(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 922 068**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **07 58149**

(51) Int Cl[8] : ***H 04 L 12/56*** *(2006.01), H 04 L 12/46*

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt :** 08.10.07.

(30) **Priorité :**

(43) **Date de mise à la disposition du public de la demande :** 10.04.09 Bulletin 09/15.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Se reporter à la fin du présent fascicule*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *CANON KABUSHIKI KAISHA* — JP.

(72) **Inventeur(s) :** ROUSSEAU PASCAL et BARON STEPHANE.

(73) **Titulaire(s) :**

(74) **Mandataire(s) :** CABINET PATRICE VIDON.

(54) PROCEDE DE NOTIFICATION A UN DISPOSITIF SOURCE D'UNE TAILLE LIMITE DE PAQUETS DE DONNEES, PRODUIT PROGRAMME D'ORDINATEUR, MOYEN DE STOCKAGE ET TETE DE TUNNEL CORRESPONDANTS.

(57) L'invention concerne un procédé de notification à un dispositif source (109, 110, 111, 112) d'une taille limite de paquets de données destinées à être transmises dans un tunnel (100) comprenant une pluralité de canaux de transmission. Le dispositif source est relié à une tête de tunnel (101) via un réseau de communication (107). Le procédé est mis en oeuvre par la tête de tunnel.

Selon l'invention, un tel procédé de notification comprend les étapes suivantes, pour chaque flux de données destinées à être transmises dans le tunnel et provenant du dispositif source:

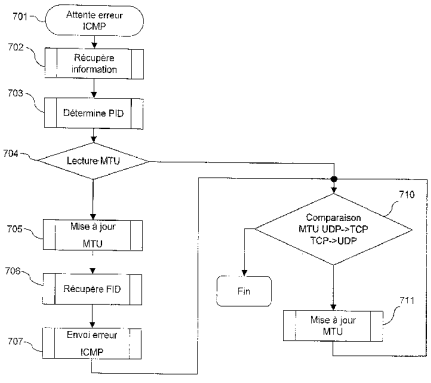
a) détection d'un événement déclencheur lié à un nouveau canal associé au flux pour le transport du flux dans le tunnel;

b) obtention d'une première taille limite de paquets associée à un canal précédent associé au flux pour le transport du flux dans le tunnel;

c) obtention d'une deuxième taille limite de paquets associée au nouveau canal;

d) détection d'un changement de taille limite de paquets (701), par comparaison de la première taille limite de paquets avec la deuxième taille limite de paquets;

e) en cas de détection positive, transmission (707) d'un message de changement de taille limite de paquets vers le dispositif source.



FR 2 922 068 - A1

Procédé de notification à un dispositif source d'une taille limite de paquets de données, produit programme d'ordinateur, moyen de stockage et tête de tunnel correspondants.

1. DOMAINE DE L'INVENTION

Le domaine de l'invention est celui des réseaux de communication.

Plus précisément, l'invention concerne une technique de notification à un dispositif source d'une taille limite (désignée ci-après par « MTU » pour « Maximum Transmission Unit » en anglais) des données utiles pouvant être transmises dans un paquet de données (aussi appelés datagrammes) transitant dans un tunnel de communication.

L'invention s'applique notamment, mais non exclusivement, à des dispositifs tels que, par exemple, les télévisions, les systèmes de type home cinéma, les caméscopes, les imprimantes, les appareils photo ou tout autre équipement photo et audio/vidéo numérique pour le grand public.

La démocratisation d'Internet haut débit d'une part et l'apparition d'équipements audiovisuels grand public ayant une connectivité réseau d'autre part vont créer de nouveaux comportements des utilisateurs. Parmi ces nouveaux comportements, il fait peu de doute que nous verrons apparaître des individus appartenant à des groupes de personnes ayants des domaines d'intérêts communs (loisirs, famille…) que nous pourrions appeler « en liaison permanente ». Ceux-ci établiront des connections quasi permanentes avec les autres individus d'un même domaine d'intérêt en établissement des communications audio et/ou vidéo et partageant des informations de tout type (audio, vidéo, photo, texte …).

La technologie des Réseaux Privés Virtuels (RPV, ou VPN pour « Virtual Private Network » en anglais) offre une solution intéressante pour répondre à cette attente. En effet, elle permet de communiquer de manière transparente en temps réel, de manière sécurisée entre des individus partageant un même domaine d'intérêt tout en utilisant l'infrastructure réseau Internet peu sûr mais bon marché. Les RPVs (VPN) sont fréquemment utilisés pour interconnecter deux réseaux locaux domestiques (appelés ci-après réseaux LAN, pour « Local Area Network » en anglais) afin de créer un réseau local virtuel composé de l'union des deux réseaux LAN originaux. Il existe de

nombreuses technologies pour mettre en œuvre un RPV. On trouve généralement ces technologies dans les équipements d'infrastructure des réseaux des opérateurs et les « passerelles Internet » pour le grand public. Ces technologies peuvent être également mises en œuvre sur un ordinateur à l'aide de logiciels spécifiques, par exemple, openVPN, SoftEther, L2TP, etc. Une configuration typique de RPV basé sur une technique de tunnellisation est illustrée sur la figure 1a (décrite en détail par la suite). Dans cet exemple, les têtes de tunnel (« Tunnel End Point » en anglais) ne sont pas intégrées aux passerelles. Le tunnel est établi entre deux têtes de tunnel, et chaque paquet (aussi appelé trame) envoyé à un équipement connecté au réseau LAN distant est encapsulé par la tête de tunnel locale, puis envoyé à la tête de tunnel distante qui va le désencapsuler et l'envoyer sur le réseau LAN distant. Les équipements de ces réseaux LAN sont virtuellement connectés à un même réseau LAN. Une communication entre deux équipements via le tunnel est appelée communication de bout en bout (« end-to-end communication » en anglais).

Pour communiquer de manière transparente et s'affranchir des adresses non routables, les RPVs utilisent une encapsulation particulière (appelée « tunnellisation », ou « tunneling » en anglais) qui crée ce que l'on appelle un tunnel. Cette opération consiste à encapsuler un protocole de niveau A (protocole embarqué) dans un protocole de niveau B (protocole de transport) grâce à un protocole d'encapsulation C, B étant un protocole de niveau supérieur au protocole de niveau A dans un modèle en couche tel le modèle OSI (pour « Open Systems Interconnection » en anglais) qui décrit les services offerts par chacune de ces couches et leurs interactions.

Dans la suite de la description, on considère, à titre d'exemple uniquement, les RPVs de niveau 2, c'est-à-dire comportant un tunnel de niveau 2 (tunnel de niveau 2 signifie que le protocole embarqué est un protocole de la couche 2 du modèle OSI).

2. ARRIÈRE-PLAN TECHNOLOGIQUE

Une opération d'encapsulation a pour conséquence de réduire la quantité d'information utilisateur pouvant être envoyée dans une trame émise sur un support physique donné. En effet, les longueurs des en-têtes du protocole embarqué et du protocole d'encapsulation viennent réduire d'autant la longueur des données utilisateurs dans la trame à transmettre.

