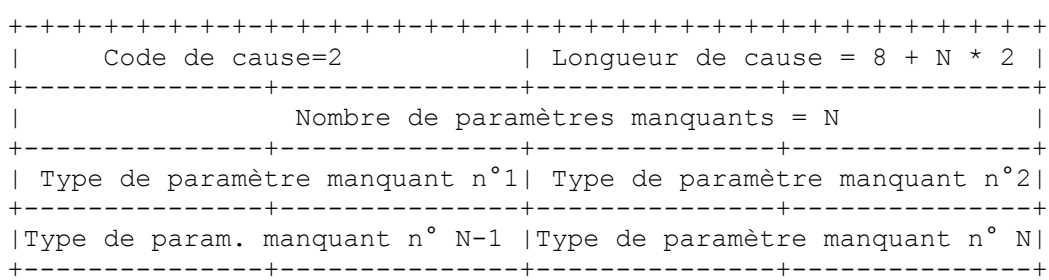


Identifiant de flux : 16 bits (entier non signé). Contient l'identifiant de flux du tronçon DATA reçu erroné.

Réservé : 16 bits. Ce champ est réservé. Il est réglé tout à 0 à l'émission et ignoré à réception.

3.3.10.2 Paramètre obligatoire manquant (2)

Indique que un ou plusieurs paramètres TLV obligatoires manquent dans un tronçon INIT ou INIT ACK reçu.

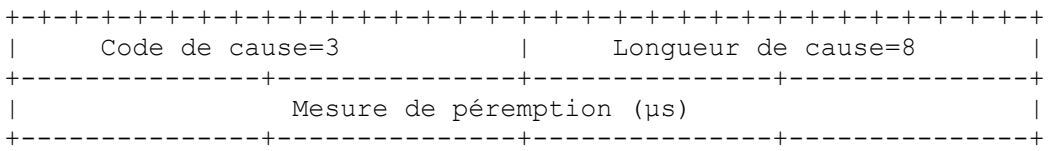


Nombre de paramètres manquants : 32 bits (entier non signé). Ce champ contient le nombre de paramètres contenus dans le champ Informations spécifiques de cause.

Type de paramètre manquant : 16 bits (entier non signé). Chaque champ contient le numéro du paramètre obligatoire manquant.

3.3.10.3 Mouchard périmé (3)

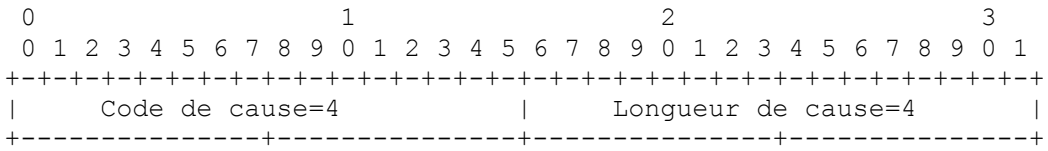
Indique la réception d'un mouchard d'état valide qui a expiré.



Mesure de péremption : 32 bits (entier non signé). Ce champ contient la différence, arrondie en microsecondes, entre l'heure actuelle et l'heure où le mouchard d'état a expiré. L'envoyeur de cette cause d'erreur PEUT choisir de rapporter depuis combien de temps le mouchard d'état est expiré en incluant une valeur non zéro dans le champ Mesure de péremption. Si l'envoyeur ne souhaite pas fournir la mesure de péremption, il DEVRAIT régler le champ Mesure de péremption à zéro.

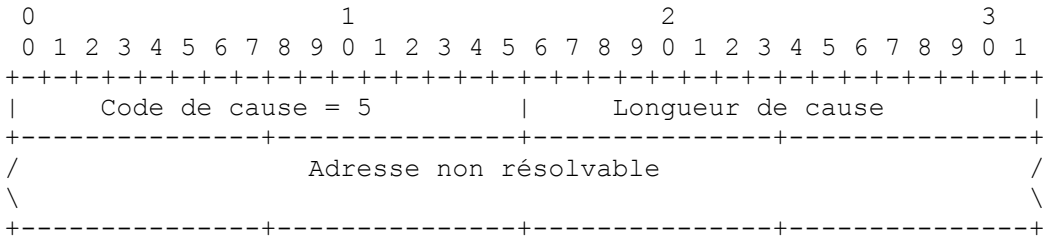
3.3.10.4 Plus de ressource (4)

Indique que l'envoyeur est à bout de ressources. Ceci est généralement envoyé combiné avec ou dans un tronçon ABORT.



3.3.10.5 Adresse non résolvable (5)

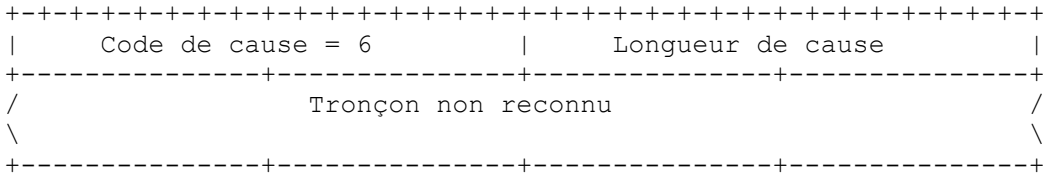
Indique que l'envoyeur n'est pas capable de résoudre le paramètre Adresse spécifié (par exemple, le type d'adresse n'est pas pris en charge par l'envoyeur). Ceci est généralement envoyé combiné avec ou dans un tronçon ABORT.



Adresse non résolvable : longueur variable. Le champ Adresse non résolvable contient le Type, Longueur, et Valeur complet du paramètre Adresse (ou paramètre Nom d'hôte) qui contient l'adresse ou nom d'hôte non résolvable.

3.3.10.6 Type de tronçon non reconnu (6)

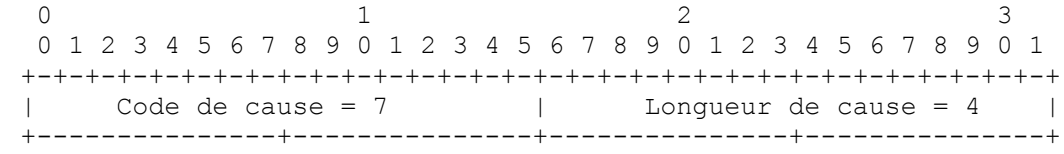
Cette cause d'erreur est retournée à l'origine du tronçon si le receveur ne comprend pas le tronçon et si les bits de poids fort du "Type de tronçon" sont réglés à 01 ou 11.



Tronçon non reconnu : longueur variable. Le champ Tronçon non reconnu contient le tronçon non reconnu du paquet SCTP complet avec Type de tronçon, Fanions de tronçon, et Longueur de tronçon.

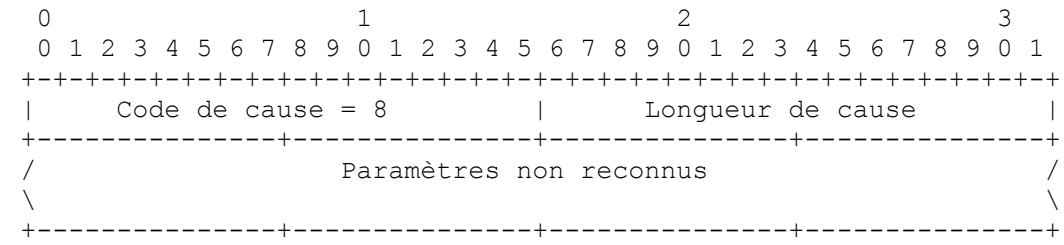
3.3.10.7 Paramètre obligatoire invalide (7)

Cette cause d'erreur est retournée à l'origine d'un tronçon INIT ou INIT ACK quand un des paramètres obligatoires est réglé à une valeur invalide.



3.3.10.8 Paramètres non reconnus (8)

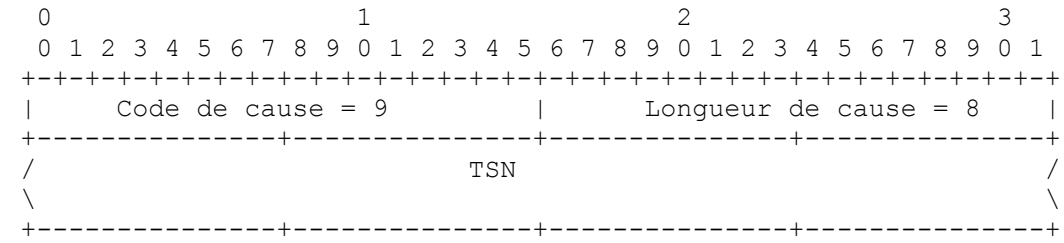
Cette cause d'erreur est retournée à l'origine du tronçon INIT ACK si le receveur ne reconnaît pas un ou plusieurs paramètres TLV facultatifs dans le tronçon INIT ACK.



Paramètres non reconnus : longueur variable. Les champs Paramètres non reconnus contiennent les paramètres non reconnus copiés du tronçon INIT ACK complet avec la TLV. Cette cause d'erreur est normalement contenue dans un tronçon ERROR regroupé avec le tronçon COOKIE ECHO quand on répond au tronçon INIT ACK, quand l'envoyeur du tronçon COOKIE ECHO souhaite faire rapport de paramètres non reconnus.

3.3.10.9 Pas de données d'utilisateur (9)

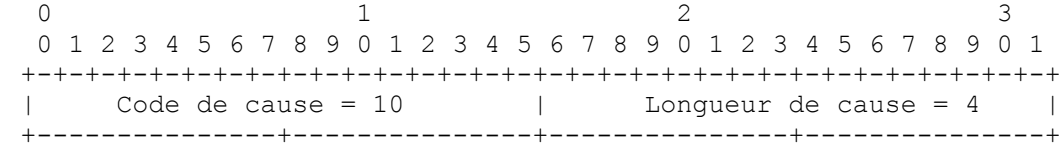
Cette cause d'erreur est retournée à l'origine d'un tronçon DATA si un tronçon DATA reçu n'a pas de données d'utilisateur.



TSN : 32 bits (entier non signé). Ce paramètre contient le TSN du tronçon DATA reçu sans champ Données d'utilisateur. Ce code de cause est normalement retourné dans un tronçon ABORT (voir au paragraphe 6.2).

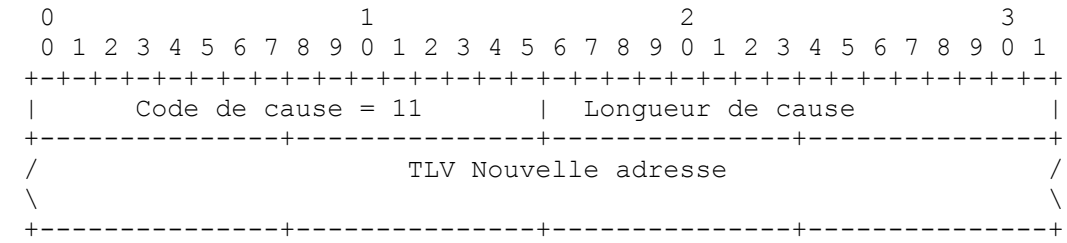
3.3.10.10 Mouchard reçu lors de la fermeture (10)

Un tronçon COOKIE ECHO a été reçu alors que le point d'extrémité était dans l'état SHUTDOWN-ACK-SENT. Cette erreur est généralement retournée dans un tronçon ERROR regroupé avec le tronçon SHUTDOWN ACK retransmis.



3.3.10.11 Redémarrage d'une association avec de nouvelles adresses (11)

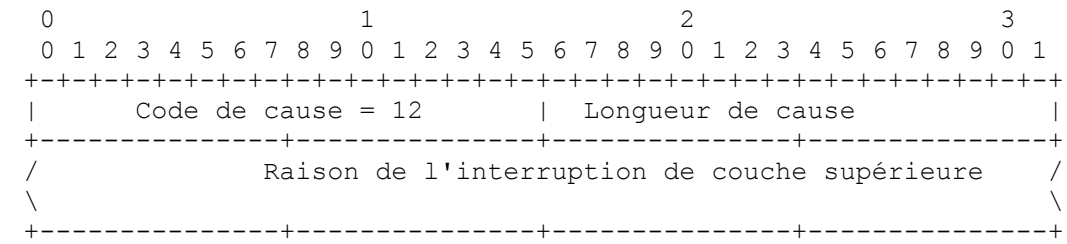
Un tronçon INIT a été reçu sur une association existante. Mais le tronçon INIT a ajouté des adresses à l'association qui ne faisaient précédemment pas partie de l'association. Les nouvelles adresses sont mentionnées dans la cause d'erreur. Cette cause d'erreur est normalement envoyée au titre d'un tronçon ABORT refusant le tronçon INIT (voir au paragraphe 5.2).



Note : chaque TLV Nouvelle adresse est une copie exacte de la TLV trouvée dans le tronçon INIT qui était nouvelle, incluant le type de paramètre et la longueur de paramètre.

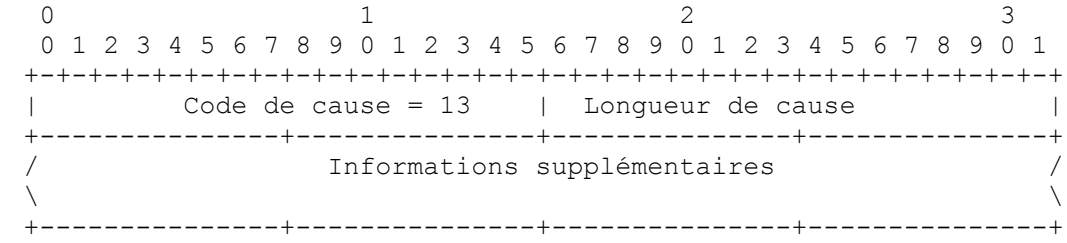
3.3.10.12 Interruption à l'initiative de l'utilisateur (12)

Cette cause d'erreur PEUT être incluse dans les tronçons ABORT qui sont envoyés à cause d'une demande de la couche supérieure. La couche supérieure peut spécifier une raison d'interruption de couche supérieure qui est transportée de façon transparente par SCTP et PEUT être livrée au protocole de couche supérieure de l'homologue.



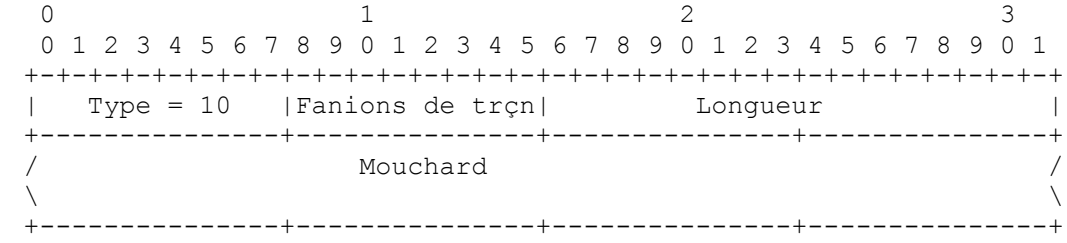
3.3.10.13 Violation du protocole (13)

Cette cause d'erreur PEUT être incluse dans les tronçons ABORT qui sont envoyés parce que un point d'extrémité SCTP détecte une violation de protocole de l'homologue qui n'est pas couverte par les causes d'erreur décrites des paragraphes 3.3.10.1 à 3.3.10.12. Une mise en œuvre PEUT fournir des informations supplémentaires spécifiant quelle sorte de violation de protocole a été détectée.



3.3.11 Écho de mouchard (COOKIE ECHO) (10)

Ce tronçon est utilisé seulement durant l'initialisation d'une association. Il est envoyé par l'initiateur d'une association à son homologue pour achever le processus d'initialisation. Ce tronçon DOIT précéder tout tronçon DATA envoyé dans l'association, mais PEUT être regroupé avec un ou plusieurs tronçons DATA dans le même paquet.



Fanions de tronçon : 8 bit. Réglé à zéro à l'émission et ignoré à réception.

Longueur : 16 bits (entier non signé). Réglé à la taille du tronçon en octets, incluant les 4 octets de l'en-tête de tronçon et la

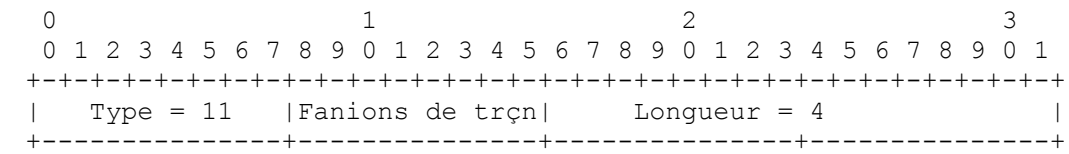
taille du mouchard.

Mouchard : taille variable. Ce champ DOIT contenir le mouchard exact reçu dans le paramètre Mouchard d'état du précédent tronçon INIT ACK. Une mise en œuvre DEVRAIT rendre le mouchard aussi petit que possible pour assurer l'interopérabilité.

Note : un Cookie Echo NE contient PAS de paramètre Mouchard d'état ; à la place, les données dans la valeur du paramètre Mouchard d'état deviennent les données dans la valeur de tronçon du Cookie Echo. Cela permet à une mise en œuvre de changer seulement les deux premiers octets du paramètre Mouchard d'état pour qu'il devienne un tronçon COOKIE ECHO.

3.3.12 Accusé de réception de mouchard (COOKIE ACK) (11)

Ce tronçon est utilisé seulement durant l'initialisation d'une association. Il est utilisé pour accuser réception d'un tronçon COOKIE ECHO. Ce tronçon DOIT précéder tout tronçon DATA ou SACK envoyé dans l'association, mais PEUT être regroupé avec un ou plusieurs tronçons DATA ou SACK dans le même paquet SCTP.

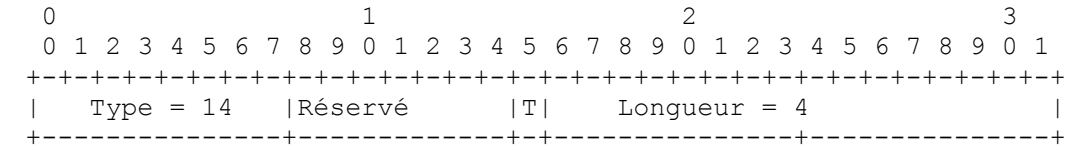


Fanions de tronçon: 8 bits. Réglé à zéro à l'émission et ignoré à réception.

3.3.13 Fermeture achevée (SHUTDOWN COMPLETE) (14)

Ce tronçon DOIT être utilisé pour accuser réception du tronçon SHUTDOWN ACK à l'achèvement du processus de fermeture ; voir les détails au paragraphe 9.2.

Le paramètre SHUTDOWN COMPLETE n'a pas de paramètre.



Fanions de tronçon : 8 bits

Réservé : 7 bits. Réglé à zéro à l'émission et ignoré à réception.

Bit T : 1 bit. Le bit T est réglé à 0 si l'envoyeur a rempli l'étiquette de vérification attendue par l'homologue. Si l'étiquette de vérification est reflétée, le bit T DOIT être réglé à 1. Refléter signifie que l'étiquette de vérification envoyée est la même que celle reçue.

Note : des règles spéciales s'appliquent à ce tronçon pour la vérification, voir les détails au paragraphe 8.5.1.

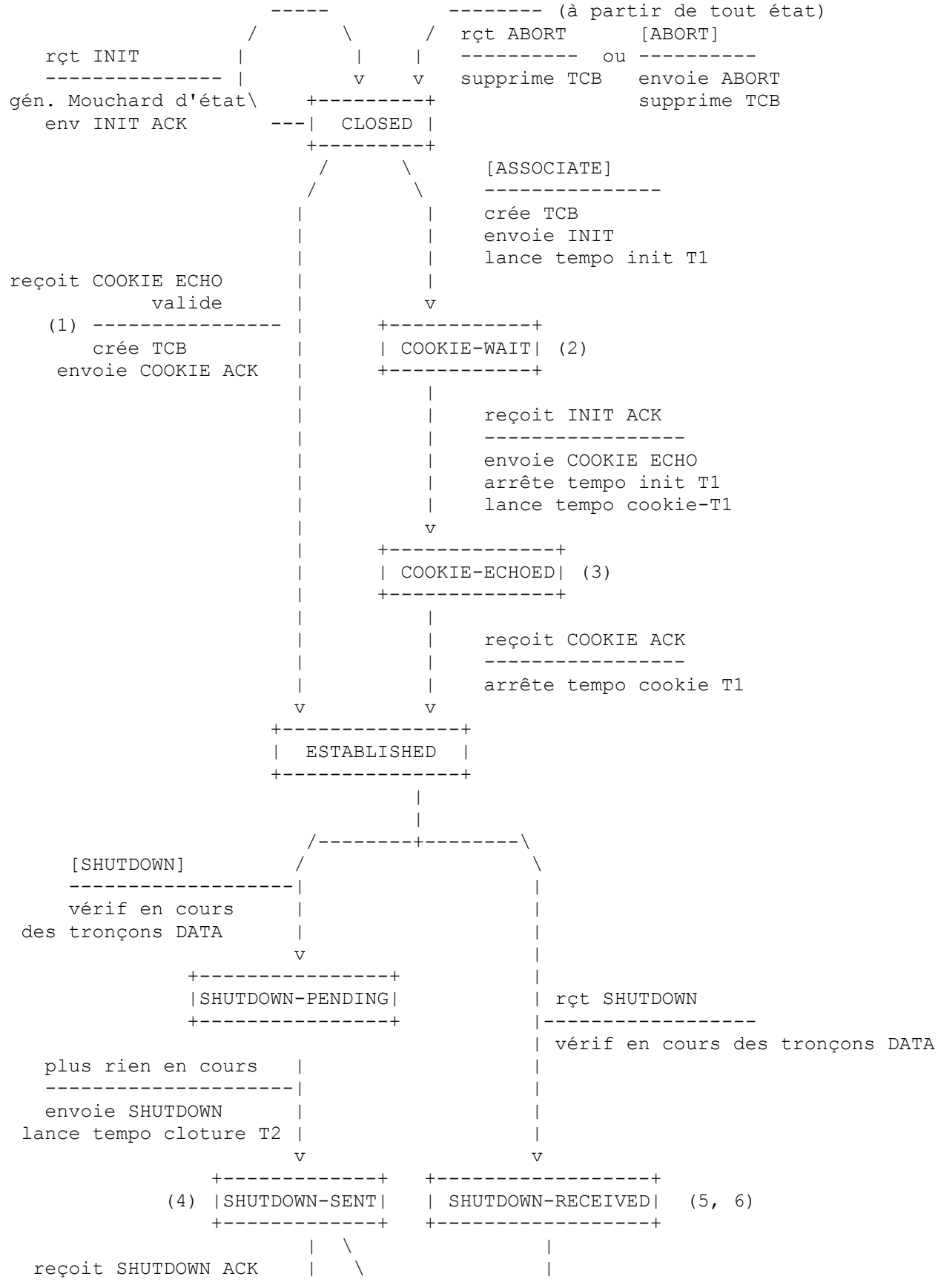
4. Diagramme d'état d'association SCTP

Durant le temps de vie d'une association SCTP, l'association de point d'extrémité SCTP progresse d'un état à un autre en réponse à divers événements. Les événements qui peuvent faire avancer l'état d'une association sont :

- l'invocation d'une primitive d'utilisateur SCTP, par exemple, [ASSOCIATE], [SHUTDOWN], [ABORT],
- la réception de INIT, COOKIE ECHO, ABORT, SHUTDOWN, etc., et de tronçons de contrôle, ou
- des événements de fin de temporisation.

Le diagramme d'états dans les figures ci-dessous illustre les changements d'état, ainsi que les événements qui les causent et les actions qui en résultent. Noter que certaines des conditions d'erreur ne sont pas montrées dans le diagramme d'états. Les descriptions complètes de tous les cas particuliers se trouvent dans le texte.

Note : les noms des tronçons sont tous donnés en lettres majuscules, tandis que les noms de paramètres ont leur première lettre en majuscule, par exemple, type de tronçon COOKIE ECHO et paramètre Mouchard d'état. Si plus d'un événement/message peuvent causer une transition d'état, ils sont marqués (A), (B), etc.



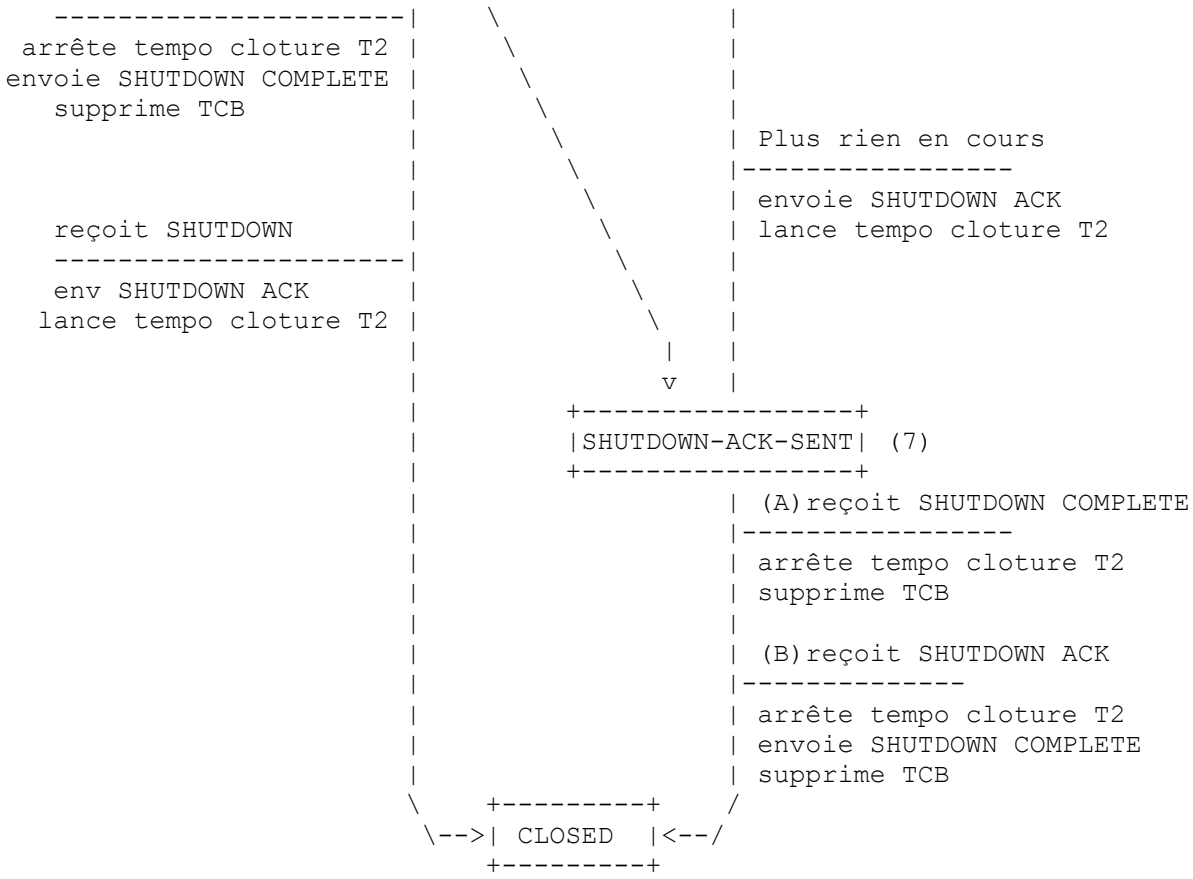


Figure 3 : Diagramme de transitions d'états de SCTP

Notes :

1) Si le Mouchard d'état dans le tronçon COOKIE ECHO reçu est invalide (c'est-à-dire, échoue à la vérification d'intégrité) le receveur DOIT éliminer en silence le paquet. Ou, si le mouchard d'état reçu est expiré (voir au paragraphe 5.1.5) le receveur DOIT renvoyer un tronçon ERROR. Dans l'un et l'autre cas, le receveur reste dans l'état CLOSED.

2) Si le temporisateur T1-init expire, le point d'extrémité DOIT retransmettre le tronçon INIT et redémarrer le temporisateur T1-init. Le point d'extrémité reste dans l'état COOKIE-WAITt. Ceci DOIT être répété jusqu'à "Max.Init.Retransmits" fois. Après cela, le point d'extrémité DOIT interrompre le processus d'initialisation et rapporter l'erreur à l'utilisateur SCTP.

3) Si le temporisateur T1-cookie expire, le point d'extrémité DOIT retransmettre le tronçon COOKIE ECHO et redémarrer le temporisateur T1-cookie. Le point d'extrémité reste dans l'état COOKIE ECHOED. Ceci DOIT être répété jusqu'à "Max.Init.Retransmits" fois. Après cela, le point d'extrémité DOIT interrompre le processus d'initialisation et rapporter l'erreur à l'utilisateur SCTP.

4) Dans l'état SHUTDOWN-SENT, le point d'extrémité DOIT accuser réception de tout tronçon DATA reçu sans délai.

5) Dans l'état SHUTDOWN-RECEIVED, le point d'extrémité NE DOIT PAS accepter de nouvelle demande d'envoi de son utilisateur SCTP.

6) Dans l'état SHUTDOWN-RECEIVED, le point d'extrémité DOIT transmettre ou retransmettre les données et quitter cet état quand toutes les données dans la file d'attente sont transmises.

7) Dans l'état SHUTDOWN-ACK-SENT, le point d'extrémité NE DOIT PAS accepter de nouvelle demande d'envoi de son utilisateur SCTP.

L'état CLOSED est utilisé pour indiquer qu'une association n'est pas créée (c'est-à-dire, n'existe pas).

5. Initialisation d'association

Avant que la première transmission de données puisse avoir lieu d'un point d'extrémité SCTP ("A") à un autre point d'extrémité SCTP ("Z") les deux points d'extrémité DOIVENT achever un processus d'initialisation afin d'établir une association SCTP entre eux.

L'utilisateur SCTP à un point d'extrémité devrait utiliser la primitive ASSOCIATE pour initier une association SCTP avec un autre point d'extrémité SCTP.

Note de mise en œuvre : du point de vue d'un utilisateur SCTP, une association peut être implicitement ouverte, sans qu'une primitive ASSOCIATE (voir au paragraphe 10.1.2) soit invoquée, par l'envoi par le point d'extrémité initiateur des premières données d'utilisateur au point d'extrémité de destination. Le SCTP initiateur va supposer les valeurs par défaut pour tous les paramètres obligatoires et facultatifs pour le tronçon INIT/INIT ACK.

Une fois que l'association est établie, des flux unidirectionnels sont ouverts pour le transfert des données aux deux extrémités (voir au paragraphe 5.1.1).

5.1 Établissement normal d'une association

Le processus d'initialisation consiste en les étapes suivantes (en supposant que le point d'extrémité SCTP "A" essaye d'établir une association avec le point d'extrémité SCTP "Z" et que "Z" accepte la nouvelle association):

A) D'abord, "A" construit un TCB et envoie un tronçon INIT à "Z". Dans le tronçon INIT, "A" DOIT fournir son étiquette de vérification (Tag_A) dans le champ Étiquette Initier. Tag_A DEVRAIT être un nombre aléatoire dans la gamme de 1 à 4 294 967 295 (voir au paragraphe 5.3.1 le choix de la valeur de l'étiquette). Après l'envoi du tronçon l'INIT, "A" lance le temporisateur T1-init et entre dans l'état COOKIE-WAIT.

B) "Z" répond immédiatement avec un tronçon INIT ACK. L'adresse de destination IP du tronçon INIT ACK DOIT être réglée à l'adresse IP de source du tronçon INIT auquel ce tronçon INIT ACK répond. Dans la réponse, en plus de remplir les autres paramètres, "Z" doit établir le champ Étiquette de vérification à Tag_A, et aussi fournir sa propre étiquette de vérification (Tag_Z) dans le champ Étiquette Initier. De plus, "Z" DOIT générer et envoyer avec le tronçon INIT ACK un mouchard d'état. Voir au paragraphe 5.1.3 la génération du mouchard d'état. Après l'envoi du tronçon INIT ACK avec le paramètre mouchard d'état, "Z" NE DOIT PAS allouer de ressources ou conserver d'état pour la nouvelle association. Autrement, "Z" serait vulnérable à des attaques sur les ressources.

C) À réception du tronçon INIT ACK de "Z", "A" arrête le temporisateur T1-init et quitte l'état COOKIE-WAIT. "A" envoie alors le mouchard d'état reçu dans le tronçon INIT ACK dans un tronçon COOKIE ECHO, démarre le temporisateur T1-cookie, et entre dans l'état COOKIE-ECHOED. Le tronçon COOKIE ECHO peut être regroupé avec tous tronçons DATA sortants en cours, mais il DOIT être le premier tronçon dans le paquet et jusqu'à ce que le tronçon COOKIE ACK soit retourné, l'envoyeur NE DOIT PAS envoyer d'autre paquet à l'homologue.

D) À réception du tronçon COOKIE ECHO, le point d'extrémité "Z" répond par un tronçon COOKIE ACK après la construction d'un TCB et passe à l'état ESTABLISHED. Un tronçon COOKIE ACK PEUT être groupé avec tout tronçon DATA en cours (et/ou tronçon SACK) mais le tronçon COOKIE ACK DOIT être le premier tronçon dans le paquet.

Note de mise en œuvre : une mise en œuvre peut choisir d'envoyer la notification Communication active à l'utilisateur SCTP à réception d'un tronçon COOKIE ECHO valide.

E) À réception du tronçon COOKIE ACK, le point d'extrémité "A" passe de l'état COOKIE-ECHOED à l'état ESTABLISHED, arrêtant le temporisateur T1-cookie. Il peut aussi notifier son ULP du succès de l'établissement de l'association avec une notification Communication active (voir la Section 11).

Un tronçon INIT ou INIT ACK NE DOIT PAS être regroupé avec un autre tronçon. Il DOIT être le seul tronçon présent dans le paquet SCTP qui le porte.

