Groupe de travail Réseau
Request for Comments : 998
RFC rendue obsolète : RFC 969
Traduction Claude Brière de L'Isle

David D. Clark, MIT
Mark L. Lambert, MIT
Lixia Zhang, MIT
mars 1987

NETBLT : Protocole de transfert de données en vrac

1. Statut

Le présent document est une description et la spécification du protocole NETBLT. C'est une révision de la spécification publiée dans la RFC-969 du NIC. Le protocole a été révisé après des recherches extensives sur les performances de NETBLT sur des canaux satellites à fort délai et large bande passante. La plupart des changements de la spécification du protocole ont à voir avec le calcul et l'utilisation des temporisateurs de données dans le modèle de transfert de données avec mises en mémoire tampon multiples.

Le présent document est publié pour discussion et commentaire, et ne constitue pas une norme. La proposition peut changer et certaines parties du protocole n'ont pas encore été spécifiées ; la mise en œuvre du présent document n'est donc pas conseillée.

2. Introduction

NETBLT (NETwork BLock Transfer) est un protocole de niveau transport destiné au transfert rapide d'une grosse quantité de données entre des ordinateurs. Il assure un transfert fiable et à flux contrôlé, et il est conçu pour fournir un débit maximum sur une large variété de réseaux. Bien que NETBLT fonctionne actuellement par dessus le protocole Internet (IP) il devrait être capable de fonctionner par dessus tout protocole de datagrammes aux fonctions similaires à celles de IP.

La motivation de NETBLT' est de réaliser un débit supérieur à celui que les autres protocoles peuvent offrir. Le protocole atteint cet objectif en essayant de minimiser les effets de plusieurs problèmes en rapport avec le réseau : l'encombrement du réseau, les délais sur les liaisons satellites, et la perte de paquet.

Ses algorithmes de contrôle du taux de transmission s'accommodent bien de l'encombrement du réseau ; sa capacité de mises en mémoire tampon multiples permet un haut débit sur des canaux satellite à fort délai, et ses divers algorithmes de temporisation/retransmission minimisent les effets des pertes de paquets durant un transfert. Plus important, les caractéristiques de NETBLT donnent de bonnes performances sur les canaux à long délai sans abaisser les performances sur les LAN à haut débit.

Le protocole fonctionne en ouvrant une connexion entre deux "clients" ("l'envoyeur" et "le receveur") et transfère les données dans une série de gros agrégats de données appelés "mémoires tampons", et en fermant ensuite la connexion. Comme la quantité de données à transférer peut être très importante, le client n'est pas obligé de fournir toutes les données en une seule fois au module de protocole. Au lieu de cela, les données sont fournies par le client dans des mémoires tampons. La couche NETBLT transfère chaque mémoire tampon sous la forme d'une séquence de paquets ; comme chaque mémoire tampon se compose d'un grand nombre de paquets, l'interaction par mémoire tampon entre NETBLT et son client est bien plus efficace que ne le serait une interaction par paquet.

Dans sa forme la plus simple, un transfert NETBLT fonctionne comme suit : le client envoyeur charge une mémoire tampon de données et invoque la couche NETBLT pour la transférer. La couche NETBLT casse la mémoire tampon en paquets et envoie ces paquets à travers le réseau dans des datagrammes Internet. La couche NETBLT receveuse charge ces paquets dans une mémoire tampon correspondante fournie par le client receveur. Lorsque le dernier paquet de la mémoire tampon est arrivé, le NETBLT receveur vérifie que tous les paquets de la mémoire tampon ont été correctement reçus. Si certains paquets manquent, le NETBLT receveur demande qu'ils soient renvoyés. Lorsque la mémoire tampon a été complètement transmise, le client receveur en est notifié par sa couche NETBLT. Le client receveur dispose de la mémoire tampon et fournit une nouvelle mémoire tampon pour recevoir plus de données. Le NETBLT receveur notifie à l'envoyeur que la nouvelle mémoire tampon est prête, et l'envoyeur prépare et envoie la nouvelle mémoire tampon de la même manière. Cela continue jusqu'à ce que toutes les données aient été envoyées ; à ce moment, l'envoyeur notifie au receveur que la transmission est achevée. La connexion est alors close.

Comme décrit ci-dessus, le protocole NETBLT est en mode rigide *(lock-step)*. L'action se termine après la transmission

d'une mémoire tampon, et recommence après réception de la confirmation du receveur des données. NETBLT fournit des mémoires tampon multiples, un modèle de transfert dans lequel le NETBLT envoyeur peut transmettre de nouvelles mémoires tampon alors que des mémoires tampon antérieures sont en attente de confirmation par le NETBLT receveur. Les mémoires tampon multiples rendent le flux de paquets essentiellement continu et cela améliore sensiblement les performances.

Le reste du présent document décrit NETBLT en détail. Les prochaines sections décrivent la philosophie qui est derrière un certain nombre de caractéristiques du protocole : la mise en paquet, le contrôle de flux, la fiabilité du transfert, et la gestion de la connexion. Les sections finales décrivent les formats de paquet de NETBLT.

3. Mémoires tampons et paquets

NETBLT est conçu pour permettre le transfert de très grosses quantités de données entre deux clients. Durant l'établissement de la connexion, le NETBLT envoyeur peut informer le NETBLT receveur de la taille du transfert ; la longueur maximale du transfert est de 2**32 octets. Cette limite devrait permettre toutes les applications pratiques. Le paramètre de taille de transfert est à utiliser par le client receveur ; le NETBLT receveur n'en a aucune utilisation. Un NETBLT receveur accepte les données jusqu'à ce que l'envoyeur lui dise que le transfert est terminé.

Les données à envoyer doivent être coupées en mémoires tampon par le client. Chaque mémoire tampon doit être de la même taille, sauf la dernière mémoire tampon. Durant l'établissement de la connexion, les NETBLT envoyeur et receveur négocient la taille de la mémoire tampon, sur la base des limites fournies par les clients. Les tailles de mémoire tampon sont seulement en octets ; le client est responsable du placement des données dans les mémoires tampon sur les limites d'octets.

NETBLT a été conçu et devrait être mis en œuvre avec des mémoires tampon de toutes les tailles. La seule limitation fondamentale sur la taille de mémoire tampon devrait être la quantité de mémoire disponible pour le client. Les mémoires tampon devraient être aussi grandes que possible car cela minimise le nombre de transmissions de mémoire tampon et améliore donc les performances.

NETBLT est conçu pour exiger une quantité minimum de mémoire, permettant au client d'allouer autant de mémoire que possible à la mémorisation dans les mémoires tampon. En particulier, NETBLT ne garde pas de copies des mémoires tampon pour les besoins de la retransmission. Les données à retransmettre sont plutôt recopiées directement à partir de la mémoire tampon du client. Cela signifie que le client ne peut pas libérer pièce par pièce les mémoires tampon à mesure qu'elles sont envoyées, mais cela n'a pas posé de problème dans les mises en œuvre préliminaires de NETBLT.

Les mémoires tampon sont découpées par la couche NETBLT en séquences de paquets de données. Comme avec la taille de mémoire tampon, la taille du paquet de données est négociée entre les NETBLT envoyeur et receveur durant l'établissement de la connexion. À la différence de la taille de la mémoire tampon, la taille du paquet de données est seulement visible par la couche NETBLT.
Tous les paquets de données sauf le dernier paquet d'une mémoire tampon doivent avoir la même taille. Les paquets devraient être aussi grands que possible, car les performances de NETBLT sont en rapport direct avec la taille du paquet. En même temps, les paquets ne devraient pas être si grands qu'ils causent une fragmentation inter réseau, car cela cause normalement une dégradation des performances.

Toutes les mémoires tampon sauf la dernière doivent avoir la même taille ; la dernière peut avoir toute taille requise pour achever le transfert. Comme le NETBLT receveur ne connaît pas la taille du transfert à l'avance, il a besoin d'un moyen pour identifier le dernier paquet de chaque mémoire tampon. Pour cette raison, le dernier paquet de chaque mémoire tampon n'est pas un paquet DATA mais plutôt un paquet LDATA. Les paquets DATA et LDATA sont identiques sauf pour le type de paquet.