En effet, bien qu'un datagramme puisse atteindre une longueur maximale de 64 ko, la plupart des interfaces de communication autorisent une taille limite de paquets qui est dépendante de la technologie de communication et qui est très inférieure à cette valeur.

Pour s'accommoder de la différence entre le MTU d'un réseau local dont est issu le datagramme et le MTU du tunnel, une solution consiste à fragmenter le datagramme. En d'autres termes, on découpe le datagramme en fragments de taille inférieure au MTU du tunnel lors de sa transition vers le tunnel (le tunnel présentant un MTU inférieur au MTU du réseau local).

Cette solution de l'art antérieur qui consiste à fragmenter un datagramme à l'entrée d'un tunnel et opérer un réassemblage à la sortie de ce tunnel présente un certain nombre d'inconvénients.

Tout d'abord, les opérations de fragmentation et de réassemblage génèrent une augmentation de la charge de l'unité de traitement (« CPU ») ainsi qu'une utilisation moins efficace de la mémoire. En effet, les traitements associés à la fragmentation ou au réassemblage augmentent l'utilisation de l'unité de traitement. Par ailleurs, la mémoire associée à un datagramme ne peut être libérée que lorsque tous les fragments associés à ce datagramme ont été reçus avec succès pour alors opérer une opération de réassemblage, ce qui entraîne une diminution de l'efficacité.

Un autre inconvénient majeur de cette technique connue réside dans le fait que les équipements d'infrastructure (par exemple les routeurs Internet) ne sont pas adaptés à ces opérations de fragmentation et de réassemblage, leur rôle étant de réaliser au plus vite une transition entre deux réseaux.

En outre, l'efficacité globale de la communication diminue car la perte d'un fragment nécessite la retransmission de l'ensemble des fragments, ce qui entraîne une augmentation de la probabilité de retransmission.

Les inventeurs ont également constaté que dans un environnement « IP v6 » (ou « Internet Protocol version 6 ») seul l'ordinateur source peut fragmenter le datagramme. Les équipements d'infrastructure tels que les routeurs Internet sur le trajet ne le peuvent pas.

Pour s'accommoder des différences de MTU et éviter la fragmentation dans les réseaux de type IP (pour « Internet Protocol » en anglais), il est proposé dans le document « IETF RFC1191, "Path MTU Discovery", J. Mogul from DECWRL, S. Deering from Standford University, November 1990 » une technique d'adaptation dynamique du MTU sur un chemin Internet arbitraire. Un chemin est défini par la combinaison des informations suivantes : une adresse source, une adresse destination et un type de service IP et accessoirement un niveau de sécurité.

Cette technique pour découvrir la valeur maximale du MTU sur un chemin appelé « PMTU » (pour « Path MTU » en anglais) est basée d'une part sur l'utilisation d'un bit du type « DF » (pour « Don't fragment » en anglais) contenu dans l'en-tête IP du datagramme et d'autre part sur l'utilisation d'un message d'erreur du type « ICMP » (pour « Internet Control Message Protocol » en anglais) retourné par les équipements d'infrastructure de type routeur lorsque la taille du message à faire suivre sur un sous-réseau excède le MTU de celui-ci.

Ainsi, une machine hôte générant un datagramme avec le bit DF à 1 se verra retourner un message d'erreur ICMP (type = 3, code =4) avec la valeur du MTU du sous-réseau suivant à atteindre, le datagramme étant quant à lui détruit. La machine hôte à l'origine du datagramme de taille trop grande peut alors adopter différentes stratégies : par exemple réduire la valeur du PMTU et donc des datagrammes suivants avec la valeur retournée dans le message d'erreur ICMP ou cesser de positionner le bit DF à 1 dans les datagrammes relatifs au PMTU.

Dans le document de brevet US 5,959,974, il est également proposé de sonder le chemin à considérer en utilisant un message de contrôle du type « ICMP Echo Request », ce qui permet d'évaluer la valeur du MTU sans perte de données utilisateur par les équipements d'infrastructure (par exemple les routeurs Internet).

On connaît plusieurs techniques permettant d'améliorer davantage l'efficacité (c'est-à-dire réduire davantage le temps de détermination du PMTU).

Une première technique connue consiste à mettre en œuvre un mécanisme de retour de message d'erreur ICMP en cas de datagramme trop grand et un mécanisme de détermination du PMTU sur le chemin restant à parcourir par le datagramme à l'origine

de cette erreur. Cette première technique est notamment présentée dans le document de brevet US2003/0188015.

Une deuxième technique, notamment présentée dans le document de brevet US2003/0185208, propose pour un environnement « IP v6 » une amélioration de la méthode de découverte d'un PMTU en émettant un message de découverte d'un PMTU dont l'en-tête IP contient la valeur courante du PMTU qui est mise à jour par les routeurs mettant en œuvre le mécanisme de découverte. La machine hôte qui réceptionne ce message de découverte extrait la valeur du PMTU à la réception pour le retourner à la machine hôte dont est issu ce message.

Dans le document « IETF internet draft, " Packetization Layer Path MTU Discovery " , M. Mathis, J. Heffner from PSC, September 25, 2006 », il est décrit une méthode de découverte robuste du MTU sur un chemin basée sur l'émission de messages de découverte à partir d'une valeur initiale du MTU puis en augmentant sa valeur jusqu'à l'échec de la transmission afin de déterminer la valeur limite du MTU pour ce chemin.

Ces techniques connues permettent dans le cas général de déterminer la valeur du MTU sur un chemin entre deux machines hôtes d'un réseau IP. Cependant la présence d'un tunnel sur le chemin reliant deux machines hôtes nécessite des opérations particulières des têtes de tunnel. En effet, l'encapsulation du datagramme original dans un tunnel a pour effet de masquer l'adresse de la véritable machine hôte dont est issu ce datagramme par l'adresse de la tête de tunnel. Ainsi, un routeur en cas de problème de taille de MTU retournera un message d'erreur ICMP non pas à la machine hôte source du datagramme mais à la tête de tunnel.

Pour résoudre les problèmes précités, il est traditionnellement envisagé de mettre en œuvre un mécanisme de fragmentation à l'entrée du tunnel ou un mécanisme de découverte du MTU dans le tunnel.

Cependant, un inconvénient majeur de cette solution connue réside dans le fait qu'elle ne permet pas de relayer les messages d'erreur ICMP pour des RPVs de niveau 2. En effet, dans le cas d'un RPV de niveau 2, compte tenu de la méthode d'encapsulation réalisée, les informations contenues dans le message d'erreur ICMP, issu du routeur ayant détecté une erreur, ne permettent pas la reconstruction d'un

message d'erreur ICMP en tête de tunnel. Il en résulte un délai d'attente qui peut atteindre la valeur d'épuisement du temporisateur du protocole de transport de la machine hôte entre le moment où la tête de tunnel est informée du nouveau MTU et sa prise en compte par la machine hôte. A titre d'exemple, pour le protocole TCP (pour « Transmission Control Protocol » en anglais, « protocole de contrôle de transmissions » en français), la valeur du temporisateur avant retransmission, qui dépend du temps d'aller retour sur le réseau (ou « RTT », pour « Round Trip Time » en anglais), est bornée entre 1 et 64 secondes.