Un point d'extrémité DOIT envoyer le tronçon INIT ACK à l'adresse IP de laquelle il a reçu le tronçon INIT.

Les temporisateurs T1-init et T1-cookie DEVRAIENT suivre les mêmes règles que données au paragraphe 6.3. Si

l'application a fourni plusieurs adresses IP de l'homologue, il DEVRAIT y avoir un temporisateur T1-init et T1-cookie pour chaque adresse de l'homologue. Les retransmissions des tronçons INIT et COOKIE ECHO DEVRAIENT utiliser les adresses de l'homologue similaires aux retransmissions des tronçons DATA.

Si un point d'extrémité reçoit un tronçon INIT, INIT ACK, ou COOKIE ECHO mais décide de ne pas établir la nouvelle association à cause de paramètres obligatoires manquants dans le tronçon INIT ou INIT ACK reçu, de valeurs de paramètre invalides, ou du manque de ressources locales, il DEVRAIT répondre par un tronçon ABORT. Il DEVRAIT aussi spécifier la cause de l'interruption, comme le type des paramètres obligatoires manquants, etc., en incluant une cause d'erreur dans le tronçon ABORT. Le champ Étiquette de vérification dans l'en-tête commun du paquet SCTP sortant contenant le tronçon ABORT DOIT être réglé à la valeur de Étiquette Initier du tronçon INIT ou INIT ACK auquel ce tronçon ABORT répond.

Noter qu'un tronçon COOKIE ECHO qui ne réussit pas la vérification d'intégrité n'est pas considéré comme un "paramètre obligatoire invalide" et requiert un traitement particulier ; voir au paragraphe 5.1.5.

Après la réception du premier tronçon DATA dans une association, le point d'extrémité DOIT immédiatement répondre par un SACK pour accuser réception du tronçon DATA. Les accusés de réception suivants DEVRAIENT être faits comme décrit au paragraphe 6.2.

Quand le TCB est créé, chaque point d'extrémité DOIT régler son point interne d'accusé de réception de TSN cumulatif à la valeur de son TSN initial transmis moins un.

Note de mise en œuvre: les adresses IP et l'accès SCTP sont généralement utilisés comme clés pour trouver le TCB dans une instance SCTP.

5.1.1 Traitement des paramètres de flux

Dans les tronçons INIT et INIT ACK, l'envoyeur du tronçon DOIT indiquer le nombre de flux sortants (OS, *outbound flux*) qu'il souhaite avoir dans l'association, ainsi que le maximum de flux entrants (MIS) qu'il va accepter de l'autre point d'extrémité.

Après réception des informations de configuration de flux de l'autre côté, chaque point d'extrémité DOIT effectuer la vérification suivante : si le MIS de l'homologue est inférieur à l'OS du point d'extrémité, ce qui signifie que l'homologue est incapable de prendre en charge tous les flux sortants que le point d'extrémité veut configurer, le point d'extrémité DOIT utiliser le MIS de flux sortants et PEUT rapporter toute pénurie à la couche supérieure. La couche supérieure peut alors choisir d'interrompre l'association si la pénurie de ressources est inacceptable.

Après l'initialisation de l'association, la gamme d'identifiants de flux sortants valide pour l'un et l'autre point d'extrémité DOIT être de 0 à min(OS local, MIS distant) - 1.

5.1.2 Traitement des paramètres d'adresse

Durant l'initialisation de l'association, un point d'extrémité utilise les règles suivantes pour découvrir et collecter la ou les adresses de transport de destination de son homologue.

A) Si aucun paramètre d'adresse n'est présent dans le tronçon INIT ou INIT ACK reçu, le point d'extrémité DOIT prendre l'adresse IP de source d'où arrive le tronçon et l'enregistrer, en combinaison avec le numéro d'accès SCTP de source, comme seule adresse de transport de destination pour cet homologue.

B) Si un paramètre Nom d'hôte est présent dans le tronçon INIT ou INIT ACK reçu, le point d'extrémité DOIT immédiatement envoyer un tronçon ABORT et PEUT inclure la cause d'erreur "Adresse non résolvable" à son homologue. Le tronçon ABORT DEVRAIT être envoyé à l'adresse IP de source de laquelle le dernier paquet de l'homologue a été reçu.

C) Si seules des adresses IPv4/IPv6 sont présentes dans le tronçon INIT ou INIT ACK reçu, le receveur DOIT déduire et enregistrer toutes les adresses de transport du tronçon reçu ET l'adresse IP de source qui a envoyé le tronçon INIT ou INIT ACK. Les adresses de transport sont déduites par la combinaison du numéro d'accès de source SCTP (provenant de l'en-tête commun) et du ou des paramètres Adresse IP portées dans le tronçon INIT ou INIT ACK et de l'adresse IP de source du datagramme IP. Le receveur DEVRAIT utiliser seulement ces adresses de transport comme adresses de destination de transport quand il envoie les paquets suivants à son homologue.

D) Un tronçon INIT ou INIT ACK DOIT être traité comme appartenant à une association déjà établie (ou en cours d'établissement) si l'utilisation d'un des paramètres d'adresse valides contenus dans le tronçon identifierait un TCB existant.

Note de mise en œuvre : dans certains cas (par exemple, quand la mise en œuvre ne contrôle pas l'adresse IP de source qui est utilisée pour la transmission) un point d'extrémité pourrait avoir besoin d'inclure dans son tronçon INIT ou INIT ACK toutes les adresses IP possibles d'où les paquets de l'homologue pourraient être transmis.

Après que toutes les adresses de transport sont déduites du tronçon INIT ou INIT ACK en utilisant les règles ci-dessus, le point d'extrémité choisit une des adresses de transport comme chemin principal initial.

Le paquet contenant le tronçon INIT ACK DOIT être envoyé à l'adresse de source du paquet contenant le tronçon INIT.

L'envoyeur du tronçon INIT PEUT inclure un paramètre "Types d'adresses pris en charge" dans le tronçon INIT pour indiquer quels types d'adresses sont acceptables.

Note de mise en œuvre : dans le cas où le receveur d'un tronçon INIT ACK échoue à résoudre le paramètre Adresse à cause d'un type non pris en charge, il peut interrompre le processus d'initiation et ensuite tenter une réinitialisation en utilisant un paramètre "Types d'adresses pris en charge" dans le nouveau tronçon INIT pour indiquer quels types d'adresses il préfère.

Si un point d'extrémité SCTP qui prend seulement en charge soit IPv4 soit IPv6 reçoit des adresses IPv4 et IPv6 dans un tronçon INIT ou INIT ACK de son homologue, il DOIT utiliser toutes les adresses appartenant à la famille d'adresses prise en charge. Les autres adresses PEUVENT être ignorées. Le point d'extrémité NE DEVRAIT PAS répondre avec quelque indication d'erreur que ce soit.

Si un point d'extrémité SCTP mentionne dans le paramètre "Types d'adresses pris en charge" soit IPv4 soit IPv6, mais utilise l'autre famille pour l'envoi du paquet contenant le tronçon INIT, ou si il mentionne aussi des adresses de l'autre famille dans le tronçon INIT, alors la famille d'adresses qui n'est pas mentionnée dans le paramètre "Types d'adresses pris en charge" DEVRAIT aussi être considérée comme prise en charge par le receveur du tronçon INIT. Le receveur du tronçon INIT NE DEVRAIT PAS répondre avec quelque indication d'erreur que ce soit.

5.1.3 Génération de mouchard d'état

Quand il envoie un INIT ACK comme réponse à un tronçon INIT, l'envoyeur du tronçon INIT ACK crée un mouchard d'état et l'envoie dans le paramètre Mouchard d'état du tronçon INIT ACK. Dans ce mouchard d'état, l'envoyeur DOIT inclure un MAC (voir un exemple dans la [RFC2104]) pour assurer la protection de l'intégrité du mouchard d'état. Le mouchard d'état DEVRAIT aussi inclure un horodatage de la création du mouchard d'état, et la durée de vie du mouchard d'état, avec toutes les informations nécessaires pour qu'il établisse l'association, incluant les numéros d'accès et les étiquettes de vérification.

La méthode utilisée pour générer le MAC est strictement une affaire privée pour le receveur du tronçon INIT. L'utilisation d'un MAC est obligatoire pour prévenir les attaques de déni de service. Les algorithmes de MAC peuvent avoir des performances différentes selon la plate-forme. Choisir un algorithme de MAC à hautes performances augmente la résistance contre les attaques d'inondation de mouchards. Un MAC avec des propriétés de sécurité acceptables DEVRAIT être utilisé. La clé secrète DEVRAIT être aléatoire (La [RFC4086] fournit des informations sur les lignes directrices pour l'aléatoire). Les clés secrètes doivent avoir une taille appropriée. La clé secrète DEVRAIT être changée raisonnablement souvent (par exemple, toutes les heures) et l'horodatage dans le mouchard d'état PEUT être utilisé pour déterminer quelle clé est utilisée pour vérifier le MAC.

Si le mouchard d'état n'est pas chiffré, il NE DOIT PAS contenir d'informations dont il n'est pas envisagé qu'elles soient partagées.

Une mise en œuvre DEVRAIT rendre les mouchard aussi petits que possible pour assurer l'interopérabilité.

5.1.4 Traitement de mouchard d'état

Quand un point d'extrémité (dans l'état COOKIE-WAIT) reçoit un tronçon INIT ACK avec un paramètre Mouchard d'état, il DOIT immédiatement envoyer un tronçon COOKIE ECHO à son homologue avec le mouchard d'état reçu. L'envoyeur PEUT aussi ajouter au paquet tous les tronçons DATA en instance après le tronçon COOKIE ECHO.

Le point d'extrémité DOIT aussi lancer le temporisateur T1-cookie après l'envoi du tronçon COOKIE ECHO. Si le temporisateur arrive à expiration, le point d'extrémité DOIT retransmettre le tronçon COOKIE ECHO et relancer le temporisateur T1-cookie. Ceci est répété jusqu'à ce qu'un tronçon COOKIE ACK soit reçu ou que "Max.Init.Retransmits" (voir la Section 16) soit atteint, causant le marquage du point d'extrémité homologue comme injoignable (et donc l'association entre dans l'état CLOSED).

5.1.5 Authentification de mouchard d'état

Quand un point d'extrémité reçoit un tronçon COOKIE ECHO d'un autre point d'extrémité avec lequel il n'a pas d'association, il entreprend les actions suivantes :

1) Calculer un MAC en utilisant les informations portées dans le mouchard d'état et la clé secrète. L'horodatage du mouchard d'état PEUT être utilisé pour déterminer quelle clé secrète utiliser. Si les secrets sont conservés seulement pour une durée limitée et si la clé secrète à utiliser n'est plus disponible, le paquet contenant le tronçon COOKIE ECHO DOIT être éliminé en silence. La [RFC2104] peut être utilisée pour les lignes directrices sur la génération de MAC.

2) Authentifier le mouchard d'état comme un de ceux qu'il a précédemment généré en comparant le MAC calculé à celui porté dans le mouchard d'état. Si cette comparaison échoue, le paquet SCTP, incluant le tronçon COOKIE ECHO et tous les tronçons DATA, DOIT être éliminé en silence.

3) Comparer les numéros d'accès et l'étiquette de vérification contenus dans le tronçon COOKIE ECHO aux numéros d'accès et l'étiquette de vérification réels dans l'en-tête commun SCTP du paquet reçu. Si ces valeurs ne correspondent pas, le paquet DOIT être éliminé en silence.

4) Comparer l'horodatage de création dans le mouchard d'état à l'heure locale actuelle. Si le temps écoulé est plus long que la durée de vie portée dans le mouchard d'état, le paquet, incluant le tronçon COOKIE ECHO et tous les tronçons DATA rattachés, DEVRAIT alors être éliminé, et le point d'extrémité DOIT transmettre un tronçon ERROR avec une cause d'erreur "Mouchard périmé" au point d'extrémité homologue.

5) Si le mouchard d'état est valide, créer une association avec l'envoyeur du tronçon COOKIE ECHO avec les informations dans le mouchard d'état portées dans le tronçon COOKIE ECHO et entrer dans l'état ESTABLISHED.

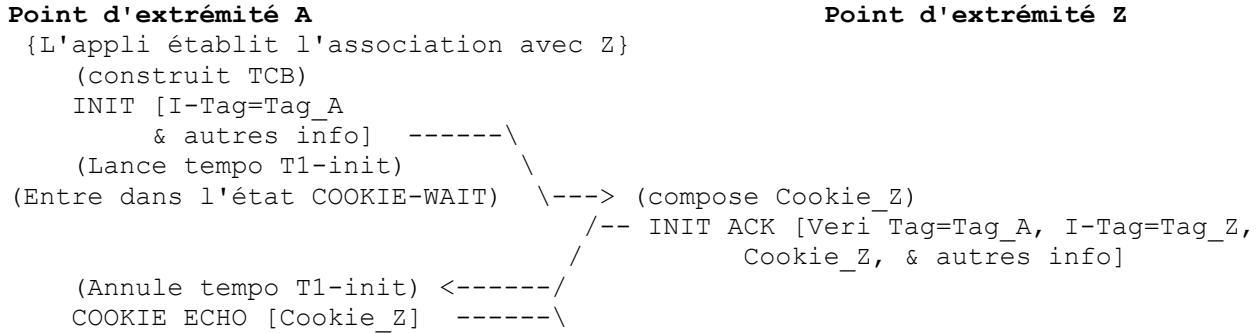
6) Envoyer un tronçon COOKIE ACK à l'homologue pour accuser réception du tronçon COOKIE ECHO. Le tronçon COOKIE ACK PEUT être groupé avec un tronçon DATA ou SACK sortant ; cependant, le tronçon COOKIE ACK DOIT être le premier tronçon dans le paquet SCTP.

7) Accuser immédiatement réception de tout tronçon DATA groupé avec le tronçon COOKIE ECHO par un tronçon SACK (les accusés de réception de tronçons DATA suivants DEVRAIENT suivre les règles définies au paragraphe 6.2). Comme mentionné à l'étape 6, si le tronçon SACK est groupé avec le tronçon COOKIE ACK, le tronçon COOKIE ACK DOIT apparaître en premier dans le paquet SCTP.

Si un tronçon COOKIE ECHO est reçu d'un point d'extrémité avec lequel le receveur du tronçon COOKIE ECHO a une association existante, les procédures du paragraphe 5.2 DEVRAIENT être suivies.

5.1.6 Exemple d'établissement normal d'association

Dans l'exemple qui suit, "A" initie l'association puis envoie un message d'utilisateur à "Z" ; ensuite, "Z" envoie deux messages d'utilisateur à "A" (en supposant qu'aucun groupement ni fragmentation ne survient) :



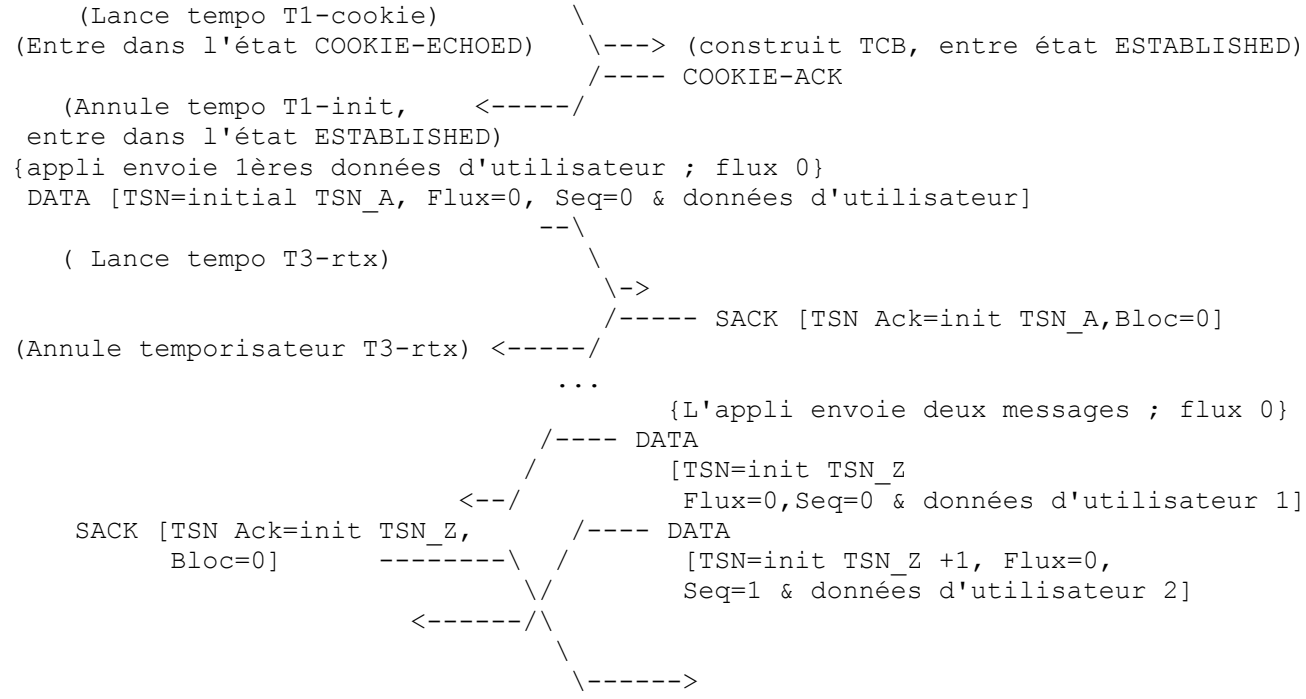


Figure 4 : Exemple d'établissement

Si le temporisateur T1-init expire à "A" après l'envoi des tronçons INIT ou COOKIE ECHO, le même tronçon INIT ou COOKIE ECHO avec la même étiquette Initier (c'est-à-dire, Tag_A) ou Mouchard d'état devra être retransmis et le temporisateur relancé. Cela est répété Max.Init.Retransmits fois avant que "A" considère que "Z" est injoignable et rapporte l'échec à sa couche supérieure (et donc l'association entre dans l'état CLOSED).

Quand il retransmet le tronçon INIT, le point d'extrémité DOIT suivre les règles définies au paragraphe 6.3 pour déterminer la valeur de temporisateur appropriée.

5.2 Traitement de tronçons INIT, INIT ACK, COOKIE ECHO, et COOKIE ACK dupliqués ou inattendus

Durant la vie d'une association (dans un des états possibles) un point d'extrémité peut recevoir de son point d'extrémité homologue un des tronçons d'établissement (INIT, INIT ACK, COOKIE ECHO, ou COOKIE ACK). Le receveur traite un tel tronçon d'établissement comme un dupliqué et comme décrit dans ce paragraphe.

Note : un point d'extrémité ne va recevoir le tronçon que si celui-ci a été envoyé à une adresse de transport SCTP et provient d'une adresse de transport SCTP associée à ce point d'extrémité. Donc, le point d'extrémité traite un tel tronçon comme faisant partie de son association en cours.

Les scénarios suivants peuvent causer des tronçons dupliqués ou inattendus :

A) L'homologue a eu une défaillance non détectée, s'est redémarré, et a envoyé un nouveau tronçon INIT pour essayer de restaurer l'association.
B) Les deux côtés essayent d'initialiser l'association en même temps.
C) Le tronçon provient d'un paquet périmé qui a été utilisé pour établir la présente association ou une association passée qui n'existe plus.
D) Le tronçon est un faux paquet généré par un attaquant, ou
E) L'homologue n'a jamais reçu le tronçon COOKIE ACK et retransmet son tronçon COOKIE ECHO.

Les règles des paragraphes suivants sont appliquées afin d'identifier et traiter correctement ces cas.

5.2.1 Tronçon INIT reçu dans l'état COOKIE-WAIT ou COOKIE-ECHOED (élément B)

Cela indique généralement une collision d'initialisation, c'est-à-dire, chaque point d'extrémité tente, à peu près en même temps d'établir une association avec l'autre point d'extrémité.

À réception d'un tronçon INIT dans l'état COOKIE-WAIT, un point d'extrémité DOIT répondre par un tronçon INIT ACK en utilisant les mêmes paramètres qu'envoyés dans son tronçon INIT original (incluant son étiquette Initier inchangée). Quand il répond, les règles suivantes DOIVENT être appliquées :

1) Le paquet contenant le tronçon INIT ACK DOIT être seulement envoyé à une adresse passée par la couche supérieure dans la demande d'initialisation de l'association.
2) Le paquet contenant le tronçon INIT ACK DOIT être seulement envoyé à une adresse rapportée dans le tronçon INIT entrant.
3) Le paquet contenant le tronçon INIT ACK DEVRAIT être envoyé à l'adresse de source du paquet reçu qui contient le tronçon INIT.

À réception d'un INIT dans l'état COOKIE-ECHOED, un point d'extrémité DOIT répondre par un tronçon INIT ACK en utilisant les mêmes paramètres qu'envoyés dans son tronçon INIT original (incluant son étiquette Initier inchangée) pourvu qu'aucune nouvelle adresse n'ait été ajoutée à l'association en formation. Si le tronçon INIT indique qu'une nouvelle adresse a été ajoutée à l'association, le tronçon INIT entier DOIT alors être éliminé, et l'état de l'association existante NE DEVRAIT PAS être changé. Un tronçon ABORT DEVRAIT être envoyé en réponse qui PEUT inclure la cause d'erreur "Redémarrage d'une association avec de nouvelles adresses". L'erreur DEVRAIT faire la liste des adresses ajoutées à l'association qui redémarre.

Quand il répond dans l'un ou l'autre état COOKIE-WAIT ou COOKIE-ECHOED avec un tronçon INIT ACK, les paramètres originaux sont combinés avec ceux du tronçon INIT nouvellement reçu. Le point d'extrémité DOIT aussi générer un mouchard d'état avec le tronçon INIT ACK. Le point d'extrémité utilise les paramètres envoyés dans son tronçon INIT pour calculer le mouchard d'état.

Après cela, le point d'extrémité NE DOIT PAS changer son état, le temporisateur T1-init DOIT continuer de courir , et le TCB correspondant NE DOIT PAS être détruit. Les procédures normale de traitement des mouchards d'état quand un TCB existe vont résoudre les tronçons INIT dupliqués à une seule association.

Pour un point d'extrémité qui est dans l'état COOKIE-ECHOED, il DOIT remplir ses étiquettes de lien dans le TCB d'association et dans le mouchard d'état (voir au paragraphe 5.2.2 la description des étiquettes de lien).

5.2.2 Tronçon INIT inattendu dans des états autres que CLOSED, COOKIE-ECHOED, COOKIE-WAIT, et SHUTDOWN-ACK-SENT

Sauf mention contraire, à réception d'un tronçon INIT inattendu pour cette association, le point d'extrémité DOIT générer un tronçon INIT ACK avec un mouchard d'état. Avant de répondre, le point d'extrémité DOIT vérifier si le tronçon INIT inattendu ajoute de nouvelles adresses à l'association. Si aucune nouvelle adresse n'est ajoutée à l'association, le point d'extrémité DOIT répondre avec un tronçon ABORT, copiant l'étiquette Initier du tronçon INIT inattendu dans l'étiquette de vérification du paquet sortant qui porte le tronçon ABORT. Dans le tronçon ABORT, la cause d'erreur PEUT être réglée à "Redémarrage d'une association avec de nouvelles adresses". L'erreur DEVRAIT faire la liste des adresses ajoutées à l'association qui redémarre. Si aucune nouvelle adresse n'est ajoutée, quand il répond au tronçon INIT dans le tronçon INIT ACK sortant, le point d'extrémité DOIT copier ses étiquettes de lien en cours à un endroit réservé dans le mouchard d'état et le TCB de l'association. On se réfère à ces localisations dans le mouchard comme étiquette de lien d'homologue et étiquette de lien locale. On se réfèrera à la copie au sein du TCB d'une association sous les noms de étiquette locale et étiquette d'homologue. Le paquet SCTP sortant qui contient ce tronçon INIT ACK DOIT porter une valeur d'étiquette de vérification égale à l'étiquette Initier trouvée dans le tronçon INIT inattendu. Et le tronçon INIT ACK DOIT contenir une nouvelle étiquette Initier (générée au hasard ; voir au paragraphe 5.3.1). Les autres paramètres pour le point d'extrémité DEVRAIENT être copiés des paramètres existants de l'association (par exemple, nombre de flux sortants) dans le tronçon INIT ACK et le mouchard.

Après l'envoi du tronçon INIT ACK ou ABORT, le point d'extrémité NE DOIT PAS prendre d'autre action ; c'est-à-dire, l'association existante, incluant son état actuel, et le TCB correspondant NE DOIT PAS être changée.

C'est seulement quand un TCB existe et que l'association n'est pas dans l'état COOKIE-WAIT ou SHUTDOWN-ACK-SENT que les étiquettes de lien sont remplies avec une valeur autre que 0. Pour un tronçon INIT d'association normale (c'est-à-dire, où le point d'extrémité est dans l'état CLOSED) les étiquettes de lien DOIVENT être réglées à 0 (indiquant qu'il n'existait pas de TCB précédent).

5.2.3 INIT ACK inattendu

Si un tronçon INIT ACK est reçu par un point d'extrémité dans tout état autre que l'état COOKIE-WAIT ou CLOSED, le

point d'extrémité DEVRAIT éliminer le tronçon INIT ACK. Un tronçon INIT ACK inattendu indique généralement le traitement d'un tronçon INIT ancien ou dupliqué.

5.2.4 Traitement de COOKIE ECHO quand un TCB existe

Quand un tronçon COOKIE ECHO est reçu par un point d'extrémité dans tout état pour un association existante (c'est-à-dire, pas dans l'état CLOSED) les règles suivantes sont appliquées :

1) Calculer un MAC comme décrit à l'étape 1 du paragraphe 5.1.5,

2) Authentifier le mouchard d'état comme décrit à l'étape 2 du paragraphe 5.1.5 (c'est le cas C ou D ci-dessus).

3) Comparer l'horodatage dans le mouchard d'état à l'heure actuelle. Si le mouchard d'état est plus vieux que la durée de vie portée dans le mouchard d'état et si les étiquettes de vérification contenues dans le mouchard d'état ne correspondent pas aux étiquettes de vérification de l'association actuelle, le paquet, incluant le tronçon COOKIE ECHO et tous tronçons DATA, devrait être éliminé. Le point d'extrémité DOIT aussi transmettre un tronçon ERROR avec une cause d'erreur "Mouchard périmé" au point d'extrémité homologue (c'est le cas C ou D au paragraphe 5.2). Si les deux étiquettes de vérification dans le mouchard d'état correspondent aux étiquettes de vérification de l'association courante, on considère que le mouchard d'état est valide (c'est le cas E du paragraphe 5.2) même si la durée de vie est excédée.

4) Si le mouchard d'état est trouvé valide, déballer le TCB dans un TCB temporaire.

5) Voir dans le Tableau 12 l'action correcte à entreprendre.

Étiquette locale	Étiquette d'homologue	Étiq. lien locale	Étiq. lien homo.	Action
X	X	M	M	(A)
M	X	A	A	(B)
M	0	A	A	(B)
X	M	0	0	(C)
M	M	A	A	(D)

Tableau 12 : Traitement d'un tronçon COOKIE ECHO quand un TCB existe

Légende :
X - L'étiquette ne correspond pas au TCB existant.
M - L'étiquette correspond au TCB existant.
0 - Étiquette inconnue (l'étiquette d'homologue n'est pas encore connue / pas d'étiquette de lien dans le mouchard).
A - Tous les cas, c'est-à-dire, M, X, ou 0.

Pour tout cas non montré dans le Tableau 2, le mouchard DEVRAIT être éliminé en silence.

Action :

A) Dans ce cas, l'homologue peut avoir redémarré. Quand le point d'extrémité reconnaît ce "redémarrage" potentiel, la session existante est traitée de la même façon que si il avait reçu un tronçon ABORT suivi par un nouveau tronçon COOKIE ECHO, sauf si :
- tous les tronçons DATA SCTP PEUVENT être conservés (c'est une option spécifique de la mise en œuvre) ;
- une notification RESTART DEVRAIT être envoyée à l'ULP au lieu d'une notification "COMMUNICATION PERDUE".

Tous les paramètres de contrôle d'encombrement (par exemple, cwnd, ssthresh) relatifs à cet homologue DOIVENT être remis à leurs valeurs initiales (voir au paragraphe 6.2.1).
Après cela, le point d'extrémité entre dans l'état ESTABLISHED.
Si le point d'extrémité est dans l'état SHUTDOWN-ACK-SENT et reconnaît que l'homologue a redémarré (Action A) il NE DOIT PAS établir une nouvelle association mais plutôt renvoyer le tronçon SHUTDOWN ACK et envoyer un tronçon ERROR avec une cause d'erreur "Mouchard reçu pendant la fermeture" à son homologue.

B) Dans ce cas, les deux côtés peuvent être en train de tenter de démarrer une association en même temps, mais le point d'extrémité homologue a commencé son tronçon INIT après avoir répondu au tronçon INIT du point d'extrémité local. Donc, il peut avoir pris une nouvelle étiquette de vérification, ne sachant pas l'étiquette précédente qu'avait envoyé ce point d'extrémité. Le point d'extrémité DEVRAIT rester ou entrer dans l'état ESTABLISHED, mais il DOIT mettre à jour l'étiquette de vérification de son homologue provenant du mouchard d'état, arrêter tout temporisateur T1-init ou T1-

cookie qui pourrait être en cours, et envoyer un tronçon COOKIE ACK.

C) Dans ce cas, le mouchard du point d'extrémité local est arrivé en retard. Avant qu'il arrive, le point d'extrémité local a envoyé un tronçon INIT et reçu un tronçon INIT ACK et a finalement envoyé un tronçon COOKIE ECHO avec la même étiquette que l'homologue mais une nouvelle étiquette à lui. Le mouchard DEVRAIT être éliminé en silence. Le point d'extrémité NE DEVRAIT PAS changer d'état et DEVRAIt laisser courir tous les temporisateurs.

D) Quand les deux étiquettes, locale et distante, correspondent, le point d'extrémité DEVRAIT entrer dans l'état ESTABLISHED si il est dans l'état COOKIE-ECHOED. Il DEVRAIT arrêter tout temporisateur T1-cookie qui pourrait être en cours et envoyer un tronçon COOKIE ACK.

Note : l'étiquette de vérification de l'homologue est celle reçue dans le champ Étiquette Initier du tronçon INIT ou INIT ACK.

5.2.4.1 Exemple de redémarrage d'une association

Dans l'exemple qui suit, "A" initie l'association après un redémarrage. Le point d'extrémité "Z" n'a pas connaissance du redémarrage jusqu'à l'échange (c'est-à-dire, des battements de cœur n'ont pas encore détecté la défaillance de "A") (en supposant qu'il n'y a pas de groupement ou de fragmentation):

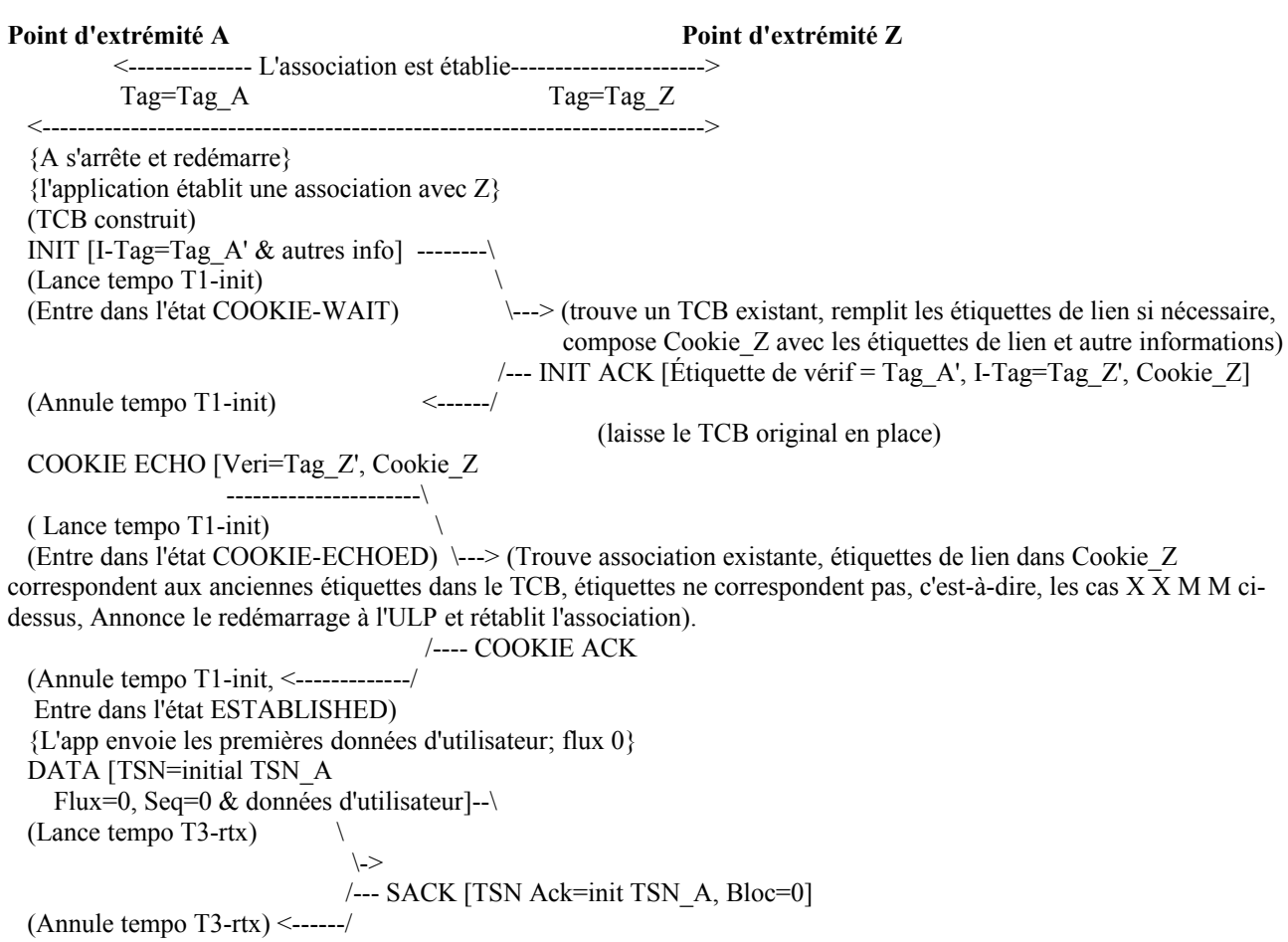


Figure 5 : Exemple de redémarrage

5.2.5 Traitement de tronçon COOKIE-ACK. dupliqué

Dans tout état autre que COOKIE-ECHOED, un point d'extrémité DEVRAIT éliminer en silence un tronçon COOKIE ACK reçu.