4. Contrôle du flux

NETBLT utilise deux stratégies pou le contrôle de flux, une interne et une au niveau du client.

Les NETBLT envoyeur et receveur transmettent les données dans les mémoires tampon ; le contrôle de flux du client est donc au niveau de la mémoire tampon. Avant qu'une mémoire tampon puisse être transmise, NETBLT confirme que les deux clients ont établi des mémoires tampon correspondantes, que l'un est prêt à envoyer les données, et que l'autre est prêt à recevoir les données. L'un et l'autre client peuvent donc contrôler le flux de données en ne fournissant pas une nouvelle mémoire tampon. Les clients ne peuvent pas stopper un transfert de mémoire tampon une fois qu'il est en cours.

Comme les mémoires tampon ne peuvent pas être très grandes, il faut qu'il y ait une autre méthode utilisée pour le contrôle de flux durant un transfert de mémoire tampon. La couche NETBLT fournit cette forme de contrôle de flux.

Plusieurs problèmes de contrôle de flux peuvent survenir lors de la transmission d'une mémoire tampon. Si le NETBLT envoyeur transfère des données plus vite que le NETBLT ne peut les traiter, la capacité du receveur à mettre en mémoire tampon les paquets non traités pourrait être débordée, causant des pertes de paquets. De même, une passerelle ou un réseau intermédiaire lent peut causer une accumulation de paquets et le débordement de l'espace de mémoire tampon de paquets dans le réseau. Des paquets vont alors être perdus au sein du réseau. Ce problème est particulièrement aigu pour NETBLT parce que les mémoires tampon NETBLT vont généralement être assez grandes, et donc composées de nombreux paquets.

Une solution traditionnelle du contrôle de flux de paquets est un système de fenêtre, dans lequel l'extrémité émettrice n'a la permission d'envoyer qu'un certain nombre de paquets pendant un certain temps. Malheureusement, le contrôle de flux qui utilise des fenêtres tend à avoir pour résultat un faible débit. Les fenêtres doivent rester petites afin d'éviter le débordement chez les hôtes et les routeurs, et elles ne peuvent pas être facilement mises à jour, car un échange de bout en bout est nécessaire pour chaque changement de fenêtre.

Pour permettre un haut débit sur divers réseaux et passerelles, NETBLT utilise une nouvelle méthode de contrôle de flux : le contrôle de taux. Le taux de transmission est négocié par les NETBLT envoyeur et receveur durant l'établissement de connexion et après chaque transmission de mémoire tampon. L'envoyeur utilise des temporisateurs, plutôt que des messages du receveur, pour maintenir le taux négocié.

Dans sa forme la plus simple, le contrôle de taux spécifie un délai minimum pour la transmission de paquet. Cela peut causer des problèmes de performances pour plusieurs raisons. D'abord, le temps de transmission pour un seul paquet est très court, fréquemment, plus court que la granularité du mécanisme de temporisation. Aussi, la redondance nécessaire pour entretenir les mécanismes de temporisation sur la base du paquet est relativement élevée et diminue les performances.

La solution est de contrôler le taux de transmission des groupes de paquets, plutôt que des paquets individuels. L'envoyeur transmet une salve de paquets sur un intervalle de temps négocié, puis envoie une autre salve. De cette façon, la redondance diminue d'un facteur égal à la taille de la salve, et le temps de transmission par salve est assez long pour que le mécanisme de transmission fonctionne correctement. Le contrôle de taux de NETBLT a donc deux parties, une taille de salve et un taux de salve, avec (taille de salve)/(taux de salve) égal au temps de transmission moyen par paquet.

La taille de salve et le taux de salve devraient se fonder non seulement sur la vitesse de transmission et de traitement de paquet que chaque extrémité peut traiter, mais aussi sur les capacités de tout réseau ou passerelle intermédiaire. Voici quelques valeurs intuitives pour la taille de paquet, la taille de mémoire tampon, la taille de salve, et le taux de salve.

La taille de paquet peut descendre jusqu'à 128 octets. Les performances avec des paquets aussi petits sont presque toujours mauvaises, à cause de la forte redondance du traitement par paquet. Même la taille par défaut de paquet du protocole Internet de 576 octets est seulement assez grosse pour des performances moyennes. La plupart des réseaux ne prennent pas en charge des tailles de paquet beaucoup plus grandes qu'un ou deux mille octets, et les paquets de cette taille peuvent aussi être fragmentés lorsque ils voyagent sur des réseaux intermédiaires, ce qui diminue les performances.

La taille d'une mémoire tampon NETBLT n'est limitée que par la quantité de mémoire disponible pour un client. Théoriquement, des mémoires tampon de 100 koctets ou plus sont possibles. Cela voudrait dire la transmission de 50 à 100 paquets par mémoire tampon.

La taille de salve et le taux de salve sont évidement très dépendants de la machine. Il y a une certaine quantité de redondance de transmission chez les machines d'émission et de réception qui est associée à l'entretien des temporisateurs et des processus de programmation. Cette redondance peut être minimisée par l'envoi des paquets en grosses salves. Il y a aussi des limitations imposées sur la taille des salves par le nombre de mémoires tampon de paquet disponibles dans le noyau du système d'exploitation. Sur les systèmes d'exploitation des plus modernes, une taille de salve d'entre cinq et dix paquets devrait réduire la redondance à un niveau acceptable. Une mise en œuvre NETBLT préliminaire pour le système IBM PC/AT envoie les paquets en salves de cinq. Elle pourrait en envoyer plus, mais est limitée par la mémoire disponible.

Le taux de salve est en partie déterminé par la granularité du mécanisme de temporisation de l'envoyeur, et en partie par la vitesse de traitement du receveur et de tout routeur intermédiaire. Il est aussi directement en relation avec la taille de salve. Des taux de salve de 20 à 45 millisecondes par salve de cinq paquets ont été essayées sur le IBM PC/AT et des mises en œuvre NETBLT Symbolics 3600 avec de bons résultats au sein d'un seul réseau de zone locale. Cette valeur dépend clairement de la bande passante du réseau et de la mise en mémoire tampon de paquet disponibles.

Tous les paramètres de contrôle de flux NETBLT (taille de paquet, taille de mémoire tampon, taille de salve, et taux de salve) sont négociés durant l'établissement de la connexion. Le processus de négociation est le même pour tous les paramètres. Le client qui initie la connexion (l'extrémité active) propose et envoie un ensemble de valeurs pour chaque

paramètre dans sa demande de connexion. L'autre client (l'extrémité passive) compare ces valeurs aux plus hautes valeurs de performances qu'il peut accepter. L'extrémité passive peut alors modifier tous les paramètres, mais seulement en les rendant plus restrictifs. Les paramètres modifiés sont alors renvoyés à l'extrémité active dans son message de réponse.

La taille de salve et le taux de salve peuvent aussi être renégociés après chaque transmission de mémoire tampon pour ajuster le taux de transfert aux performances observées lors du transfert de la mémoire tampon précédente. L'extrémité receveuse envoie les valeurs de taille et de taux de salve dans ses messages OK (décrits plus loin). L'envoyeur compare ces valeurs à celles qu'il peut accepter. Là encore, il peut modifier n'importe lesquels des paramètres, mais seulement en les rendant plus restrictifs. Les paramètres modifiés sont alors communiqués au receveur dans un paquet NULL-ACK, décrit plus loin.

Il est évident que chaque paramètre dépend de nombreux facteurs – vitesse de traitement de routeurs et des hôtes, mémoire disponible, granularité des temporisateurs – dont certains ne peuvent pas être vérifiés par l'un ou l'autre client. Chaque client doit donc essayer de faire les meilleures prévisions comme il peut, en réglant les performances sur les transferts suivants.