3. OBJECTIFS DE L'INVENTION

L'invention, dans au moins un mode de réalisation, a notamment pour objectif de pallier ces différents inconvénients de l'état de la technique.

Plus précisément, dans au moins un mode de réalisation de l'invention, un objectif est de fournir une technique de notification à un dispositif source d'une taille limite de datagrammes transitant dans un tunnel, permettant d'éviter la fragmentation des datagrammes.

Au moins un mode de réalisation de l'invention a également pour objectif de fournir une telle technique qui soit simple à mettre en œuvre et peu coûteuse.

Un autre objectif d'au moins un mode de réalisation de l'invention est de fournir une telle technique pouvant être mise en œuvre dans les têtes de tunnel, et qui soit donc transparente pour les équipements source et destination.

Un objectif complémentaire d'au moins un mode de réalisation de l'invention est de fournir une telle technique qui soit notamment bien adaptée au cas d'un tunnel mettant en œuvre une sélection dynamique du protocole de transport.

4. EXPOSÉ DE L'INVENTION

Dans un mode de réalisation particulier de l'invention, il est proposé un procédé de notification à un dispositif source d'une taille limite de paquets de données destinées à être transmises dans un tunnel comprenant une pluralité de canaux de transmission, ledit dispositif source étant relié à une tête de tunnel via un réseau de communication, ledit procédé étant mis en oeuvre par ladite tête de tunnel formant un point d'entrée dudit tunnel.

Selon l'invention, le procédé de notification comprend les étapes suivantes, pour chaque flux de données destinées à être transmises dans ledit tunnel et provenant dudit dispositif source :

a) détection d'un événement déclencheur lié à un nouveau canal associé audit flux pour le transport dudit flux dans ledit tunnel ;

b) obtention d'une première taille limite de paquets associée à un canal précédent associé audit flux pour le transport dudit flux dans ledit tunnel ;

c) obtention d'une deuxième taille limite de paquets associée audit nouveau canal ;

d) détection d'un changement de taille limite de paquets, par comparaison de ladite première taille limite de paquets avec ladite deuxième taille limite de paquets ;

e) en cas de détection positive, transmission d'un message de changement de taille limite de paquets vers ledit dispositif source.

Le principe général de l'invention consiste donc à retourner un message de changement de taille limite de paquets vers le dispositif source de manière à ne pas mettre en œuvre un mécanisme de fragmentation à l'entrée du tunnel. Ainsi, l'invention permet de résoudre le problème du masquage d'adresse du dispositif source par le mécanisme d'encapsulation de la tête de tunnel et donc de notifier plus rapidement un changement de taille limite de paquets au dispositif source.

Selon l'invention, la transmission du message de changement de taille limite de paquets est conditionnée par la détection successive d'un événement déclencheur et d'un changement de taille limite de paquets, c'est-à-dire la diminution ou l'augmentation de taille limite de paquets.

De façon avantageuse, un premier événement déclencheur est un basculement de transmission, pour ledit flux, dudit canal précédent vers ledit nouveau canal, qui est un canal distinct dudit canal précédent.

On a par exemple un basculement d'un canal TCP (canal précédent) vers un canal UDP (nouveau canal). D'une part, ces canaux peuvent emprunter des chemins différents pour relier un premier réseau de communication local (LAN) à un second réseau de communication local (LAN) par un tunnel, ce qui fait que les tailles maximales de paquets autorisées associées à ces chemins diffèrent. De plus, le fait de basculer d'un canal précédent utilisant un premier protocole d'encapsulation et de

transport vers un nouveau canal utilisant un second protocole d'encapsulation et de transport engendre un changement de taille d'entête de paquet ajoutée par la tête de tunnel ce qui provoque un changement de taille maximale de paquets autorisée pour les flux subissant ce basculement de canal.

Avantageusement, le tunnel comprend des canaux de transmission réels et des canaux de transmission virtuels, un canal réel étant un canal dans lequel les flux transportés sont encapsulés par un protocole de transport unique. Le premier événement déclencheur est un basculement appartenant au groupe comprenant :

- un basculement depuis un premier canal réel vers un second canal réel ;
- un basculement depuis un canal réel vers un canal virtuel ;
- un basculement depuis un canal virtuel vers un canal réel ;
- un basculement depuis un premier canal virtuel vers un second canal virtuel.

Un canal virtuel associé à un flux donné pour le transport dudit flux dans ledit tunnel est défini par :

- deux canaux réels dudit tunnel, et
- un mécanisme de basculement progressif de l'un vers l'autre desdits canaux réels, permettant pour chaque paquet dudit flux donné de sélectionner de façon dynamique un canal effectif parmi lesdits canaux réels.

Le canal réel peut être par exemple un canal réel TCP ou un canal réel UDP.

Le canal virtuel peut être par exemple un canal virtuel TCP vers UDP ou un canal virtuel UDP vers TCP.

Le mécanisme de basculement progressif permet d'éviter de brutales variations de la quantité de donnée à transmettre sur un canal, qui auraient pour conséquences une détérioration de la transmission.

Par exemple, lors d'un basculement d'un canal UDP vers un canal TCP (c'est-à-dire d'un canal dont le protocole de transport est le protocole UDP vers un canal dont le protocole de transport est le protocole TCP), si l'on bascule immédiatement tous les paquets sur le canal TCP, sans prendre garde de respecter la fenêtre (de congestion) TCP du canal TCP, les paquets ne pouvant être transmis immédiatement vont être temporisés (bufferisés), créant une augmentation artificielle du RTT (« Round Trip Time », ou « temps d'aller-retour de bout en bout dans le réseau ») pour ces paquets, pouvant aller

jusqu'à une retransmission de certains paquets qui aurait des effets catastrophiques dans le cas de où le protocole embarqué est le protocole TCP. En clair, tous les bénéfices attendus d'un basculement du canal UDP vers le canal TCP seraient perdus, et l'on aurait juste fait empirer les choses en introduisant des perturbations artificielles.

De même, un basculement d'un canal TCP vers un canal UDP sans contrôle pourrait engorger le medium de transmission, car n'oublions pas que les différents canaux de transmission partagent le même accès physique à l'Internet. Dans le cas d'un basculement d'un canal TCP vers un canal UDP, des paquets temporisés sur le canal TCP et destinés à être transmis sur le canal TCP seraient fortement pénalisés par une augmentation rapide du débit sur le canal UDP. Il faut donc mettre en place un système progressif permettant de transférer l'utilisation de la bande passante par le canal TCP vers le canal UDP. Avec un tel système, les paquets temporisés sur le canal TCP ont le temps d'être « écoulés », si bien que lorsque la totalité des paquets sont transmis sur le nouveau canal (canal UDP), aucun paquet n'a été pénalisé.

Selon une caractéristique avantageuse, la taille limite de paquets associée à un canal virtuel est égale à la plus petite taille limite de paquets parmi les deux tailles limites de paquets associées aux deux canaux réels.