5.2.6 Traitement de l'erreur COOKIE périmé

La réception d'un tronçon ERROR avec une cause d'erreur "Mouchard périmé" indique un des événements possibles suivants :

A) L'association n'a pas réussi à être complètement établie avant que soit traité le mouchard d'état produit par l'envoyeur.
B) Un vieux mouchard d'état a été traité après l'achèvement de l'établissement.
C) Un vieux mouchard d'état est reçu de quelqu'un avec lequel le receveur n'est pas intéressé à avoir une association et le tronçon ABORT a été perdu.

Quand il traite un tronçon ERROR avec une cause d'erreur "Mouchard périmé", un point d'extrémité DEVRAIT d'abord examiner si une association est en cours d'établissement, c'est-à-dire, si l'association est dans l'état COOKIE-ECHOED. Dans tous les cas, si l'association n'est pas dans l'état COOKIE-ECHOED, le tronçon ERROR devrait être éliminé en silence.

Si l'association est dans l'état COOKIE-ECHOED, le point d'extrémité PEUT choisir entre les trois solutions de remplacement suivantes.

1) Envoyer un nouveau tronçon INIT au point d'extrémité pour générer un nouveau mouchard d'état et tenter à nouveau la procédure d'établissement.

2) Éliminer le TCB et rapporter à la couche supérieure l'incapacité à établir l'association.

3) Envoyer un nouveau tronçon INIT au point d'extrémité, en ajoutant un paramètre Préservation de mouchard demandant une extension de la durée de vie du mouchard d'état. Quand elle calcule l'extension de durée, une mise en œuvre DEVRAIT utiliser les informations de RTT mesurées sur la base du précédent échange de tronçon COOKIE ECHO/ERROR et DEVRAIT n'ajouter pas plus d'une seconde au delà du RTT mesuré, du fait que de longues durées de vie de mouchard d'état rendent le point d'extrémité plus susceptible d'une attaque en répétition.

5.3 Autres questions d'initialisation

5.3.1 Choix d'une valeur d'étiquette

Les valeurs d'étiquette Initier DEVRAIENT être choisies dans la gamme de 1 à 2**32 - 1. Il est très important que la valeur de l'étiquette Initier soit aléatoire pour aider à se protéger contre les attaques hors chemin. Les méthodes décrites dans la [RFC4086] peuvent être utilisées pour rendre aléatoire l'étiquette Initier. Un choix attentif des étiquettes Initier est aussi nécessaire pour empêcher que de vieux paquets dupliqués d'associations précédentes soient traités par erreur comme appartenant à l'association en cours.

De plus, la valeur de l'étiquette de vérification utilisée par l'un et l'autre point d'extrémité dans une association donnée NE DOIT PAS changer durant la vie d'une association. Une nouvelle valeur d'étiquette de vérification DOIT être utilisée chaque fois que le point d'extrémité supprime et ensuite rétablit une association avec le même homologue.

5.4 Vérification de chemin

Durant l'établissement d'association, les deux homologues échangent une liste d'adresses. Dans le cas prédominant, ces listes représentent précisément les adresses possédées par chaque homologue. Cependant, il est possible qu'un homologue au mauvais comportement fournisse des adresses qui ne lui appartiennent pas. Pour empêcher cela, les règles suivantes sont appliquées à toutes les adresses de la nouvelle association :

1) Toute adresse passée à l'envoyeur du tronçon INIT par sa couche supérieure dans la demande d'initialisation d'une association est automatiquement considérée comme CONFIRMED.

2) Pour le receveur du tronçon COOKIE ECHO, la seule adresse CONFIRMED est celle à laquelle le paquet contenant le tronçon INIT-ACK a été envoyé.

3) Toutes les autres adresses non couvertes par les règles 1 et 2 sont considérées UNCONFIRMED et sont soumises à une épreuve pour vérification.

Pour éprouver une adresse aux fins de vérification, un point d'extrémité va envoyer des tronçons HEARTBEAT incluant un nom occasionnel aléatoire de 64 bits et un indicateur de chemin (pour identifier l'adresse à laquelle le tronçon

HEARTBEAT est envoyé) au sein du paramètre Informations de battement de cœur.

À réception du tronçon HEARTBEAT ACK, une vérification est faite que le nom occasionnel inclus dans le paramètre Informations de battement de cœur est celui envoyé à l'adresse indiquée dans le paramètre Informations de battement de cœur. Quand il y a correspondance, l'adresse à laquelle le HEARTBEAT original a été envoyé est maintenant considéré CONFIRMED et disponible pour un transfert normal de données.

Ces procédures d'épreuve démarrent quand une association passe à l'état ESTABLISHED et se terminent quand tous les chemins sont confirmés.

Dans chaque RTO, une épreuve PEUT être envoyée sur un chemin actif UNCONFIRMED pour tenter de le faire passer à l'état CONFIRMED. Si durant cette épreuve le chemin devient inactif, ce taux est diminué au taux normal de HEARTBEAT. À l'expiration du temporisateur RTO, le compteur d'erreur de tout chemin qui a été éprouvé mais non CONFIRMED est incrémenté de un et soumis à la détection de défaillance de chemin, comme défini au paragraphe 8.2. Cependant, quand on éprouve des adresses UNCONFIRMED, le compte d'erreur global de l'association N'EST PAS incrémenté.

Le nombre de paquets contenant des tronçons HEARTBEAT envoyés à chaque RTO DEVRAIT être limité par le paramètre HB.Max.Burst. C'est une décision de la mise en œuvre de comment distribuer les paquets contenant des tronçons HEARTBEAT aux adresses de l'homologue pour la vérification de chemin.

Chaque fois qu'un chemin est confirmé, une indication PEUT être donnée à la couche supérieure.

Un point d'extrémité NE DOIT PAS envoyer de tronçons à une adresse UNCONFIRMED, avec les exceptions suivantes :
- Un tronçon HEARTBEAT incluant un nom occasionnel PEUT être envoyé à une adresse UNCONFIRMED.
- Un tronçon HEARTBEAT ACK PEUT être envoyé à une adresse UNCONFIRMED.
- Un tronçon COOKIE ACK PEUT être envoyé à une adresse UNCONFIRMED, mais il DOIT être groupé avec un HEARTBEAT incluant un nom occasionnel. Une mise en œuvre qui ne prend pas en charge le groupement NE DOIT PAS envoyer un tronçon COOKIE ACK à une adresse UNCONFIRMED.
- Un tronçon COOKIE ECHO PEUT être envoyé à une adresse UNCONFIRMED, mais il DOIT être groupé avec un tronçon HEARTBEAT incluant un nom occasionnel, et la taille du paquet SCTP NE DOIT PAS excéder la PMTU. Si la mise en œuvre ne prend pas en charge le groupement ou si le tronçon COOKIE ECHO groupé plus le tronçon HEARTBEAT (incluant le nom occasionnel) résulterait en un paquet SCTP excédant la PMTU, la mise en œuvre NE DOIT alors PAS envoyer de COOKIE ECHO à une adresse UNCONFIRMED.

6. Transfert des données d'utilisateur

La transmission des données DOIT seulement se produire dans les états ESTABLISHED, SHUTDOWN-PENDING, et SHUTDOWN-RECEIVED. La seule exception est qu'il est permis que les tronçons DATA soient groupés avec un tronçon COOKIE ECHO sortant quand on est dans l'état COOKIE-WAIT.

Les tronçons DATA DOIVENT seulement être reçus en accord avec les règles ci-dessous dans les états ESTABLISHED, SHUTDOWN-PENDING, et SHUTDOWN-SENT. Un tronçon DATA reçu dans l'état CLOSED sort de nulle part et DEVRAIT être traité selon le paragraphe 8.4. Un tronçon DATA reçu dans tout autre état DEVRAIT être éliminé.

Un tronçon SACK DOIT être traité dans les états ESTABLISHED, SHUTDOWN-PENDING, et SHUTDOWN-RECEIVED. Un tronçon SACK entrant PEUT être traité dans l'état COOKIE-ECHOED. Un tronçon SACK dans l'état CLOSED sort de nulle part et DEVRAIT être traité en accord avec les règles du paragraphe 8.4. Un tronçon SACK reçu dans tout autre état DEVRAIT être éliminé.

Pour l'efficacité de la transmission, SCTP définit des mécanismes pour grouper de petits messages d'utilisateur et fragmenter les grands messages d'utilisateur. Le diagramme suivant décrit le flux de messages d'utilisateur à travers SCTP.

Dans ce paragraphe, le terme "envoyeur de données" se réfère au point d'extrémité qui transmet un tronçon DATA et le terme "receveur de données" se réfère au point d'extrémité qui reçoit un tronçon DATA. Un receveur de données va transmettre des tronçons SACK.

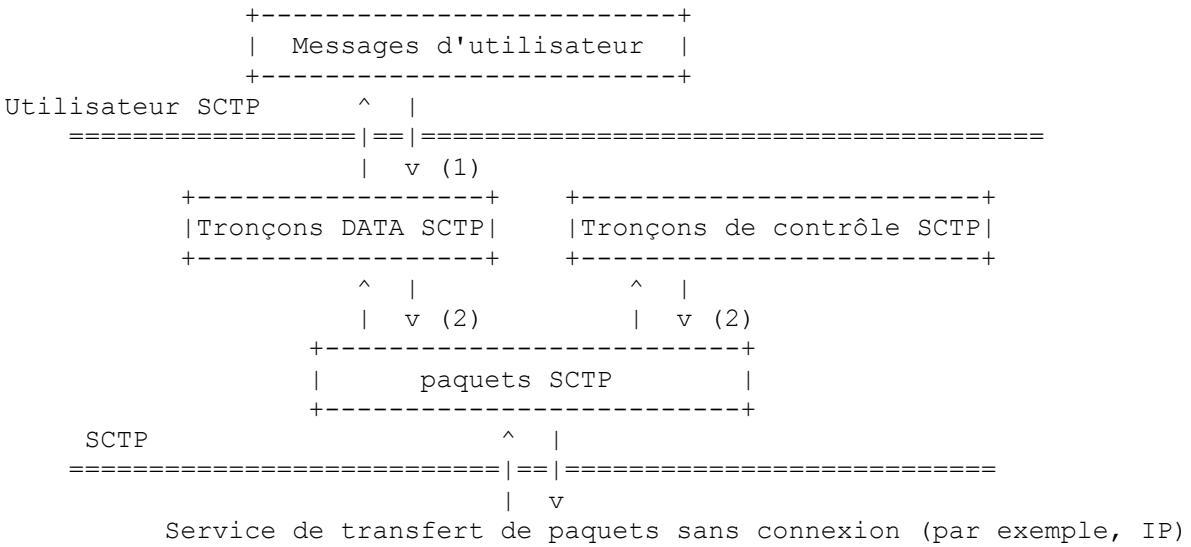


Figure 6 : Illustration du transfert des données d'utilisateur

Notes :

1) Quand il convertit des messages d'utilisateur en tronçons DATA, un point d'extrémité DOIT fragmenter les grands messages d'utilisateur en plusieurs tronçons DATA. La taille de chaque tronçon DATA DEVRAIT être inférieure ou égale à la taille maximum de tronçon DATA d'association (AMDCS, *Association Maximum DATA Chunk Size*). Le receveur des données va normalement réassembler le message fragmenté à partir des tronçons DATA avant de le livrer à l'utilisateur (voir les détails au paragraphe 6.9).
2) Plusieurs tronçons DATA et de contrôle PEUVENT être groupés par l'envoyeur dans un seul paquet SCTP pour la transmission, tant que la taille finale du paquet n'excède pas la PMTU courante. Le receveur va dégrouper le paquet en les tronçons d'origine. Les tronçons de contrôle DOIVENT venir avant les tronçons DATA dans le paquet.

Les mécanismes de fragmentation et groupage, comme précisés aux paragraphes 6.9 et 6.10, sont de mise en œuvre FACULTATIVE par l'envoyeur des données, mais ils DOIVENT être mis en œuvre par le receveur des données, c'est-à-dire, un point d'extrémité DOIT recevoir et traiter correctement les données groupées ou fragmentées.

6.1 Transmission des tronçons DATA

Ce paragraphe spécifie les règles de l'envoi des tronçons DATA. En particular, il définit la vérification de fenêtre zéro, qui est exigée pour éviter le blocage infini d'une association en cas de perte de paquets contenant des tronçons SACK effectuant la mise à jour de la fenêtre.

Le présent document est spécifié comme si il y avait un seul temporisateur de retransmission par adresse de transport de destination, mais les mises en œuvre PEUVENT avoir un temporisateur de retransmission pour chaque tronçon DATA.

Les règles générales suivantes DOIVENT être appliquées par l'envoyeur des données pour la transmission et/ou retransmission des tronçons DATA sortants :

A) À aucun moment, l'envoyeur des données NE DOIT transmettre de nouvelles données à une adresse de transport de destination si la rwnd de son homologue indique qu'il n'a pas d'espace de mémoire tampon (c'est-à-dire, si rwnd est plus petit que la taille du prochain tronçon DATA ; voir au paragraphe 6.2.1) sauf pour les épreuves de fenêtre zéro. Une épreuve de fenêtre zéro est un tronçon DATA envoyé quand le receveur n'a pas d'espace de mémoire tampon. Cette règle permet à l'envoyeur de mettre en évidence un changement de la rwnd que l'envoyeur a manqué à cause de la perte de tronçons SACK dans le transit entre receveur et envoyeur des données. Une épreuve de fenêtre zéro DOIT seulement être envoyée quand c'est permis par la cwnd (voir la règle B, ci-dessous). Une épreuve de fenêtre zéro DEVRAIT seulement être envoyée quand tous les tronçons DATA en instance ont été acquittés cumulativement et qu'aucun tronçon DATA n'est en transit. Les envoyeurs DOIVENT prendre en charge l'épreuve de fenêtre zéro. Si l'envoyeur continue de recevoir des tronçons SACK de l'homologue pendant qu'il fait l'épreuve de fenêtre zéro, les épreuves de fenêtre non acquittées NE DEVRAIENT PAS incrémenter le compteur d'erreur pour l'association ou toute adresse de transport de destination. C'est parce que le receveur pourrait garder sa fenêtre fermée pendant un temps indéfini. Voir au paragraphe 6.2 le comportement du receveur quand il annonce une fenêtre de zéro. L'envoyeur DEVRAIT envoyer la

première épreuve de fenêtre zéro après un RTO quand il détecte que le receveur a fermé sa fenêtre et DEVRAIT ensuite augmenter de façon exponentielle l'intervalle d'épreuves. On notera aussi que la cwnd DEVRAIT être ajustée selon le paragraphe 7.2.1. L'épreuve de fenêtre zéro n'affecte pas le calcul de cwnd. L'envoyeur DOIT aussi avoir un algorithme pour l'envoi de nouveaux tronçons DATA pour éviter le syndrome de la fenêtre folle (SWS, *silly window syndrome*) décrit dans la [RFC0813]. L'algorithme peut être similaire à celui décrit au paragraphe 4.2.3.4 de la [RFC1122].

B) À aucun moment, l'envoyeur NE DOIT transmettre de nouvelles données à une adresse de transport si elle a cwnd + (PMDCS - 1) ou plus d'octets de données en instance pour cette adresse de transport. Si des données sont disponibles, l'envoyeur DEVRAIT dépasser cwnd de jusqu'à (PMDCS - 1) octets sur une nouvelle transmission de données si la taille en vol n'a pas encore atteint cwnd. L'outrepassement de cwnd DOIT consister en un seul paquet.

C) Quand vient pour l'envoyeur le moment de transmettre, avant d'envoyer de nouveaux tronçons DATA, il DOIT d'abord transmettre tous les tronçons DATA qui sont marqués pour retransmission (limité par la cwnd en cours).

D) Quand vient pour l'envoyeur le moment de transmettre de nouveaux tronçons DATA, le paramètre de protocole Max.Burst *(salve maximale)* DEVRAIT être utilisé pour limiter le nombre de paquets envoyés. La limite PEUT être appliquée en ajustant temporairement cwnd comme suit :
si((flightsize + Max.Burst * PMDCS) < cwnd) cwnd = flightsize + Max.Burst * PMDCS
ou elle PEUT être appliquée en limitant strictement le nombre de paquets émis par le sous programme de sortie. Quand on calcule le nombre de paquets à transmettre, et particulièrement, quand on utilise la formule ci-dessus, la cwnd NE DEVRAIT PAS être changée de façon permanente.

E) Ensuite, l'envoyeur peut envoyer autant de nouveaux tronçons DATA que les règles A et B le permettent.

Plusieurs tronçons DATA promis à la transmission PEUVENT être groupés dans un seul paquet. De plus, les tronçons DATA à retransmettre PEUVENT être groupés avec de nouveaux tronçons DATA, pour autant que la taille du paquet SCTP résultant n'excède pas la PMTU. Un ULP peut demander qu'aucun groupage ne soit effectué, mais cela supprime seulement les délais qu'une mise en œuvre de SCTP peut utiliser pour augmenter l'efficacité du groupage. Cela n'arrête pas par lui-même tout groupage de se produire (c'est-à-dire, en cas d'encombrement ou de retransmission).

Avant qu'un point d'extrémité transmette un tronçon DATA, si un tronçon DATA reçu n'a pas encore été acquitté (par exemple, à cause d'un accusé de réception retardé) l'envoyeur devrait créer un tronçon SACK et le grouper avec le tronçon DATA sortant, pour autant que la taille du paquet SCTP final n'excède pas la PMTU actuelle (voir le paragraphe 6.2).

Quand la fenêtre est pleine (c'est-à-dire, quand la transmission est interdite par la règle A et/ou B) l'envoyeur PEUT quand même accepter les demandes envoyées de sa couche supérieure, mais il NE DOIT PAS transmettre d'autres tronçons DATA jusqu'à ce que certains ou tous les tronçons DATA en instance soient acquittés et que la transmission soit à nouveau permise par les règles A et B.

Chaque fois qu'une transmission ou retransmission est faite à une adresse, si le temporisateur T3-rtx de cette adresse n'est pas actuellement lancé, l'envoyeur DOIT le lancer. Si le temporisateur pour cette adresse est déjà lancé, l'envoyeur DOIT le redémarrer si le tronçon DATA en instance envoyé le plus tôt (c'est-à-dire, le TSN le plus bas) à cette adresse est retransmis. Autrement, l'envoyeur des données NE DOIT PAS relancer le temporisateur.

Lors du démarrage ou redémarrage du temporisateur T3-rtx, la valeur du temporisateur DEVRAIT être ajustée en accord avec les règles de temporisateur définies aux paragraphes 6.3.2 et 6.3.3.

L'envoyeur des données NE DOIT PAS utiliser un TSN de plus de 2**31 - 1 au dessus du TSN de début de la fenêtre d'envoi actuelle.

Pour chaque flux, l'envoyeur des données NE DOIT PAS avoir plus de 2^16 - 1 messages d'utilisateur ordonnés dans la fenêtre d'envoi en cours.

Chaque fois que l'envoyeur d'un tronçon DATA peut bénéficier du renvoi sans délai du tronçon SACK correspondant, l'envoyeur PEUT établir le bit I dans l'en-tête du tronçon DATA. Noter que la raison pour laquelle l'envoyeur a établi le bit I n'a pas de signification pour le receveur.

Les raisons pour établir le bit I incluent, sans s'y limiter, les suivantes (voir à la Section 4 de la [RFC7053] la discussion des avantages) :

* L'application demande que le bit I du dernier tronçon DATA d'un message d'utilisateur soit établi quand elle fournit le message d'utilisateur à la mise en œuvre SCTP (voir au paragraphe 11.1).

* L'envoyeur est dans l'état SHUTDOWN-PENDING.
* L'envoi d'un tronçon DATA remplit la fenêtre d'encombrement ou de receveur.

6.2 Accusé de réception à réception de tronçons DATA

Le point d'extrémité SCTP DOIT toujours accuser réception de chaque tronçon DATA valide quand le tronçon DATA reçu est dans sa fenêtre de réception.

Quand la fenêtre annoncée du receveur est 0, le receveur DOIT éliminer tout nouveau tronçon DATA entrant avec un TSN plus grand que le plus grand TSN reçu jusque là. Aussi, si le nouveau tronçon DATA entrant contient une valeur de TSN inférieure à la plus grande valeur de TSN reçue jusque là, le receveur DEVRAIT alors éliminer le plus grand TSN détenu pour réordonner et accepter le nouveau tronçon DATA entrant. Dans l'un et l'autre cas, un tel tronçon DATA est éliminé, le receveur DOIT immédiatement renvoyer un SACK avec la fenêtre de réception courante montrant seulement les tronçons DATA reçus et acceptés jusque là. Le ou les tronçons DATA éliminés NE DOIVENT PAS être inclus dans le tronçon SACK, car ils n'ont pas été acceptés. Le receveur DOIT aussi avoir un algorithme pour annoncer sa fenêtre de réception pour éviter le syndrome de fenêtre folle du receveur, comme décrit dans la [RFC1122]. L'algorithme peut être similaire à celui décrit au paragraphe 4.2.3.3 de la [RFC1122].

Les lignes directrices sur l'algorithme d'accusé de réception retardé spécifiées au paragraphe 4.2 de la [RFC5681] DEVRAIENT être suivies. Spécifiquement, un accusé de réception DEVRAIT être généré pour au moins chaque second paquet (pas chaque second tronçon DATA) reçu, et DEVRAIT être généré dans les 200 ms de l'arrivée de tout tronçon DATA non acquitté. Dans certaines situations, il peut être avantageux pour un émetteur SCTP d'être plus prudent que ce que permettent les algorithmes détaillés dans le présent document. Cependant, un émetteur SCTP NE DOIT PAS être plus agressif que ce que permettent les algorithmes suivants.

Un receveur SCTP NE DOIT PAS générer plus d'un tronçon SACK pour chaque paquet entrant, autrement que pour mettre à jour la fenêtre offerte lorsque l'application receveuse consomme les nouvelles données. Quand la fenêtre s'ouvre, un receveur SCTP DEVRAIT envoyer des tronçons SACK supplémentaires pour mettre à jour la fenêtre même si aucune nouvelle donnée n'est reçue. Le receveur DOIT éviter d'envoyer un grand nombre de mises à jour de fenêtre -- en particulier en grosses salves. Un façon de réaliser cela est de n'envoyer de mise à jour de fenêtre que si la fenêtre peut être augmentée d'au moins un quart de la taille de mémoire tampon de réception de l'association.

Note de mise en œuvre : le retard maximum pour générer un accusé de réception PEUT être configuré par l'administrateur SCTP, statiquement ou dynamiquement, afin de satisfaire l'exigence de temps spécifique du protocole porté.

Une mise en œuvre NE DOIT PAS permettre que le retard maximum (paramètre de protocole SACK.Delay) soit configuré comme étant plus de 500 ms. En d'autres termes, une mise en œuvre PEUT diminuer cette valeur en dessous de 500 ms mais NE DOIT PAS l'élever à plus de 500 ms.

Les accusés de réception DOIVENT être envoyés dans les tronçons SACK sauf si la fermeture a été demandée par l'ULP, et dans ce cas, un point d'extrémité PEUT envoyer un accusé de réception dans le tronçon SHUTDOWN. Un tronçon SACK peut accuser la réception de multiples tronçons DATA. Voir au paragraphe 3.3.4 le format de tronçon SACK. En particulier, le point d'extrémité SCTP DOIT remplir le champ Accusé de réception de TSN cumulatif pour indiquer le dernier TSN en séquence (d'un tronçon DATA valide) qu'il a reçu. Tout tronçon DATA reçu avec un TSN supérieur à la valeur dans le champ Accusé de réception de TSN cumulatif est rapporté dans le champ Bloc de trou d'accusé de réception. Le point d'extrémité SCTP DOIT rapporter autant de blocs de trou d'accusé de réception qu'il peut tenir dans un seul tronçon SACK de telle sorte que le paquet SCTP n'excède pas la PMTU en cours.

Le tronçon SHUTDOWN ne contient pas de champ Bloc de trou d'accusé de réception. Donc, le point d'extrémité DEVRAIT utiliser un tronçon SACK au lieu du tronçon SHUTDOWN pour accuser réception de tronçons DATA reçus déclassés.

À réception d'un paquet SCTP contenant un tronçon DATA avec le bit I établi, le receveur NE DEVRAIT PAS retarder l'envoi du tronçon SACK correspondant, c'est-à-dire, le receveur DEVRAIT répondre immédiatement avec le tronçon SACK correspondant.

Quand un paquet arrive avec un ou des tronçons DATA dupliqués et sans nouveau tronçon DATA, le point d'extrémité DOIT immédiatement et sans retard envoyer un tronçon SACK. Si un paquet arrive avec un ou des tronçons DATA dupliqués groupés avec de nouveaux tronçons DATA, le point d'extrémité PEUT immédiatement envoyer un tronçon SACK. Normalement, la réception de tronçons DATA dupliqués va survenir quand le tronçon SACK original a été perdu et

que le RTO de l'homologue a expiré. Le ou les numéros de TSN dupliqués DEVRAIENT être rapportés dans le tronçon SACK comme dupliqués.

Quand un point d'extrémité reçoit un tronçon SACK, il PEUT utiliser les informations de TSN dupliqué pour déterminer si une perte de tronçon SACK se produit. D'autres utilisations de ces données feront l'objet d'études ultérieures.

Le receveur des données est responsable du maintien de ses mémoires tampon de réception. Le receveur des données DEVRAIT notifier en temps utile à l'envoyeur des données les changements de sa capacité à recevoir des données. La façon dont une mise en œuvre gère ses mémoires tampon de réceptio dépend de nombreux facteurs (par exemple, le système d'exploitation, le système de gestion de la mémoire, la quantité de mémoire, etc.). Cependant, la stratégie de l'envoyeur des données définie au paragraphe 6.2.1 se fonde sur l'hypothèse d'un fonctionnement du receveur similaire à ce qui suit :

A) À l'initialisation de l'association, le point d'extrémité dit à l'homologue combien d'espace de mémoire tampon de réception il a alloué à l'association dans le tronçon INIT ou INIT ACK. Le point d'extrémité règle a_rwnd à cette valeur.

B) Lorsque des tronçons DATA sont reçus et mis en mémoire tampon, on décrémente a_rwnd du nombre d'octets reçus et mis en mémoire tampon. Ceci a pour effet, de clore rwnd chez l'envoyeur des données et de restreindre la quantité de données qu'il peut transmettre.

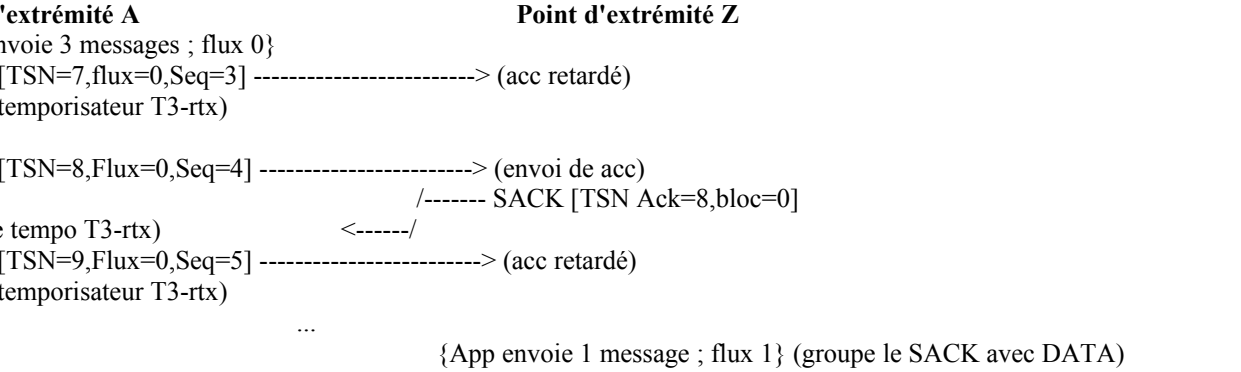
C) Lorsque des tronçons DATA sont livrés à l'ULP et libérés des mémoires tampon de réception, on incrémente a_rwnd du nombre d'octets livrés à la couche supérieure. Cela a pour effet, d'ouvrir rwnd chez l'envoyeur des données et de lui permettre d'envoyer plus de données. Le receveur des données NE DEVRAIT PAS incrémenter a_rwnd sauf si il a libéré des octets de sa mémoire tampon de réception. Par exemple, si le receveur détient des tronçons DATA fragmentés dans une file d'attente de réassemblage, il ne devrait pas incrémenter a_rwnd.

D) Quand il envoie un tronçon SACK, le receveur des données DEVRAIT placer la valeur courante de a_rwnd dans le champ a_rwnd. Le receveur des données DEVRAIT tenir compte de ce que l'envoyeur des données ne va pas retransmettre les tronçons DATA qui ont été acquittés via l'accusé de réception de TSN cumulatif (c'est-à-dire, va les éliminer de sa file d'attente de retransmission).

Dans certaines circonstances, le receveur des données peut avoir besoin d'éliminer des tronçons DATA qu'il a reçus mais n'a pas libérés de ses mémoires tampon de réception (c'est-à-dire, livrés à l'ULP). Ces tronçons DATA peuvent avoir été acquittés dans des blocs de trou d'accusé de réception. Par exemple, le receveur des données peut détenir des données dans ses mémoires tampon de réception pendant qu'il réassemble un message d'utilisateur fragmenté provenant de son homologue quand il se trouve à bout d'espace de mémoire tampon de réception. Il peut éliminer ces tronçons DATA même si il les a acquittés dans un bloc de trou d'accusé de réception. Si un receveur de données élimine des tronçons DATA, il NE DOIT PAS les inclure dans des blocs de trou d'accusé de réception dans des tronçons SACK suivants jusqu'à ce qu'ils les ait reçus de nouveau via une retransmission. De plus, le point d'extrémité DEVRAIT tenir compte des données éliminées quand il calcule son a_rwnd.

Un point d'extrémité NE DEVRAIT PAS révoquer un tronçon SACK et des données éliminées. C'est seulement dans des circonstances extrêmes qu'un point d'extrémité pourrait utiliser cette procédure (comme d'être à bout d'espace de mémoire tampon). Le receveur des données DEVRAIT tenir compte de ce que l'élimination de données qui ont été acquittées dans un bloc de trou d'accusé de réception peut résulter en des stratégies de retransmission sous optimales chez l'envoyeur des données et donc en des performances sous optimales.

L'exemple suivant illustre l'utilisation d'accusés de réception retardés :



```
                                   /----- SACK [TSN Acc=9,bloc=0] \
                                  /       DATA [TSN=6,Flux=1,Seq=2]
(Annule tempo T3-rtx)        <--------/       (lance tempo T3-rtx)
(acc retardé) (envoie acc)
SACK [TSN Acc=6,bloc=0] ----------------------> (Annule tempo T3-rtx)
```

Figure 7 : Exemple d'accusé de réception retardé

Si un point d'extrémité reçoit un tronçon DATA sans données d'utilisateur (c'est-à-dire, le champ Longueur est réglé à 16) il DEVRAIT envoyer un tronçon ABORT avec la cause d'erreur "Pas de données d'utilisateur".

Un point d'extrémité NE DEVRAIT PAS envoyer de tronçon DATA sans une partie Données d'utilisateur. Cela évite qu'il ait besoin d'être capable de retourner un message de longueur zéro dans l'API, en particulier dans l'API de prise, comme spécifié en détails dans la [RFC6458].

6.2.1 Traitement du tronçon SACK reçu

Chaque tronçon SACK que reçoit un point d'extrémité contient une valeur a_rwnd. Cette valeur représente l'espace de mémoire tampon que le receveur des données, au moment de la transmission du tronçon SACK, a laissé dans son espace total de mémoire tampon de réception (comme spécifié dans le tronçon INIT/INIT ACK). En utilisant a_rwnd, un accusé de réception de TSN cumulatif, et des blocs de trou d'accusé de réception, l'envoyeur des données peut développer une représentation de l'espace de mémoire tampon de réception de l'homologue.

Un des problèmes dont l'envoyeur des données doit tenir compte quand il traite un tronçon SACK est que celui-ci peut être reçu décalé. C'est-à-dire, un tronçon SACK envoyé par le receveur des données peut passer avant un tronçon SACK antérieur et être reçu en premier par l'envoyeur des données. Si un tronçon SACK est reçu décalé, l'envoyeur des données peut développer une vue incorrecte de l'espace de mémoire tampon de réception de l'homologue.

Comme il n'y a pas d'identifiant explicite qui puisse être utilisé pour détecter des tronçons SACK déclassés, l'envoyeur des données doit utiliser une heuristique pour déterminer si un tronçon SACK est nouveau.

Un point d'extrémité DEVRAIT utiliser les règles suivantes pour calculer le rwnd, en utilisant la valeur de a_rwnd, l'accusé de réception de TSN cumulatif, et les blocs de trou d'accusé de réception dans un tronçon SACK reçu.