5. Modèle de transfert NETBLT

Chaque transfert NETBLT a trois étapes, l'établissement de la connexion, le transfert des données, et la fermeture de la connexion. Les étapes sont décrites en détail ci-dessous, avec les méthodes pour assurer que chaque étape se déroule de façon fiable.

5.1 Établissement de la connexion

Une connexion NETBLT est établie par un échange de deux paquets entre le NETBLT actif et le NETBLT passif. Noter que l'un ou l'autre NETBLT peut envoyer ou recevoir des données; les mots "actif" et "passif ne sont utilisés que pour différencier l'extrémité qui fait la demande de connexion de l'extrémité qui répond à la demande de connexion. L'extrémité active envoie un paquet OPEN ; l'extrémité passive accuse réception du paquet OPEN d'une des deux façons suivantes. Elle peut soit envoyer un paquet REFUSED, indiquant que la connexion ne peut pas être réalisée pour une certaine raison, ou elle peut réaliser l'établissement de la connexion en envoyant un paquet RESPONSE. À ce point, le transfert peut commencer.

Comme discuté dans la section précédente, les paquets OPEN et RESPONSE sont utilisés pour négocier les paramètres de contrôle de flux. Les autres paramètres utilisés dans le transfert de données sont aussi négociés. Ces paramètres sont (1) le nombre maximum de mémoires tampon qui peuvent envoyer à la fois, et (2) si il y aura ou non une somme de contrôle sur les paquets DATA. NETBLT effectue automatiquement une somme de contrôle sur tous les paquets non DATA/LDATA. Si le fanion Somme de contrôle négociée est mis à VRAI (1), l'en-tête et les données d'un paquet DATA/LDATA font l'objet d'une somme de contrôle ; si il est réglé à FAUX (0), seul l'en-tête fait l'objet d'une somme de contrôle. La valeur de la somme de contrôle est la négation au bit près de la somme des compléments à un des mots de 16 bits sur lesquels sont faits la somme de contrôle.

Finalement, chaque extrémité transmet sa valeur de temporisation de fin de vie en secondes soit dans le paquet OPEN, soit dans le paquet RESPONSE. La valeur de fin de vie sera utilisée pour déterminer la fréquence à laquelle envoyer les paquets KEEPALIVE durant les périodes d'inactivité d'une connexion ouverte (les temporisateurs de fin de vie et les paquets KEEPALIVE sont décrits au paragraphe suivant).

L'extrémité active spécifie un client passif au moyen d'un numéro d'accès "bien connu" de 16 bits spécifique du client sur lequel écoute l'extrémité passive. L'extrémité active s'identifie au moyen d'une adresse Internet de 32 bits et un numéro d'accès unique de 16 bits.

Afin de permettre aux extrémités active et passive de communiquer diverses informations utiles et non structurées, un champ de longueur variable est fourni dans les paquets OPEN et RESPONSE pour chaque information spécifique de client qui peut être nécessaire. De plus, un champ "raison du refus" est fourni dans les paquets REFUSED.

La récupération des paquets OPEN et RESPONSE perdus est assurée par l'utilisation des temporisateurs. L'extrémité active établit un temporisateur lorsque elle envoie un paquet OPEN. Lorsque le temporisateur arrive à expiration, un autre paquet OPEN est envoyé, jusqu'à un nombre maximum prédéterminé de paquets OPEN. Le temporisateur est arrêté à réception d'un paquet RESPONSE.

Pour empêcher la duplication des paquets OPEN et RESPONSE, le paquet OPEN contient un identifiant de connexion univoque de 32 bits qui doit être retourné dans le paquet RESPONSE. Cela empêche l'initiateur de confondre la réponse à

la demande en cours avec la réponse à une demande de connexion antérieure (il ne peut y avoir qu'une seule connexion entre deux accès donnés). Tout paquet OPEN ou RESPONSE avec un accès de destination qui correspond à celui d'une connexion ouverte a son identifiant univoque vérifié. Si l'identifiant univoque du paquet correspond à l'identifiant univoque de la connexion, le type du paquet est alors vérifié. Si c'est un paquet RESPONSE, il est traité comme un duplicata et ignoré. Si c'est un paquet OPEN, le NETBLT passif envoie une autre RESPONSE (supposant qu'un paquet RESPONSE précédent a été envoyé et perdu, amenant le NETBLT initiateur à retransmettre son paquet OPEN). Un identifiant univoque non correspondant doit être traité comme une tentative d'ouvrir une seconde connexion entre la même paire d'accès et est rejeté par l'envoi d'un message ABORT.

5.2 Transfert de données

Le plus simple modèle de transfert de données se passe comme suit. Le client envoyeur établit une mémoire tampon pleine de données. Le NETBLT receveur envoie un message GO à l'intérieur d'un paquet CONTROL à l'envoyeur, signifiant qu'il a aussi établi une mémoire tampon et qu'il est prêt à recevoir les données. Une fois que le message GO est reçu, l'envoyeur transmet la mémoire tampon comme une série de paquets DATA suivie par un paquet LDATA. Lorsque le dernier paquet de la mémoire tampon a été reçu, le receveur envoie un message RESEND à l'intérieur d'un paquet CONTROL contenant une liste des paquets qui n'ont pas été reçus. L'envoyeur renvoie ces paquets. Ce processus continue jusqu'à ce qu'il n'y ait plus de paquet manquant. À ce moment, le receveur envoie un message OK à l'intérieur d'un paquet CONTROL, établit une autre mémoire tampon pour recevoir des données, et envoie un autre message GO. L'envoyeur, ayant reçu le message OK, établit une autre mémoire tampon, attend le message GO, et répète le processus.

Le modèle de transfert de données ci-dessus est effectivement un protocole en mode rigide, et amène une perte de temps pendant l'envoi lorsque NETBLT attend la permission d'envoyer une nouvelle mémoire tampon. Un modèle de transfert plus efficace utilise plusieurs mémoires tampon pour augmenter les performances. La mémoire tampon multiple est une technique dans laquelle l'envoyeur et le receveur allouent et transmettent les mémoires tampon d'une manière qui permet la récupération d'erreur ou la confirmation de la réussite de la transmission des mémoires tampon précédentes concurremment avec la transmission de la mémoire tampon en cours.

Durant la phase d'établissement de la connexion, un des paramètres négociés est le nombre de mémoires tampon concurrentes permises durant le transfert. Si il y a plus d'une mémoire tampon disponible, le transfert de la prochaine mémoire tampon peut commencer juste après la fin de celui de la mémoire tampon en cours. Ceci est illustré dans l'exemple suivant :

Supposons deux mémoires tampon A et B dans un transfert à plusieurs mémoires tampon, avec A précédant B. Lorsque A a été transféré et que le NETBLT envoyeur attend un message soit OK, soit RESEND pour lui, le NETBLT envoyeur peut commencer immédiatement à envoyer B, gardant le flux de données à un taux stable. Si le receveur des données envoie un OK pour A, tout va bien ; si il reçoit un RESEND, les paquets spécifiés manquants dans le message RESEND sont retransmis.

Dans le modèle de transfert à mémoires tampon multiples, tous les paquets à envoyer sont réordonnés par numéro de mémoire tampon (le plus petit numéro en premier) avec le taux de transfert spécifié par la taille de salve et le taux de salve. Comme les numéros de mémoire tampon augmentent de façon monotone, les paquets provenant d'une mémoire tampon antérieure vont toujours précéder les paquets provenant d'une mémoire ultérieure.

Avoir plusieurs mémoires tampon transmises concurremment n'est en fait pas beaucoup plus compliqué que de transmettre une seule mémoire tampon à la fois. La clé est de visualiser chaque mémoire tampon comme un automate à états finis ; plusieurs mémoires tampon sont simplement un groupe d'automates à états finis, chacune dans un état parmi plusieurs possibles. Le processus de transfert consiste à déplacer les mémoires tampon à travers divers états jusqu'à ce que la transmission entière soit achevée.