Avantageusement, un second événement déclencheur est une réception d'un message d'erreur indiquant un changement de taille limite de paquets pour un canal de transport du flux qui constitue à la fois le canal précédent et le nouveau canal, lesdites première et deuxième tailles limites de paquets étant respectivement une précédente et une nouvelle taille limite de paquets dudit canal de transmission.

Selon une caractéristique avantageuse, le procédé de notification comprend :

- une phase de configuration comprenant une première étape d'association d'une information de type de paquet, à une information de canal de transmission, à une information de taille limite de paquets, et à une liste d'identifiants de flux transmis par ledit canal de transmission et composés de paquets dudit type de paquet ;
- une phase de mise à jour, quand un changement de taille limite de paquets est détecté pour la transmission d'un type de paquet donné, de l'information de taille limite de paquets associée audit type de paquet.

Avantageusement, si ledit nouveau canal est un canal réel, la phase de mise à jour comprend une étape de mise à jour de l'information de taille limite de paquets associée à des types de paquets transmis par un canal virtuel défini par un couple de canaux réels comprenant ledit nouveau canal.

Selon un mode de réalisation particulier, la phase de configuration comprend en outre une deuxième étape d'association d'un identifiant de flux à une adresse de dispositif source. La phase de mise à jour comprend en outre les étapes suivantes, en cas de détection du second événement déclencheur :

- récupération d'un identifiant de flux dans le message d'erreur, ledit identifiant de flux étant inséré par la tête de tunnel dans les paquets dudit flux pendant une étape de transmission desdits paquets dans le tunnel ;
- récupération de l'adresse de dispositif source associée à l'identifiant de flux récupéré ;
- construction d'un message de changement de taille limite de paquets à partir de l'information de taille limite de paquets mise à jour ;
- transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

Dans une variante de réalisation, le procédé de notification comprend les étapes suivantes :

- récupération d'un ensemble d'identifiants de flux à partir d'un identifiant dudit nouveau canal ;

pour chaque identifiant de flux de l'ensemble récupérée :

- récupération de l'adresse de dispositif source associée audit identifiant de flux;
- construction d'un message de changement de taille limite de paquets à partir de ladite information de taille limite de paquets mise à jour ;
- transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

Dans un autre mode de réalisation, l'invention concerne un produit programme d'ordinateur téléchargeable depuis un réseau de communication et/ou enregistré sur un support lisible par ordinateur et/ou exécutable par un processeur, le programme comprenant des instructions de code de programme pour l'exécution des étapes du

procédé de notification tel que précédemment décrit, lorsque ledit programme est exécuté sur un ordinateur.

Dans un autre mode de réalisation, l'invention concerne un moyen de stockage lisible par un ordinateur, stockant un jeu d'instructions exécutables par ledit ordinateur pour mettre en œuvre le procédé de notification tel que précédemment décrit.

Dans un autre mode de réalisation, l'invention concerne une tête de tunnel destinée à mettre en œuvre un procédé de notification à un dispositif source d'une taille limite de paquets de données destinées à être transmises dans un tunnel comprenant une pluralité de canaux de transmission, ledit dispositif source étant relié à ladite tête de tunnel via un réseau de communication, ladite tête de tunnel comprenant des moyens de transmission dans ledit tunnel de flux de données provenant dudit dispositif source.

Selon l'invention, la tête de tunnel comprend :

- des moyens de détection d'un événement déclencheur lié à un nouveau canal associé à un flux donné pour le transport dudit flux donné dans ledit tunnel ;
- des moyens d'obtention d'une première taille limite de paquets associée à un canal précédent associé audit flux donné pour le transport dudit flux donné dans ledit tunnel ;
- des moyens d'obtention d'une deuxième taille limite de paquets associée audit nouveau canal ;
- des moyens de détection d'un changement de taille limite de paquets, par comparaison de ladite première taille limite de paquets avec ladite deuxième taille limite de paquets ;
- des moyens de transmission d'un message de changement de taille limite de paquets vers ledit dispositif source.

Les avantages des produit programme d'ordinateur, moyen de stockage et tête de tunnel sont sensiblement les mêmes que ceux du procédé de notification et ne sont donc pas repris ci-après.

De façon avantageuse, un premier événement déclencheur est un basculement de transmission, pour ledit flux donné, dudit canal précédent vers ledit nouveau canal, qui est un canal distinct dudit canal précédent.

Préférentiellement, le tunnel comprend des canaux de transmission réels et des canaux de transmission virtuels, un canal réel étant un canal dans lequel les flux transportés sont encapsulés par un protocole de transport unique. Le premier événement déclencheur est un basculement appartenant au groupe comprenant :

- un basculement depuis un premier canal réel vers un second canal réel ;
- un basculement depuis un canal réel vers un canal virtuel ;
- un basculement depuis un canal virtuel vers un canal réel ;
- un basculement depuis un premier canal virtuel vers un second canal virtuel ;

Un canal virtuel associé à un flux donné pour le transport dudit flux donné dans ledit tunnel est défini par :

- deux canaux réels dudit tunnel, et
- un mécanisme de basculement progressif de l'un vers l'autre desdits canaux réels, permettant pour chaque paquet dudit flux donné de sélectionner de façon dynamique un canal effectif parmi lesdits canaux réels.

Avantageusement, la taille limite de paquets associée à un canal virtuel est égale à la plus petite taille limite de paquets parmi les deux tailles limites de paquets associées aux deux canaux réels.

Selon un mode de réalisation préférentiel de l'invention, un second événement déclencheur est une réception d'un message d'erreur indiquant un changement de taille limite de paquets pour un canal de transport du flux donné qui constitue à la fois le canal précédent et le nouveau canal, lesdites première et deuxième tailles limites de paquets étant respectivement une précédente et une nouvelle taille limite de paquets dudit canal de transmission.

Avantageusement, la tête de tunnel comprend :

- des moyens de configuration comprenant des premiers moyens d'association d'une information de type de paquet, à une information de canal de transmission, à une information de taille limite de paquets, et à une liste d'identifiants de flux transmis par ledit canal de transmission et composés de paquets dudit type de paquet ;
- des moyens de mise à jour de l'information de taille limite de paquets associée audit type de paquet.

Préférentiellement, la tête de tunnel comprend des moyens de mise à jour de l'information de taille limite de paquets associée à des types de paquets transmis par un canal virtuel défini par un couple de canaux réels comprenant ledit nouveau canal.