A) À l'établissement de l'association, le point d'extrémité initialise la rwnd au crédit de fenêtre de receveur annoncée (a_rwnd) que l'homologue a spécifié dans le tronçon INIT ou INIT ACK.

B) Chaque fois qu'un tronçon DATA est transmis (ou retransmis) à un homologue, le point d'extrémité soustrait de la rwnd de cet homologue la taille des données du tronçon.

C) Chaque fois qu'un tronçon DATA est marqué pour retransmission, via l'expiration du temporisateur T3-rtx (paragraphe 6.3.3) ou via la retransmission rapide (paragraphe 7.2.4) on ajoute la taille des données de ces tronçons à la rwnd.

D) Chaque fois qu'un tronçon SACK arrive, le point d'extrémité effectue ce qui suit :
   i) Si l'accusé de réception de TSN cumulatif est inférieur au point d'accusé de réception de TSN cumulatif, éliminer le tronçon SACK. Comme l'accusé de réception de TSN cumulatif est à accroissement monotone, un tronçon SACK dont l'accusé de réception de TSN cumulatif est inférieur au point d'accusé de réception de TSN cumulatif indique un tronçon SACK déclassé.
   ii) Régler rwnd égal au nouveau a_rwnd reçu moins le nombre d'octets encore en instance après le traitement de l'accusé de réception de TSN cumulatif et des blocs de trou d'accusé de réception.
   iii) Si il manque au tronçon SACK un TSN qui a été précédemment acquitté via un bloc de trou d'accusé de réception (par exemple, le receveur des données a renié les données) alors on considère le tronçon DATA correspondant qui pourrait être éventuellement manquant : on compte une indication de manque à l'égard de la retransmission rapide comme décrit au paragraphe 7.2.4, et si aucun temporisateur de retransmission ne court pour l'adresse de destination à laquelle le tronçon DATA a été transmis à l'origine, alors T3-rtx est lancé pour cette adresse de destination.
   iv) Si l'accusé de réception de TSN cumulatif correspond ou excède le point de sortie de récupération rapide (paragraphe 7.2.4) on sort de la récupération rapide.

6.3 Gestion du temporisateur de retransmission

Un point d'extrémité SCTP utilise un temporisateur de retransmission T3-rtx pour s'assurer de la livraison des données en l'absence de retours de son homologue. La durée de ce temporisateur est appelée le RTO *(retransmission timeout)*.

Quand l'homologue d'un point d'extrémité est multi rattachements, le point d'extrémité va calculer un RTO séparé pour chaque adresse de transport de destination différente de son point d'extrémité homologue.

Le calcul et la gestion du RTO dans SCTP suit de près la façon dont TCP gère son temporisateur de retransmission. Pour calculer le RTO en cours, un point d'extrémité tient deux variables d'état par adresse de transport de destination : le temps d'aller retour lissé (SRTT, *smoothed round-trip time*) et la variation de temps d'aller retour (RTTVAR, *round-trip time variation*).

6.3.1 Calcul du RTO

Les règles gouvernant le calcul de SRTT, RTTVAR, et RTO sont les suivantes :

C1) Jusqu'à ce qu'une mesure de RTT ait été faite pour un paquet envoyé à l'adresse de transport de destination donnée, régler le RTO au paramètre de protocole "RTO.Initial".

C2) Quand la première mesure de RTT est faite, régler SRTT = R, RTTVAR = R/2, et RTO = SRTT + 4 * RTTVAR.

C3) Quand une nouvelle mesure de RTT R' est faite, régler RTTVAR = (1 - RTO.Beta) * RTTVAR + RTO.Beta * | SRTT - R'| et SRTT = (1 - RTO.Alpha) * SRTT + RTO.Alpha * R'.

Note : La valeur de SRTT utilisée pour la mise à jour de RTTVAR est sa valeur avant la mise à jour de SRTT lui-même en utilisant la seconde allocation.

Après le calcul, mettre à jour RTO = SRTT + 4 * RTTVAR.

C4) Quand les données sont en transit et quand c'est permis par la règle C5 ci-dessous, une nouvelle mesure de RTT DOIT être faite à chaque aller-retour. De plus, de nouvelles mesures de RTT NE DEVRAIENT PAS être faites plus d'une fois par aller-retour pour une adresse de transport de destination donnée. Il y a deux raisons à cette recommandation : d'abord, il apparaît que mesurer plus fréquemment ne donne souvent en pratique aucun avantage significatif [ALLMAN99] ; ensuite, si des mesures sont faites plus souvent, les valeurs de RTO.Alpha et RTO.Beta dans la règle C3 ci-dessus DEVRAIENT alors être ajustées afin que SRTT et RTTVAR suivent les changements à peu près au même rythme (en termes de combien d'allers-retours cela prend pour refléter les nouvelles valeurs) qu'ils feraient en effectuant seulement une mesure par aller-retour et en utilisant RTO.Alpha et RTO.Beta comme donnés dans la règle C3. Cependant, la nature exacte de ces ajustements reste un sujet de recherches.

C5) A1gorithme de Karn : les mesures de RTT NE DOIVENT PAS être faites en utilisant des paquets qui ont été retransmis (et donc pour lesquels il n'est pas clair si la réponse est pour la première instance ou une instance suivante du tronçon). Les mesures de RTT DEVRAIENT seulement être faites en utilisant un tronçon DATA avec un TSN r si aucun tronçon DATA avec un TSN inférieur ou égal à r n'est retransmis depuis le premier envoi d'un tronçon DATA avec le TSN r.

C6) Chaque fois que RTO est calculé, si il est moins de RTO.Min secondes, il est alors arrondi à RTO.Min secondes. La raison de cette règle est que les RTO qui n'ont pas une valeur minimum élevée sont susceptibles de fins de temporisation non nécessaires [ALLMAN99].

C7) Une valeur maximum PEUT être donnée au RTO pourvu qu'elle soit au moins de RTO.max secondes.

Il n'y a pas d'exigence sur la granularité d'horloge, G, utilisée pour calculer les mesures de RTT et les différentes variables d'état, autres que :

G1) Chaque fois que RTTVAR est calculé, si RTTVAR == 0, ajuster alors RTTVAR = G.

L'expérience [ALLMAN99] a montré que des granularités d'horloge plus fines (inférieures à 100 ms) fonctionnent un peu mieux que des granularités plus grossières. Voir à la Section 16 les valeurs de paramètres suggérées.

6.3.2 Règles du temporisateur de retransmission

Les règles de la gestion du temporisateur de retransmission sont les suivantes :

R1) Chaque fois qu'un tronçon DATA est envoyé à une adresse (incluant une retransmission) si le temporisateur T3-rtx de cette adresse n'est pas en cours, le lancer afin qu'il expire après le RTO de cette adresse. Le RTO utilisé ici est celui obtenu après un doublement dû aux expirations précédentes du temporisateur T3-rtx sur l'adresse de destination correspondante comme discuté à la règle E2 ci-dessous.

R2) Chaque fois que toutes les données en instance envoyées à une adresse ont été acquittées, arrêter le temporisateur T3-rtx de cette adresse.

R3) Chaque fois qu'un tronçon SACK est reçu qui accuse réception du tronçon DATA avec le plus ancien TSN en instance pour cette adresse, redémarrer le temporisateur T3-rtx pour cette adresse avec le RTO courant (si il y a encore des données en instance sur cette adresse).

R4) Chaque fois qu'un tronçon SACK est reçu où manque un TSN précédemment acquitté via un bloc de trou d'accusé de réception, démarrer le temporisateur T3-rtx pour l'adresse de destination à laquelle le tronçon DATA a été transmis à l'origine si il ne court pas déjà.

L'exemple suivant montre l'utilisation des diverses règles de temporisateur (en supposant que le receveur utilise des accusés de réception retardés).

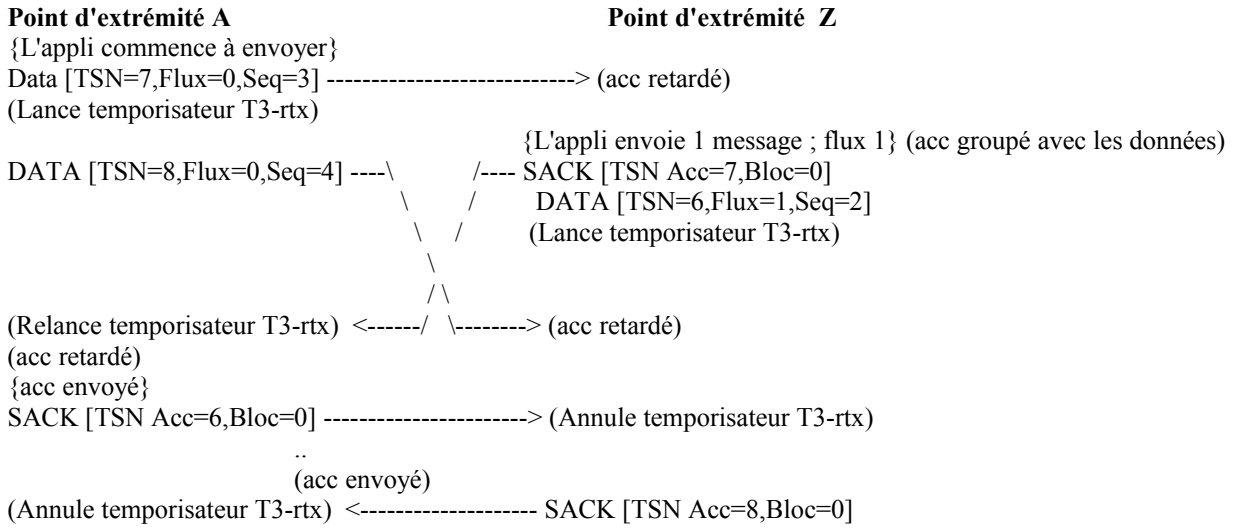


Figure 8 : Exemples de règle de temporisateur

6.3.3 Traitement de l'expiration de T3-rtx

Chaque fois que le temporisateur de retransmission T3-rtx expire pour une adresse de destination, faire ce qui suit :

E1) Pour l'adresse de destination pour laquelle le temporisateur expire, ajuster son ssthresh avec les règles définies au paragraphe 7.2.3 et régler la cwnd = PMDCS.

E2) Pour l'adresse de destination pour laquelle le temporisateur expire, régler RTO = RTO * 2 ("décote du temporisateur"). La valeur maximum discutée à la règle C7 ci-dessus (RTO.max) PEUT être utilisée pour fournir une limite supérieure à cette opération de doublement.

E3) Déterminer combien des plus anciens (c'est-à-dire, de plus faible TSN) tronçons DATA en instance pour l'adresse pour laquelle T3-rtx a expiré vont tenir dans un seul paquet SCTP, sous réserve de la PMTU correspondant à l'adresse de transport de destination à laquelle la retransmission est envoyée (qui peut être différente de l'adresse pour laquelle le temporisateur expire ; voir au paragraphe 6.4). On appelle cette valeur K. On groupe et retransmet ces K tronçons DATA dans un seul paquet au point d'extrémité de destination.

E4) Démarrer le temporisateur de retransmission T3-rtx sur l'adresse de destination à laquelle la retransmission est

envoyée, si la règle R1 ci-dessus indique de le faire. Le RTO à utiliser pour démarrer T3-rtx DEVRAIT être celui pour l'adresse de destination à laquelle la retransmission est envoyée, qui, quand le receveur est multi rattachements, pourrait être différente de l'adresse de destination pour laquelle le temporisateur a expiré (voir au paragraphe 6.4).

Après la retransmission, une fois qu'une nouvelle mesure de RTT est obtenue (qui ne peut arriver que quand de nouvelles données ont été envoyées et acquittées, selon la règle C5, ou pour une mesure faite à partir d'un tronçon HEARTBEAT ; voir au paragraphe 8.3) le calcul de la règle C3 est effectué, incluant le calcul du RTO, qui peut résulter en "l'effondrement" du RTO après qu'il ait été l'objet d'un doublement (règle E2).

Tout tronçon DATA envoyé à l'adresse pour laquelle le temporisateur T3-rtx a expiré mais qui ne tient pas dans un paquet SCTP de taille inférieure ou égale à la PMTU (règle E3) DEVRAIT être marqué pour retransmission et envoyé aussitôt que cwnd le permet (normalement, quand un tronçon SACK arrive).

La règle finale pour gérer le temporisateur de retransmission concerne la reprise sur défaillance (voir au paragraphe 6.4.1):

F1) Chaque fois qu'un point d'extrémité passe de l'adresse de transport de destination en cours à une autre, les temporisateurs de retransmission en cours sont maintenus. Aussitôt que le point d'extrémité transmet un paquet contenant un ou des tronçons DATA à la nouvelle adresse de transport, démarrer le temporisateur sur cette adresse de transport, en utilisant la valeur de RTO de l'adresse de destination à laquelle les données sont envoyées, si la règle R1 indique de le faire.

6.4 Points d'extrémité SCTP multi rattachements

Un point d'extrémité SCTP est considéré comme multi rattachements si il y a plus d'une adresse de transport qui peut être utilisée comme adresse de destination pour accéder à ce point d'extrémité.

De plus, l'ULP d'un point d'extrémité devra choisir une des multiples adresses de destination d'un point d'extrémité homologue multi rattachements comme chemin principal (voir les détails aux paragraphes 5.1.2 et 11.1).

Par défaut, un point d'extrémité DEVRAIT toujours transmettre au chemin principal, sauf si l'utilisateur SCTP spécifie explicitement l'adresse de transport de destination (et éventuellement l'adresse de transport de source) à utiliser.

Un point d'extrémité DEVRAIT transmettre les tronçons de réponse (par exemple, INIT ACK, COOKIE ACK, et HEARTBEAT ACK) aux tronçons de contrôle à la même adresse de transport de destination d'où ils ont reçu le tronçon de contrôle auquel ils répondent.

Le choix d'une adresse de transport de destination pour les paquets contenant des tronçons SACK dépend de la mise en œuvre. Cependant, un point d'extrémité NE DEVRAIT PAS changer l'adresse de transport de destination d'un tronçon SACK quand il reçoit des tronçons DATA provenant de la même adresse de source.

Cependant, quand il accuse réception de plusieurs tronçons DATA reçus dans des paquets provenant de différentes adresses de source dans un seul tronçon SACK, celui-ci PEUT être transmis à une des adresses de transport de destination d'où les tronçons DATA ou de contrôle acquittés ont été reçus.

Quand un receveur d'un tronçon DATA dupliqué envoie un tronçon SACK à un point d'extrémité multi rattachements, il PEUT être avantageux de faire varier l'adresse de destination et de ne pas utiliser l'adresse de source du tronçon DATA. La raison en est que la réception d'un dupliqué provenant d'un point d'extrémité multi rattachements pourrait indiquer que le chemin de retour (spécifé dans l'adresse de source du tronçon DATA) pour le SACK n'est plus valide.

De plus, quand son homologue est multi rattachements, un point d'extrémité DEVRAIT essayer de retransmettre un tronçon qui est arrivé en fin de temporisation à une adresse de transport de destination active différente de la dernière adresse de destination à laquelle le tronçon a été envoyé.

Quand son homologue est multi rattachements, un point d'extrémité DEVRAIT envoyer des retransmissions rapides à la même adresse de transport de destination que celle où les données originales ont été envoyées. Si le chemin principal a été changé et si les données d'origine étaient envoyées à l'ancien chemin principal avant la retransmission rapide, la mise en œuvre PEUT l'envoyer au nouveau chemin principal.

Les retransmissions n'affectent pas le total du compte de données en instance. Cependant, si le tronçon DATA est retransmis sur une adresse de destination différente, les comptes de données en instance sur la nouvelle adresse de

destination et sur l'ancienne adresse de destination à laquelle le tronçon de données a été envoyé en dernier devront être ajustés en conséquence.

6.4.1 Reprise sur défaillance d'une adresse de destination inactive

Certaines des adresses de transport d'un point d'extrémité SCTP multi rattachements peuvent devenir inactives à cause de l'occurrence de certaines conditions d'erreur (voir au paragraphe 8.2) ou d'ajustements par l'utilisateur SCTP.

Quand il y a des données sortantes à envoyer et que le chemin principal devient inactif (par exemple, à cause de défaillances) ou quand l'utilisateur SCTP demande explicitement à envoyer des données à une adresse de transport de destination inactive, avant de rapporter une erreur à son ULP, le point d'extrémité SCTP DEVRAIT essayer d'envoyer les données à une autre adresse de transport de destination active si il en existe une.

Quand il retransmet des données dont la temporisation a expiré, si le point d'extrémité est multi rattachements, il devrait considérer chaque paire d'adresses de source/destination dans sa politique de choix de retransmission. Quand il retransmet des données dont la temporisation a expiré, le point d'extrémité DEVRAIT tenter de prendre la paire d'adresses de source/destination la plus divergente de la paire d'adresses de source/destination originale à laquelle le paquet était transmis.

Note : les règles pour prendre la paire d'adresses de source/destination la plus divergente sont une décision de la mise en œuvre et ne sont pas spécifiées dans le présent document.

6.5 Identifiant de flux et numéro de séquence de flux

Chaque tronçon DATA DOIT porter un identifiant de flux valide. Si un point d'extrémité reçoit un tronçon DATA avec un identifiant de flux invalide, il DEVRAIT accuser réception du tronçon DATA suivant la procédure normale, envoyer immédiatement un tronçon ERROR avec la cause réglée à "Identifiant de flux invalide" (voir au paragraphe 3.3.10) et éliminer le tronçon DATA. Le point d'extrémité PEUT grouper le tronçon ERROR dans le même paquet que le SACK.

Le numéro de séquence de flux dans tous les flux DOIT commencer à 0 quand l'association est établie. Le numéro de séquence de flux d'un flux sortant DOIT être incrémenté de 1 pour chaque message d'utilisateur ordonné envoyé sur ce flux sortant. En particulier, quand le numéro de séquence de flux atteint la valeur 65535, le prochain numéro de séquence de flux DOIT être réglé à 0. Pour les messages d'utilisateur non ordonnés, le numéro de séquence de flux NE DOIT PAS être changé.

6.6 Livraison ordonnée et non ordonnée

Au sein d'un flux, un point d'extrémité DOIT livrer à la couche supérieure les tronçons DATA reçus avec le fanion U réglé à 0 conformément à l'ordre de leur numéro de séquence de flux. Si des tronçons DATA arrivent déclassés par rapport à leur numéro de séquence de flux, le point d'extrémité DOIT retenir les tronçons DATA reçu sans les livrer à l'ULP jusqu'à ce qu'ils soient réordonnés.

Cependant, un point d'extrémité SCTP peut indiquer qu'il n'exige pas une livraison ordonnée pour un tronçon DATA particulier transmis dans le flux en établissent le fanion U du tronçon DATA à 1.

Quand un point d'extrémité reçoit un tronçon DATA avec le fanion U établi à 1, il doit outrepasser le mécanisme d'ordre et livrer immédiatement les données à la couche supérieure (après réassemblage si les données d'utilisateur sont fragmentées par l'envoyeur des données).

Cela fournit un moyen efficace de transmettre des données "hors bande" dans un certain flux. Aussi, un flux peut être utilisé comme flux "non ordonné" en établissant simplement le fanion U à 1 dans tous les tronçons DATA envoyés dans ce flux.

Note de mise en œuvre : quand elle envoie un tronçon DATA non ordonné, une mise en œuvre PEUT choisir de placer le tronçon DATA dans un paquet sortant qui est à la tête de la file d'attente de transmission sortante si possible.

Le champ Numéro de séquence de flux dans un tronçon DATA avec un fanion U établi à 1 n'a pas de signification. L'envoyeur peut remplir le "Numéro de séquence de flux" avec une valeur arbitraire, mais le receveur DOIT ignorer le champ.

Note : quand il transmet des données ordonnées et non ordonnées, un point d'extrémité n'incrémente pas son numéro de séquence de flux quand il transmet un tronçon DATA avec le fanion U établi à 1.

6.7 Rapport de trous dans les TSN de données reçus

À réception d'un nouveau tronçon DATA, un point d'extrémité examine la continuité des TSN reçus. Si le point d'extrémité détecte un trou dans la séquence de tronçons DATA reçus, il DEVRAIT envoyer immédiatement un tronçon SACK avec des blocs de trou d'accusé de réception. Le receveur des données continue d'envoyer un tronçon SACK après la réception de chaque paquet SCTP qui ne comble pas le trou. Sur la base du bloc de trou d'accusé de réception provenant du tronçon SACK reçu, le point d'extrémité peut calculer les tronçons DATA manquants et prendre des décisions sur leur retransmission (voir les détails au paragraphe 6.2.1). Plusieurs trous peuvent être rapportés dans un seul SACK (voir au paragraphe 3.3.4).

Quand son homologue est multi rattachements, le point d'extrémité SCTP DEVRAIT toujours essayer d'envoyer le tronçon SACK à la même adresse de destination de laquelle le dernier tronçon DATA a été reçu.

À réception d'un tronçon SACK, le point d'extrémité DOIT retirer de sa file d'attente de transmission tous les tronçons DATA qui ont été acquittés par l'accusé de réception de TSN cumulatif du tronçon SACK provenant de sa file d'attente de transmission. Tous les tronçons DATA avec des TSN non inclus dans les blocs de trou d'accusé de réception qui sont inférieurs au plus au TSN acquittté rapporté dans le tronçon SACK DOIVENT être traités comme "manquants" par le point d'extrémité envoyeur. Le nombre de rapports "manquants" pour chaque tronçon DATA en instance DOIT être enregistré par l'envoyeur des données afin de prendre les décisions de retransmission. Voir les détails au paragraphe 7.2.4.

L'exemple suivant montre l'utilisation du tronçon SACK pour rapporter un trou.

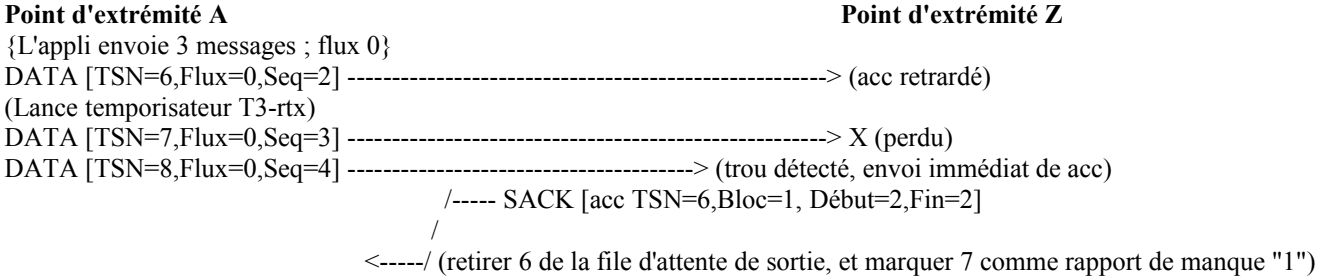


Figure 9 : Rapport d'un trou en utilisant SACK

Le nombre maximum de blocs de trou d'accusé de réception qui peuvent être rapportés dans un seul tronçon SACK est limité par la PMTU en cours. Quand un seul tronçon SACK ne peut pas couvrir le rapport de tous les blocs de trou d'accusé de réception nécessaires à cause des limites de PMTU, le point d'extrémité DOIT seulement envoyer un tronçon SACK. Ce seul tronçon SACK DOIT rapporter les blocs de trou d'accusé de réception du TSN du plus bas au plus élevé, dans la limite de taille établie par la PMTU, et laisser les plus hauts numéros de TSN restants non acquittés.

6.8 Calcul de la somme de contrôle CRC32c

Quand il envoie un paquet SCTP, le point d'extrémité DOIT renforcer l'intégrité des données de la transmission en incluant la valeur de la somme de contrôle CRC32c calculée sur le paquet, comme décrit ci-dessous.

Après la construction du paquet (contenant l'en-tête commun SCTP et un ou plusieurs tronçons de contrôle ou DATA) l'émetteur DOIT :

1) remplir l'étiquette de vérification appropriée dans l'en-tête commun SCTP et initialiser le champ Somme de contrôle à 0,
2) calculer la somme de contrôle CRC32c sur le paquet entier, incluant l'en-tête commun SCTP et tous les tronçons (se référer à l'Appendice B pour les détails de l'algorithme de CRC32c) ; et
3) mettre la valeur résultante dans le champ Somme de contrôle dans l'en-tête commun, et laisser le reste des bits inchangé.

Quand un paquet SCTP est reçu, le receveur DOIT d'abord vérifier la somme de contrôle CRC32c comme suit :

1) Mémoriser la valeur de la somme de contrôle CRC32c reçue à part.

2) Remplacer les 32 bits du champ Somme de contrôle dans le paquet SCTP reçu par 0 et calculer la valeur de la somme de contrôle CRC32c du paquet reçu entier.
3) Vérifier que la somme de contrôle CRC32c calculée est la même que la somme de contrôle CRC32c reçue. Si elle ne l'est pas, le receveur DOIT traiter le paquet comme un paquet SCTP invalide.

La procédure par défaut pour traiter les paquets SCTP invalides est de les éliminer en silence.

Toute mise en œuvre de matériel DEVRAIT permettre une autre façon de vérifier le CRC dans le logiciel.

6.9 Fragmentation et réassemblage

Un point d'extrémité PEUT prendre en charge la fragmentation quand il envoie des tronçons DATA, mais il DOIT prendre en charge le réassemblage quand il reçoit des tronçons DATA. Si un point d'extrémité prend en charge la fragmentation, il DOIT fragmenter un message d'utilisateur si la taille du message d'utilisateur à envoyer cause le dépassement de la PMTU en cours par la taille du paquet SCTP sortant. Un point d'extrémité qui ne prend pas en charge la fragmentation et à qui il est demandé d'envoyer un message d'utilisateur tel que la taille du paquet SCTP sortant va excéder la PMTU courante DOIT retourner une erreur à sa couche supérieure et NE DOIT PAS tenter d'envoyer le message d'utilisateur.

Une mise en œuvre SCTP PEUT fournir un mécanisme à la couche supérieure qui désactive la fragmentation lors de l'envoi de tronçons DATA. Quand la fragmentation des tronçons DATA est désactivée, la mise en œuvre SCTP DOIT se comporter de la même façon qu'une mise en œuvre qui ne prend pas en charge la fragmentation, c'est-à-dire, elle rejette les appels qui résulteraient en l'envoi de paquets SCTP qui excèdent la PMTU courante.

Note de mise en œuvre : dans ce cas d'erreur, la primitive SEND discutée au paragraphe 11.1 va devoir retourner une erreur à la couche supérieure.

Si son homologue est multi rattachements, le point d'extrémité DEVRAIT choisir une taille taille de tronçon DATA inférieure ou égale à l'AMDCS.

Une fois un message d'utilisateur fragmenté, il ne peut pas être re-fragmenté. À la place, si la PMTU a été réduite, la fragmentation IP DOIT alors être utilisée. Donc, une association SCTP peut échouer si la fragmentation IP ne fonctionne sur aucun chemin. Voir au paragraphe 7.3 les détails de la découverte de PMTU.

Quand elle détermine quand fragmenter, la mise en œuvre SCTP DOIT tenir compte de l'en-tête du paquet SCTP ainsi que du ou des en-têtes de tronçon DATA. La mise en œuvre DOIT aussi tenir compte de l'espace requis pour un tronçon SACK si elle groupe un tronçon SACK avec le tronçon DATA.

Les étapes de la fragmentation sont les suivantes :

1) L'envoyeur des données DOIT couper le message d'utilisateur en une série de tronçons DATA. L'envoyeur DEVRAIT choisir une taille de tronçon DATA inférieure ou égale à l'AMDCS.

2) L'émetteur DOIT ensuite allouer, en séquence, un TSN séparé à chacun des tronçons DATA de la série. L'émetteur alloue le même numéro de séquence de flux à chaque tronçon DATA. Si l'utilisateur indique que le message d'utilisateur est à livrer en utilisant une livraison non ordonnée, le fanion U de chaque tronçon DATA du message d'utilisateur DOIT alors être établi à 1.

3) L'émetteur DOIT aussi régler les bits B/E du premier tronçon DATA de la série à '10', les bits B/E du dernier tronçon DATA de la série à '01', et les bits B/E de tous les autres tronçons DATA de la série à '00'.

Un point d'extrémité DOIT reconnaître les tronçons DATA fragmentés en examinant les bits B/E dans chacun des tronçons DATA reçus, et mettre en file d'attente les tronçons DATA fragmentés pour le réassemblage. Une fois que le message d'utilisateur est réassemblé, SCTP passe le message d'utilisateur réassemblé au flux spécifique pour un réarrangement éventuel et la distribution finale.

Si le receveur des données est à bout d'espace de mémoire tampon alors qu'il attend encore d'autres fragments pour achever le réassemblage du message, il DEVRAIT distribuer une partie de son message entrant par une API de livraison partielle (voir la Section 11) libérant une partie de son espace de mémoire tampon de réception afin que le reste du message puisse être reçu.

6.10 Regroupement

Un point d'extrémité regroupe les tronçons simplement en incluant plusieurs tronçons dans un paquet SCTP sortant. La taille totale du paquet SCTP résultant DOIT être inférieure ou égale à la PMTU courante.

Si son point d'extrémité homologue est multi rattachements, le point d'extrémité envoyeur DEVRAIT choisir une taille non supérieure à la PMTU du chemin principal actuel.

Quand il regroupe des tronçons de contrôle avec des tronçons DATA, un point d'extrémité DOIT placer les tronçons de contrôle en premier dans le paquet SCTP sortant. L'émetteur DOIT transmettre les tronçons DATA au sein d'un paquet SCTP en ordre croissant de TSN.

Note : Comme les tronçons de contrôle sont placés en premier dans un paquet et que les tronçons DATA sont transmits avant les tronçons SHUTDOWN ou SHUTDOWN ACK, les tronçons DATA ne peuvent pas être regroupés avec les tronçons SHUTDOWN ou SHUTDOWN ACK.

Des tronçons partiels NE DOIVENT PAS être placés dans un paquet SCTP. Un tronçon partiel est un tronçon qui n'est pas complètement contenu dans le paquet SCTP ; c'est-à-dire que le paquet SCTP est trop court pour contenir tous les octets du tronçon comme indiqué par la longueur du tronçon.

Un point d'extrémité DOIT traiter les tronçons reçus dans leur ordre dans le paquet. Le receveur utilise le champ Longueur de tronçon pour déterminer la fin d'un tronçon et le début du tronçon suivant, en tenant compte du fait que tous les tronçons se terminent sur une frontière de quatre octets. Si le receveur détecte un tronçon partiel, il DOIT éliminer le tronçon.

Un point d'extrémité NE DOIT PAS regrouper de tronçons INIT, INIT ACK, ou SHUTDOWN COMPLETE avec d'autres tronçons.

7. Contrôle d'encombrement

Le contrôle d'encombrement est une des fonctions de base dans SCTP. Pour gérer l'encombrement, les mécanismes et algorithmes de cette section sont à employer.

Note de mise en œuvre : pour autant que les exigences spécifiques de performances sont satisfaites, une mise en œuvre est toujours libre d'adopter un algorithme de contrôle d'encombrement plus prudent que celui défini ci-dessous.

Les algorithmes de contrôle d'encombrement utilisés par SCTP se fondent sur la [RFC5681]. Cette section décricrit comment les algorithmes définis dans la [RFC5681] sont adaptés pour être utilisés dans SCTP. On fait d'abord la liste des différences dans la conception des protocoles TCP et SCTP et ensuite on décrit le schéma de contrôle d'encombrement de SCTP. La description utilise la même terminologie que dans le contrôle d'encombrement de TCP chaque fois qu'approprié.

Le contrôle d'encombrement SCTP est toujours appliqué à l'association entière et non aux flux individuels.

7.1 Différences entre le contrôle d'encombrement SCTP et TCP

Les blocs de trou d'accusé de réception dans le tronçon SACK SCTP ont la même sémantique que le SACK de TCP. TCP considère les informations portées dans le SACK seulement comme des informations. SCTP considère les informations portées dans les blocs de trou d'accusé de réception dans le tronçon SACK comme des conseils. Dans SCTP, tout tronçon DATA qui a été acquitté par un tronçon SACK, incluant les tronçons DATA qui sont arrivées déclassées à l'extrémité receveuse, n'est pas considéré comme pleinement livré jusqu'à ce que le point d'accusé de réception de TSN cumulatif passe le TSN du tronçon DATA (c'est-à-dire, le tronçon DATA a été acquitté par le champ Accusé de réception de TSN cumulatif dans le tronçon SACK). Par conséquent, la valeur de cwnd contrôle la quantité de données en instance, plutôt que (comme dans le cas de TCP non SACK) la limite supérieure entre le plus haut numéro de séquence acquitté et le dernier tronçon DATA qui peut être envoyé dans la fenêtre d'encombrement. Le SACK SCTP conduit à des mises en œuvre de la retransmission rapide et de la récupréation rapide différentes de celles de TCP non SACK. Pour un exemple, voir [FALL96].

Cependant, la plus grande différence entre SCTP et TCP est le multi rattachements. SCTP est conçu pour établir des associations de communication robustes entre deux points d'extrémité, dont chacun peut être accessible par plus d'une adresse de transport. Des adresses différentes peuvent potentiellement conduire à des chemins de données différents entre

les deux points d'extrémité ; donc, idéalement, on a besoin d'un ensemble séparé de paramètres de contrôle d'encombrement pour chacun des chemins. Le traitement du contrôle d'encombrement pour les receveurs multi rattachements est ici nouveau avec SCTP et pourrait exiger des précisions à l'avenir. Les algorithmes actuels font les hypothèses suivantes :

* L'envoyeur utilise généralement la même adresse de destination jusqu'à ce qu'il ait reçu pour instruction de la couche supérieure de faire autrement ; cependant, SCTP PEUT changer pour une autre destination dans le cas où une adresse est marquée inactive (voir le paragraphe 8.2). Aussi, SCTP PEUT retransmettre à une adresse de transport différente de celle de la transmission d'origine.