Il y a plusieurs fautes évidentes dans le modèle de transfert de données décrit ci-dessus. D'abord, que ce passe t-il si les messages GO, OK, ou RESEND sont perdus ? L'envoyeur ne peut pas agir sur un paquet qu'il n'a pas reçu, de sorte que le protocole va rester en panne. Ensuite, si un paquet LDATA est perdu, comment le receveur sait-il quand la mémoire tampon a été transmise ? Des solutions pour chacun de ces problèmes sont présentées ci-dessous.

5.2.1 Récupération après des messages de perte de contrôle

NETBLT résout le problème des messages OK, GO, et RESEND perdus de deux façons. Premièrement, il peut utiliser un temporisateur de contrôle. Le receveur peut envoyer un ou plusieurs messages de contrôle (OK, GO, ou RESEND) au sein d'un seul paquet CONTROL. Chaque fois que le receveur envoie un paquet de contrôle, il établit un temporisateur de contrôle. Ce temporisateur est soit "rétablir" (établir à nouveau) soit "éliminé" (désactivé) sous les conditions suivantes :

Lorsque le temporisateur de contrôle arrive à expiration, le NETBLT receveur renvoie le paquet de contrôle et rétablit le temporisateur. Le NETBLT receveur continue de renvoyer les paquets de contrôle en réponse à l'expiration du temporisateur de contrôle jusqu'à ce que soit le temporisateur de contrôle soit retiré, soit que le temporisateur de fin de vie du NETBLT receveur (décrit plus loin) arrive à expiration (moment auquel il ferme la connexion).

Chaque message de contrôle comporte un numéro de séquence qui commence à un et augmente de un pour chaque message de contrôle envoyé. Le NETBLT envoyeur vérifie le numéro de séquence de chaque message de contrôle entrant par rapport à tous les autres numéros de séquence qu'il a reçus. Il mémorise le plus grand numéro de séquence en dessous duquel tous les autres numéros de séquence reçus sont consécutifs (dans les paragraphes suivants ceci est appelé le plus fort numéro de séquence acquitté) et il retourne ce numéro dans chaque paquet qui retourne vers le receveur. Le receveur a la permission de retirer son temporisateur de contrôle lorsque il reçoit un paquet de l'envoyeur avec un plus fort numéro de séquence acquitté supérieur ou égal au plus fort numéro de séquence dans le paquet de contrôle juste envoyé.

Idéalement, une mise en œuvre NETBLT devrait être capable de s'arranger de messages de contrôle hors séquence, peut-être en les collectant pour un traitement ultérieur, ou même en les traitant immédiatement. Si un message de contrôle entrant "comble" un "trou" dans un groupe de numéros de séquence de messages, la mise en œuvre pourrait même être assez habile pour le détecter et ajuster en conséquence sa valeur de numéro de séquence sortant.

Le NETBLT envoyeur, à réception d'un paquet CONTROL, devrait agir aussi rapidement que possible sur le paquet. Soit il établit une nouvelle mémoire tampon (à réception d'un message OK pour une mémoire tampon précédente) marque les données à renvoyer (à réception d'un message RESEND) soit il prépare une mémoire tampon à envoyer (à réception d'un message GO). Si le NETBLT envoyeur n'est pas en mesure d'envoyer des données, il devrait envoyer un paquet NULL-ACK, qui contient son plus fort numéro de séquence acquitté (cela permet au NETBLT receveur d'accuser réception de tout les messages de contrôle en instance) et attendre qu'il puisse envoyer plus de données. Dans tous ces cas, la redondance du système pour une réponse au message de contrôle entrant devrait être petite et relativement constante.

La petite quantité de redondance de traitement de message permet d'établir des temporisateurs de contrôle précis pour tous les types de messages de contrôle avec un seul algorithme simple – le temps de transit d'un aller-retour réseau, plus un facteur de variance. Ceci est plus efficace que les schémas utilisés par les autres protocoles, où le calcul de la valeur du temporisateur a été un problème parce que le temps de traitement pour un certain paquet peut varier considérablement en fonction du type de paquet.

L'estimation de la valeur du temporisateur de contrôle est extrêmement importante dans un protocole à hautes performances comme NETBLT. Un temporisateur de contrôle long fait attendre le NETBLT receveur pendant de longues périodes de temps avant de retransmettre les messages non acquittés. Une valeur courte de temporisateur de contrôle cause la réception par le NETBLT envoyeur de nombreux messages de contrôle dupliqués (qu'il peut rejeter, mais qui prennent du temps).

En plus de l'utilisation des temporisateurs de contrôle, NETBLT réduit la perte des messages de contrôle en utilisant un seul paquet de contrôle à longue durée de vie ; le paquet est traité comme une file d'attente FIFO *(premier entré, premier sorti)* avec de nouveaux messages de contrôle ajoutés à la fin et les messages de contrôle acquittés retirés du début. La mise en œuvre place les messages de contrôle dans le paquet de contrôle et transmet la totalité du paquet de contrôle, consistant en tous les messages de contrôle non acquittés plus les nouveaux messages juste ajoutés. Le paquet de contrôle entier est aussi transmis chaque fois que le temporisateur de contrôle arrive à expiration. Comme les transmissions de paquets de contrôle sont très fréquentes, les messages non acquittés peuvent être transmis plusieurs fois avant qu'ils soient enfin acquittés. Cette transmission redondante des messages de contrôle assure la récupération automatique de la plupart des pertes de message de contrôle sur un canal à pertes.

Ce schéma fait peser une certaine charge sur le receveur des messages de contrôle. Il doit être capable de rejeter rapidement les messages de contrôle dupliqués, car un certain message peut être retransmis plusieurs fois avant que son accusé de réception soit reçu et qu'il soit retiré du paquet de contrôle. Ceci est normalement très facile à faire ; l'envoyeur de données élimine simplement tous les messages de contrôle qui ont des numéros de séquence inférieurs à son plus fort numéro de séquence acquitté.

Un autre problème de ce schéma est que le paquet de contrôle peut devenir plus grand que la taille maximum de paquet admissible si trop de messages de contrôle sont placés dedans. Cela n'a pas été un problème dans les mises en œuvre actuelles de NETBLT : la taille normale d'un paquet de contrôle est de 1000 octets ; les messages de contrôle RESEND ont en moyenne environ 20 octets, les messages GO font 8 octets, et les messages OK font 16 octets. Cela permet de placer 50 à 80 messages de contrôle dans le paquet de contrôle, plus qu'assez pour les transferts raisonnables. D'autres mises en œuvre peuvent produire plusieurs paquets de contrôle si un seul n'est pas suffisant.

La valeur du temporisateur de contrôle doit être estimée avec soin. Il peut avoir pour valeur initiale un nombre arbitraire. Les paquets de contrôle suivants devraient avoir leurs valeurs de temporisateur fondées sur le temps de transit d'un aller-

retour du réseau (c'est à dire, le temps écoulé entre l'envoi du paquet de contrôle et la réception de l'accusé de réception de tous les messages dans le paquet de contrôle) plus un facteur de variance. La valeur du temporisateur devrait être continuellement mise à jour sur la base d'une moyenne lissée des temps d'aller-retour de transit collectés.

5.2.2 Récupération de la perte de paquets LDATA

NETBLT résout le problème de la perte du paquet LDATA en utilisant un temporisateur de données pour chaque mémoire tampon à l'extrémité réceptrice. Le plus simple modèle de temporisateur de données a un temporisateur de données qui est établi lorsque une mémoire tampon est prête à être reçue ; si le temporisateur de données arrive à expiration, le NETBLT receveur va supposer une perte du paquet LDATA et va envoyer un message RESEND demandant tous les paquets DATA manquants dans la mémoire tampon. Lorsque tous les paquets ont été reçus, le temporisateur est supprimé.