Avantageusement, les moyens de configuration comprennent en outre des deuxièmes moyens d'association d'un identifiant de flux à une adresse de dispositif source. Les moyens de mise à jour comprennent :

- des moyens de récupération d'un identifiant de flux dans le message d'erreur, ledit identifiant de flux étant inséré par la tête de tunnel dans les paquets dudit flux pendant une étape de transmission desdits paquets dans le tunnel ;
- des moyens de récupération de l'adresse de dispositif source associée à l'identifiant de flux récupéré ;
- des moyens de construction d'un message de changement de taille limite de paquets à partir de l'information de taille limite de paquets mise à jour ;
- des moyens de transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

Préférentiellement, la tête de tunnel comprend :

- des moyens de récupération d'un ensemble d'identifiants de flux à partir d'un identifiant dudit nouveau canal ;
- des moyens de récupération de l'adresse de dispositif source associée audit identifiant de flux;
- des moyens de construction d'un message de changement de taille limite de paquets à partir de ladite information de taille limite de paquets mise à jour ;
- des moyens de transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

5. LISTE DES FIGURES

D'autres caractéristiques et avantages de modes de réalisation de l'invention apparaîtront à la lecture de la description suivante, donnée à titre d'exemple indicatif et non limitatif (tous les modes de réalisation de l'invention ne sont pas limités aux

caractéristiques et avantages des modes de réalisation décrits ci-après), et des dessins annexés, dans lesquels :

- la figure 1a illustre une configuration typique de réseau privé virtuel (VPN) mettant en œuvre un tunnel ;
- la figure 1b présente un diagramme de séquences illustrant un scénario relatif à l'émission d'un datagramme TCP de taille trop grande ;
- la figure 2a présente un exemple de modèle en couche classique d'une tête de tunnel dans laquelle peut être mis en œuvre le procédé selon l'invention ;
- la figure 2b présente un exemple de format d'encapsulation d'une trame Ethernet véhiculant un paquet tunnel de niveau 2 ;
- la figure 3 présente un format, selon un mode de réalisation particulier de l'invention, d'un datagramme IP ;
- la figure 4 présente un format, selon un mode de réalisation particulier de l'invention, d'un message d'erreur ICMP ;
- la figure 5 présente une table de flux de données selon un mode de réalisation particulier de l'invention ;
- la figure 6 présente une table des canaux de transport selon un mode de réalisation particulier de l'invention ;
- la figure 7 présente un organigramme d'un algorithme de génération d'un deuxième message d'erreur ICMP par la tête de tunnel ;
- la figure 8 présente un organigramme d'un algorithme de gestion du mode de transport courant et de fenêtres de transmission pour un type de paquet, selon un mode de réalisation particulier du procédé selon l'invention ;
- la figure 9a présente un organigramme d'un algorithme de gestion d'un mode transitoire « UDP-vers-TCP », selon un mode de réalisation particulier du procédé selon l'invention ;
- la figure 9b présente un organigramme d'un algorithme de gestion d'un mode transitoire « TCP-vers-UDP », selon un mode de réalisation particulier du procédé selon l'invention ;
- la figure 10 présente la structure d'un appareil de communication (tête de tunnel) selon un mode de réalisation particulier de l'invention ;

- la figure 11 présente un organigramme d'un algorithme de sortie, exécuté par la tête de tunnel qui transmet via le tunnel, selon un mode de réalisation particulier du procédé selon l'invention ;
- la figure 12 présente un organigramme d'un algorithme d'entrée, exécuté par la tête de tunnel qui reçoit via le tunnel, selon un mode de réalisation particulier du procédé selon l'invention ;
- la figure 13 présente un organigramme d'un algorithme de sélection d'un canal effectif (détail de l'étape 3 de la figure 11), selon un mode de réalisation particulier du procédé selon l'invention.

6. DESCRIPTION DÉTAILLÉE

Sur toutes les figures du présent document, les éléments et étapes identiques sont désignés par une même référence numérique.

La présente invention concerne donc une technique permettant de notifier à un dispositif source une taille limite de paquet associée à un canal de transport, pour éviter une fragmentation de datagrammes.

Le principe général de l'invention consiste à utiliser un identifiant de flux pour retrouver l'adresse du dispositif source à laquelle un message de changement de taille limite de paquets (comprenant une information relative à la taille limite de paquet associée au canal de transport) doit être transmis.

La **figure 1a** illustre une configuration typique de réseau privé virtuel (VPN) mettant en œuvre un tunnel 100 entre une tête de tunnel locale 101 et une tête de tunnel distante 102, à travers un réseau de communication 107 (Internet par exemple). Ce tunnel 100 interconnecte deux réseaux locaux : LAN A 103 et LAN B 104. Chacun des réseaux LAN 103 et 104 comporte un équipement d'accès Internet haut débit (passerelle domestique, ou « Home Gateway », pouvant intégrer un pare-feu (« Firewall »)) 105 et 106, des équipements de type PC 109 et 111, des serveurs 110 et 113 permettant le stockage et le partage des media numériques (de type audio, vidéo, photo), ainsi que des équipements de restitutions des médias numériques 108 et 112. Une tête de tunnel peut être intégrée dans un équipement audiovisuel comme un téléviseur numérique. Elle peut aussi être présente dans un équipement de type PC sous la forme d'un programme réalisant les fonctions associées à celle-ci.

Une fois que le tunnel 100 est établi, les équipements 108, 109, et 110, connectés au réseau LAN A 103, sont capables de communiquer avec les équipements 111, 112 et 113, connectés au réseau LAN B 104. Par exemple, le client 108 connecté au réseau LAN A 103 peut communiquer avec le serveur 113 connecté au réseau LAN B 104.

Dans cette figure 1a, on a représenté un réseau de communication simple avec un seul tunnel, mais il est bien entendu qu'une même tête de tunnel peut être amenée à gérer plusieurs tunnels (à destination d'autant de têtes de tunnel) pour interconnecter un premier réseau LAN à plusieurs autres réseaux LAN. En outre, dans un souci de simplification, n'ont pas été représentés les équipements d'infrastructure dans le réseau Internet tels que les routeurs Internet.

En relation avec la **figure 1b**, nous allons maintenant décrire un diagramme de séquences illustrant un scénario relatif à l'émission d'un datagramme TCP de taille trop grande transitant dans un tunnel de niveau 2 classique. Pour une meilleure compréhension de ce diagramme, on a repris dans la partie haute de cette figure 1b, les éléments importants de la figure 1a à laquelle on a ajouté deux routeurs Internet 114 et 115.

Dans une première phase 120, un datagramme 150, par exemple, de longueur 1500 octets (« bytes » en anglais) est issu du serveur d'application 110 du réseau LAN A 103 à destination du terminal 112 du réseau LAN B 104. La tête de tunnel compare la taille de ce datagramme 150 et la valeur du MTU du tunnel 100. Bien que l'interface de communication de la tête de tunnel 101 soit la même que celle du serveur 110, compte tenu du mécanisme d'encapsulation réalisé par la tête de tunnel, la valeur du MTU du tunnel 100 est, dans cet exemple, de 1420 octets. La taille du datagramme 150 de longueur 1500 octets est donc supérieure au MTU du tunnel 100.