* L'envoyeur conserve un ensemble de paramètres de contrôle d'encombrement séparé pour chacune des adresses de destination à laquelle il peut envoyer (pas pour chaque paire source- destination mais pour chaque destination). Les paramètres DEVRAIENT dépérir si l'adresse n'est pas utilisée pendant une période assez longue. La [RFC5681] spécifie cette période comme une fin de temporisation de retransmission.

* Pour chaque adresse de destination, un point d'extrémité fait le démarrage lent à la première transmission à cette adresse.

Note : TCP garantit la livraison en séquence des données à son protocole de couche supérieure au sein d'une seule session TCP. Cela signifie que quand TCP remarque un trou dans les numéros de séquence reçus, il attend que le trou soit comblé avant de livrer les données reçues avec des numéros de séquence supérieurs à ceux des données manquantes. Par ailleurs, SCTP peut livrer des données à son protocole de couche supérieure, même si il y a un trou dans le TSN si les numéros de séquence de flux sont en séquence pour un flux particulier (c'est-à-dire que les tronçons DATA manquants sont pour un flux différent) ou si la livraison non ordonnée est indiquée. Bien que cela n'affecte pas la cwnd, cela pourrait affecter le calcul de rwnd.

7.2 Démarrage lent SCTP et évitement d'encombrement

Les algorithmes de démarrage lent et d'évitement d'encombrement DOIVENT être utilisés par un point d'extrémité pour contrôler la quantité de données injectées dans le réseau. Le contrôle d'encombrement dans SCTP est employé à l'égard de l'association, non d'un flux individuel. Dans certaines situations, il pourrait être avantageux pour un envoyeur SCTP d'être plus prudent que ce que permettent les algorithmes ; cependant, un envoyeur SCTP NE DOIT PAS être plus agressif que ce que les algorithmes suivants permettent.

Comme TCP, un point d'extrémité SCTP utilise les trois variables de contrôle suivantes pour réguler son taux de transmission .

* Taille de fenêtre de réception annoncée (rwnd, en octets) qui est réglée par le receveur sur la base de son espace de mémoire tampon disponible pour les paquets entrants.

  Note : Cette variable est conservée pour l'association entière.

* Fenêtre de contrôle d'encombrement (cwnd, en octets) qui est ajustée par l'envoyeur sur la base des conditions de réseau observées.

  Note : Cette variable est tenue sur la base de l'adresse de destination.

* Seuil de démarrage lent (ssthresh, *Slow-start threshold*, en octets) qui est utilisé par l'envoyeur pour distinguer les phases de démarrage lent et d'évitement d'encombrement.

  Note : Cette variable est tenue par adresse de destination.

SCTP exige aussi une variable de contrôle suppélmentaire, partial_octets_acked, qui est utilisée durant la phase d'évitement d'encombrement pour faciliter l'ajustement de la cwnd.

À la différence de TCP, un envoyeur SCTP DOIT garder un ensemble des variables de contrôle cwnd, ssthresh, et partial_octets_acked pour CHAQUE adresse de destination de son homologue (quand son homologue est multi rattachements). Quand on calcule une de ces variables, la longueur du tronçon DATA, incluant le bourrage, DEVRAIT être utilisée.

Une seule rwnd est gardée pour l'association entière (que l'homologue soit multi rattachements ou ait une seule adresse).

7.2.1 Démarrage lent

Commencer la transmission des données dans un réseau dont les conditions ne sont pas connues ou après une période de repos suffisamment longue exige que SCTP sonde le réseau pour déterminer la capacité disponible. L'algorithme de démarrage lent est utilisé à cette fin au début d'un transfert ou après la réparation de pertes détectées par le temporisateur de retransmission.

* La cwnd initiale avant la transmission des données DOIT être réglée à min(4 * PMDCS, max(2 * PMDCS, 4404)) octets si l'adresse de l'homologue est une adresse IPv4 et à min(4 * PMDCS, max(2 * PMDCS, 4344)) octets si l'adresse de l'homologue est une adresse IPv6.

* La cwnd initiale après une fin de temporisation de retransmission DOIT être inférieure ou égale à la PMDCS, et il est permis à un seul paquet d'être en vol jusqu'à son accusé de réception réussi.

* La valeur initiale de ssthresh DEVRAIT être arbitrairement élevée (par exemple, la taille de la plus grande fenêtre annonçée possible).

* Chaque fois que la cwnd est supérieure à zéro, il est permis au point d'extrémité d'avoir cwnd octets de données en instance sur cette adresse de transport. Une sur-affectation limitée, comme décrit dans la règle B du paragraphe 6.1 DEVRAIT être acceptée.

* Lorsque cwnd est inférieure ou égale à ssthresh, un point d'extrémité SCTP DOIT n'utiliser l'algorithme de démarrage lent pour augmenter cwnd que si la fenêtre d'encombrement courante est pleinement utilisée et si l'envoyeur des données n'est pas en récupération rapide. C'est seulement quand ces deux conditions sont satisfaites que la cwnd peut être augmentée ; autrement, la cwnd NE DOIT PAS être augmentée. Si ces conditions sont satisfaites, cwnd DOIT alors être augmentée de, au plus, le moindre de
  1. la taille totale du ou des tronçons DATA précédemment en instance qui ont été acquittés et
  2. L fois la PMDCS de destination.

  La première limite supérieure protège contre l'attaque d'éclatement de ACK décrite dans [SAVAGE99]. L'entier positif L DEVRAIT être 1 et PEUT être supérieur à 1. Voir dans la [RFC3465] les détails du choix de L.
  Dans les instances où le point d'extrémité homologue est multi rattachements, si un point d'extrémité reçoit un tronçon SACK qui résulte en la mise à jour de la cwnd, il DEVRAIT alors mettre à jour sa ou ses cwnd rapportées aux adresses de destination auxquelles il a transmis les données acquittées.

* Pendant que le point d'extrémité ne transmet pas de données sur une certaine adresse de transport, la cwnd de l'adresse de transport DEVRAIT être ajustée à max(cwnd/2, 4 * PMDCS) une fois par RTO. Avant le premier ajustement de cwnd, le ssthresh de l'adresse de transport DEVRAIT être réglé à la cwnd.

7.2.2. Évitement d'encombrement

Quand la cwnd est supérieure à ssthresh, cwnd DEVRAIT être incrémentée de PMDCS par RTT si l'envoyeur a cwnd ou plus d'octets de données en instance pour l'adresse de transport correspondante. Les recommandations de base pour incrémenter cwnd durant l'évitement d'encombrement sont les suivantes :

* SCTP PEUT incrémenter cwnd de PMDCS.
* SCTP DEVRAIT incrémenter cwnd de PMDCS une fois par RTT quand l'envoyeur a cwnd ou plus octets de données en instance pou l'adresse de transport correspondante.
* SCTP NE DOIT PAS incrémenter cwnd de plus de PMDCS par RTT.

En pratique, une mise en œuvre peut atteindre cet objectif de la façon suivante :

* partial_octets_acked est initialisé à 0.
* Chaque fois que cwnd est supérieur à ssthresh, à chaque arrivée de tronçon SACK, augmenter partial_octets_acked du nombre total d'octets (incluant l'en-tête de tronçon et le bourrage) de tous les nouveaux tronçons DATA acquittés dans ce tronçon SACK, incluant les tronçons acquittés par le nouvel accusé de réception de TSN cumulatif, par les blocs de trou d'accusé de réception, et par le nombre d'octets des tronçons dupliqués rapportés dans les TSN dupliqués.
* Quand (1) partial_octets_acked est supérieur à cwnd et (2) avant l'arrivée du tronçon SACK l'envoyeur a moins de cwnd octets de données en instance (c'est-à-dire, avant l'arrivée du tronçon SACK, la taille en vol *(flightsize)* était inféieure à la cwnd) remettre partial_octets_acked à cwnd.
* Quand (1) partial_octets_acked est égal ou supérieur à cwnd et (2) avant l'arrivée du tronçon SACK l'envoyeur avait cwnd ou plus octets de données en instance (c'est-à-dire, avant l'arrivée du tronçon SACK, flightsize était supérieure ou

égale à cwnd) partial_octets_acked est remis à (partial_octets_acked - cwnd). Ensuite, cwnd est augmenté de PMDCS.

* Comme dans le démarragre lent, quand l'envoyeur ne transmet pas de tronçons DATA sur une certaine adresse de transport, la cwnd de l'adresse de transport DEVRAIT être ajustée à max(cwnd/2, 4 * PMDCS) par RTO.
* Quand toutes les données transmises par l'envoyeur ont été acquittées par le receveur, partial_octets_acked est initialisé à 0.

7.2.3 Contrôle d'encombrement

À la détection de pertes de paquets à partir de tronçons SACK (voir au paragraphe 7.2.4) un point d'extrémité DEVRAIT faire ce qui suit :

ssthresh = max(cwnd/2, 4 * PMDCS)
cwnd = ssthresh
partial_octets_acked = 0

Fondamentalement, une perte de paquet cause la division par deux de la cwnd.

Quand le temporisateur T3-rtx arrive à expiration sur une adresse, SCTP DEVRAIT effectuer le démarrage lent :

ssthresh = max(cwnd/2, 4 * PMDCS)
cwnd = PMDCS
partial_octets_acked = 0

et s'assurer que pas plus d'un paquet SCTP ne va être en vol pour cette adresse jusqu'à ce que le point d'extrémité reçoive l'accusé de réception de la réussite de la livraison des données à cette adresse.

7.2.4 Retransmission rapide sur rapports de trous

En l'absence de perte de données, un point d'extrémité effectue l'accusé de réception retardé. Cependant, chaque fois qu'un point d'extrémité remarque un trou dans la séquence de TSN arrivants, il DEVRAIT commencer à renvoyer un tronçon SACK chaque fois qu'arrive un paquet portant des données jusqu'à ce que le trou soit comblé.

Chaque fois qu'un point d'extrémité reçoit un tronçon SACK qui indique que des TSN sont manquants, il DEVRAIT attendre deux autres indications de manques (via des tronçons SACK suivants pour un total de trois rapports de manque) sur les mêmes TSN avant d'entreprendre une action à l'égard de la retransmission rapide.

Les indications de manque DEVRAIENT suivre l'algorithme du plus haut TSN nouvellement acquitté (HTNA, *Highest TSN Newly Acknowledged*). Pour chaque tronçon SACK entrant, les indications de manques ne sont incrémentées que pour les TSN manquants avant le HTNA dans le tronçon SACK. Un tronçon DATA nouvellement acquitté est celui qui n'a pas été acquitté précédemment dans un tronçon SACK. Si un point d'extrémité est en récupération rapide et si des tronçons SACK arrivent qui avancent le point d'accusé de réception de TSN cumulatif, les indications de manque sont incrémentées pour tous les TSN rapportés manquants dans le tronçon SACK.

Quand la troisième indication consécutive de manque est reçue pour un ou plusieurs TSN, l'envoyeur des données fait ce qui suit :

1) Marquer pour retransmission le ou les tronçons DATA avec trois indications de manque.

2) Si on n'est pas en récupération rapide, ajuster le ssthresh et la cwnd de la ou des adresses de destination auxquelles les tronçons DATA manquants ont été envoyés en dernier, selon la formule décrite au paragraphe 7.2.3.

3) Si on n'est pas en récupération rapide, déterminer combien des plus anciens (c'est-à-dire, de plus bas TSN) tronçons DATA marqués pour retransmission vont tenir dans un seul paquet, selon les contraintes de PMTU de l'adresse de transport de destination à laquelle le paquet est envoyé. On appelle K cette valeur. On retransmet ces K tronçons DATA dans un seul paquet. Quand une retransmission rapide est effectuée, l'envoyeur DEVRAIT ignorer la valeur de cwnd et NE DEVRAIT PAS retarder la retransmission pour ce seul paquet.

4) Ne redémarrer le temporisateur T3-rtx que si le dernier tronçon SACK a acquitté le plus bas numéro de TSN en instance envoyé à cette adresse ou si le point d'extrémité retransmet le premier tronçon DATA en instance envoyé à cette adresse.

5) Marquer le ou les tronçons DATA comme étant en retransmission rapide et donc inéligibles pour une retransmission rapide suivante. Les TSN marqués pour retransmission rapide, dus à ce que l'algorithme de retransmission rapide ne les faisait pas tenir dans le datagramme envoyé portant K autres TSN, sont aussi marqués comme inéligibles pour une

retransmission rapide suivante. Cependant, comme ils sont marqués pour retransmission, ils vont être retransmis plus tard aussitôt que la cwnd le permet.

6)) Si on n'est pas en récupération rapide, entrer en récupération rapide et marquer le plus haut TSN en instance comme le point de sortie de récupération rapide. Quand un tronçon SACK accuse réception de tous lesTSN jusqu'à et y compris ce point de sortie, la récupération rapide est quittée. Pendant la récupération rapide, le ssthresh et la cwnd NE DEVRAIENT changer pour aucune destination du fait d'un événement de récupération rapide suivant (c'est-à-dire, on NE DEVRAIT PAS réduire plus la cwnd à cause d'une récupération rapide suivante).

Note : Avant les ajustements ci-dessus, si le tronçon SACK reçu accuse aussi réception de nouveaux tronçons DATA et avance le point d'accusé de réception de TSN cumulatif, les règles d'ajustement de cwnd définies aux paragraphes 7.2.1 et 7.2.2 DOIVENT être d'abord appliquées.

7.2.5 Réinitialisation

Durant la vie d'une association SCTP, des événements peuvent se produire qui résultent en l'utilisation du réseau dans de nouvelles conditions inconnues. Quand elles sont détectées par une mise en œuvre de SCTP, le contrôle d'encombrement DOIT être réinitialisé.

7.2.5.1 Changement des codets de service différencié

Les mises en œuvre de SCTP PEUVENT permettre à une application de configurer les codets de service différencié (DSCP, *Differentiated Services Code Point*) utilisés pour l'envoi de paquets. Si un changement de DSCP pourrait résulter en ce que des paquets sortants soient mis en file d'attente dans des files d'attente différentes, les paramètres de contrôle d'encombrement pour toutes les adresses de destination affectées DOIVENT être remis à leurs valeurs initiales.

7.2.5.2 Changement des chemins

Les mises en œuvre de SCTP PEUVENT être informées des changements d'acheminement qui affectent les paquets envoyés à une adresse de destination. En particulier, cela inclut le choix d'une adresse de source différente utilisée pour l'envoi des paquets à une adresse de destination. Si un tel changement d'acheminement se produit, les paramètres de contrôle d'encombrement pour les adresses de destination affectées DOIVENT être remis à leurs valeurs initiales.

7.3 Découverte de la PMTU

Les [RFC1191], [RFC8201], et [RFC8899] spécifient la "découverte de MTU de chemin de couche de mise en paquet", par laquelle un point d'extrémité conserve une estimation de la PMTU le long d'un certain chemin Internet et s'abstient d'envoyer le long de ce chemin des paquets qui excèdent la PMTU, sauf des tentatives occasionnelles de sonder si il y a eu un changement de PMTU. La [RFC8899] fournit une discussion complète du mécanisme de découverte de la PMTU et des stratégies pour déterminer le réglage de la PMTU courant de bout en bout ainsi que la détection des changements de sa valeur.

Un point d'extrémité DEVRAIT appliquer ces techniques et DEVRAIT le faire sur la base de l'adresse de destination.

Deux points importants spécifiques de SCTP concernent la découverte de la PMTU :

1) Les associations SCTP peuvent s'étendre sur plusieurs adresses. Un point d'extrémité DOIT conserver une estimation séparée de PMTU pour chaque adresse de destination de son homologue.

2) L'envoyeur DEVRAIT tracer un AMDCS qui va être la plus petite PMDCS découverte pour toutes les adresses de destination de l'homologue. Lors de la fragmentation de messages en plusieurs parties, cet AMDCS DEVRAIT être utilisé pour calculer la taille de chaque tronçon DATA. Cela va permettre que les retransmissions soient envoyées sans coupure à une autre adresse quand on rencontre une fragmentation IP.

8. Gestion des fautes

8.1 Détection de la défaillance d'un point d'extrémité

Un point d'extrémité DEVRAIT tenir un compteur du nombre total de retransmissions consécutives à son homologue (cela

inclut les retransmissions de données à toutes les adresses de transport de destination de l'homologue si il est multi rattachements) incluant le nombre de tronçons HEARTBEAT non acquittés observés sur le chemin actuellement utilisé pour le transfert des données. Les tronçons HEARTBEAT non acquittés observés sur des chemins différents du chemin actuellement utilisé pour le transfert des données NE DEVRAIENT PAS incrémenter le compteur d'erreurs de l'association, car cela pourrait conduite à la cloture de l'association même si le chemin qui est actuellement utilisé pour le transfert de données est disponible (mais au repos). Si la valeur de ce compteur excède la limite indiquée dans le paramètre de protocole "Association.Max.Retrans", le point d'extrémité DEVRAIT considérer le point d'extrémité homologue comme inaccessible et DEVRA arrêter de lui transmettre des données (et donc l'association entre dans l'état CLOSED). De plus, le point d'extrémité DEVRAIT rapporter la défaillance à la couche supérieure et facultativement rapporter toutes les données d'utilisateur en instance restant dans sa file d'attente de sortie. L'association est automatiquement close quand le point d'extrémité homologue devient inaccessible.

Le compteur utilisé pour la détection de la défaillance du point d'extrémité DOIT être réinitialisé chaque fois qu'un tronçon DATA envoyé à ce point d'extrémité homologue est acquitté (par la réception d'un tronçon SACK). Quand un tronçon HEARTBEAT ACK est reçu du point d'extrémité homologue, le compteur DEVRAIT aussi être réinitialisé. Le receveur du tronçon HEARTBEAT ACK PEUT choisir de ne pas remettre le compteur à zéro si il y a des données en instance sur l'association. Cela permet le traitement de possibles différences d'accessibilité sur la base des tronçons DATA et HEARTBEAT.

8.2 Détection de la défaillance d'un chemin

Quand son point d'extrémité homologue est multi rattachements, un point d'extrémité DEVRAIT tenir un compteur d'erreurs pour chacune des adresses de transport de destination du point d'extrémité homologue.

Chaque fois que le temporisateur T3-rtx arrive à expiration sur une adresse, ou quand un tronçon HEARTBEAT envoyé à une adresse inactive n'est pas acquitté dans un RTO, le compteur d'erreurs de cette adresse de destination va être incrémenté. Quand la valeur du compteur d'erreur excède le paramètre de protocole "Path.Max.Retrans" de cette adresse de destination, le point d'extrémité DEVRAIT marquer l'adresse de transport de destination comme inactive, et une notification DEVRAIT être envoyée à la couche supérieure.

Quand un TSN en instance est acquitté ou quand un tronçon HEARTBEAT envoyé à cette adresse est acquitté avec un tronçon HEARTBEAT ACK, le point d'extrémité DEVRAIT mettre à zéro le compteur d'erreurs de l'adresse de transport de destination à laquelle le tronçon DATA a été envoyé en dernier (ou à laquelle le tronçon HEARTBEAT a été envoyé) et DEVRAIT aussi rapporter à la couche supérieure quand une adresse de destination inactive est marquée comme active. Quand le point d'extrémité homologue est multi rattachements et quand le dernier tronçon qui lui a été envoyé était une retransmission à une adresse de remplacement, il existe une ambiguïté quant au point de savoir si l'accusé de réception a pu ou non être crédité à l'adresse du dernier tronçon envoyé. Cependant, cette ambiguïté ne semble pas avoir de conséquences significatives pour le comportement de SCTP. Si cette ambiguïté est indésirable, l'émetteur PEUT choisir de ne pas remettre à zéro le compteur d'erreurs si le dernier tronçon envoyé était une retransmission.

Note : Quand on configure le point d'extrémité SCTP, l'utilisateur devrait éviter d'avoir la valeur de "Association.Max.Retrans" plus grande que la sommae des "Path.Max.Retrans" de toutes les adresses de destination pour le point d'extrémité distant. Autrement, toutes les adresses de destination pourraient devenir inactives alors que le point d'extrémité considère encore le point d'extrémité homologue comme accessible. Quand cette condition se produit, la façon dont SCTP choisit de fonctionner est spécifique de la mise en œuvre.

Quand le chemin principal est marqué inactif (par exemple, à cause de retransmissions excessives) l'envoyeur PEUT automatiquement transmettre de nouveaux paquets à une adresse de destination de remplacement si il en existe une et qu'elle est active. Si plus d'une adresse de remplacement est active quand le chemin principale est marqué inactif, une seule adresse de transport DEVRAIT être choisie et utilisée comme nouvelle adresse de transport de destination.

8.3. Battement de cœur de chemin

Par défaut, un point d'extrémité SCTP DEVRAIT surveiller l'accessibilité de la ou des adresses de transport de destination inactives de son homologue en envoyant un tronçon HEARTBEAT périodiquement à la ou aux adresses de transport de destination. L'envoi de tronçons HEARTBEAT PEUT commencer en atteignant l'état ESTABLISHED et est arrêté après l'envoi d'un tronçon SHUTDOWN ou SHUTDOWN ACK. Un receveur d'un tronçon HEARTBEAT DOIT répondre au tronçon HEARTBEAT avec un tronçon HEARTBEAT ACK après l'entrée dans l'état COOKIE-ECHOED (envoyeur du tronçon INIT) ou dans l'état ESTABLISHED (receveur du tronçon INIT) jusqu'à atteindre l'état SHUTDOWN-SENT

(envoyeur du tronçon SHUTDOWN) ou l'état SHUTDOWN-ACK-SENT (receveur du tronçon SHUTDOWN).

Une adresse de transport de destination est considérée comme "inactive" si aucun nouveau tronçon qui puisse être utilisé pour mettre à jour le RTT du chemin (généralement incluant d'abord des transmissions de tronçons DATA, INIT, COOKIE ECHO, ou HEARTBEAT, etc.) et aucun tronçon HEARTBEAT ne lui a été envoyé dans la période courante de battement de cœur de cette adresse. Cela s'applique aux adresses de destination actives et inactives.

La couche supérieure peut facultativement initier les fonctions suivantes :
A) Désactiver le battement de cœur sur une adresse de transport de destination spécifique d'une certaine association,
B) Changer l'intervalle de battement de cœur.
C) Réactiver le battement de cœur sur une adresse de transport de destination spécifique d'une certaine association, et
D) Demander l'envoi d'un tronçon HEARTBEAT à la demande sur une adresse de transport de destination spécifique d'une certaine association.

Le point d'extrémité DEVRAIT incrémenter le compteur d'erreurs de l'adresse de transport de destination chaque fois qu'un tronçon HEARTBEAT est envoyé à cette adresse et non acquitté dans un RTO.

Quand la valeur de ce compteur excède le paramètre de protocole "Path.Max.Retrans", le point d'extrémité DEVRAIT marquer l'adresse de destination correspondante comme inactive si elle n'est pas marquée ainsi et DEVRAIT aussi rapporter à la couche supérieure le changement d'accessibilité de cette adresse de destination. Après cela, le point d'extrémité DEVRAIT continuer d'envoyer des tronçons HEARTBEAT sur cette adresse de destination mais DEVRAIT arrêter d'augmenter le compteur.

L'envoyeur du tronçon HEARTBEAT DEVRAIT inclure dans le champ Informations de battement de cœur du tronçon l'heure courante d'envoi du paquet et l'adresse de destination à laquelle le paquet est envoyé.

Note de mise en œuvre : une autre mise en œuvre du mécanisme de battement de cœur qui peut être utilisée est d'incrémenter la variable de compteur d'erreurs chaque fois qu'un tronçon HEARTBEAT est envoyé à une destination. Chaque fois qu'arrive un tronçon HEARTBEAT ACK, l'envoyeur DEVRAIT remetrre à zéro le compteur d'erreurs de la destination sur laquelle le tronçon HEARTBEAT a été envoyé. Cela supprimerait en effet les erreurs précédemment marquées (et aussi tous les autres comptes d'erreur).

Le receveur du tronçon HEARTBEAT DEVRAIT immédiatement répondre avec un tronçon HEARTBEAT ACK contenant la TLV Informations de battement de cœur, ainsi que toutes autres TLV reçues, copiées inchangées du tronçon HEARTBEAT reçu.

À réception du tronçon HEARTBEAT ACK, l'envoyeur du tronçon HEARTBEAT DEVRAIT remettre à zéro le compteur d'erreur de l'adresse de transport de destination à laquelle a été envoyé le tronçon HEARTBEAT et marquer l'adresse de transport de destination comme active si elle ne l'est pas déjà. Le point d'extrémité DEVRAIT rapporter à la couche supérieure quand une adresse de destination inactive est marquée comme active à cause de la réception du dernier tronçon HEARTBEAT ACK. Le receveur du tronçon HEARTBEAT ACK DEVRAIT aussi remettre à zéro le compte global d'erreurs de l'association (comme défini au paragraphe 8.1).

Le receveur du tronçon HEARTBEAT ACK DEVRAIT aussi effectuer une mesure de RTT pour cette adresse de transport de destination en utilisant la valeur d'heure portée dans le tronçon HEARTBEAT ACK.

Sur une adresse de destination inactive qui a la permission de faire des battements de cœur, il est RECOMMANDÉ qu'un tronçon HEARTBEAT soit envoyé une fois par RTO de cette adresse de destination plus le paramètre de protocole "HB.interval", avec une gigue de +/- 50 % de la valeur de RTO et un repli exponentiel du RTO si le tronçon HEARTBEAT précédent n'est pas acquitté.

Une primitive est fournie pour que l'utilisateur SCTP change l'intervalle de battement de cœur *(HB.interval)* et active ou désactive le battement de cœur sur une adresse de destination donnée. Le paramètre "HB.interval" établi par l'utilisateur SCTP est ajouté au RTO de cette destination (incluant tout repli exponentiel). Un seul battement de cœur DEVRAIT être envoyé chaque fois que le temporisateur de battement de cœur expire (si plusieurs destinations sont inactives). C'est une décision de mise en œuvre de choisir à laquelle des destinations inactives candidates envoyer le battement de cœur (si plus d'une destination est inactive).

Pour le réglage de l'intervalle de battement de cœur, un effet colatéral DEVRAIT être pris en compte. Quand cette valeur est augmentée, c'est-à-dire, le temps entre l'envoi des tronçons HEARTBEAT est plus long, la détection de tronçons ABORT perdus prend aussi plus longtemps. Si un point d'extrémité homologue envoie un tronçon ABORT pour une raison

quelconque et si le tronçon ABORT est perdu, le point d'extrémité local va seulement découvrir le tronçon ABORT perdu en envoyant un tronçon DATA ou HEARTBEAT (causant donc l'envoi par l'homologue d'un autre tronçon ABORT). Ceci est à prendre en compte lors du réglage du temporisateur de battement de cœur. Si l'envoyeur des tronçons HEARTBEAT est désactivé, seul l'envoi de tronçons DATA à l'association va découvrir la perte d'un tronçon ABORT chez l'homologue.

8.4 Traitement des paquets "sortants de nulle part"

Un paquet SCTP est dit "sorti de nulle part" (OOTB, *Out of the Blue*) si il est correctement formé (c'est-à-dire, que la vérification de CRC32c du receveur a réussi ; voir au paragraphe 6.8) mais le receveur n'est pas capable d'identifier l'association à laquelle appartient ce paquet.

Le receveur d'un paquet OOTB fait ce qui suit :

1) Si le paquet OOTB est pour ou provient d'une adresse qui n'est pas d'envoi individuel, un receveur DEVRAIT éliminer en silence le paquet. Autrement,
2) Si le paquet OOTB contient un tronçon ABORT, le receveur DOIT éliminer en silence le paquet OOTB et ne rien faire d'autre. Autrement,
3) Si le paquet contient un tronçon INIT avec une étiquette de vérification réglée à 0, il DEVRAIT être traité comme décrit au paragraphe 5.1. Si, pour une raison quelconque, le tronçon INIT ne peut pas être traité normalement et si un tronçon ABORT doit être envoyé en réponse, l'étiquette de vérification du paquet contenant le tronçon ABORT DOIT être l'étiquette Initier du tronçon INIT reçu, et le bit T du tronçon ABORT doit être réglé à 0, indiquant que l'étiquette de vérification n'est pas reflétée. Autrement,
4) Si le paquet contient un tronçon COOKIE ECHO comme premier tronçon, il DOIT être traité comme décrit au paragraphe 5.1. Autrement,
5) Si le paquet contient un tronçon SHUTDOWN ACK, le receveur DEVRAIT répondre à l'envoyeur du paquet OOTB avec un tronçon SHUTDOWN COMPLETE. Quand il envoie le tronçon SHUTDOWN COMPLETE, le receveur du paquet OOTB DOIT remplir le champ Étiquette de vérification du paquet sortant avec l'étiquette de vérification reçue dans le tronçon SHUTDOWN ACK et établir le bit T dans les fanions de tronçon pour indiquer que l'étiquette de vérification est reflétée. Autrement,
6) Si le paquet contient un tronçon SHUTDOWN COMPLETE, le receveur DEVRAIT éliminer en silence le paquet et ne rien faire d'autre. Autrement,
7 Si le paquet contient un tronçon ERROR avec la cause d'erreur "Mouchard périmé" ou un tronçon COOKIE ACK, le paquet SCTP DEVRAIT être éliminé en silence. Autrement,
8) Le receveur DEVRAIT répondre à l'envoyeur du paquet OOTB avec un tronçon ABORT. Quand il envoie le tronçon ABORT, le receveur du paquet OOTB DOIT remplir le champ Étiquette de vérification du paquet sortant avec la valeur trouvée dans le champ Étiquette de vérification du paquet OOTB et établir le bit T dans les fanions de tronçon pour indiquer que l'étiquette de vérification est reflétée. Après l'envoi de ce tronçon ABORT, le receveur du paquet OOTB DOIT éliminer le paquet OOTB et NE DOIT PAS prendre d'autre action.

8.5 Étiquette de vérification

Les règles d'étiquette de vérification définies dans ce paragraphe s'appliquent lors de l'envoi ou la réception de paquets SCTP qui ne contiennent pas de tronçon INIT, SHUTDOWN COMPLETE, COOKIE ECHO (voir le paragraphe 5.1), ABORT, ou SHUTDOWN ACK. Les règles d'envoi et de réception de paquets SCTP contenant un de ces types de tronçon sont discutées séparémena au paragraphe 8.5.1.

Quand il envoie un paquet SCTP, le point d'extrémité DOIT remplir le champ Étiquette de vérification du paquet sortant avec la valeur d'étiquette dans le paramètre Étiquette Initier du tronçon INIT ou INIT ACK reçu de son homologue.

Quand il reçoit un paquet SCTP, le point d'extrémité DOIT s'assurer que la valeur dans le champ Étiquette de vérification du paquet SCTP reçu correspond à celle de sa propre étiquette. Si la valeur de l'étiquette de vérification reçue ne correspond pas à celle de l'étiquette du receveur, celui-ci DOIT éliminer en silence le paquet et NE DOIT PAS le traiter, sauf dans les cas mentionnées au paragraphe suivant.

8.5.1 Exceptions aux règles d'étiquette de vérification

A) Règles pour les paquets portant un tronçon INIT :
   * L'envoyeur DOIT régler l'étiquette de vérification du paquet à 0.
   * Quand un point d'extrémité reçoit un paquet SCTP avec l'étiquette de vérification réglée à 0, il DEVRAIT vérifier que le paquet contient seulement un tronçon INIT. Autrement, le receveur DOIT éliminer en silence le paquet.

B) Règles pour les paquets portant un tronçon ABORT:
   * Le point d'extrémité DOIT toujours remplir le champ Étiquette de vérification du paquet sortant avec la valeur d'étiquette du point d'extrémité de destination si elle est connue.
   * Si le tronçon ABORT est envoyé en réponse à un paquet OOTB, le point d'extrémité DOIT suivre la procédure décrite au paragraphe 8.4.
   * Le receveur d'un tronçon ABORT DOIT accepter le paquet si le champ Étiquette de vérification du paquet correspond à sa propre étiquette et si le bit T n'est pas établi OU si il est établi dans l'étiquette de son homologue et si le bit T est établi dans les fanions de tronçon. Autrement, le receveur DOIT éliminer en silence le paquet et ne rien faire d'autre.

C) Règles pour les paquets portant un tronçon SHUTDOWN COMPLETE :
   * Quand il envoie un tronçon SHUTDOWN COMPLETE, si le receveur du tronçon SHUTDOWN ACK a un TCB, alors l'étiquette du point d'extrémité de destination DOIT être utilisée et le bit T NE DOIT PAS être établi. C'est seulement lorsque il n'existe pas de TCB que l'envoyeur DEVRAIT utiliser l'étiquette de vérification provenant du tronçon SHUTDOWN ACK et DOIT établir le bit T.
   * Le receveur d'un tronçon SHUTDOWN COMPLETE accepte le paquet si le champ Étiquette de vérification du paquet correspond à sa propre étiquette et si le bit T n'est pas établi OU si il est établi à l'étiquette de son homologue et le bit T est établi dans les fanions de tronçon. Autrement, le receveur DOIT éliminer en silence le paquet et ne rien faire d'autre. Un point d'extrémité DOIT ignorer le tronçon SHUTDOWN COMPLETE si il n'est pas dans l'état SHUTDOWN-ACK-SENT.

D) Règles pour les paquets portant un tronçon COOKIE ECHO :
   * Quand il envoie un tronçon COOKIE ECHO, le point d'extrémité DOIT utiliser la valeur de l'étiquette Initier reçue dans le tronçon INIT ACK.
   * Le receveur d'un tronçon COOKIE ECHO suit les procédures de la Section 5.

E) Règles pour les paquets portant un tronçon SHUTDOWN ACK :
   * Si le receveur est dans l'état COOKIE-ECHOED ou COOKIE-WAIT, les procédures du paragraphe 8.4 DEVRAIENT être suivies ; en d'autres termes, il est traité comme un paquet OOTB.