Les valeurs de temporisateur de données ne se fondent pas sur le temps de transit d'aller-retour réseau ; elles s'appuient plutôt sur le temps nécessaire pour transférer une mémoire tampon (comme déterminé par le nombre de salves de paquets DATA dans la mémoire tampon multiplié par le taux de salve) plus un facteur de variance <1>.
Il est évident qu'une estimation précise de la valeur du temporisateur de données est très importante. Un petite valeur du temporisateur de données cause l'envoi inutile de paquets RESEND par le NETBLT receveur. Cela cause une sérieuse dégradation des performances car le NETBLT envoyeur doit arrêter ce qu'il fait et renvoyer un certain nombre de paquets DATA.

L'établissement et la suppression des temporisateurs de données se révèlent être assez compliqués, en particulier dans un modèle de transfert à mémoires tampon multiples. Pour comprendre comment et quand les temporisateurs de données sont établis et supprimés il est utile de voir chaque mémoire tampon comme un automate à états finis et de regarder les divers états.

La séquence d'états pour l'envoi d'une mémoire tampon est simple. Lorsque un message GO est reçu pour la mémoire tampon, celle-ci est créée, remplie avec les données, et placée dans un état SENDING. Lorsque un OK a été reçu pour cette mémoire tampon, elle passe dans un état SENT et est envoyée.

La séquence d'état pour la réception d'une mémoire tampon est un petit peu plus compliquée. Supposons l'existence d'une mémoire tampon A. Lorsque est envoyé un message de contrôle pour A, la mémoire tampon passe à l'état ACK-WAIT (elle attend l'accusé de réception du message de contrôle).

Aussitôt que le message de contrôle a été acquitté, la mémoire tampon A passe de l'état ACK-WAIT à l'état ACKED (elle attend maintenant que les paquets DATA arrivent). À ce moment, le temporisateur de données de A est établi et le message de contrôle est retiré du paquet de contrôle. L'estimation de la valeur du temporisateur de données à ce moment est assez difficile. Dans un modèle de transfert à plusieurs mémoires tampon, le NETBLT receveur peut envoyer plusieurs messages GO à la fois. Un seul paquet DATA provenant du NETBLT envoyeur pourrait accuser réception de tous les messages GO, provoquant le lancement des temporisateurs de données par plusieurs mémoires tampon. Il est clair que chacun des temporisateurs de données doit être établi d'une manière qui prenne en compte la place de chaque mémoire tampon dans l'ordre de transmission. Les paquets pour une mémoire tampon A - 1 seront toujours transmis avant les paquets dans A, de sorte que le temporisateur de données de A doit prendre en compte l'arrivée de tout paquet DATA de A – 1 ainsi que de l'arrivée de ses propres paquets DATA. Cela signifie que les valeurs de temporisateur deviennent de moins en moins précises pour les mémoires tampon de numéros supérieurs. Comme ces valeurs de temporisateur de données peuvent être assez imprécises, on l'appelle un temporisateur de données "lâche". La valeur du temporisateur de données lâche est recalculée ultérieurement (en utilisant le même algorithme, mais avec des informations mises à jour) donnant un temporisateur "strict", comme décrit ci-dessous.

Lorsque le premier paquet DATA arrive pour A, A passe de l'état ACKED à l'état RECEIVING et son temporisateur de données est établi à une nouvelle valeur "stricte". La valeur de temporisateur stricte est calculée de la même manière que celle du temporisateur lâche, mais elle est plus précise car du temps a passé et ces mémoires tampon ont duré moins longtemps que ce que A avait d'abord pensé (ou leurs paquets sont arrivés avant ceux de A) laissant moins de paquets arriver entre l'établissement du temporisateur de données et l'arrivés du dernier paquet DATA en A.

Le NETBLT receveur établit aussi les temporisateurs de données stricts de toutes les mémoires tampon de numéro inférieur à A qui sont aussi dans l'état ACKED. Ceci est fait comme une optimisation : on sait que les mémoires tampon sont traitées dans l'ordre, le plus faible numéro en premier. Si une mémoire tampon B qui a un numéro inférieur à celui de A est dans l'état ACKED, ses paquets DATA devraient arriver avant ceux de A. Comme ceux de A sont arrivés en premier, ceux de B doivent avoir été perdus. Comme le temporisateur lâche de données de B n'est pas arrivé à expiration (il aurait alors envoyé un message RESEND et serait dans l'état ACK-WAIT) on établit le temporisateur strict, permettant aux paquets manquants d'être détectés plus tôt. Un RESEND immédiat n'est pas envoyé parce que il est possible que le paquet de A ait été réordonné avant celui de B par le réseau, et que les paquets de B arrivent sous peu.

Lorsque tous les paquets DATA pour A ont été reçus, il passe de l'état RECEIVING à l'état RECEIVED et est supprimé. Si des paquets avaient été manquants, le temporisateur de données de A serait arrivé à expiration et A serait passé à l'état ACK-WAIT après l'envoi d'un message RESEND. La progression d'état se ferait alors comme dans l'exemple ci-dessus.

Le système de temporisateur de contrôle et de données peut être résumé comme suit : normalement, le NETBLT receveur fonctionne avec un des deux types de temporisateurs, un temporisateur de contrôle ou un temporisateur de données. Il y a un temporisateur de données par transmission de mémoire tampon et un temporisateur de contrôle par paquet de contrôle. Le temporisateur de données est actif lorsque sa mémoire tampon est soit dans l'état ACKED (la valeur lâche de temporisateur de données est utilisée) soit dans l'état RECEIVING (la valeur stricte de temporisateur de données est utilisée) ; un temporisateur de contrôle est actif chaque fois que le NETBLT receveur a des messages de contrôle non acquittés dans son paquet de contrôle.

5.2.3 Temporisateurs de vie et paquets de maintien en vie

Le système ci-dessus pose encore quelques problèmes. Si le NETBLT envoyeur n'est pas prêt à envoyer, il envoie un seul paquet NULL-ACK pour supprimer tous les temporisateurs de contrôle en activité à l'extrémité de réception. Après cela, le receveur va attendre. Le NETBLT envoyeur pourrait cesser d'exister et le receveur, avec ses temporisateurs de contrôle supprimés, resterait en attente. Aussi, le système ci-dessus ne met de temporisateurs que chez le NETBLT receveur. Le NETBLT d'envoi n'a pas de temporisateurs ; si le NETBLT receveur cesse d'exister, le NETBLT envoyeur va rester en attente que des messages de contrôle arrivent.

La solution des deux problème ci-dessus est d'utiliser un temporisateur de durée de vie et un paquet de maintien en vie pour les deux NETBLT, l'envoyeur et le receveur. Aussitôt que la connexion est ouverte, chaque extrémité établit un temporisateur de vie ; ce temporisateur est remis à zéro chaque fois qu'un paquet est reçu. Lorsque le temporisateur de vie d'un NETBLT arrive à expiration, il peut supposer que l'autre extrémité est morte et peut clore la connexion.

Il est possible que le NETBLT envoyeur ou receveur doive attendre pendant de longues périodes alors que leurs clients respectifs ont de l'espace de mémoire tampon et chargent leurs mémoires tampons avec des données. Comme un NETBLT qui attend de l'espace de mémoire tampon est dans un état parfaitement valide, le protocole doit avoir une méthode pour empêcher que le temporisateur de vie de l'autre extrémité n'arrive à expiration. La solution est d'utiliser un paquet KEEPALIVE, qui est envoyé de façon répétée à des intervalles fixes lorsque un NETBLT ne peut pas envoyer d'autres paquets. Comme le temporisateur de vie est remis à zéro chaque fois qu'un paquet est reçu, il ne va jamais arriver à expiration tant que l'autre extrémité envoie des paquets.

La fréquence à laquelle sont transmis les paquets KEEPALIVE est calculée comme suit : au démarrage de la connexion, chaque NETBLT choisit une valeur de temporisateur de vie et l'envoie à l'autre extrémité dans le paquet OPEN ou dans le paquet RESPONSE. L'autre extrémité prend la valeur de temporisation de vie et l'utilise pour calculer une fréquence à laquelle envoyer les paquets KEEPALIVE. La fréquence des KEEPALIVE devrait être assez élevée pour que plusieurs paquets KEEPALIVE puissent être perdus avant que le temporisateur de vie de l'autre extrémité n'arrive à expiration (par exemple, la valeur du temporisateur de vie divisée par quatre).