Il s'ensuit alors une deuxième phase 121, dans laquelle la tête de tunnel rejette le datagramme 150 et émet un message d'erreur ICMP 151 (code=3, type=4) à destination du serveur 110 (à l'origine du datagramme de taille trop grande), en lui précisant la valeur du MTU du tunnel (soit MTU = 1420 octets) prenant en compte les éléments nécessaires à l'encapsulation et à la signalisation dans le tunnel ajoutés par la tête de tunnel. Sur réception de ce message, le serveur 110 peut ajuster la longueur de ses datagrammes à cette nouvelle valeur de MTU. Ainsi dans une troisième phase 122 il

effectue une nouvelle transmission 152 en tenant compte de la nouvelle valeur de MTU. La taille de ce datagramme 152 répond ainsi à la limite du MTU imposée par le tunnel. La tête de tunnel réalise donc son opération d'encapsulation et transfère le nouveau datagramme 153 ainsi formé dans le tunnel. Cependant, dans cette quatrième phase, alors que le datagramme 153 transite dans le tunnel 110 à travers Internet 107, il entre dans un premier routeur Internet 114 pour être routé vers un deuxième routeur Internet 115 sur le chemin de sa destination. Ce premier routeur compare à son tour la taille du datagramme 153 avec la valeur du MTU sur le chemin menant vers le deuxième routeur. Dans cet exemple, la taille du datagramme 153 est supérieure au MTU du chemin menant vers le deuxième routeur Internet 115. Le routeur Internet 114 exécute alors une cinquième phase dans laquelle il rejette ce datagramme 153 et émet à son tour un message d'erreur ICMP 154 (code=3, type=4), en précisant la valeur du MTU pour le chemin à destination du deuxième routeur 115. Du fait de l'encapsulation, ce message d'erreur ICMP 154 est émis à destination de la tête de tunnel 101 dont est issu ce datagramme 153 comportant lui-même le datagramme original 152 encapsulé. Sur réception de ce message d'erreur ICMP 154, la tête de tunnel 101 met à jour le MTU du tunnel 100 (soit MTU = 920 octets) en tenant compte de la nouvelle valeur du MTU contenue dans le message d'erreur ICMP et l'opération d'encapsulation du tunnel 100.

Dans une sixième phase 125, le serveur 110, ne recevant pas de message d'acquittement de son datagramme 152 après une temporisation prédéterminée (par exemple le délai d'attente de retransmission « RTO » (pour « Retransmission Timeout » en anglais) du protocole TCP), fait une nouvelle tentative d'émission 155 de son datagramme 152 de longueur 1420 octets. A l'entrée du tunnel, la taille de ce datagramme 155 est à nouveau comparée à la nouvelle valeur du MTU du tunnel 100 (soit MTU = 920 octets). La taille du datagramme 155 est 1420 octets et est donc supérieure au nouveau MTU du tunnel 100. Ensuite, dans une septième phase 126, le datagramme 155 est rejeté et un message d'erreur ICMP 156 (code=3, type=4) est retourné au serveur 110 avec la nouvelle valeur de MTU du tunnel 100 (soit MTU = 920 octets). Enfin, dans une huitième phase 127 (comme dans la troisième phase 122), suite à la réception du message d'erreur ICMP 156, le serveur émet un nouveau datagramme 157 de taille inférieure (soit 920 octets) au datagramme 155 rejeté. Celui-ci est

encapsulé dans un nouveau datagramme 158 par la tête de tunnel pour être transféré dans le tunnel à travers le réseau de communication 107. Le datagramme 157 sera donc reçu avec succès et pourra être acquitté conformément au protocole de transport utilisé par le serveur.

Comme on peut le constater à la lecture de ce diagramme de séquences, l'émission d'un datagramme 150 de taille trop importante provoque plusieurs retransmissions 152 et 155 ainsi qu'un délai significatif entre la transmission du datagramme 150 initial (ayant échoué) et la transmission finale du datagramme 157 (résultant des opérations décrites ci-dessus). Ce délai peut être considérable lorsque la sixième phase 125 résulte de l'échéance d'une temporisation avant retransmission, comme le délai d'attente de retransmission « RTO » du protocole TCP qui peut être compris entre 1 et 64 secondes en fonction du temps d'aller-retour sur le réseau (ou « Round Trip Time » en anglais).

En relation avec la **figure 2a**, nous allons désormais décrire le cheminement d'une trame Ethernet issue d'un des équipements 108, 109, 110 (connectés au réseau LAN A 103) et qui va entrer dans le tunnel 100. Pour ce faire, on va utiliser un modèle en couche décrivant les couches de protocoles nécessaires à la mise en œuvre de ce tunnel 100. Dans ce modèle ne sont pas représentés les éléments de protocole nécessaires aux fonctions autres que la mise en œuvre du tunnel. Par exemple, ne sont pas représentés les éléments de protocole associés à une architecture UPnP lorsqu'une tête de tunnel 101 est intégrée dans un équipement conforme à la norme UPnP.

La tête de tunnel 101 comporte une interface physique Ethernet 208 qui remet les trames Ethernet issues des équipements 108, 109, 110 à la couche liaison 207 pour routage : vers la couche réseau 206, pour les trames Ethernet à destination de l'équipement comportant la tête de tunnel, ou vers la couche de pont (« bridge ») 209, pour les trames Ethernet à destination du réseau LAN B 104. La couche de pont 209 réalise les opérations classiques d'un pont Ethernet telles que le filtrage des trames Ethernet et le relais de ces trames vers le ou les port(s) Ethernet de sortie approprié(s). Sur le pont sont attachées une interface Ethernet 207 et au moins une interface virtuelle 210 simulant un contrôleur Ethernet. Une interface virtuelle 210 est créée pour chaque tunnel instancié par l'application 200 à qui elle remet les trames Ethernet qui doivent

transiter sur les tunnels respectivement établis. D'une manière générale, le protocole d'encapsulation du tunnel représenté par l'application 200 réalise les opérations nécessaires à la mise en œuvre de chaque tunnel, parmi lesquelles on trouvera notamment la configuration, le filtrage et l'encapsulation (par la suite, on entendra par encapsulation la formation d'un paquet tunnel) et l'extraction d'une trame.

Les trames reçues de l'interface virtuelle 210, après traitement par l'application 200, sont remises sous la forme de paquets à travers une interface applicative (« socket » en anglais) 201 à un protocole de transport fiable TCP 203 ou non fiable UDP 205, respectivement sécurisés par les protocoles SSL 202 et DTLS 204. Après traitement par un protocole de transport pour former le paquet tunnel 250 (cf. figure 2a), on passe celui-ci à la couche réseau 206. Le datagramme IP ainsi formé avec le paquet tunnel peut maintenant être transmis sur le réseau LAN à travers les couches liaison 207 et physique 208.

La réception d'une trame en provenance du tunnel 100 suivra dans la tête de tunnel le cheminement inverse à celui présenté ci-dessus.

La **figure 2b** présente un exemple de format d'encapsulation d'une trame Ethernet (référencée 260), transitant par exemple sur le réseau LAN A 103 de la figure 1a entre la tête de tunnel 101 et la passerelle domestique 105 (destiné à être émis sur Internet ou reçu d'Internet), et qui comporte : un champ d'en-tête Ethernet 261, un premier datagramme IP véhiculant lui-même un paquet tunnel de niveau 2 (référencé 250), et un champ FCS (« Frame Check Sequence », ou « séquence de contrôle de trame »).