9. Terminaison d'association

Un point d'extrémité DEVRAIT terminer son association quand il quitte le service. Une association peut être terminée par interruption ou par fermeture. L'interruption d'une association est interruptive par définition en ce que toutes les données en instance sur l'un et l'autre côté de l'association sont éliminées et non livrées à l'homologue. Une fermeture d'une association est considérée comme une cloture en douceur où toutes les données en file d'attente dans l'un et l'autre point d'extrémité sont livrées aux homologues respectifs. Cependant, dans le cas d'une fermeture, SCTP ne prend pas en charge un état semi ouvert (comme le fait TCP) dans lequel un côté peut continuer d'envoyer des données alors que l'autre côté est fermé. Quand l'un ou l'autre point d'extrémité effectue une fermeture, l'association chez chaque homologue va arrêter d'accepter de nouvelles données provenant de son utilisateur et seulement livrer les données dans la file d'attente au moment de l'envoi ou de la réception du tronçon SHUTDOWN.

9.1 Interruption d'une association

Quand un point d'extrémité décide d'interrompre une association existante, il DOIT envoyer un tronçon ABORT à son point d'extrémité homologue. L'envoyeur DOIT remplir l'étiquette de vérification de l'homologue dans le paquet sortant et NE DOIT PAS grouper de tronçon DATA avec le tronçon ABORT. Si l'association est interrompue sur demande de la couche supérieure, une cause d'erreur "Interruption à l'initiative de l'utilisateur" (voir au paragraphe 3.3.10.12) DEVRAIT être présente dans le tronçon ABORT.

Un point d'extrémité NE DOIT répondre à aucun paquet reçu qui contient un tronçon ABORT (voir aussi le paragraphe 8.4).

Un point d'extrémité qui reçoit un tronçon ABORT DOIT appliquer les règles spéciales de vérification d'étiquette de vérification décrites au paragraphe 8.5.1.

Après la vérification de l'étiquette de vérification, le point d'extrémité receveur DOIT retirer l'association de ses

enregistrements et DEVRAIT rapporter la terminaison à sa couche supérieure. Si une cause d'erreur "Interruption à l'initiative de l'utilisateur" est présente dans le tronçon ABORT, la raison de l'interruption de couche supérieure DEVRAIT être mise à la disposition de la couche supérieure.

9.2 Fermeture d'une association

En utilisant la primitive SHUTDOWN (voir le paragraphe 11.1) la couche supérieure d'un point d'extrémité dans une association peut fermer en douceur l'association. Cela va permettre que tous les tronçons DATA en instance en provenance de l'homologue de l'initiateur de la fermeture soient livrés avant la terminaison de l'association.

À réception de la primitive SHUTDOWN provenant de sa couche supérieure, le point d'extrémité entre dans l'état SHUTDOWN-PENDING et y reste jusqu'à ce que toutes les données en instance aient été acquittées par son homologue. Le point d'extrémité n'accepte aucune nouvelle donnée de sa couche supérieure mais retransmet les données au point d'extrémité homologue si nécessaire pour boucher les trous.

Une fois que toutes les données en instance ont été acquittées, le point d'extrémité envoie un tronçon SHUTDOWN à son homologue, incluant dans le champ Accusé de réception de TSN cumulatif le dernier TSN en séquence reçu de l'homologue. Il DEVRAIT alors lancer le temporisateur T2-shutdown et entrer dans l'état SHUTDOWN-SENT. Si le temporisateur expire, le point d'extrémité DOIT renvoyer le tronçon SHUTDOWN avec le dernier TSN séquentiel mis à jour reçu de son homologue.

Les règles du paragraphe 6.3 DOIVENT être suivies pour déterminer la valeur de temporisateur appropriée pour T2-shutdown. Pour indiquer des trous dans le TSN, le point d'extrémité PEUT aussi grouper un tronçon SACK avec le tronçon SHUTDOWN dans le même paquet SCTP.

Un point d'extrémité DEVRAIT limiter le nombre de retransmissions du tronçon SHUTDOWN au paramètre de protocole "Association.Max.Retrans". Si ce seuil est dépassé, le point d'extrémité DEVRAIT détruire le TCB et DEVRAIT rapporter le point d'extrémité homologue comme inaccessible à la couche supérieure (et donc l'association entre dans l'état CLOSED). La réception d'un paquet provenant de son homologue (c'est-à-dire, lorsque l'homologue envoie tous ses tronçons DATA mis en file d'attente) DEVRAIT remettre à zéro le compte de retransmissions du point d'extrémité et redémarrer le temporisateur T2-shutdown, donnant à son homologue une ample opportunité de transmettre tous les tronçons DATA en file d'attente qu'il n'a pas encore envoyés.

À réception du tronçon SHUTDOWN, le point d'extrémité homologue fait ce qui suit :

* il entre dans l'état SHUTDOWN-RECEIVED,
* il cesse d'accepter de nouvelles données de son utilisateur SCTP, et
* il vérifie, en examinant le champ Accusé de réception de TSN cumulatif du tronçon, que tous ses tronçons DATA en instance ont été reçus par le tronçon SHUTDOWN envoyeur.

Une fois qu'un point d'extrémité a atteint l'état SHUTDOWN-RECEIVED, il DOIT ignorer les demandes de cloture provenant de l'ULP mais DOIT continuer de répondre aux tronçons SHUTDOWN venant de son homologue.

Si il reste encore des tronçons DATA en instance, le receveur du tronçon SHUTDOWN DOIT continuer de suivre les procédures normales de transmission des données définies à la Section 6, jusqu'à ce que tous les tronçons DATA en instance soient acquittés ; cependant, le receveur du tronçon SHUTDOWN NE DOIT PAS accepter de nouvelles données venant de son utilisateur SCTP.

Lorsque il est dans l'état SHUTDOWN-SENT, l'envoyeur du tronçon SHUTDOWN DOIT répondre immédiatement à chaque paquet reçu contenant un ou plusieurs tronçons DATA avec un tronçon SHUTDOWN et redémarrer le temporisateur T2-shutdown. Si un tronçon SHUTDOWN ne peut pas par lui-même accuser réception de tous les tronçons DATA reçus (c'est-à-dire, il y a des TSN qui ne peuvent pas être acquittés qui sont plus grands que le TSN cumulatif et donc des trous existent dans la séquence des TSN) ou si des TSN dupliqués ont été reçus, alors un tronçon SACK DOIT aussi être envoyé.

L'envoyeur du tronçon SHUTDOWN PEUT aussi démarrer un temporisateur de garde global T5-shutdown-guard pour limiter la durée globale de la séquence de fermeture. À l'expiration de ce temporisateur, l'envoyeur DEVRAIT interrompre l'association en envoyant un tronçon ABORT. Si le temporisateur T5-shutdown-guard est utilisé, il DEVRAIT être réglé à la valeur RECOMMANDÉE de 5 fois "RTO.Max".

Si le receveur du tronçon SHUTDOWN n'a plus de tronçons DATA en instance, le receveur du tronçon SHUTDOWN

DOIT envoyer un tronçon SHUTDOWN ACK et démarrer de lui-même un temporisateur T2-shutdown, entrant dans l'état SHUTDOWN-ACK-SENT. Si le temporisateur expire, le point d'extrémité DOIT envoyer à nouveau le tronçon SHUTDOWN ACK.

L'envoyeur du tronçon SHUTDOWN ACK DEVRAIT limiter le nombre de retransmissions du tronçon SHUTDOWN ACK au paramètre de protocole "Association.Max.Retrans". Si ce seuil est dépassé, le point d'extrémité DEVRAIT détruire le TCB et DEVRAIT rapporter le point d'extrémité homologue comme injoignable à la couche supérieure (et donc l'association entre dans l'état CLOSED).

À réception du tronçon SHUTDOWN ACK, l'envoyeur du tronçon SHUTDOWN DOIT arrêter le temporisateur T2-shutdown, envoyer un tronçon SHUTDOWN COMPLETE à son homologue, et supprimer tous les enregistrements de l'association.

À réception du tronçon SHUTDOWN COMPLETE, le point d'extrémité vérifie qu'il est dans l'état SHUTDOWN-ACK-SENT ; si il ne l'est pas, le tronçon DEVRAIT être éliminé. Si le point d'extrémité est dans l'état SHUTDOWN-ACK-SENT, le point d'extrémité DEVRAIT arrêter le temporisateur T2-shutdown et supprimer toutes les informations relatives à l'association (et donc, l'association entre dans l'état CLOSED).

Un point d'extrémité DEVRAIT s'assurer que tous ses tronçons DATA en instance ont été acquittés avant d'initier la procédure de fermeture.

Un point d'extrémité DEVRAIT rejeter toute demande de nouvelles données provenant de sa couche supérieure si il est dans l'état SHUTDOWN-PENDING, SHUTDOWN-SENT, SHUTDOWN-RECEIVED, ou SHUTDOWN-ACK-SENT.

Si un point d'extrémité est dans l'état SHUTDOWN-ACK-SENT et reçoit un tronçon INIT (par exemple, si le tronçon SHUTDOWN COMPLETE a été perdu) avec des adresses de transport de source et destination (dans les adresses IP ou dans le tronçon INIT) qui appartiennent à cette association, il DEVRAIT éliminer le tronçon INIT et retransmettre le tronçon SHUTDOWN ACK.

Note : La réception d'un paquet contenant un tronçon INIT avec les mêmes adresses IP de source et destination qu'utilisées dans les adresses de transport allouées à un point d'extrémité mais avec un numéro d'accès différent indique l'initialisation d'une association séparée.

L'envoyeur du tronçon INIT ou COOKIE ECHO DEVRAIT répondre à la réception d'un tronçon SHUTDOWN ACK avec un tronçon SHUTDOWN COMPLETE autonome dans un paquet SCTP avec le champ Étiquette de vérification de son en-tête commun réglé à la même étiquette que reçue dans le paquet contenant le tronçon SHUTDOWN ACK. Ceci est considéré comme un paquet OOTB comme défini au paragraphe 8.4. L'envoyeur du tronçon INIT laisse T1-init continuer de tourner et reste dans l'état COOKIE-WAIT ou COOKIE-ECHOED. L'expiration normale du temporisateur T1-init va causer la retransmission du tronçon INIT ou COOKIE et donc le démarrage d'une nouvelle association.

Si un tronçon SHUTDOWN est reçu dans l'état COOKIE-WAIT ou COOKIE ECHOED, le tronçon SHUTDOWN DEVRAIT être éliminé en silence.

Si un point d'extrémité est dans l'état SHUTDOWN-SENT et reçoit de son homologue un tronçon SHUTDOWN, le point d'extrémité DEVRAIT respondre immédiatement avec un tronçon SHUTDOWN ACK à son homologue et passer à l'état SHUTDOWN-ACK-SENT, redémarrant son temporisateur T2-shutdown.

Si un point d'extrémité est dans l'état SHUTDOWN-ACK-SENT et reçoit un SHUTDOWN ACK, il DOIT arrêter le temporisateur T2-shutdown, envoyer un tronçon SHUTDOWN COMPLETE à son homologue, et supprimer tous les enregistrements de l'association.

10. Traitement de ICMP

Chaque fois qu'un message ICMP est reçu par un point d'extrémité SCTP, les procédures suivantes DOIVENT être suivies pour assurer une utilisation appropriée des informations fournies par la couche 3.

ICMP1) Une mise en œuvre PEUT ignorer tous les messages ICMPv4 où le champ Type n'est pas réglé à "Destination injoignable".

ICMP2) Une mise en œuvre PEUT ignorer tous les messages ICMPv6 où le champ Type n'est pas réglé à "Destination injoignable", "Problème de paramètre", ou "Paquet trop gros".

ICMP3) Une mise en œuvre DEVRAIT ignorer tout message ICMP où le code indique "Accès injoignable".

ICMP4) Une mise en œuvre PEUT ignorer tous les messages ICMPv6 de type "Problème de paramètre" si le code n'est pas "Type de prochain en-tête rencontré inconnu".

ICMP5) Une mise en œuvre DOIT utiliser la charge utile du message ICMP (v4 ou v6) pour localiser l'association qui a envoyé le message auquel ICMP répond. Si l'association ne peut pas être trouvée, une mise en œuvre DEVRAIT ignorer le message ICMP.

ICMP6) Une mise en œuvre DOIT valider que l'étiquette de vérification contenue dans le message ICMP correspond à l'étiquette de vérification de l'homologue. Si l'étiquette de vérification n'est pas 0 et ne correspond pas, éliminer le message ICMP. Si elle est 0 et si le message ICMP contient assez d'octets pour vérifier que le type de tronçon est un tronçon INIT et que l'étiquette Initier correspond à l'étiquette de l'homologue, continuer avec ICMP7. Si le message ICMP est trop court ou si le type de tronçon ou l'étiquette Initier ne correspond pas, éliminer en silence le paquet.

ICMP7) Si le message ICMP est soit un message ICMPv6 de type "Paquet trop gros" soit un message ICMPv4 de type "Destination injoignable" et de code "Fragmentation nécessaire", une mise en œuvre DEVRAIT traiter cette information comme défini pour la découverte de la PMTU.

ICMP8) Si le code ICMP est "Type du prochain en-tête rencontré non reconnu" ou "Protocole injoignable", une mise en œuvre DOIT traiter ce message comme une interruption avec le bit T établi si il ne contient pas un tronçon INIT. Si il contient un tronçon INIT et si l'association est dans l'état COOKIE-WAIT, traiter le message ICMP comme un tronçon ABORT.

ICMP9) Si le type ICMP est "Destination injoignable", la mise en œuvre PEUT pssser la destination à l'état injoigable ou, autrement, incrémenter le compteur d'erreur de chemin. SCTP PEUT fournir des informations à la couche supérieure indiquant la réception des messages ICMP quand elle rapporte un changement de l'état du réseau.

Ces procédures diffèrent de celles de la [RFC1122] et des exigences pour le traitement des messages d'accès injoignable et de l'exigence qu'une mise en œuvre DOIT interrompre les associations en réponse à un message Protocole injoignable. Les messages Protocole injoignable ne sont pas traités, car une mise en œuvre va envoyer un tronçon ABORT, et non un message Protocole injoignable. Le traitement plus strict du message Protocole injoignable est dû à des problèmes de sécurité pour les hôtes qui ne prennent pas en charge SCTP.

11. Interface avec la couche supérieure

Les protocoles de couche supérieure (ULP) demandent des services en passant des primitives à SCTP et reçoivent des notifications de SCTP pour divers événements.

Les primitives et notifications décrites dans cette section peuvent être utilisées comme des lignes directrices pour mettre en œuvre SCTP. La description fonctionnelle suivante des primitives d'interface de l'ULP est montrée à des fins d'illustration. Des mises en œuvre différentes de SCTP peuvent avoir des interfaces d'ULP différentes. Cependant, toutes les mises en œuvre de SCTP sont supposées fournir un certain ensemble minimum de services pour garantir que toutes les mises en œuvre de SCTP peuvent prendre en charge la même hiérarchie de protocoles.

Noter que cette section est seulement pour information.

La [RFC6458] et la Section 7 ("Considérations d'API de prise") de la [RFC7053] définissent une extension de l'API de prise pour SCTP comme décrit dans le présent document.

11.1 De ULP à SCTP

Les paragraphes qui suivent caractérisent les fonctions d'une interface ULP/SCTP. La notation utilisée est similaire à celle de l'invocation de la plupart des procédures ou fonctions dans les langages de haut niveau.

Les primitives d'ULP décrites ci-dessous spécifient les fonctions de base qu'effectue SCTP pour prendre en charge la communication entre les processus. Les mises en œuvre individuelles définissent leur propre format exact et fournissent des combinaisons ou sous ensembles des fonctions de base dans les invocations seules.

11.1.1 "Initialize"

INITIALIZE([accès local],[liste d'adresses locales éligibles]) -> nom de l'instance SCTP locale

Cette primitive permet à SCTP d'initialiser ses structures internes de données et alloue les ressources nécessaires pour établir l'environnement de fonctionnement. Une fois que SCTP est initialisé, l'ULP peut communiquer directement avec les autres points d'extrémité sans réinvoquer cette primitive.

SCTP va retourner un nom d'instance locale de SCTP à l'ULP.

Attributs obligatoires : aucun.

Attributs facultatifs :
accès local : numéro d'accès SCTP, si l'ULP veut qu'il soit spécifié.
liste d'adresses locales éligibles : liste des adresses que le point d'extrémité local SCTP lie. Par défaut, si une liste d'adresses n'est pas incluse, toutes les adresses IP allouées à l'hôte sont utilisées par le point d'extrémité local.

Note de mise en œuvre : Si cet attribut facultatif n'est pas pris en charge par une mise en œuvre, il va être de la responsabilité de la mise en œuvre d'appliquer que le champ Adresse IP de source de tous les paquets SCTP envoyés par ce point d'extrémité contiennne une des adresses IP indiquées dans la liste d'adresses locales éligibles.

11.1.2 "Associate"

ASSOCIATE(nom local d'instance SCTP, liste initiale d'adresses de transport de destination, compte de flux sortants) -> identifiant d'association [,liste d'adresses de transport de destination] [,compte de flux sortants]

Cette primitive permet à la couche supérieure d'initier une association à un point d'extrémité homologue spécifique.

Le point d'extrémité homologue est spécifié par une ou plusieurs des adresses de transport qui définissent le point d'extrémité (voir le paragraphe 1.3). Si l'instance SCTP locale n'a pas été initialisée, le ASSOCIATE est considéré comme étant une erreur.

Un identifiant d'association, qui est une bride locale vers l'association SCTP, va être retourné à l'établissement réussi de l'association. Si SCTP n'est pas capable d'ouvrir l'association SCTP avec le point d'extrémité homologue, une erreur est retournée.

D'autres paramètres d'association peuvent être retournés, incluant les adresses complètes de transport de destination de l'homologue ainsi que le compte de flux sortants du point d'extrémité local. Une des adresses de transport des adresses de destination retournées va être choisie par le point d'extrémité local comme chemin principal par défaut pour l'envoi des paquets SCTP à cet homologue. La "liste d'adresses de transport de destination" retournée peut être utilisée par l'ULP pour changer le chemin principal par défaut ou pour forcer l'envoi d'un paquet à une adresse de transport spécifique.

Note de mise en œuvre : Si la primitive ASSOCIATE est mise en œuvre comme une invocation de fonction de blocage, la primitive ASSOCIATE peut retourner des paramètres d'association en plus de l'identifiant d'association à la réussite de l'établissement. Si la primitive ASSOCIATE est mise en œuvre comme une invocation non bloquante, seul l'identifiant d'association est retourné et les paramètres d'association sont passés en utilisant la notification COMMUNICATION UP.

Attributs obligatoires :
nom local d'instance SCTP : obtenu de l'opération INITIALIZE.
liste initiale d'adresses de transport de destination : liste non vide des adresses de transport du point d'extrémité homologue avec lequel l'association est à établir.
compte de flux sortants : nombre de flux sortants que l'ULP voudrait ouvrir vers ce point d'extrémité homologue.

Attributs facultatifs : aucun.

11.1.3 "Shutdown"

SHUTDOWN(identifiant d'association) -> résultat

Ferme en douceur une association. Toutes les données d'utilisateur mises en file d'attente en local vont être livrées à l'homologue. L'association ne va être terminée qu'après que l'homologue a accusé réception de tous les paquets SCTP envoyés. Un code de succès va être retourné à la terminaison réussie de l'association. Si la tentative de terminaison de l'association résulte en un échec, un code d'erreur est retourné.

Attributs obligatoires : identifiant d'association : bride locale vers l'association SCTP.

Attributs facultatifs : aucun.

11.1.4 "Abort"

ABORT(identifiant d'association [, Raison de l'interruption de couche supérieure]) -> résultat

Cloture sans douceur une association. Toutes les données d'utilisateur mises en mémoire tampon en local vont être éliminées, et an tronçon ABORT est envoyé à l'homologue. Un code de succès va être retourné à la réussite de l'interruption de l'association. Si la tentative d'interruption de l'association résulte en un échec, un code d'erreur est retourné.

Attributs obligatoires : identifiant d'association : bride locale vers l'association SCTP.

Attributs facultatifs : Raison d'interruption de couche supérieure : raison de l'interruption à passer à l'homologue.

11.1.5 "Send"

SEND(identifiant d'association, adresse de mémoire tampon, compte d'octets [,contexte]
[,identifiant de flux] [,durée de vie] [,adresse de transport de destination]
[,fanion non ordonné] [,fanion pas de groupement] [,identifiant de protocole de charge utile]
[,fanion sack immédiat]) -> résultat

C'est la principale méthode pour envoyer des données d'utilisateur via SCTP.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de mémoire tampon : localisation de l'endroit où le message d'utilisateur à transmettre est mémorisé.
compte d'octets : la taille des données d'utilisateur en nombre d'octets.

Attributs facultatifs :
contexte : informations facultatives fournies qui vont être portées dans la notification SEND FAILURE à l'ULP si le transport de ce message d'utilisateur échoue.
identifiant de flux : indique sur quel flux envoyer les données. Si il n'est pas spécifié, le flux 0 va être utilisé.
durée de vie : spécifie la durée de vie des données d'utilisateur. Les données d'utilisateur nevont pas être envoyées par SCTP après l'expiration de la durée de vie. Ce paramètre peut être utilisé pour éviter des efforts pour transmettre des messages d'utilisateur périmés. SCTP notifie à l'ULP si les données ne peuvent pas être initiées pour le transport (c'est-à-dire, envoyées à la destination via la primitive SEND de SCTP) dans la variable Durée de vie. Cependant, les données d'utilisateur vont être transmises si SCTP a tenté de transmettre un tronçon avant l'expiration de la durée de vie.

Note de mise en œuvre : afin de mieux prendre en charge l'option de durée de vie des données, l'émetteur peut retenir l'allocation du numéro de TSN d'un tronçon DATA sortant au dernier moment. Et, pour la simplicité de la mise en œuvre, une fois qu'un numéro de TSN a été alloué, l'envoyeur considére l'envoi de ce tronçon DATA comme engagé, outrepasant toute option de durée de vie attachée au tronçon DATA.

adresse de transport de destination : spécifiée comme une des adresses de transport de destination du point d'extrémité homologue auquel ce paquet est envoyé. Chaque fois que possible, SCTP utilise cette adresse de transport de destination pour l'envoi des paquets, plutôt que le chemin principal courant.

fanion non ordonné : ce fanion, si il est présent, indique que l'utilisateur aimerait que les données soient livrées de façon

non ordonnée à l'homologue (c'est-à-dire, le fanion U est établi à 1 sur tous les tronçons DATA portant ce message).

fanion pas de groupement : il donne pour instruction à SCTP de ne pas retarder l'envoi des tronçons DATA pour ces données d'utilisateur juste pour leur permettre d'être groupées avec d'autres tronçons DATA sortants. Quand il y a de l'encombrement dans le réseau, SCTP pourrait quand même grouper les données, même quand ce fanion est présent.

identifiant de charge utile de protocole : entier non signé de 32 bits qui est à passer à l'homologue, indiquant le type de données de protocole de charge utile transmises. Noter que la couche supérieure est chargée par l'hôte de convertir ce champ dans l'ordre des octets du réseau, qui est passé par SCTP comme 4 octets de données opaques.

fanion sack immédiat : établit le bit I sur le dernier tronçon DATA utilisé pour le message d'utilisateur à transmettre.

11.1.6 "Set Primary"

SETPRIMARY(identifiant d'association, adresse de transport de destination, [adresse de transport de source]) -> résultat

Ordonne au SCTP local d'utiliser l'adresse de transport de destination spécifiée comme chemin principal pour l'envoi des paquets.

Le résultat de la tentative d'opération est retourné. Si l'adresse de transport de destination spécifiée n'est pas présente dans la "liste d'adresses de transport de destination" retournée précédemment dans une primitive ASSOCIATE ou une notification COMMUNICATION UP, une erreur est retournée.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.

adresse de transport de destination : spécifiée comme une des adresses de transport du point d'extrémité homologue, qui est utilisée comme l'adresse principale pour l'envoi des paquets. Elle outrepasse les informations d'adresse principale courante conservées par le point d'extrémité local SCTP.

Attributs facultatifs :
adresse de transport de source : facultative, certaines mises en œuvre peuvent permettre de régler l'adresse de source par défaut placée dans tous les datagrammes IP sortants.

11.1.7 "Receive"

RECEIVE(identifiant d'association, adresse de mémoire tampon, taille de mémoire tampon [,identifiant de flux]) -> compte d'octets [,adresse de transport] [,identifiant de flux] [,numéro de séquence de flux] [,fanion partiel] [,identifiant de protocole de charge utile]

Cette primitive lit le premier message d'utilisateur dans la file d'attente SCTP dans la mémoire tampon spécifiée par l'ULP, si il en est une disponible. La taille du message lu, en octets, va être retournée. Elle peut aussi, selon la mise en œuvre spécifique, retourner d'autres informations, comme l'adresse de l'envoyeur, l'identifiant du flux sur lequel il est reçu, si il y a plus de messages disponibles à restituer, etc. Pour des messages ordonnés, leur numéro de séquence de flux pourrait aussi être retourné.

Selon la mise en œuvre, si cette primitive est invoquée quand aucun message n'est disponible, la mise en œuvre retourne une indication de cette condition ou bloque le processus d'invocation jusqu'à ce que des données deviennent disponibles.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de mémoire tampon : localisation de la mémoire indiquée par l'ULP pour mémoriser le message reçu.
taille de mémoire tampon : taille maximum des données à recevoir, en octets.

Attributs facultatifs :
identifiant de flux : pour indiquer sur quel flux recevoir les données.
numéro de séquence de flux : il est alloué par l'homologue SCTP envoyeur.
fanion partiel : si ce fanion retourné est établi à 1, cette primitive contient alors une livraison partielle du message entier. Quand ce fanion est établi, l'identifiant de flux et le numéro de séquence de flux accompagnent cette primitive. Quand ce fanion est réglé à 0, il indique qu'aucune livraison supplémentaire ne sera reçue pour ce numéro de séquence de flux.

identifiant de protocole de charge utile : entier non signé de 32 bits qui est reçu de l'homologue pour indiquer le type de protocole de charge utile des données reçues. Noter que la couche supérieure est chargée pour l'hôte de la conversion de ce champ dans l'ordre des octets du réseau, qui est passé par SCTP comme 4 octets de données opaques.

11.1.8 "Status"

STATUS(identifiant d'association) -> données d'état

Cette primitive retourne un bloc de données contenant les informations suivantes :
* état de connexion de l'association,
* liste des adresses de transport de destination,
* état d'accessibilité des adresses de transport de destination,
* taille actuelle de la fenêtre du receveur,
* tailles actuelles des fenêtres d'encombrement,
* nombre de tronçons DATA non acquittés,
* nombre de tronçons DATA en instance de réception,
* chemin principal,
* plus récent SRTT sur le chemin principal ,
* RTO sur le chemin principal,
* SRTT et RTO sur les autres adresses de destination, etc.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.

Attributs facultatifs : aucun.

11.1.9 "Change Heartbeat"

CHANGE HEARTBEAT(identifiant d'association, adresse de transport de destination, nouvel état [,intervalle]) -> résultat

Ordonne au point d'extrémité local d'activer ou désactiver le battement de cœur sur l'adresse de transport de destination spécifiée.

Le résultat de cette tentative d'opération est retourné.

Note : Même quand il est activé, le battement de cœur ne va avoir lieu que si l'adresse de transport de destination n'est pas inactive.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de transport de destination : spécifiée comme une des adresses de transport du point d'extrémité homologue.
nouvel état : le nouvel état de battement de cœur pour cette adresse de transport de destination (activé ou désactivé).

Attributs facultatifs :
intervalle : si il est présent, indique la fréquence du batttement de cœur si il est pour activer le batttement de cœur sur une adresse de transport de destination. Cette valeur est ajoutée au RTO de l'adresse de transport de destination. Cette valeur, si elle est présente, affecte toutes les destinations.

11.1.10 "Request Heartbeat"

REQUESTHEARTBEAT(identifiant d'association, adresse de transport de destination) -> résultat

Ordonne au point d'extrémité local d'effectuer un battement de cœur sur l'adresse de transport de destination spécifiée de l'association concernée. Le résultat retourné indique si la transmission du tronçon HEARTBEAT à l'adresse de destination est réussie.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de transport de destination : adresse de transport de l'association sur laquelle le battement de cœur est produit.

Attributs facultatifs : aucun.

11.1.11 "Get SRTT Report"

GETSRTTREPORT(identifiant d'association, adresse de transport de destination) -> résultat srtt

Ordonne au SCTP local de faire rapport de la mesure de SRTT courante sur l'adresse de transport de destination spécifiée de l'association concernée. Le résultat retourné peut être un entier contenant le plus récent SRTT en millisecondes.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de transport de destination : adresse de transport de l'association sur laquelle la mesure de SRTT est à rapporter.

Attributs facultatifs : aucun.

11.1.12 "Set Failure Threshold"

SETFAILURETHRESHOLD(identifiant d'association, adresse de transport de destination, seuil d'échec) -> résultat

Cette primitive permet au SCTP local de personnaliser le seuil de détection d'échec d'accessibilité "Path.Max.Retrans" pour l'adresse de destination spécifiée. Noter que cela peut aussi être fait en utilisant la primitive SETPROTOCOLPARAMETERS (paragraphe 11.1.13).

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
adresse de transport de destination : adresse de transport de l'association sur laquelle le seuil de détection d'échec est à établir.
seuil d'échec : nouvelle valeur de "Path.Max.Retrans" pour l'adresse de destination.

Attributs facultatifs : aucun.

11.1.13 "Set Protocol Parameters"

SETPROTOCOLPARAMETERS(identifiant d'association, [adresse de transport de destination,] liste de paramètres de protocole) -> résultat

Cette primitive permet au SCTP local de personnaliser les paramètres de protocole.

Attributs obligatoires :
identifiant d'association : bride locale vers l'association SCTP.
liste de paramètres de protocole : noms et valeurs spécifiques des paramètres de protocole (par exemple, "Association.Max.Retrans" (Section 16) ou d'autres paramètres comme le DSCP) que l'utilisateur SCTP souhaite personnaliser.

Attributs facultatifs :
adresse de transport de destination : certains des paramètres de protocole pourraient être réglés sur la base de l'adresse de transport de destination.

11.1.14 Message "Receive Unsent"

RECEIVE_UNSENT(identifiant de restitution des données, adresse de mémoire tampon, taille de mémoire tampon, [,identifiant de flux] [, numéro de séquence de flux] [,fanion partiel] [,identifiant de protocole de charge utile])

Cette primitive lit un message d'utilisateur qui n'a jamais été envoyé dans la mémoire tampon spécifiée par l'ULP.

Attributs obligatoires :
identifiant de restitution des données : identification passée à l'ULP dans la notification SEND FAILURE.
adresse de mémoire tampon : localisation de la mémoire indiquée par l'ULP pour mémoriser le message reçu.
taille de mémoire tampon : taille maximum des données à recevoir, en octets.

Attributs facultatifs :
identifiant de flux : c'est une valeur de retour qui est réglée pour indiquer à quel flux les données ont été envoyées.
numéro de séquence de flux : cette valeur est retournée pour indiquer le numéro de séquence de flux qui a été associé au message.
fanion partiel : si ce fanion retourné est établi à 1, alors ce message est une livraison partielle du message entier. Quand ce fanion est établi, l'identifiant de flux et le numéro de séquence de flux accompagnent cette primitive. Quand ce fanion est réglé à 0, il indique qu'aucune autre livraison ne sera reçue pour ce numéro de séquence de flux.
identifiant de protocole de charge utile : entier non signé de 32 bits qui a été établi pour être envoyé à l'homologue, indiquant le type de protocole de charge utile des données reçues.

11.1.15 "Message Receive Unacknowledged"

RECEIVE_UNACKED(identifiant de restitution des données, adresse de mémoire tampon, taille de mémoire tampon, [,identifiant de flux] [,numéro de séquence de flux] [,fanion partiel] [,identifiant de protocole de charge utile])

Cette primitive lit un message d'utilisateur qui a été envoyé et n'a pas été acquitté par l'homologue dans la mémoire tampon spécifiée par l'ULP.

Attributs obligatoires :
identifiant de restitution des données : l'identification passée à l'ULP dans la notification SEND FAILURE.
adresse de mémoire tampon : localisation de la mémoire indiquée par l'ULP pour mémoriser le message reçu.
taille de mémoire tampon : taille maximum des données à recevoir, en octets.

Attributs facultatifs :
identifiant de flux : c'est une valeur retournée qui est réglée à indiquer à quel flux les données ont été envoyées.
numéro de séquence de flux : cette valeur est retournée, indiquant le numéro de séquence de flux qui était associé au message.
fanion partiel : si ce fanion retourné est établi à 1, ce message est une livraison partielle du message entier. Quand ce fanion est établi, l'identifiant de flux et le numéro de séquence de flux accompagnent cette primitive. Quand ce fanion est réglé à 0, il indique qu'aucune livraison supplémentaire ne sera reçue pour ce numéro de séquence de flux.
identifiant de protocole de charge utile : entier non signé de 32 bits qui a été envoyé à l'homologue, indiquant le type de protocole de charge utile des données reçues.

11.1.16 "Destroy SCTP Instance"

DESTROY(nom d'instance locale SCTP)

Attributs obligatoires :
nom d'instance locale SCTP : c'est la valeur qui a été passée à l'application dans la primitive "Initialize" et elle indique quelle instance de SCTP est à détruire.

Attributs facultatifs : aucun.

11.2 De SCTP à l'ULP

Il est supposé que le système d'exploitation ou l'environnement d'application fournit un moyen pour que SCTP signale asynchroniquement le processus d'ULP. Quand SCTP signale un processus d'ULP, certaines informations sont passées à l'ULP.

Note de mise en œuvre : dans certains cas, ceci pourrait être fait par une prise ou canal d'erreur séparé.