La valeur du temporisateur de vie est relativement facile à estimer. Comme il est continuellement remis à zéro, il n'a pas besoin de se fonder sur la taille de transfert. Il devrait plutôt être fondé, au moins en partie, sur le type d'application qui utilise NETBLT. Les applications d'usager devraient avoir de plus petites valeurs de temporisation de vie pour éviter de forcer les humains à attendre pendant de longues périodes que survienne une fin de temporisation de vie. Les applications machines peuvent avoir de plus longues valeurs de temporisation.

5.3 Fermeture de la connexion

Il y a trois façons de clore une connexion : une clôture de connexion, un "quit", ou un "abort".

5.3.1 Transfert réussi

Après un transfert de données réussi, NETBLT ferme la connexion. Lorsque l'envoyeur transmet la dernière mémoire tampon de données, il établit un fanion "last-buffer" sur chaque paquet DATA dans la mémoire tampon. Cela signifie qu'aucunes NOUVELLES données ne seront transmises. Le receveur sait que le transfert s'est achevé avec succès lorsque tout ce qui suit est vrai : (1) il a reçu des paquets DATA avec un fanion "last-buffer" établi, (2) tous ses messages de contrôle ont été acquittés, et (3) il n'a pas de mémoire tampon en instance avec des paquets manquants. À ce moment, il est permis au receveur de clore sa moitié de connexion. L'envoyeur sait que le transfert s'est achevé lorsque ce qui suit est vrai : (1) il a transmis des paquets DATA avec un fanion "last-buffer" établi et (2) il a reçu des messages OK pour toutes ses mémoires tampon. À ce moment, il "lambine" pendant un délai prédéterminé avant de clore sa moitié de la connexion. Si le

paquet NULL-ACK qui accuse réception du dernier message OK du receveur a été perdu, le receveur a du temps pour retransmettre le message OK, recevoir un nouveau NULL-ACK, et reconnaître un transfert réussi. La valeur du temps de lambinage DOIT se fonder sur la valeur du temporisateur de contrôle du receveur ; elle doit être assez longue pour permettre au temporisateur de contrôle du receveur d'arriver à expiration afin que le message OK puisse être envoyé à nouveau. Pour cette raison, tous les messages OK contiennent (en plus des valeurs de nouvelle taille de salve et de taux de salve) la valeur du temporisateur de contrôle actuel du receveur en millisecondes. L'envoyeur utilise cette valeur pour calculer sa valeur de temporisateur de lambinage.

Comme la valeur du temporisateur de lambinage peut être assez élevée, il est permis au NETBLT receveur de "court circuiter" le temporisateur de lambinage du NETBLT envoyeur en transmettant un paquet DONE *(fait)*. Le paquet DONE est transmis lorsque le receveur sait que le transfert s'est achevé avec succès. Lorsque l'envoyeur reçoit un paquet DONE, il lui est permis de supprimer son temporisateur de lambinage et de clore immédiatement sa moitié de connexion. Le paquet DONE n'est pas transmis de façon fiable, car manquer à le recevoir signifie seulement que le NETBLT envoyeur prendra plus longtemps pour clore sa moitié de la connexion (car il attend pendant la durée de son temporisateur de lambinage pour la supprimer).

5.3.2 QUIT du client

Durant un transfert NETBLT, un client peut envoyer un paquet QUIT à l'autre si il pense que l'autre client fonctionne mal. Comme le QUIT survient au niveau client, la transmission de QUIT ne peut survenir qu'entre les transmissions de mémoire tampon. Le NETBLT qui reçoit le paquet QUIT peut ne prendre aucune autre action que de notifier immédiatement son client et de transmettre un paquet QUITACK. L'envoyeur du QUIT doit attendre la fin de temporisation et retransmettre jusqu'à ce qu'un QUITACK ait été reçu ou que son temporisateur de vie soit arrivé à expiration. L'envoyeur du QUITACK lambine avant de quitter, de sorte qu'il peut répondre à une retransmission de QUIT.

5.3.3 NETBLT ABORT

Un ABORT a lieu lorsque une couche NETBLT pense que lui ou son correspondant fonctionne mal. Comme le ABORT est généré dans la couche NETBLT, il peut être envoyé à tout moment. Le ABORT implique que la couche NETBLT fonctionne mal, de sorte qu'aucune fiabilité de transmission n'est espérée, et que l'envoyeur peut clore immédiatement sa connexion.

6. Structure de mise en couches du protocole

NETBLT est mis en œuvre directement par dessus le protocole Internet (IP). Il lui a été alloué un numéro officiel de protocole de 30 (en décimal).

7. Améliorations prévues

Comme il est actuellement spécifié, NETBLT n'a pas d'algorithme pour déterminer ses paramètres de contrôle de taux (taux de salve, taille de salve, etc.). Dans la vérification des performances initiales, ces paramètres ont été établis par la personne qui effectue l'essai. On va maintenant explorer les façons d'établir et ajuster automatiquement les paramètres de contrôle de taux de NETBLT.

8. Formats de paquet

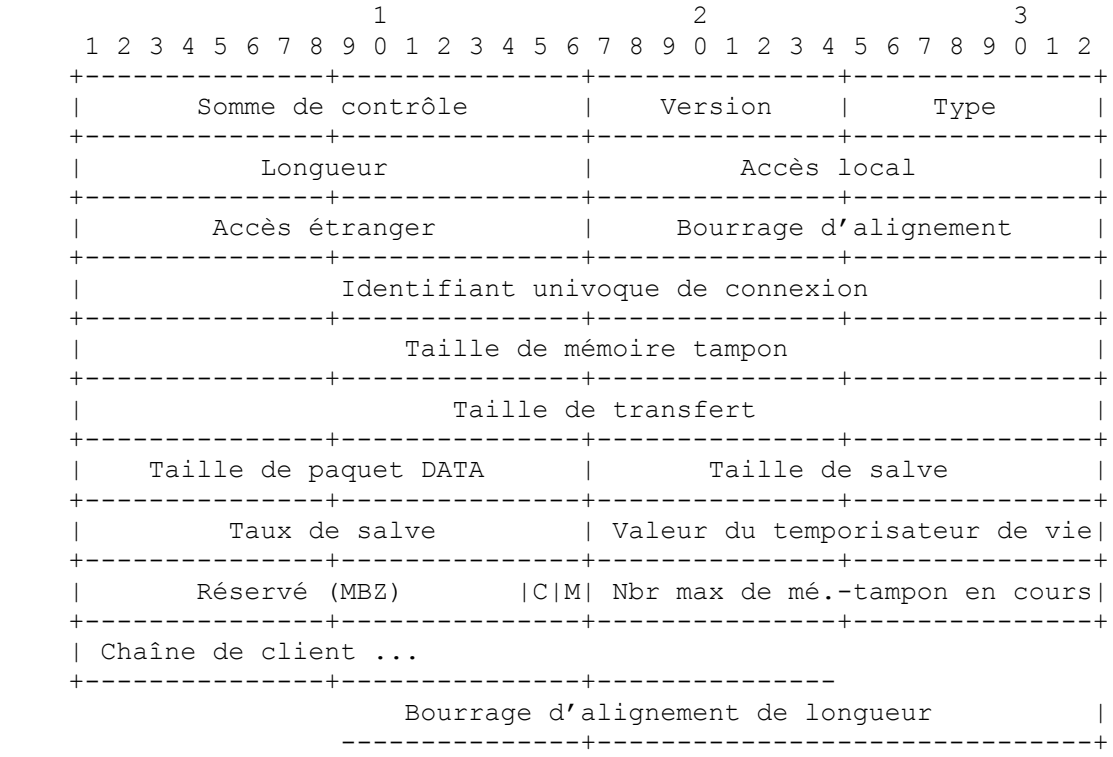
Les paquets NETBLT se divisent en trois catégories, qui partagent toutes un en-tête commun de paquet. Il y a d'abord les paquets qui ne voyagent que de l'envoyeur au receveur des données ; ils contiennent le plus fort numéro de séquence acquitté que le receveur utilise pour la fiabilité de la transmission du message de contrôle. Ces paquets sont NULL-ACK, DATA, et LDATA. Ensuite, il y a un paquet qui ne voyage que du receveur à l'envoyeur. C'est le paquet CONTROL ; chaque paquet CONTROL peut contenir un nombre arbitraire de messages de contrôle (GO, OK, ou RESEND) dont chacun a son propre numéro de séquence. Finalement, il y a les paquets qui ont soit une façon particulière d'assurer la fiabilité, soit ne sont pas transmis de façon fiable. Ce sont les paquets OPEN, RESPONSE, REFUSED, QUIT, QUITACK, DONE, KEEPALIVE, et ABORT. Parmi eux, tous sauf le paquet DONE peuvent être envoyés aussi bien par le NETBLT envoyeur que receveur.