Le paquet tunnel 250 comporte quatre parties :

- un champ d'en-tête du protocole de transport 251 (à savoir TCP ou UDP dans cet exemple),
- un champ d'en-tête du protocole d'encapsulation 252 (à savoir L2TP ou TLS dans cet exemple, qui sont décrits notamment dans les documents suivants : « IETF RFC3931, "Layer two tunneling protocol – version 3 (L2TPv3)", J. Lau and all, March 2005 » et « IETF RFC2246, "The TLS Protocol Version 1.0" »,

- un champ d'en-tête du protocole embarqué 253 (à savoir Ethernet dans cet exemple), et enfin
- un champ de données utilisateurs 254, qui lui-même comporte un second datagramme IP complet si aucune fragmentation n'a été opérée lors de son transit depuis l'équipement source.

On décrit maintenant, en relation avec la **figure 3**, un format, selon un mode de réalisation particulier de l'invention, d'un datagramme IP (version 4) 300 qui comporte : un champ d'en-tête IP fixe 350 (par exemple de 20 octets), un champ d'en-tête optionnel 309 (peu usité, de longueur variable), et un champ de données 310.

Le champ d'en-tête IP fixe 350 comporte :

- un champ d'identification 302 (« Identification »),
- un champ d'adresse source 307 qui identifie l'origine du datagramme,
- un champ d'adresse de destination 308 qui identifie le destinataire du datagramme,
- un champ de type de service (ou « TOS » pour « Type Of Service ») 301 qui précise le type de service demandé,
- un champ de protocole (« Protocol ») 306 qui permet le démultiplexage vers l'application appropriée en réception, et
- un champ de temps de vie (ou « TTL » pour « Time To Live ») 312 qui limite le nombre de routeurs que le datagramme peut traverser.

Le champ d'en-tête IP fixe 350 comporte en outre un ensemble de champs non référencés car non utilisés de manière spécifique dans la présente invention.

Selon l'invention, on interdit la fragmentation du datagramme entrant dans le tunnel, en positionnant un bit de non fragmentation (ou « DF » pour « Don't Fragment ») 311 dans le champ « Flags » 303 (noté F). En outre, on recopie dans l'en-tête IP de la trame tunnel 260 les champs « indentification » 302, « TOS » 301, et « TTL » 312 décrémenté du datagramme entrant dans le tunnel.

Dans un premier mode de réalisation particulier de l'invention, il est possible d'envisager d'utiliser l'option STREAMID du champ d'en-tête optionnel 309 afin d'identifier l'origine du datagramme envoyé dans le tunnel. Dans une variante, compte tenu du fait que le tunnel ne réalise pas d'opération de fragmentation, il est également

possible d'envisager d'utiliser le champ « Fragment Offset » 304 (noté FO). Cette variant permet un traitement plus rapide du datagramme par les routeurs du réseau du fait de l'absence du champ d'en-tête optionnel 309 qui simplifie et accélère le traitement de routage.

La **figure 4** présente un format, selon un mode de réalisation particulier de l'invention, d'un message d'erreur ICMP 410 retourné par un routeur Internet sur réception d'un datagramme IP 300 (décrit précédemment en relation avec la figure 3) n'autorisant pas la fragmentation (le bit de non fragmentation « DF » est à 1) et de taille trop grande. Dans ce mode de réalisation particulier, le routeur Internet 114 (cf. figure 1b) va générer un message d'erreur ICMP 410 comportant un champ d'en-tête ICMP 400 et trois recopies de champ du datagramme IP 300 (ayant amené la génération du message d'erreur ICMP 410), à savoir : les recopies du champ d'en-tête IP fixe 350, du champ d'en-tête optionnel (s'il existe) 309, et des 8 premiers octets 411 du champ de données 310 (cf. figure 3).

Plus précisément, le champ d'en-tête ICMP 400 comprend un premier champ « Type » 401 indiquant que la destination n'est pas accessible (Type = 3), un deuxième champ « Code » 402 indiquant que la fragmentation du datagramme 300 est nécessaire mais interdite du fait que le bit de non fragmentation « DF » est à 1 dans le champ « Flags » 303 du champ d'en-tête IP fixe 350 (Code = 4), un troisième champ « Prochain MTU » (ou « Next Hop MTU ») 404 précisant la valeur du MTU du sous-réseau par lequel doit transiter le datagramme 300 par la suite, et un quatrième champ « checksum » 403 relatif à un code de contrôle.

Dans un premier cas, si le datagramme IP 300 (ayant amené la génération du message d'erreur ICMP 410) encapsule un paquet UDP, alors les 8 premiers octets 411 du champ de données 310 comprennent les champs port source et port destination du paquet UDP transporté, ainsi que la longueur de ce paquet UDP et le code de contrôle qui lui est associé.

Dans un deuxième cas, si le datagramme IP 300 (ayant amené la génération du message d'erreur ICMP 410) encapsule un paquet TCP, alors les 8 premiers octets 411 du champ de données 310 comprennent les champs port source et port destination du paquet TCP transporté, ainsi que le numéro de séquence qui lui est associé.

Le rôle et les valeurs possibles de chacun des champs d'un datagramme IP, ICMP, UDP et TCP sont bien connus de l'Homme du Métier et sont notamment décrits en détail dans le document « TCP/IP illustrated, Volumes 1, 2 et 3 ", Stevens, Wright, Addison-Wesley, 1994, 1995 et 1996 ».

La **figure 5** décrit une première base d'information 500 sous la forme d'une table de flux de données qui est mise à jour notamment lors de chaque transfert d'un paquet vers le tunnel 100, chaque paquet ainsi en transit appartenant à un flux de données 510 dans cette table 500. Cette base d'information, vise à conserver les informations essentielles à la construction d'un message d'erreur ICMP 410 à destination d'une ou plusieurs sources de datagramme IP.

Selon l'invention, un deuxième message d'erreur ICMP est généré par la tête de tunnel sur réception d'un premier message d'erreur ICMP provenant du tunnel 100 ou généré suite à un changement de canal de transport du tunnel, en raison de modifications des conditions de transport sur le réseau.

Les informations contenues dans cette table d'information des flux de données 500 sont extraites des datagrammes IP entrant dans le tunnel 100. On y trouve pour chaque flux de données 510, l'adresse source 502 « SA » (correspondant au champ d'adresse source 307 du datagramme IP), l'adresse de destination 503 « DA » (correspondant au champ de destination 308 du datagramme IP), le protocole véhiculé 506 « Protocol » (correspondant au champ de protocole 306 du datagramme IP), le port source 504 « S Port », et le port de destination 505 « D Port » contenu dans les 8 premiers octets 411 du champ de données 310 du datagramme IP.

En outre, on y trouve un champ d'identification 501 « FID » des flux de données 510 qui permet d'identifier de manière unique chaque élément entré (chaque flux) dans cette table d'information 500, et un champ « PID » 550 d'identification du canal de transport dans la table de canaux de transport 600 (décrit ci-après) du tunnel, utilisée pour le transit des datagrammes appartenant au flux de données.

Le changement de canal de transport précité repose sur la mise en œuvre d'un mécanisme de sélection dynamique de protocole de transport. Le principe de ce mécanisme de sélection dynamique consiste à sélectionner, pour chaque paquet de données à transmettre via le tunnel, le meilleur canal (caractérisé typiquement par son

protocole de transport) à utiliser. La sélection est basée sur le type des données à transmettre (protocole des données utiles contenues dans ce paquet, type d'application, etc.), mais aussi sur les conditions de transmission sur le réseau (entre les deux têtes de tunnel).