11.2.1 Notification DATA ARRIVE

SCTP invoque cette notification sur l'ULP quand un message d'utilisateur est bien reçu et prêt à la restitution.

Ce qui suit pourrait facultativement être passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.
identifiant de flux : pour indiquer sur quel flux les données sont reçues.

11.2.2 Notification SEND FAILURE

Si un message ne peut pas être livré, SCTP invoque cette notification sur l'ULP.

Ce qui suit pourrait facultativement être passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.
identifiant de restitution des données : identification utilisée pour restituer des données non envoyées et non acquittées.
mode : indique si aucune partie du message n'a jamais été envoyée ou si au moins une partie en a été envoyée mais qu'il n'est pas complètement acquitté.
code de cause : indique la raison de l'échec, par exemple, taille trop grande, expiration de la durée de vie du message, etc.
contexte : informations facultatives associées à ce message (voir le paragraphe 11.1.5).

11.2.3 Notification NETWORK STATUS CHANGE

Quand une adresse de transport de destination est marquée inactive (par exemple, quand SCTP détecte un échec) ou est marquée active (par exemple, quand SCTP détecte une récupération) SCTP invoque cette notification sur l'ULP.

Ce qui suit est passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.
adresse de transport de destination : indique l'adresse de transport de destination du point d'extrémité homologue affecté par le changement.
nouvel état : cela indique le nouvel état.

11.2.4 Notification COMMUNICATION UP

Cette notification est utilisée quand SCTP devient prêt à envoyer ou recevoir des messages d'utilisateur ou quand une communication perdue avec un point d'extrémité est restaurée.

Note de mise en œuvre : Si la primitive ASSOCIATE est mise en œuvre comme invocation d'une fonction de blocage, les paramètres d'association sont retournés comme résultat de la primitive ASSOCIATE elle-même. Dans ce cas, la notification COMMUNICATION UP est facultative du côté de l'initiateur de l'association.

Ce qui suit est passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.
état : indique le type d'événement qui s'est produit.
liste d'adresses de transport de destination : ensemble complet des adresses de transport de l'homologue.
compte de flux sortants : nombre maximum de flux dont l'utilisation est permise dans cette association par l'ULP.
compte de flux entrants : nombre de flux que le point d'extrémité homologue a demandé avec cette association (ce pourrait n'être pas le même nombre que "compte de flux sortants").

11.2.5 Notification COMMUNICATION LOST

Quand SCTP perd complètement la communication avec un point d'extrémité (par exemple, via des battements de cœur) ou détecte que le point d'extrémité a effectué une opération d'interruption, il invoque cette notification sur l'ULP.

Ce qui suit est passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.
état : indique quel type d'événement s'est produit ; l'état pourrait indiquer qu'une défaillance OU un événement normal de terminaison s'est produit en réponse à une demande de fermeture ou d'interruption.

Ce qui suit pourrait être passé avec la notification :
dernier accusé de réception : dernier TSN acquitté par ce point d'extrémité homologue.
dernier envoyé : dernier TSN envoyé à ce point d'extrémité homologue.
Raison d'interruption de couche supérieure : raison d'interruption spécifiée en cas d'interruption initiée par l'utilisateur.

11.2.6 Notification COMMUNICATION ERROR

Quand SCTP reçoit un tronçon ERROR de son homologue et décide de le notifier à son ULP, il peut invoquer cette notification sur l'ULP.

Ce qui suit peut être passé avec la notification :

identifiant d'association : bride locale vers l'association SCTP.
informations d'erreur : indique le type d'erreur et facultativement des informations supplémentaires reçues avec le tronçon ERROR.

11.2.7 Notification RESTART

Quand SCTP détecte que l'homologue a redémarré, il pourrait envoyer cette notification à son ULP.

Ce qui suit peut être passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.

11.2.8 Notification SHUTDOWN COMPLETE

Quand SCTP achève les procédures de fermeture (paragraphe 9.2) cette notification est passée à la couche supérieure.

Ce qui suit peut être passé avec la notification :
identifiant d'association : bride locale vers l'association SCTP.

12. Considérations sur la sécurité

12.1. Objectifs de sécurité

Comme protocole commun de transport conçu pour porter de façon fiable des messages d'utilisateur sensibles au délai, comme des messages de facturation ou de signalisation pour les services de téléphonie, entre deux points d'extrémité sur le réseau, SCTP a les objectifs de sécurité suivants :
* disponibilité de services fiables et en temps utile de transport de données
* intégrité des informations d'utilisateur à utilisateur portées par SCTP.

12.2 Réponses de SCTP aux menaces potentielles

SCTP pourrait être utilisé dans des situations de risque très variées.Il est important pour les opérateurs de systèmes qui fonctionnent avec SCTP d'analyser leurs situations particulières et de décider des contre-mesures appropriées.

Les opérateurs de systèmes fonctionnant avec SCTP pourraient consulter la [RFC2196] pour des lignes directrices sur la sécurisation de leur site.

12.2.1 Contrer les attaques par interposition

Les principes de la [RFC2196] pourraient être appliqués pour minimiser le risque de vol d'informations ou de sabotage par des infiltrés. Ces procédures incluent de publier les politiques de sécurité, le contrôle de l'accès aux niveaux physique, logiciel, et de réseau, et la séparation des services.

12.2.2 Protection contre la corruption des données dans le réseau

Lorsque le risque d'erreurs non détectées dans les datagrammes livrés par les service de transport de couche inférieure est considéré comme étant trop grand, une protection supplémentaire de l'intégrité est requise. Si cette protection supplémentaire devait être fournie dans la couche d'application, l'en-tête SCTP resterait vulnérable à des attaques délibérées contre l'intégrité. Bien que les mécanismes SCTP existants pour la détection des répétitions de paquet soient considérées être suffisantes en fonctionnement normal, des protections plus fortes sont nécessaires pour protéger SCTP quand l'environnement de fonctionnement contient des risques significatifs d'attaques délibérées de la part d'un adversaire sophistiqué.

L'extension d'authentification SCTP SCTP-AUTH [RFC4895] PEUT être utilisée quand l'environnement de menaces exige de plus fortes protections de l'intégrité mais n'exige pas la confidentialité.

12.2.3 Protection de la confidentialité

Dans la plupart des cas, le risque d'une violation de la confidentialité s'applique aux charge utiles de données de signalisation, et pas aux frais généraux de protocole de SCTP ou de couche inférieure. Si cela est vrai, le chiffrement des

données d'utilisateur SCTP peut être envisagé. Comme avec le service de somme de contrôle supplémentaire, le chiffrement des données d'utilisateur PEUT être effectué par l'application d'utilisateur SCTP. La [RFC6083] PEUT être utilisée pour cela. Autrement, l'application d'utilisateur PEUT utiliser une API spécificque de la mise en œuvre pour demander que l'encapsulation de charge utile de sécurité IP (ESP, *Encapsulating Security Payload*) [RFC4303] soit utilisée pour assurer la confidentialité et l'intégrité.

Particulièrement pour les utilisateurs mobiles, l'exigence de confidentialité pourrait inclure le masquage des adresses et accès IP. Dans ce cas, ESP DEVRAIT être utilisé au lieu de la confidentialité de niveau application. Si ESP est utilisé pour protéger la confidentialité du trafic SCTP, une transformation cryptographique ESP qui inclut une protection d'intégrité cryptographique DOIT être utilisée, parce que, si il y a une menace pour la confidentialité, il y aura aussi une forte menace sur l'intégrité.

Sans considération de si la protection de la confidentialité est fournie, le protocole d'échange de clé Internet version 2 (IKEv2) [RFC7296] DEVRAIT être utilisé pour la gestion de clés de ESP.

Les opérateurs peuvent consulter la [RFC4301] pour plus d'informations sur les services de sécurité disponibles à la couche de protocole Internet et immédiatement au-dessus.

12.2.4 Protection contre les attaques aveugles de déni de service

Une attaque aveugle est celle où l'attaquant n'est pas capable d'intercepter ou voir le contenu des flux de données passant de et vers le nœud SCTP cible. Les attaques de déni de service aveugles peuvent prendre la forme de l'inondation, du déguisement, ou d'une monopolisation inappropriée des services.

12.2.4.1 Inondation

L'objectif de l'inondation est de causer la perte de service et un comportement incorrect des systèmes cibles par l'épuisement des ressources, l'interférence avec les transactions légitimes, et l'exploitation de fautes de logiciel en relation avec la mémoire tampon. L'inondation peut être dirigée contre le nœud SCTP ou contre des ressources dans les liaisons d'accès IP intermédiaires ou l'Internet. Lorsque ces dernières entités sont la cible, l'inondation va se manifester par la perte de services du réseau, incluant une rupture potentielle de tous les pare-feu en place.

En général, la protection contre l'inondation commence au niveau de la conception de l'équipement, où elle inclut des mesures comme :

* éviter d'engager des ressources limitées avant de déterminer que la demande de service est légitime,
* donner la priorité à l'achèvement des traitements en cours sur l'acceptation de nouvelles tâches,
* identifier et supprimer les demandes de service dupliquées ou périmées en file d'attente,
* ne pas répondre à des paquets inattendus envoyés à des adresses qui ne sont pas d'envoi individuel.

L'équipement du réseau est supposé être capable de générer une alarme et un enregistrement si une augmentation suspecte du trafic se produit. L'enregistrement fournit des informations, comme l'identité de la liaison entrante et la ou les adresses de source utilisées, ce qui va aider l'opérateur du réseau ou du système SCTP à prendre des mesures protectrices. Des procédures sont supposées être en place pour que l'opérateur agisse sur de telles alarmes si un schéma clair d'abus se fait jour.

La conception de SCTP est résistante aux attaques d'inondation, en particulier par son utilisation d'une prise de contact en quatre étapes au démarrage, son utilisation d'un mouchard pour différer l'engagement des resources au nœud SCTP qui répond jusqu'à ce que la prise de contact soit achevée, et son utilisation d'une étiquette de vérification pour empêcher l'insertion de paquets étrangers dans le flux d'une association établie.

ESP pourrait être utile pour réduire le risque de certaines sortes d'attaques de déni de service.

La prise en charge du paramètre Adresse de nom d'hôte a été supprimée du protocole. Les points d'extrémité qui reçoivent des tronçons INIT ou INIT ACK contenant le paramètre Adresse de nom d'hôte DOIVENT envoyer un tronçon ABORT en réponse et PEUVENT inclure une cause d'erreur "Adresse non résolvable".

12.2.4.2 Déguisement aveugle

Le déguisement peut être utilisé pour dénier le service de plusieurs façons :

* en liant des ressources au nœud SCTP cible auxquelles le nœud déguisé a un accès limité. Par exemple, le nœud cible peut par sa politique permettre un maximum d'une association SCTP avec le nœud SCTP déguisé. L'attaquant déguisé peut tenter d'établir une association se donnant pour provenir du nœud prétendu afin que ce dernier ne puisse pas le faire quand il en a besoin.

* en permettant délibérément que le déguisement soit détecté, provoquant par là des contre-mesures qui causent le verrouillage du nœud travesti au nœud SCTP cible.

* en interférant avec une association établie en insérant un contenu étranger comme un tronçon SHUTDOWN.

SCTP réduit le risque d'attaque de déguisement aveugle par l'usurpation d'identité IP en utilisant la prise de contact de démarage en quatre phases. Parce que l'échange initial est sans mémoire, aucun mécanisme de verrouillage n'est déclenché par des attaques de déguisement aveugle. De plus, le paquet contenant le tronçon INIT ACK avec le mouchard d'état est retransmis à l'adresse IP d'où il a reçu le paquet contenant le tronçon INIT. Donc, l'attaquant ne pourra pas recevoir le tronçon INIT ACK contenant le mouchard d'état. SCTP protège contre l'insertion de paquets étrangers dans le flux d'une association établie par l'utilisation de l'étiquette de vérification.

L'enregistrement des tronçons INIT reçus et des anomalies, comme des tronçons INIT ACK inattendus, pourrait être considéré comme un moyen de détecter des schémas d'activité hostiles. Cependant, l'utilité potentielle d'un tel enregistrement doit être soupesée par rapport au poids de l'augmentation du traitement de démarrage de SCTP qu'il implique, rendant le nœud SCTP plus vulnérable aux attaques d'inondation. L'enregistrement est inutile sans l'établissement de procédures de fonctionnement pour revoir et analyser les enregistrements d'une façon régulière.

12.2.4.3 Monopolisation inappropriée des services

Les attaques à ce titre sont effectuées ouvertement et légitimement par l'attaquant. Elles sont dirigées contre des utilisateurs amis du nœud SCTP cible ou des ressources partagées entre l'attaquant et le nœud cible. Des attaques possibles incluent l'ouverture d'un grand nombre d'associations entre le nœud attaquant et la cible ou le transfert de gros volumes d'informations au sein d'une association légitimement établie.

Des limites de politique sont supposées être établies sur le nombre d'associations par nœud SCTP adjacent. Les applications d'utilisateur SCTP sont supposées être capables de détecter les gros volumes de messages illégitimes ou "non fonctionnels" au sein d'une certaine association et enregistrer ou terminer l'association en conséquence, sur la base de la politique locale.

12.3 Interactions de SCTP avec les pare-feu

Il est utile à certains pare-feu de pouvoir inspecter juste le premier fragment d'un paquet SCTP fragmenté et de déterminer sans ambiguïté si il correspond à un tronçon INIT (pour plus d'informations, se référer à la [RFC1858]). En conséquence, on souligne les exigences, mentionnées au paragraphe 3.1, que (1) un tronçon INIT NE DOIT PAS être groupé avec un autre tronçon dans un paquet et (2) un paquet contenant un tronçon INIT DOIT avoir une étiquette de vérification de zéro. Le receveur d'un tronçon INIT DOIT éliminer en silence le tronçon INIT et tous les tronçons suivants si le tronçon INIT est groupé avec d'autres tronçons ou si le paquet a une étiquette de vérification non zéro.

12.4 Protection des hôtes sans capacité SCTP

Pour fournir à un hôte sans capacité SCTP le même niveau de protection contre des attaques que celui de ceux qui ont la capacité SCTP, toutes les mises en œuvre de SCTP DOIVENT mettre en œuvre le traitement ICMP décrit à la Section 10.

Quand une mise en œuvre de SCTP reçoit un paquet contenant plusieurs tronçons DATA ou de contrôle et que le traitement du paquet résulterait en l'envoi de plusieurs tronçons en réponse, l'envoyeur du ou des tronçons de réponse NE DOIT PAS envoyer plus d'un paquet contenant des tronçons autres que des tronçons DATA. Cette exigence protège le réseau du déclenchement d'une salve de paquets en réponse à un seul paquet. Si le groupement est pris en charge, plusieurs tronçons de réponse qui tiennent dans un seul paquet PEUVENT être groupés dans un seul paquet de réponse. Si le groupement n'est pas pris en charge, alors l'envoyeur NE DOIT PAS envoyer plus d'un tronçon de réponse et DOIT éliminer toutes les autres réponses. Noter que cette règle ne s'applique pas à un tronçon SACK, car un tronçon SACK est, par lui-même, une réponse aux tronçons DATA, et un tronçon SACK n'exige pas de réponse de plus de tronçons DATA.

Une mise en œuvre de SCTP DOIT interrompre l'association si elle reçoit un tronçon SACK accusant réception d'un TSN qui n'a pas été envoyé.

Une mise en œuvre de SCTP qui reçoit un tronçon INIT qui exigerait un grand paquet en réponse, à cause de l'inclusion de plusieurs paramètres "Paramètre non reconnu"PEUT (à sa discrétion) choisir d'omettre certains ou tous les paramètres "Paramètre non reconnu" pour réduire la taille du tronçon INIT ACK. Du fait d'une combinaison de la taille du paramètre Mouchard d'état et du nombre d'adresses qu'un receveur d'un tronçon INIT indique à un homologue, il est toujours possible que le tronçon INIT ACK soit plus grand que le tronçon INIT d'origine. Une mise en œuvre de SCTP DEVRAIT tenter de rendre le tronçon INIT ACK aussi petit que possible pour réduire les possibilités d'attaques d'amplification d'octets.

13. Considérations sur la gestion du réseau

Le module de MIB pour SCTP défini dans la [RFC3873] s'applique pour la version du protocole spécifiée dans le présent document.

14. Paramètres recommandés de bloc de contrôle de transmission (TCB)

Cette Section précise un ensemble de paramètres qui sont supposés être contenus dans le TCB pour une mise en œuvre. Cette section est à des fins d'illustration et n'est pas considérée comme des exigences pour une mise en œuvre ou comme une liste exhaustive de tous les paramètres dans une TCB SCTP. Chaque mise en œuvre peut avoir besoin de ses propres paramètres supplémentaires pour une optimisation.

14.1 Paramètres nécessaires pour l'instance SCTP

Associations : une liste des associations courantes et des transpositions en les consommateurs de données pour chaque association. Ce pourrait être sous la forme d'un tableau de hachage ou autre structure dépendant de la mise en œuvre. Le consommateur des données pourrait être des informations d'identification de traitement, comme des descripteurs de fichiers, un pointeur de tuyau désigné, ou des pointeurs de tableau dépendant de la façon dont SCTP est mis en œuvre.

Clé secrète : une clé secrète utilisée par ce point d'extrémité pour calculer le MAC. Ce DEVRAIT être un nombre aléatoire de qualité cryptographique d'une longueur suffisante. L'exposé de la [RFC4086] peut être utile pour le choix d'une clé.

Liste d'adresses : la liste des adresses IP liées à cette instance. Cette information est passée à l'homologue dans les tronçons INIT et INIT ACK.

Accès SCTP : le numéro d'accès local SCTP auquel est lié le point d'extrémité.

14.2 Paramètres nécessaires par association (c'est-à-dire, le TCB)

Étiquette de vérification de l'homologue : la valeur de l'étiquette à envoyer dans chaque paquet et qui est reçue dans le tronçon INIT ou INIT ACK.

Mon étiquette de vérification : étiquette attendue dans chaque paquet entrant et envoyée dans le tronçon INIT ou INIT ACK.

État : COOKIE-WAIT, COOKIE-ECHOED, ESTABLISHED, SHUTDOWN-PENDING, SHUTDOWN-SENT, SHUTDOWN-RECEIVED, SHUTDOWN- ACK-SENT.

Note : aucun état "CLOSED" n'est illustré, car si une association est "CLOSED", son TCB DEVRAIT être supprimé.

Liste d'adresses de transport de l'homologue : liste des adresses de transport SCTP auxquelles l'homologue est lié. Cette information est déduite du tronçon INIT ou INIT ACK et est utilisée pour associer un paquet entrant à une certaine association. Normalement, cette information est hachée ou chiffrée pour une recherche et accès rapides du TCB.

Chemin principal : c'est l'adresse de transport de destination principale courante du point d'extrémité homologue. Il pourrait aussi spécifier une adresse de transport de source sur ce point d'extrémité.

Compte d'erreur global : compte d'erreur global de l'association.

Seuil d'erreur global : seuil pour cette association qui, si le compte d'erreur global l'atteint, va causer la suppression de cette association.

Fenêtre de réception d'homologue *(Peer Rwnd)* : valeur calculée actuellle de la fenêtre de réception de l'homologue.

Prochain TSN : numéro du prochain TSN à allouer à un nouveau tronçon DATA. Il est envoyé dans le tronçon INIT ou INIT ACK à l'homologue et incrémenté chaque fois qu'un TSN est alloué à un tronçon DATA (normalement, juste avant de le transmettre ou durant la fragmentation).

Dernier TSN reçu : c'est le dernier TSN reçu en séquence. Cette valeur est réglée initialement en prenant le TSN initial de l'homologue, reçu dans le tronçon INIT ou INIT ACK, et en y soustrayant un.

Matrice de transposition : matrice de bits ou d'octets indiquant quels TSN déclassés ont été reçus (par rapport au dernier TSN reçu). Si il n'existe pas de trou, c'est-à-dire, aucun paquet déclassé n'a été reçu, cette matrice va être remplie de zéros. Cette structure pourrait être sous la forme d'une mémoire tampon circulaire ou d'une matrice de bits.

État Ack : ce fanion indique si le prochain paquet reçu doit recevoir en réponse un tronçon SACK. C'est initialisé à 0. Quand un paquet est reçu, il est incrémenté. Si cette valeur atteint 2 ou plus, un tronçon SACK est envoyé et la valeur est remise à 0. Note : ceci n'est utilisé que quand aucun tronçon DATA n'est reçu déclassé. Quand des tronçons DATA sont déclassés, les tronçons SACK ne sont pas retardés (voir la Section 6).

Flux entrants : dispositifs de structures pour retracer les flux entrants, incluant normalement le prochain numéro de séquence attendu et éventuellement le numéro de flux.

Flux sortants : dispositifs de structures pour retracer les flux sortants, incluant normalement le prochain numéro de séquence à envoyer sur le flux.

Reasm Queue : file d'attente de réassemblage.

Mémoire tampon de réception : mémoire tampon pour mémoriser les données d'utilisateur reçues qui n'ont pas encore été livrées à la couche supérieure.

Liste des adresses de transport locales : liste des adresses IP locales liées à cette association.

Taille maximum de tronçon DATA de l'association : plus petite taille maximum de tronçon DATA du chemin de toutes les adresses de destination.

14.3 Données par adresse de transport

Pour chaque adresse de transport de destination dans le champ Liste d'adresses de l'homologue déduite du tronçon INIT ou INIT ACK, un certain nombre d'éléments de données doivent être conservés, incluant :

Compte d'erreurs : compte d'erreur en cours pour cette destination.

Seuil d'erreur : seuil d'erreur courant pour cette destination, c'est-à-dire, la valeur qui marque la destination comme morte si le compte d'erreurs atteint cette valeur.

cwnd : fenêtre d'encombrement courante.

ssthresh : valeur courante du seuil de démarrage lent.

RTO : valeur courante du temporisateur de retransmission.

SRTT : valeur courante du temps d'aller-retour lissé.

RTTVAR : variation courante du délai d'aller-retour.

octets partiels acquittés : méthode de traçage pour l'augmentation de la fenêtre d'encombrement en mode d'évitement d'encombrement (voir le paragraphe 7.2.2).

état : état courant de cette destination, c'est-à-dire, DOWN, UP, ALLOW-HEARTBEAT, NO-HEARTBEAT, etc.

PMTU : la PMTU connue courante.

PMDCS (*Path Maximum DATA Chunk Size*) : taille maximum courante connue de tronçon DATA du chemin.

Temporisateur par destination : temporisateur utilisé par chaque destination.

RTO en cours : fanion utilisé pour retracer si un des tronçons DATA envoyé à cette adresse est actuellement utilisé pour calculer un RTT. Si ce fanion est 0, le prochain tronçon DATA envoyé à cette destination est supposé être utilisé pour calculer un RTT et ce fanion est supposé être établi. Chaque fois que le calcul de RTT s'achève (c'est-à-dire, quand le tronçon DATA est acquitté) on supprime ce fanion.

dernière fois : l'heure à laquelle on a envoyé pour la dernière fois à cette destination. Cela peut être utilisé pour déterminer si l'envoi d'un tronçon HEARTBEAT est nécessaire.

14.4 Paramètres généraux nécessaires

File d'attente sortante *(Out Queue)* : file d'attente de tronçons DATA sortants.

File d'attente entrante *(In Queue)* : file d'attente de tronçons DATA entrants.

15. Considérations relatives à l'IANA

Le présent document définit cinq registres tenus par l'IANA :
* par définition de types de tronçon supplémentaires,
* par définition de fanions de tronçon supplémentaires,
* par définition de types de paramètres supplémentaires,
* par définition de codes de cause supplémentaires dans les tronçons ERROR, ou par définition d'identifiants de protocole de charge utile supplémentaires.

L'IANA a effectué les mises à jour suivantes sur les cinq registres ci-dessus :

* Dans le registre "Types de tronçons", l'IANA a remplacé la référence du registre aux [RFC4960] et [RFC6096] par une référence au présent document.

De plus, dans la Section Notes, la référence au paragraphe 3.2 de la [RFC6096] a été mise à jour par une référence au paragraphe 15.2 du présent document.

Finalement, chaque référence à la [RFC4960] a été remplacée par une référence au présent document pour les types de tronçon suivants :
- Données de charge utile (DATA)
- Initiation (INIT)
- Accusé de réception d'initiation (INIT ACK)
- Accusé de réception sélectif (SACK)
- Demande de battement de cœur (HEARTBEAT)
- Accusé de réception de battement de cœur (HEARTBEAT ACK)
- Interruption (ABORT)
- Fermeture (SHUTDOWN)
- Accusé de réception de fermeture (SHUTDOWN ACK)
- Erreur de fonctionnement (ERROR)
- Mouchard d'état (COOKIE ECHO)
- Accusé de réception de mouchard (COOKIE ACK)
- Réservé pour écho de notification explicite d'encombrement (ECNE)
- Réservé pour fenêtre d'encombrement réduite (CWR, *Congestion Window Reduced*)
- Fermeture achevée (SHUTDOWN COMPLETE)
- Réservé pour extensions de tronçon définies par l'IETF

* Dans le registre "Types de paramètres de tronçon", l'IANA a remplacé la référence du registre à la [RFC4960] par une référence au présent document.

L'IANA a changé le nom du type de paramètre de tronçon "Paramètres non reconnus" par "Paramètre non reconnu" dans le registre "Types de paramètres de tronçon".

De plus, chaque référence à la [RFC4960] a été remplacée par une référence au présent document pour les types de paramètre de tronçon suivants :
- Informations de battement de cœur
- Adresse IPv4
- Adresse IPv6
- Mouchard d'état
- Paramètre non reconnu
- Préservatioin de mouchard
- Adresse de nom d'hôte
- Types d'adresses pris en charge

L'IANA a ajouté une référence au présent document pour les types de paramètres de tronçon suivants :
- Réservé pour la capacité ECN (0x8000)

Aussi, l'IANA a ajouté la valeur 65535 comme étant réservée pour les extensions définies par l'IETF.

* Dans le registre "Fanions de tronçon", l'IANA a remplacé la référence du registre à la [RFC6096] par une référence au présent document.

De plus, chaque référence à la [RFC4960] a été remplacée par une référence au présent document pour les fanions de tronçon DATA suivants :
- bit E
  bit B
- bit U

L'IANA a aussi remplacé la référence à la [RFC7053] par une référence au présent document pour le fanion tronçon DATA suivant :
- bit I

L'IANA a remplacé la référence à la [RFC4960] par une référence au présent document pour le fanion tronçon ABORT suivant :
- bit T

L'IANA a remplacé la référence à la [RFC4960] par une référence au présent document pour le fanion tronçon SHUTDOWN COMPLETE suivant :
- bit T

* Dans le registre "Codes de cause d'erreur", l'IANA a remplacé la référence de registre à la [RFC4960] par une référence au présent document.

L'IANA a changé le nom de la cause d'erreur "Interruption initiée par l'utilisateur" en "Interruption initiée par l'utilisateur" et le nom de la cause d'erreur "Erreur de mouchard périmé" en "Mouchard périmé" dans le registre "Codes de cause d'erreur".

De plus, chaque référence à la [RFC4960] a été remplacée ar une référence au présent document pour les codes de cause suivants :
- Identifiant de flux invalide
- Paramètre obligatoire manquant
- Mouchard périmé
- Plus de ressources
- Adresse non résolvable
- Type de tronçon non reconnu
- Paramètre obligatoire invalide
- Paramètres non reconnus

- Pas de données d'utilisateur
- Mouchard reçu pendant la fermeture
- Redémarrage d'une association avec des adresses nouvelles

L'IANA a aussi remplacé chaque référence à la [RFC4460] par une référence au présent document pour les codes de cause suivants :
- Interruption initiée par l'utilisateur
- Violation de protocole

* Dans le registre "Identifiants de protocole de charge utile SCTP", l'IANA a remplacé la référence du registre à la [RFC4960] par une référence au présent document.

L'IANA a remplacé la référence à la [RFC4960] par une référence au présent document pour les identifiants de protocole de charge utile SCTP suivants :
- Réservé par SCTP

SCTP exigee que le registre IANA "Numéros d'accès" soit ouvert aux enregistrements d'accès SCTP ; le paragraphe 15.6 décrit comment. Une revue d'expert appointé par l'IESG soutient l'IANA pour évaluer les demandes d'allocations d'accès SCTP.

Dans le "registre des noms de service et des numéros d'accès de protocole de transport", l'IANA a remplacé chaque référence à la [RFC4960] par une référence au présent document pour les numéros d'accès SCTP suivants :
* 9 (éliminer)
* 20 (données ftp)
* 21 (ftp)
* 22 (ssh)
* 80 (http)
* 179 (bgp)
* 443 (https)

De plus, dans le registre "Valeurs d'algorithme de résumé du protocole de transfert Hypertexte (HTTP)", l'IANA a remplacé la référence à l'Appendice B de la [RFC4960] par une référence à l'Appendice A du présent document.

De plus, dans le registre "ONC RPC Netids (Standards Action)" l'IANA a remplacé chaque référence à la [RFC4960] par une référence au présent document pour les netids suivants :
* sctp
* sctp6

Dans le registre "Éléments d'information IPFIX" L'IANA a remplacé chaque référence à la [RFC4960] par une référence au présent document pour les éléments suivants par le nom :
* sourceTransportPort
* destinationTransportPort
* collectorTransportPort
* exporterTransportPort
* postNAPTSourceTransportPort
* postNAPTDestinationTransportPort

15.1 Extension de tronçon définie par l'IETF

L'allocation de nouveaux codes de type de tronçon est faite par une action de revue par l'IETF, comme défini dans la [RFC8126]. La documentation pour un nouveau tronçon DOIT contenir les informations suivantes :
a) Un nom long et un nom court pour le nouveau type de tronçon.
b) Une description détaillée de la structure du tronçon, qui DOIT se conformer à la structure de base définie au paragraphe 3.2.
c) Une définition et description détaillées de l'utilisation prévue de chaque champ dans le tronçon, incluant les fanions de tronçon si il en est. Les fanions de tronçon définis seront utilisés comme entrées initiales dans le tableau des fanions de tronçon pour le nouveau type de tronçon.
d) Une description détaillée des procédure d'utilisation du nouveau type de tronçon dans le fonctionnement du protocole.

Le dernier type de tronçon (255) est réservé pour de futures extensions si nécessaire.

Pour chaque nouveau type de tronçon, l'IANA crée un tableau d'enregistrement pour les fanions de tronçon de ce type. La procédure pour enregistrer des fanions de tronçon particuliers est décrite au paragraphe 15.2.

15.2 Enregistrement de fanions de tronçon définis par l'IETF

L'allocation de nouveaux fanions de tronçons est faite par une action de RFC exigée, comme défini dans la [RFC8126]. La documentation des fanions de tronçon DOIT contenir les informations suivantes :

a) Un nom pour le nouveau fanion de tronçon.
b) Une description procédurale détaillée de l'utilisation du nouveau fanion de tronçon dans le fonctionnement du protocole. Il DOIT être considéré que les mises en œuvre qui ne prennent pas en charge le fanion vont envoyer 0 à l'émission et juste l'ignorer à reception.

L'IANA choisit la valeur des fanions de tronçon. Cela DOIT être une des valeurs de 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, ou 0x80, qui DOIVENT être uniques parmi les valeurs de fanion de tronçon pour le type de tronçon spécifique.

15.3 Extension de paramètre de tronçon défini par l'IETF

L'allocation de nouveaux codes de types de paramètre de tronçon est faite par une action de revue de l'IETF, comme définie dans la [RFC8126]. La documentation du paramètre de tronçon DOIT contenir les informations suivantes :

a) Nom du type de paramètre.
b) Description détaillée de la structure du champ du paramètre. Cette structure DOIT se conformer au format général de Type-Longueur-Valeur décrit au paragraphe 3.2.1.
c) Description détaillée de chaque composant de la valeur du paramètre.
d) Description détaillée de l'utilisation prévue de ce ype de paramètre et d'une indication de si et dans quelles circonstances plusieurs instances de ce type de paramètre peuvent être trouvées dans le même tronçon.
e) Chaque type de paramètre DOIT être unique parmi tous les tronçons.

15.4 Causes d'erreur supplémentaires définies par l'IETF

Des codes de cause supplémentaires peuvent être alloués par une action de spécification exigée comme défini dans la [RFC8126]. La documentation fournie DOIT inclure les informations suivantes :

a) Le nom de la condition d'erreur.
b) La description détaillée des conditions dans lesquelles un point d'extrémité SCTP produit un tronçon ERROR (ou ABORT) avec ce code de cause.
c) L'action attendue par le point d'extrémité SCTP qui reçoit un tronçon ERROR (ou ABORT) contenant ce code de cause.
d) La description détaillée de la structure et du contenu des champs de données qui accompagnent ce code de cause.

Le mot initial (32 bits) d'un paramètre de code de cause DOIT se conformer au format donné au paragraphe 3.3.10, qui est :

* les deux premiers octets contiennent la valeur du code de cause
* les deux derniers octets contiennent la longueur de la cause d'erreur.

15.5 Identifiants de protocoles de charge utile

L'allocation d'identifiants de protocole de charge utile est faite en utilisant la politique de premier arrivé, premier servi définie dans la [RFC8126].

Sauf pour la valeur 0, qui est réservée pour indiquer un identifiant de protocole de charge utile non spécifié dans un tronçon DATA, une mise en œuvre de SCTP ne va pas être responsable de la normalisation ou de la vérification des identifiants de protocole de charge utile. Une mise en œuvre de SCTP reçoit simplement l'identifiant provenant de la couche supérieure et le porte avec les données de charge utile correspondantes.

La couche supérieure, c'est-à-dire, l'utilisateur SCTP, DEVRAIT normaliser auprès de l'IANA tout identifiant de protocole spécifique si cela est désiré. L'utilisation de tout identifiant de protocole de charge utile spécifique sort du domaine d'application de la présente spécification.