Tous les paquets sont alignés sur un multiple de quatre octets et tous les champs de quatre octets commencent sur une

limite de mot de quatre octets. Tous les champs de chaînes de longueur arbitraire se terminent par au moins un octet nul, avec des octets nuls supplémentaires ajoutés à la fin pour créer un champ qui soit un multiple de quatre octets.

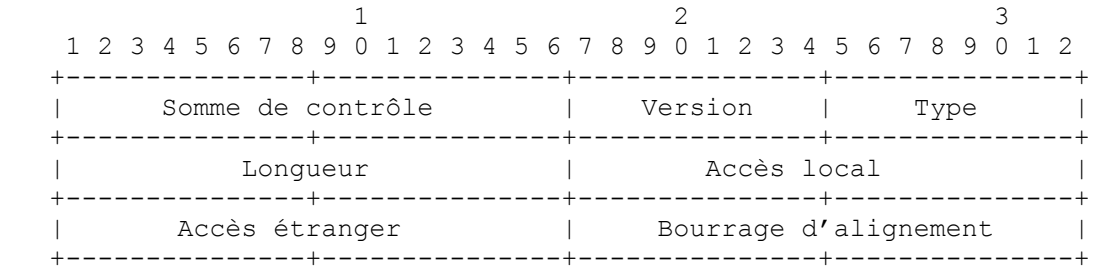
Formats de paquet pour NETBLT :

OPEN (type 0) et RESPONSE (type 1):

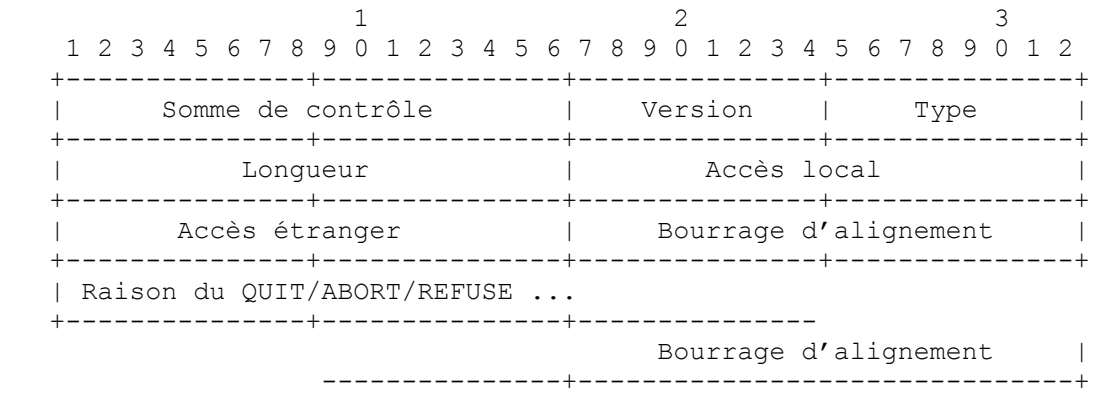


Somme de contrôle : somme de contrôle sur le paquet (l'algorithme est décrit à la section "Établissement de connexion").
Version : numéro de version du protocole NETBLT.
Type : numéro du type de paquet NETBLT (OPEN = 0, RESPONSE = 1, etc.).
Longueur : longueur totale (en-tête NETBLT plus données, s'il en est) du paquet NETBLT en octets.
Accès local : numéro de l'accès local NETBLT de 16 bits.
Accès étranger : numéro de l'accès NETBLT étranger de 16 bits.
Identifiant univoque de connexion : UID de connexion de 32 bits spécifié au paragraphe "Établissement de connexion".
Taille de mémoire tampon : taille en octets de chaque mémoire tampon NETBLT (sauf la dernière).
Taille de transfert : (facultatif) taille en octets du transfert. Ceci est seulement pour l'information du client ; le NETBLT receveur NE DEVRAIT PAS l'utiliser.
Taille du paquet de données : longueur de chaque paquet DATA en octets.
Taille de salve : nombre de paquets DATA dans une salve.
Taux de salve : temps de transmission en millisecondes d'une seule salve.
Temporisateur de vie : valeur du temporisateur de vie de l'envoyeur du paquet en secondes.
"M" : mode de transfert (0 = LECTURE, 1 = ÉCRITURE).
"C" : fanion de somme de contrôle des données du paquet DATA (0 = ne pas faire la somme de contrôle, 1 = la faire).
Nbr max de mé.-tampon en cours : nombre maximum de mémoires tampon qui peuvent être transférées avant d'attendre un message OK du NETBLT receveur.
Chaîne de client : chaîne arbitraire, terminée par des octets à zéro, alignée sur des mots de quatre octets, à utiliser par les clients NETBLT.

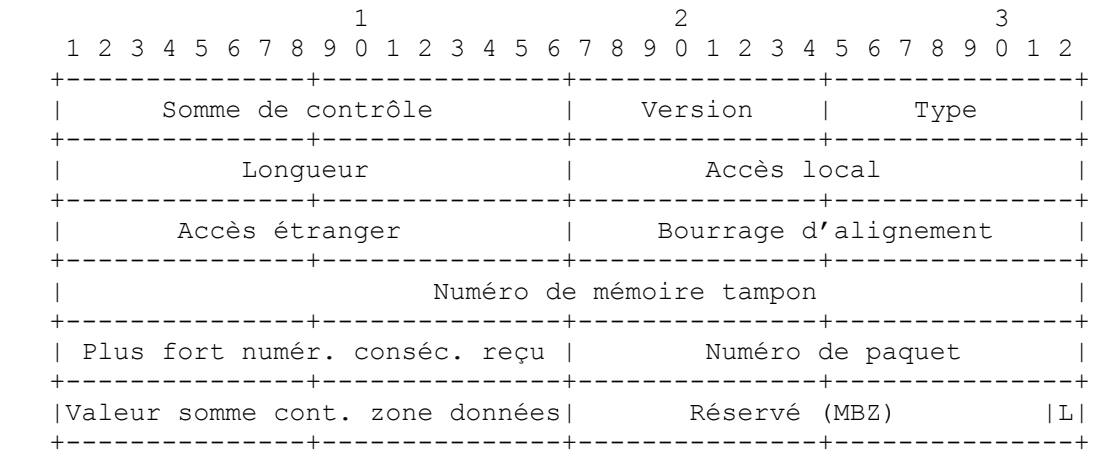
KEEPALIVE (type 2), QUITACK (type 4), et DONE (type 11)



QUIT (type 3), ABORT (type 5), et REFUSED (type 10)



DATA (type 6) et LDATA (type 7):



Numéro de mémoire tampon : numéro univoque de 32 bits alloué à chaque mémoire tampon. Les numéros augmentent de façon monotone.

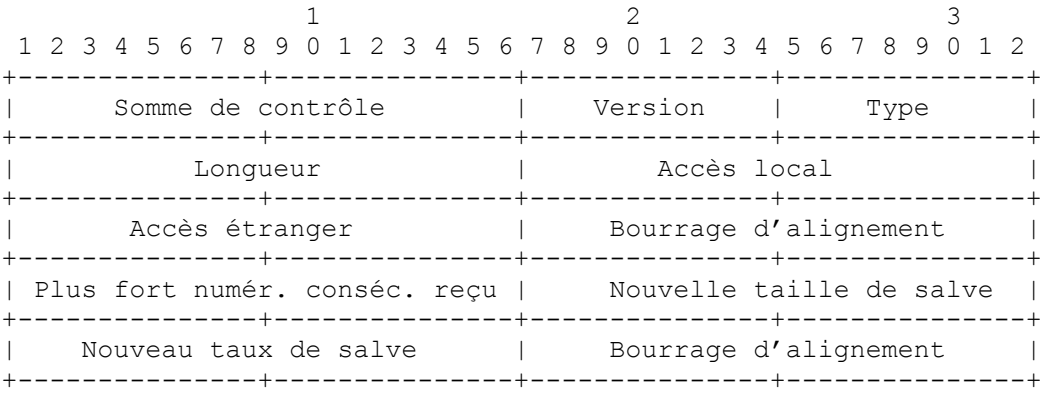
Plus fort numéro de séquence consécutif reçu : plus fort numéro de séquence de message de contrôle en dessous duquel tous les numéros de séquence reçus sont consécutifs.

Numéro de paquet : identifiant de paquet DATA à augmentation monotone.

Valeur de la somme de contrôle de la zone de données : Somme de contrôle des données du paquet DATA. L'algorithme utilisé est le même que celui utilisé pour calculer les sommes de contrôle des autres paquets NETBLT.

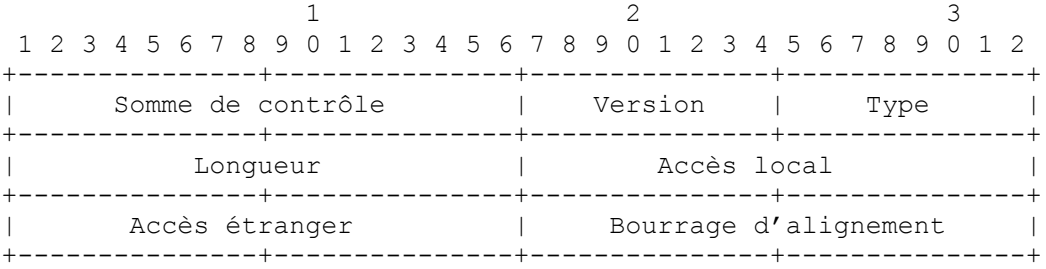
"L" est un fanion établi lorsque la mémoire tampon à laquelle appartient ce paquet DATA est la dernière mémoire tampon dans le transfert.

NULL-ACK (type 8)



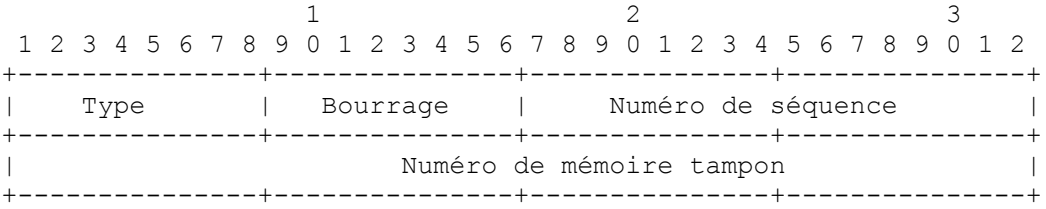
Plus fort numéro de séquence consécutif reçu : le même que dans le paquet DATA/LDATA.
Nouvelle taille de salve : taille de salve négociée à partir de la valeur donnée par le NETBLT receveur dans le message OK.
Nouveau taux de salve : taux de salve négocié à partir de la valeur donnée par le NETBLT receveur dans le message OK. La valeur est en millisecondes.

CONTROL (type 9):



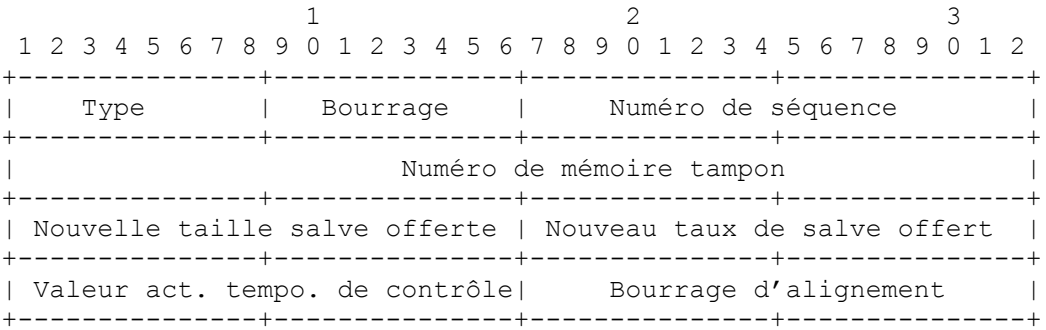
Suivi par un nombre quelconque de messages, dont chacun est aligné sur quatre octets, avec les formats suivants :

Message GO (type 0) :



Type : type du message (GO = 0, OK = 1, RESEND = 2)
Numéro de séquence : numéro de message univoque de 16 bits. Les numéros de séquence doivent avoir une augmentation monotone et commencent à 1.
Numéro de mémoire tampon : comme dans le paquet DATA/LDATA.

Message OK (type 1) :

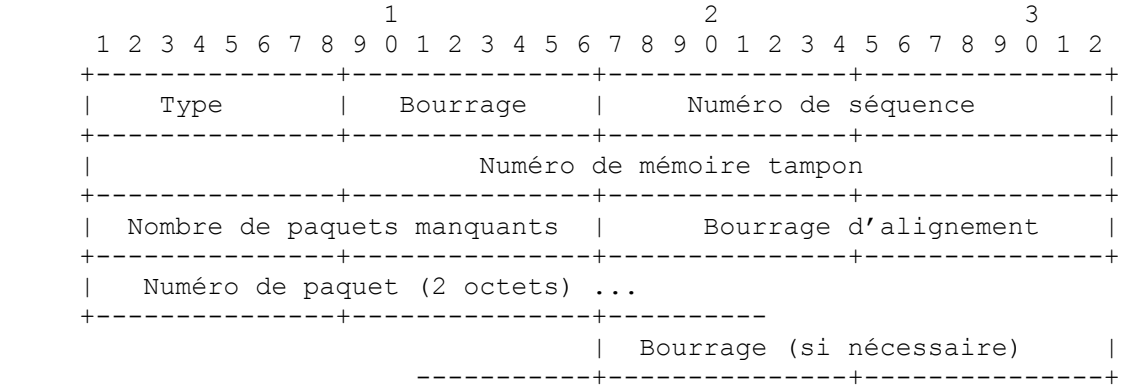


Nouvelle taille de salve offerte : taille de salve pour les transferts ultérieurs de mémoire tampon, éventuellement fondée sur les informations de performances provenant des précédents transferts de mémoires tampons.

Nouveau taux de salve offert : taux de salve pour les transferts de mémoire tampon ultérieurs, éventuellement fondé sur les informations de performances provenant des précédents transferts de mémoires tampons. Le taux est en millisecondes.

Valeur actuelle du temporisateur de contrôle : valeur en millisecondes du temporisateur de contrôle du NETBLT receveur.

Message RESEND (type 2) :



Numéro de paquet : identifiant de 16 bits de paquet de données qui se trouve dans chaque paquet DATA.

Notes :

<1> Lorsque la taille de la mémoire tampon est grande, la variance des délais d'aller retour de nombreux paquets peut s'annuler ; cela signifie que la valeur de la variance n'a pas besoin d'être très grosse. Cette hypothèse sera étudiée plus en détails avec d'autres essais.