15.6 Registre des numéros d'accès

Les services SCTP peuvent utiliser des numéros d'accès de contact pour fournir le service à des appelants inconnus, comme dans TCP et UDP. Un expert réviseur appointé par l'IESG soutient l'IANA dans l'évaluation des demandes d'allocation d'accès SCTP, en accord avec la procédure définie dans la [RFC8126]. Les détails de ce processus sont définis dans la [RFC6335].

16. Valeurs suggérées des paramètres de protocole SCTP

Les paramètres de protocole suivants sont RECOMMANDÉS :
RTO.Initial : 1 seconde
RTO.Min : 1 seconde
RTO.Max : 60 secondes
Max.Burst : 4
RTO.Alpha : 1/8
RTO.Beta : 1/4
Valid.Cookie.Life : 60 second s
Association.Max.Retrans : 10 tentatives
Path.Max.Retrans : 5 tentatives (par adresse de destination)
Max.Init.Retransmits : 8 tentatives
HB.interval : 30 secondes
HB.Max.Burst : 1
SACK.Delay : 200 millisecondse

Note de mise en œuvre : la mise en œuvre de SCTP peut permettre à l'ULP de personnaliser certains de ces paramètres de protocole (voir la Section 11).

'RTO.Min' DEVRAIT être réglé comme décrit ci-dessus.

17. Références

17.1 Références normatives

[V.42] Recommandation UIT-T V.42, "Procédures de correction d'erreur pour les équipements terminaux de circuit de données qui utilisent la conversion asynchrone à synchrone". 1994.

[RFC1122] R. Braden, "Exigences pour les hôtes Internet – couches de communication", STD 3, octobre 1989. (DOI : 10.17487/RFC1122) *(MàJ par la* RFC6633).

[RFC1123] R. Braden, éditeur, "Exigences pour les hôtes Internet – Application et prise en charge", STD 3, octobre 1989. (DOI : 10.17487/RFC1123).

[RFC1191] J. Mogul et S. Deering, "Découverte de la MTU de chemin", novembre 1990. (DOI : 10.17487/RFC1191).

[RFC1982] R. Elz, R. Bush, "Arithmétique des numéros de série", août 1996. (DOI : 10.17487/RFC1982) (*MàJ* RFC1034, RFC1035) *(P.S.)*

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. (DOI : 10.17487/RFC2119).

[RFC4291] R. Hinden, S. Deering, "Architecture d'adressage IP version 6", février 2006. (DOI : 10.17487/RFC4291) (*MàJ par* 5952 et 6052 ; *D.S.)*

[RFC4303] S. Kent, "Encapsulation de charge utile de sécurité dans IP (ESP)", décembre 2005. (DOI : 10.17487/RFC4303) (*Remplace* RFC2406) *(P.S.)*

[RFC4895] M. Tuexen et autres, "Tronçons authentifiés dans le protocole de transmission de contrôle de flux (SCTP)", août 2007. (DOI : 10.17487/RFC4895) *(P.S.)*

[RFC5681] M. Allman, V. Paxson, E. Blanton, "Contrôle d'encombrement de TCP", septembre 2009. (DOI : 10.17487/RFC5681) *(Remplace* RFC2581) *(D.S.)*

[RFC6083] M. Tüexen, R. Seggelmann et E. Rescorla, "Sécurité de la couche Transport de datagrammes (DTLS) pour le protocle de transmission de contrôle de flux (SCTP)", janvier 2011. (DOI : 10.17487/RFC6083) *(P.S.)*

[RFC6335] M. Cotton et autres, "Procédures de l'autorité d'allocation des numéros de l'Internet (IANA) pour la gestion du registre des numéros d'accès aux noms de service et protocoles de transport", août 2011. (DOI : 10.17487/RFC6335) *(MàJ les* RFC2780, RFC2782, RFC3828, RFC4340, RFC4960, RFC5595) (BCP0165)

[RFC7296] C. Kaufman, et autres, "Protocole d'échange de clé Internet version 2 (IKEv2)", octobre 2014. STD 79. (DOI : 10.17487/RFC7296) *(MàJ par* RFC7670, RFC8247)

[RFC8126] M. Cotton, B. Leiba, T. Narten, "Lignes directrices pour la rédaction d'une section de considérations relatives à l'IANA dans les RFC", juin 2017. BCP 26. (DOI : 10.17487/RFC8126) *(Remplace* RFC5226)

[RFC8174] B. Leiba, "Ambiguïté des mots clés en majuscules ou minuscules dans la RFC2119", mai 2017. BCP14. (DOI : 10.17487/RFC8174) *(MàJ* 2119)

[RFC8200] S. Deering, R. Hinden, "Spécification du protocole Internet version 6 (IPv6)", juillet 2017. STD 86. (DOI : 10.17487/RFC8200) *(Remplace* 2460)

[RFC8201] J. McCann, et autres, "Découverte de la MTU de chemin pour IPv6", juillet 2017. STD 87. (DOI : 10.17487/RFC8201) *(Remplace* RFC1981).

[RFC8899] G. Fairhurst, T. Jones, M. Tüxen, I. Rüngeler, T. Völker, "Découverte de la MTU de couche de mise en paquet pour le transport des datagrammes", septembre 2020. *(MàJ* RFC4821, RFC4960, RFC6951, RFC8085, RFC8261 *; P.S.*) (DOI : 10.17487/RFC8899).

17.2 Références pour information

[FALL96] Fall, K. and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, et SACK TCP", SIGCOMM'99 V. 26 N. 3 pp 5-21, juillet 1996.

[SAVAGE99] Savage, S., Cardwell, N., Wetherall, D., and T. Anderson, "TCP Congestion Control with a Misbehaving Receiver", ACM Computer Communications Review 29(5), octobre 1999.

[ALLMAN99] Allman, M. and Paxson, "On Estimating End-to-End Network Path Properties", SIGCOMM'99 , 1999.

[WILLIAMS93] Williams, R., "A painless guide to CRC error detection algorithms", Internet publication, http://www.geocities.com/SiliconValley/Pines/8659/crc.htm , août 1993.

[RFC0768] J. Postel, "Protocole de datagramme d'utilisateur (UDP)", (STD 6), 28 août 1980.(DOI : 10.17487/RFC0768).

[RFC0793] J. Postel (éd.), "Protocole de commande de transmission – Spécification du protocole du programme Internet DARPA", STD 7, septembre 1981.(DOI : 10.17487/RFC0793).

[RFC1858] G. Ziemba, D. Reed, P. Traina, "Considérations sur la sécurité pour le filtrage de fragment IP", octobre 1995. (DOI : 10.17487/RFC1858) *(Mise à jour par la* RFC3128*) (Information)*

[RFC2104] H. Krawczyk, M. Bellare et R. Canetti, "HMAC : Hachage de clés pour l'authentification de message", février 1997.(DOI : 10.17487/RFC2104).

[RFC2196] B. Fraser, "Manuel de la sécurité des sites", septembre 1997. (DOI : 10.17487/RFC2196) (FYI0008) *(Info.)*

[RFC2522] P. Karn, W. Simpson, "Photuris : Protocole de gestion de clé de session", mars 1999. (DOI : 10.17487/RFC2522) *(Expérimentale)*

[RFC2960] R. Stewart et autres, "Protocole de transmission de commandes de flux", octobre 2000. (*Obsolète, voir* RFC4960) *(P.S.)*

[RFC3465] M. Allman, "Contrôle d'encombrement sur TCP avec compte d'octets approprié (ABC)", février 2003. (DOI : 10.17487/RFC3465) *(Expérimentale)*

[RFC3873] J. Pastor, M. Belinchon, "Base de données d'informations de gestion pour le protocole de transmission de commandes de flux (SCTP)", septembre 2004. (DOI : 10.17487/RFC3873) *(P.S.)*

[RFC4086] D. Eastlake 3rd, J. Schiller, S. Crocker, "Exigences d'aléa pour la sécurité", juin 2005. (DOI : 10.17487/RFC4086) (*Remplace* RFC1750) (BCP0106)

[RFC4301] S. Kent et K. Seo, "Architecture de sécurité pour le protocole Internet", décembre 2005. (DOI : 10.17487/RFC4301) *(P.S. ;Remplace* RFC2401).

[RFC4460] R. Stewart et autres, "Errata et problèmes de la spécification du protocole de transmission de contrôle de flux (SCTP)", avril 2006. (DOI : 10.17487/RFC4460) *(Information)*

[RFC4960] R. Stewart, éd., "Protocole de transmission de commandes de flux (SCTP)", septembre 2007. (DOI : 10.17487/RFC4960) (*Remplace* RFC2960, RFC3309 ; *P.S. ; Remplacée par* RFC9260)

[RFC6096] M. Tuexen, R. Stewart, "Enregistrement des fanions de tronçons dans le protocole de transmission de contrôle de flux (SCTP)", janvier 2011. (*MàJ* RFC4960) *(P. S.)*

[RFC6458] R. Stewart et autres, "Extensions d'API de prises pour le protocole de transmission de contrôle de flux (SCTP)", décembre 2011. (DOI : 10.17487/RFC6458) *(Information)*

[RFC6951] M. Tüexen, R. Stewart, "Encapsulation UDP de paquets du protocole de transmission de contrôle de flux (SCTP) pour les communications entre hôtes d'extrémité", mai 2013 (DOI : 10.17487/RFC6951) *(P.S. ; MàJ par RFC8899)*

[RFC7053] M. Tüexen, I. Ruengeler, R. Stewart, "Extension SACK-IMMEDIATELY pour le protocole de transmission de contrôle de flux", novembre 2013. (DOI : 10.17487/RFC7053) (*MàJ* RFC4960) *(P.S.)*

[RFC8260] R. Stewart, et autres, "Entrelaçage de programmeurs de flux et de message d'utilisateur pour le protocole de transmission de commande de flux (SCTP)", novembre 2017. (DOI : 10.17487/RFC8260) *(P.S.)*

[RFC8261] M. Tüexen, et autres, "Encapsulation de paquets SCTP dans DTLS", novembre 2017. (DOI : 10.17487/RFC8261) *(P.S. ; MàJ par* RFC8899*)*

[RFC8540] R. Stewart, M. Tüexen, M. Proshin, "Protocole de transmission de contrôle de flux : erreurs et problèmes de la RFC 4960", février 2019. (DOI : 10.17487/RFC8540) *(Information)*

Appendice A. Calcul de la somme de contrôle CRC32c

On définit une "valeur reflétee"'comme celle qui est à l'opposé de l'ordre normal des bits de la machine. Le contrôle de redondance cyclique (CRC, *Cyclic Redundancy Check*) de 32 bits est calculé, comme décrit pour le CRC32c et utilise le code polynomial 0x11EDC6F41 (Castagnoli93) ou x^32+x^28+x^27+x^26+x^25+x^23+x^22 +x^20+x^19+x^18+x^14+x^13+x^11+x^10+x^9+x^8+x^6+x^0. Le CRC est calculé en utilisant une procédure similaire à celle de ETHERNET CRC [V.42], modifiée pour refléter l'usage de niveau transport.

Le calcul de CRC utilise la division polynomiale. Un message de chaîne binaire M est transformé en un polynome, M(X), et le CRC est calculé à partir de M(X) en utilisant l'arithmétique polynomiale.

Quand les CRC sont utilisés à la couche de liaison, le polynome est déduit de l'ordre des bits du réseau : le premier bit "du réseau" est le coefficient de plus fort poids. Comme SCTP est un protocole de niveau transport, il ne peut pas connaître l'ordre des bits réel du support de série. De plus, des liaisons différentes dans le chemin entre les points d'extrémité SCTP peuvent utiliser des ordres de bits différents au niveau de la liaison.

Une convention est donc établie pour transposer les messages de transport SCTP en polynomes pour les besoins du calcul du CRC. L'ordre des bits pour la transposition des messages SCTP en polynomes est que les octets sont pris avec ceux de poids fort en premier, mais dans chaque octet, les bits sont pris avec celui de moindre poids en premier. Le premier octet du message donne les huit coefficients les plus forts. Au sein de chaque octet, le bit SCTP de moindre poids donne le coefficient de polynome de plus fort poids au sein de cet octet, et le bit SCTP de moindre poids est le coefficient de polynome de moindre poids dans cet octet. (Cet ordre des bits est parfois appelé "miroré" ou "réfléchi" [WILLIAMS93].) Les polynomes de CRC sont à retransformer en valeurs d'octets de niveau transport SCTP, en utilisant une transposition cohérente.

La valeur de CRC de niveau transport SCTP peut être calculée comme suit :

* Les données d'entrée du CRC sont allouées à un flux d'octets, numérotés de 0 à N-1.

* Le flux d'octets de niveau transport est transposé en une valeur de polynome. Une PDU de N octets avec j octets numérotés de 0 à N-1 est considérée comme les coefficients d'un polynome M(x) d'ordre 8*N-1, avec le bit 0 de l'octet j étant le coefficient x^(8*(N-j)-8) et le bit 7 de l'octet j étant le coefficient x^(8*(N-j)-1).

* Le registre restant du CRC est initialisé avec tous les bits à un et le CRC est calculé avec un algorithme qui multiplie simultanément par x^32 et divise par le polynome du CRC.

* Le polynome est multiplié par x^32 et divisé par G(x), le générateur polynomial, produisant un reste R(x) de degré inférieur ou égal à 31.

* Les coefficients de R(x) sont considérés comme une séquence de 32 bits.

* La séquence de bits est complétée. Le résultat est le polynome du CRC.

* Le polynome du CRC est retransposé en octets de niveau transport SCTP. Le coefficient de x^31 donne la valeur du bit 7 de l'octet SCTP 0, et le coefficient de x^24 donne la valeur du bit 0 de l'octet 0. Le coefficient de x^7 donne le bit 7 de l'octet 3, et le coefficient de x^0 donne le bit 0 de l'octet 3. La séquence résultante de 4 octets de niveau transport est la valeur de 32 bits de la somme de contrôle SCTP.

Note de mise en œuvre : les documents de normes, manuels, et la littérature des fabricants sur les CRC suivent souvent une formulation différente, dans laquelle le registre utilisé pour contenir le reste de l'algorithme de longue division est initialisé à zéro plutôt qu'avec des uns, et à la place, les 32 premiers bits du message sont complétés. L'algorithme de longue division utilisé dans notre formulation est spécifié de telle façon que la multiplication initiale par 2^32 et la longue division sont combinées en une opération simultanée. Pour ces algorithmes, et pour les messages de plus de 64 bits, les deux spécifications sont exactement équivalentes. Cette équivalence est voulue par le présent document.

Les développeurs de SCTP sont informés que les deux spécifications se trouvent dans la littérature, parfois sans restriction sur l'algorithme de division longue. Le choix de formulation de ce document est de permettre un usage non SCTP, où le même algorithme de CRC peut être utilisé pour protéger les messages inférieurs à 64 bits.

Il peut y avoir un avantage pour le calcul à valider l'association à l'égard de l'étiquette de vérification, avant d'effectuer une somme de contrôle, car des étiquettes invalides vont résulter en la même action qu'une mauvaise somme de contrôle dans la plupart des cas. Les exceptions à cette technique vont être les paquets contenant des tronçons INIT et certains tronçons SHUTDOWN-COMPLETE, ainsi que des tronçons COOKIE ECHO périmés. Ces cas particuliers d'échanges représentent de petits paquets et vont minimiser l'effet du calcul de somme de contrôle.

L'échantillon non normatif de code suivant est tiré d'un générateur de code d'accès libre [WILLIAMS93], utilisant la technique du "miroir" et donnant un tableau de recherche pour le CRC32c SCTP avec 256 entrées, chacune de 32 bits. Bien que ni spécialement lent ni particulièrement rapide, comme le sont les logiciels de CRC à tableau de recherche, il a l'avantage de fonctionnenner sur les CPU grosses boutiennes et petites boutiennes, en utilisant les mêmes tableaux de recherche (dans l'ordre de l'hôte) et en utilisant seulement les opérations prédéfinies ntohl() et htonl(). Le code est un peu modifié par rapport à [WILLIAMS93] pour assurer la portabilité entre architectures grosses boutiennes et petites boutiennes, il utilise des types de taille fixe pour permettre la portabilité entre plates-formes de 32 bits et de 64 bits, et utilise les améliorations générales de code C. (Noter que si l'ordre des octets de l'architecture cible est connu pour être petit boutien, les étapes finales d'inversion de bit et l'octet peuvent être réunies en une seule opération.)

<DÉBUT DU CODE>

```
/**********************************************************************/
```

```
/* Note : Les définitions pour le générateur de tableau de Ross Williams seraient TB_WIDTH=4, TB_POLY=0x1EDC6F41, TB_REVER=TRUE. Pourle code de calcul direct de M. Williams, on utilise les réglages cm_width=32, cm_poly=0x1EDC6F41, cm_init=0xFFFFFFFF, cm_refin=TRUE, cm_refot=TRUE, cm_xorot=0x00000000.  */
/**************************************************************/

/* Exemple du fichier de tableau de crc */
#ifndef __crc32cr_h__
#define __crc32cr_h__

#define CRC32C_POLY 0x1EDC6F41UL
#define CRC32C(c,d) (c=(c>>8)^crc_c[(c^(d))&0xFF])

  uint32_t crc_c[256] = {
    0x00000000UL, 0xF26B8303UL, 0xE13B70F7UL, 0x1350F3F4UL,
    0xC79A971FUL, 0x35F1141CUL, 0x26A1E7E8UL, 0xD4CA64EBUL,
    0x8AD958CFUL, 0x78B2DBCCUL, 0x6BE22838UL, 0x9989AB3BUL,
    0x4D43CFD0UL, 0xBF284CD3UL, 0xAC78BF27UL, 0x5E133C24UL,
    0x105EC76FUL, 0xE235446CUL, 0xF165B798UL, 0x030E349BUL,
    0xD7C45070UL, 0x25AFD373UL, 0x36FF2087UL, 0xC494A384UL,
    0x9A879FA0UL, 0x68EC1CA3UL, 0x7BBCEF57UL, 0x89D76C54UL,
    0x5D1D08BFUL, 0xAF768BBCUL, 0xBC267848UL, 0x4E4DFB4BUL,
    0x20BD8EDEUL, 0xD2D60DDDUL, 0xC186FE29UL, 0x33ED7D2AUL,
    0xE72719C1UL, 0x154C9AC2UL, 0x061C6936UL, 0xF477EA35UL,
    0xAA64D611UL, 0x580F5512UL, 0x4B5FA6E6UL, 0xB93425E5UL,
    0x6DFE410EUL, 0x9F95C20DUL, 0x8CC531F9UL, 0x7EAEB2FAUL,
    0x30E349B1UL, 0xC288CAB2UL, 0xD1D83946UL, 0x23B3BA45UL,
    0xF779DEAEUL, 0x05125DADUL, 0x1642AE59UL, 0xE4292D5AUL,
    0xBA3A117EUL, 0x4851927DUL, 0x5B016189UL, 0xA96AE28AUL,
    0x7DA08661UL, 0x8FCB0562UL, 0x9C9BF696UL, 0x6EF07595UL,
    0x417B1DBCUL, 0xB3109EBFUL, 0xA0406D4BUL, 0x522BEE48UL,
    0x86E18AA3UL, 0x748A09A0UL, 0x67DAFA54UL, 0x95B17957UL,
    0xCBA24573UL, 0x39C9C670UL, 0x2A993584UL, 0xD8F2B687UL,
    0x0C38D26CUL, 0xFE53516FUL, 0xED03A29BUL, 0x1F682198UL,
    0x5125DAD3UL, 0xA34E59D0UL, 0xB01EAA24UL, 0x42752927UL,
    0x96BF4DCCUL, 0x64D4CECFUL, 0x77843D3BUL, 0x85EFBE38UL,
    0xDBFC821CUL, 0x2997011FUL, 0x3AC7F2EBUL, 0xC8AC71E8UL,
    0x1C661503UL, 0xEE0D9600UL, 0xFD5D65F4UL, 0x0F36E6F7UL,
    0x61C69362UL, 0x93AD1061UL, 0x80FDE395UL, 0x72966096UL,
    0xA65C047DUL, 0x5437877EUL, 0x4767748AUL, 0xB50CF789UL,
    0xEB1FCBADUL, 0x197448AEUL, 0x0A24BB5AUL, 0xF84F3859UL,
    0x2C855CB2UL, 0xDEEEDFB1UL, 0xCDBE2C45UL, 0x3FD5AF46UL,
    0x7198540DUL, 0x83F3D70EUL, 0x90A324FAUL, 0x62C8A7F9UL,
    0xB602C312UL, 0x44694011UL, 0x5739B3E5UL, 0xA55230E6UL,
    0xFB410CC2UL, 0x092A8FC1UL, 0x1A7A7C35UL, 0xE811FF36UL,
    0x3CDB9BDDUL, 0xCEB018DEUL, 0xDDE0EB2AUL, 0x2F8B6829UL,
    0x82F63B78UL, 0x709DB87BUL, 0x63CD4B8FUL, 0x91A6C88CUL,
    0x456CAC67UL, 0xB7072F64UL, 0xA457DC90UL, 0x563C5F93UL,
    0x082F63B7UL, 0xFA44E0B4UL, 0xE9141340UL, 0x1B7F9043UL,
    0xCFB5F4A8UL, 0x3DDE77ABUL, 0x2E8E845FUL, 0xDCE5075CUL,
    0x92A8FC17UL, 0x60C37F14UL, 0x73938CE0UL, 0x81F80FE3UL,
    0x55326B08UL, 0xA759E80BUL, 0xB4091BFFUL, 0x466298FCUL,
    0x1871A4D8UL, 0xEA1A27DBUL, 0xF94AD42FUL, 0x0B21572CUL,
    0xDFEB33C7UL, 0x2D80B0C4UL, 0x3ED04330UL, 0xCCBBC033UL,
    0xA24BB5A6UL, 0x502036A5UL, 0x4370C551UL, 0xB11B4652UL,
    0x65D122B9UL, 0x97BAA1BAUL, 0x84EA524EUL, 0x7681D14DUL,
    0x2892ED69UL, 0xDAF96E6AUL, 0xC9A99D9EUL, 0x3BC21E9DUL,
    0xEF087A76UL, 0x1D63F975UL, 0x0E330A81UL, 0xFC588982UL,
    0xB21572C9UL, 0x407EF1CAUL, 0x532E023EUL, 0xA145813DUL,
    0x758FE5D6UL, 0x87E466D5UL, 0x94B49521UL, 0x66DF1622UL,
    0x38CC2A06UL, 0xCAA7A905UL, 0xD9F75AF1UL, 0x2B9CD9F2UL,
```

```
  0xFF56BD19UL, 0x0D3D3E1AUL, 0x1E6DCDEEUL, 0xEC064EEDUL,
  0xC38D26C4UL, 0x31E6A5C7UL, 0x22B65633UL, 0xD0DDD530UL,
  0x0417B1DBUL, 0xF67C32D8UL, 0xE52CC12CUL, 0x1747422FUL,
  0x49547E0BUL, 0xBB3FFD08UL, 0xA86F0EFCUL, 0x5A048DFFUL,
  0x8ECEE914UL, 0x7CA56A17UL, 0x6FF599E3UL, 0x9D9E1AE0UL,
  0xD3D3E1ABUL, 0x21B862A8UL, 0x32E8915CUL, 0xC083125FUL,
  0x144976B4UL, 0xE622F5B7UL, 0xF5720643UL, 0x07198540UL,
  0x590AB964UL, 0xAB613A67UL, 0xB831C993UL, 0x4A5A4A90UL,
  0x9E902E7BUL, 0x6CFBAD78UL, 0x7FAB5E8CUL, 0x8DC0DD8FUL,
  0xE330A81AUL, 0x115B2B19UL, 0x020BD8EDUL, 0xF0605BEEUL,
  0x24AA3F05UL, 0xD6C1BC06UL, 0xC5914FF2UL, 0x37FACCF1UL,
  0x69E9F0D5UL, 0x9B8273D6UL, 0x88D28022UL, 0x7AB90321UL,
  0xAE7367CAUL, 0x5C18E4C9UL, 0x4F48173DUL, 0xBD23943EUL,
  0xF36E6F75UL, 0x0105EC76UL, 0x12551F82UL, 0xE03E9C81UL,
  0x34F4F86AUL, 0xC69F7B69UL, 0xD5CF889DUL, 0x27A40B9EUL,
  0x79B737BAUL, 0x8BDCB4B9UL, 0x988C474DUL, 0x6AE7C44EUL,
  0xBE2DA0A5UL, 0x4C4623A6UL, 0x5F16D052UL, 0xAD7D5351UL,
};

#endif

/* Exemple de sous programme de construction de tableau */

  #include <stdio.h>
  #include <stdlib.h>

  #define OUTPUT_FILE   "crc32cr.h"
  #define CRC32C_POLY    0x1EDC6F41UL

  static FILE *tf;

  static uint32_t
  reflect_32(uint32_t b)
  {
   int i;
   uint32_t rw = 0UL;

   for (i = 0; i < 32; i++) {
    if (b & 1)
     rw |= 1UL << (31 - i);
    b >>= 1;
   }
   return (rw);
  }

  static uint32_t
  build_crc_table (int index)
  {
   int i;
   uint32_t rb;

   rb = reflect_32(index);

   for (i = 0; i < 8; i++) {
    if (rb & 0x80000000UL)
     rb = (rb << 1) ^ (uint32_t)CRC32C_POLY;
    else
     rb <<= 1;
   }
   return (reflect_32(rb));
  }
```

```
  int
  main (void)
  {
   int i;

   printf("\nGenerating CRC32c table file <%s>.\n",
   OUTPUT_FILE);
   if ((tf = fopen(OUTPUT_FILE, "w")) == NULL) {
    printf("Unable to open %s.\n", OUTPUT_FILE);
    exit (1);
   }
   fprintf(tf, "#ifndef __crc32cr_h__\n");
   fprintf(tf, "#define __crc32cr_h__\n\n");
   fprintf(tf, "#define CRC32C_POLY 0x%08XUL\n",
    (uint32_t)CRC32C_POLY);
   fprintf(tf,
    "#define CRC32C(c,d) (c=(c>>8)^crc_c[(c^(d))&0xFF])\n");
   fprintf(tf, "\nuint32_t crc_c[256] =\n{\n");
   for (i = 0; i < 256; i++) {
    fprintf(tf, "0x%08XUL,", build_crc_table (i));
    if ((i & 3) == 3)
     fprintf(tf, "\n");
    else
     fprintf(tf, " ");
   }
   fprintf(tf, "};\n\n#endif\n");

   if (fclose(tf) != 0)
    printf("Unable to close <%s>.\n", OUTPUT_FILE);
   else
    printf("\nThe CRC32c table has been written to <%s>.\n",
     OUTPUT_FILE);
   return (0);
  }

/* Exemple d'insertion de crc */

  #include "crc32cr.h"

  uint32_t
  generate_crc32c(unsigned char *buffer, unsigned int length)
  {
   unsigned int i;
   uint32_t crc32 = 0xffffffffUL;
   uint32_t result;
   uint32_t byte0, byte1, byte2, byte3;

   for (i = 0; i < length; i++) {
    CRC32C(crc32, buffer[i]);
   }

   result = ~crc32;
```

/* Le résultat contient maintenant le reste inverse du polynome, car le tableau et l'algorithme sont "réfléchis" [williams95]. C'est-à-dire que le résultat a la même valeur que si on avait transposé le message en un polynome, calculé le reste du polynome dans l'ordre des octets de l'hôte, effectué l'inversion finale, et ensuite fait une inversion de bits de bout en bout. Noter qu'une inversion de bits sur 32 bits est identique à quatre inversions de bits sur 8 bits en place suivies par un échange d'octets de bout en bout. En d'autres termes, les bits de chaque octet sont dans le bon ordre, mais les octets ont été échangés. Donc, on fait maintenant un échange d'octets explicite. Sur une machine petite boutienne, cet échange d'octets et le ntohl final s'annulent et peuvent être élidés.

```
*/

   byte0 = result & 0xff;
   byte1 = (result>>8) & 0xff;
   byte2 = (result>>16) & 0xff;
   byte3 = (result>>24) & 0xff;
   crc32 = ((byte0 << 24) |
           (byte1 << 16) |
           (byte2 << 8)  |
           byte3);
   return (crc32);
  }

  int
  insert_crc32(unsigned char *buffer, unsigned int length)
  {
   SCTP_message *message;
   uint32_t crc32;

   message = (SCTP_message *)buffer;
   message->common_header.checksum = 0UL;
   crc32 = generate_crc32c(buffer,length);
   /* et l'insérer dans le message */
   message->common_header.checksum = htonl(crc32);
   return (1);
  }

  int
  validate_crc32(unsigned char *buffer, unsigned int length)
  {
   SCTP_message *message;
   unsigned int i;
   uint32_t original_crc32;
   uint32_t crc32;

   /* sauvegarde et somme de contrôle zéro */
   message = (SCTP_message *)buffer;
   original_crc32 = ntohl(message->common_header.checksum);
   message->common_header.checksum = 0L;
   crc32 = generate_crc32c(buffer, length);
   return ((original_crc32 == crc32) ? 1 : -1);
  }
  <FIN DE CODE>
```

Remerciements

Une entreprise comme celle représentée par ce document mis à jour n'est pas une petite affaire et représente l'apport des co-auteurs initiaux de la [RFC2960], Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, et V. Paxson. Ajouter à cela, les commentaires de tous ceux qui ont contribué à la [RFC2960], Mark Allman, R. J. Atkinson, Richard Band, Scott Bradner, Steve Bellovin, Peter Butler, Ram Dantu, R. Ezhirpavai, Mike Fisk, Sally Floyd, Atsushi Fukumoto, Matt Holdrege, Henry Houh, Christian Huitema, Gary Lehecka, Jonathan Lee, David Lehmann, John Loughney, Daniel Luan, Barry Nagelberg, Thomas Narten, Erik Nordmark, Lyndon Ong, Shyamal Prasad, Kelvin Porter, Heinz Prantner, Jarno Rajahalme, Raymond E. Reeves, Renee Revis, Ivan Arias Rodriguez, A. Sankar, Greg Sidebottom, Brian Wyld, La Monte Yarroll, et de nombreus autres avec leurs précieux commentaires.

Puis, ajouter les co-auteurs de la [RFC4460], I. Arias-Rodriguez, K. Poon, et A. Caro.

Puis, ajouter les efforts des sept essais d'interopérabilité suivants de SCTP et de ceux qui ont commenté sur la [RFC4460], comme montré dans ses remerciements : Barry Zuckerman, La Monte Yarroll, Qiaobing Xie, Wang Xiaopeng, Jonathan

Wood, Jeff Waskow, Mike Turner, John Townsend, Sabina Torrente, Cliff Thomas, Yuji Suzuki, Manoj Solanki, Sverre Slotte, Keyur Shah, Jan Rovins, Ben Robinson, Renee Revis, Ian Periam, RC Monee, Sanjay Rao, Sujith Radhakrishnan, Heinz Prantner, Biren Patel, Nathalie Mouellic, Mitch Miers, Bernward Meyknecht, Stan McClellan, Oliver Mayor, Tomas Orti Martin, Sandeep Mahajan, David Lehmann, Jonathan Lee, Philippe Langlois, Karl Knutson, Joe Keller, Gareth Keily, Andreas Jungmaier, Janardhan Iyengar, Mutsuya Irie, John Hebert, Kausar Hassan, Fred Hasle, Dan Harrison, Jon Grim, Laurent Glaude, Steven Furniss, Atsushi Fukumoto, Ken Fujita, Steve Dimig, Thomas Curran, Serkan Cil, Melissa Campbell, Peter Butler, Rob Brennan, Harsh Bhondwe, Brian Bidulock, Caitlin Bestler, Jon Berger, Robby Benedyk, Stephen Baucke, Sandeep Balani, et Ronnie Sellar.

Des remerciement particuliers pour Mark Allman, qui devrait en fait être un des co-auteurs de la [RFC4460] pour son travail sur la salve maximale mais s'est arrangé pour s'esquiver à cause d'une divergence technique.

Nous tenons aussi à remercier Lyndon Ong et Phil Conrad de leurs apports précieux et de nombreuses contributions.

De plus, on a la [RFC4960] et ceux qui ont fait des commentaires sur elle, incluant Alfred Hänes et Ronnie Sellars.

Puis ajouter le co-auteur de la [RFC8540], Maksim Proshin.

Et les personnes qui ont fait des commentaires sur la [RFC8540], Pontus Andersson, Eric W. Biederman, Cedric Bonnet, Spencer Dawkins, Gorry Fairhurst, Benjamin Kaduk, Mirja Kähelwind, Peter Lei, Gyula Marosi, Lionel Morand, Jeff Morriss, Tom Petch, Kacheong Poon, Julien Pourtet, Irene Rängeler, Michael Welzl, et Qiaobing Xie.

Et, finalement, les personnes qui ont fourni des commentaires sur le présent document, Gorry Fairhurst, Martin Duke, Benjamin Kaduk, Tero Kivinen, Eliot Lear, Marcelo Ricardo Leitner, David Mandelberg, John Preuä Mattsson, Claudio Porfiri, Maksim Proshin, Ines Robles, Timo Välker, Magnus Westerlund, et Zhouming.

Nos remerciements ne peuvent pas être exprimés de façon adéquate à tous ceux qui ont participé au codage, aux essais, et au processus de mise à jour de ce document. Tout ce qu'on peut dire est Merci !

Adresse des auteurs

Randall R. Stewart
Netflix, Inc.
2455 Heritage Green Ave
Davenport, FL 33837
United States of America
mél : randall@lakerest.net

Michael Tüxen
Münster University of Applied Sciences
Stegerwaldstrasse 39
48565 Steinfurt
Germany
mél : tuexen@fh-muenster.de

Karen E. E. Nielsen
Kamstrup A/S
Industrivej 28
DK-8660 Skanderborg
Denmark
mél : kee@kamstrup.com