On décrit ci-après, en relation avec les figures 11 à 13, des exemples d'algorithmes de sortie, d'entrée, et de sélection mis en oeuvre par le mécanisme de sélection dynamique précité.

On décrit maintenant, en relation avec la **figure 11**, un exemple d'algorithme de sortie de LAN, exécuté par la tête de tunnel (par exemple celle référencée 101 sur la figure 1a) qui transmet via le tunnel 100. Cette figure explique le traitement global de données à envoyer à l'autre tête de tunnel 102 à travers le tunnel 100.

Dans l'étape 1, on écoute sur une interface réseau, et on capture des paquets de données IP ou Ethernet, destinés à au moins un équipement connecté au réseau LAN B 104. Ceci peut être réalisé en utilisant un pont (« bridge » en anglais), et avec quelques dispositifs réseaux virtuels tels que TUN/TAP ajoutés au pont.

Dans l'étape 2, on décide si le paquet est autorisé ou non à être transmis vers le LAN B. Par exemple, un paquet reçu du LAN B ne sera pas retransmis en direction de ce même LAN. Ceci peut être réalisé par exemple en comparant l'adresse source MAC Ethernet à une liste prédéfinie d'adresses MAC autorisées.

Dans l'étape 3, on sélectionne le canal le plus approprié à utiliser pour transmettre ce paquet de données vers le réseau LAN B 104. Cette étape 3 est détaillée ci-après en relation avec les autres figures.

Dans l'étape 4 (optionnelle), un chiffrement des données peut être effectué, pour garantir le secret des données utilisateur. Cette étape peut être réalisée en utilisant un algorithme de chiffrement bien connu, comme l'algorithme AES (« Advanced Encryption Standard ») par exemple.

Dans l'étape 5, basée sur le résultat de l'étape 3, le paquet reçu est encapsulé avec un protocole d'encapsulation (aussi appelé protocole de tunnellisation) associé au canal sélectionné à l'étape 3. Ce protocole d'encapsulation ajoute des informations spécifiques (en-tête), et peut optionnellement ajouter des données supplémentaires pour fournir des caractéristiques spécifiques aux fonctionnalités du tunnel (comme par

exemple un mécanisme de « keep-alive » (« maintien ouvert ») permettant aux deux têtes de tunnel de savoir si le canal est toujours « ouvert », c'est-à-dire si la transmission est toujours possible). Ces fonctionnalités supplémentaires peuvent être dépendantes du canal. Par exemple, dans le cas d'un canal caractérisé par son protocole de transport, il peut être utile d'ajouter des données supplémentaires pour mesurer le RTT (« Round Trip Time », ou « temps d'aller-retour de bout en bout dans le réseau ») d'un canal UDP qui de manière classique ne fournit pas de mécanisme de mesure de RTT. Ceci peut être fait en ajoutant une requête pour réponse immédiate (incluant un identifiant) dans les données d'encapsulation. Quand la tête de tunnel distante reçoit une telle requête, elle répond immédiatement. A la réception de la réponse, la tête de tunnel locale peut alors déterminer le RTT. Un tel mécanisme n'a bien sûr pas besoin d'être mis en œuvre sur un canal qui met déjà en œuvre un mécanisme d'évaluation du RTT (cas par exemple d'un canal basé sur TCP) (voir le document : " TCP/IP illustrated, Volumes 1, 2 et 3 ", Stevens, Wright, Addison-Wesley, 1994, 1995 et 1996).

Dans l'étape 6, on transmet le paquet résultant de l'encapsulation sur le canal sélectionné dans l'étape 3. Ceci peut être réalisé en écrivant les données sur une interface de connexion (« socket ») configurée pour envoyer des paquets sur le tunnel. Après cette étape, le paquet aura finalement la forme de celui référencé 250 sur la figure 2b. Cette étape permet aussi de mettre à jour les statistiques de canal (retransmission, type des données transmises, etc.).

On décrit maintenant, en relation avec la **figure 12**, un exemple d'algorithme d'entrée sur un LAN, exécuté par la tête de tunnel (par exemple celle référencée 102 sur la figure 1a) qui reçoit via le tunnel 100. Cette figure explique le traitement global de données provenant de l'autre tête de tunnel 101 et reçues à travers le tunnel 100.

Dans l'étape 7, on écoute sur chaque interface de connexion (« socket ») spécifique (correspondant à chaque canal), pour recevoir des paquets.

Dans l'étape 8, on met à jour les informations relatives à la qualité réseau (retransmission, RTT, PER, congestion, etc.) du canal sur lequel on reçoit.

Dans l'étape 9, on déchiffre les données utiles (si l'étape 4 de la figure 11 a été mise en œuvre) en utilisant un algorithme de déchiffrement et des clés associées, compatibles avec ceux utilisés à l'étape 4.

Dans l'étape 10, on désencapsule le paquet de données, pour retrouver le paquet de données d'origine (préalablement capturé sur le réseau LAN A 103 par la tête de tunnel 101). Dans cette étape, on traite aussi les éventuelles données supplémentaires associées à des mécanismes supplémentaires optionnels (voir la description de l'étape 5).

Dans l'étape 11, on décide si le paquet résultant de la désencapsulation est autorisé ou non. Par exemple, un paquet dont le déchiffrement ou la désencapsulation ne donne pas de résultat satisfaisant ne sera pas autorisé à être transmis sur le LAN B afin de ne pas perturber le fonctionnement des équipements connectés sur ce LAN. Ceci peut aussi être réalisé par exemple en comparant l'adresse source MAC Ethernet à une liste prédéfinie d'adresses MAC autorisées.

Dans l'étape 12, on met à jour les statistiques de canal (bande passante, type des données transmises, etc.).

Dans l'étape 13, on envoie le paquet résultant de la désencapsulation sur le réseau LAN B 104. Ceci peut être réalisé en utilisant un dispositif réseau virtuel tel que TUN/TAP.

On décrit désormais, en relation avec la **figure 13**, un exemple d'algorithme de sélection d'un canal effectif (détail de l'étape 3 de la figure 11).

Le paquet reçu de l'étape 2 (cf. figure 11) est analysé dans l'étape 31, afin de déterminer s'il s'agit d'un paquet IP ou non (car dans ce mode de réalisation on considère uniquement les paquets IP). Ceci est réalisé en analysant le contenu du paquet (en-tête LLC…). S'il ne s'agit pas d'un paquet IP, on passe à l'étape 37 de sélection d'un canal par défaut. Ce canal par défaut peut être déterminé par l'utilisateur, et est par exemple un canal TCP. S'il s'agit d'un paquet IP, on passe à l'étape 32.

Dans l'étape 32, le paquet est classifié (on extrait toutes les informations concernant le paquet qui seront utilisées ultérieurement pour sélectionner le meilleur canal). Typiquement, pour déterminer le type du paquet (résultat de la classification) en

fonction de ses données utiles, on détermine le protocole de transport des données utiles (protocole de transport sur IP), cette information étant codée dans les 8 bits réservés de l'en-tête IP. Dans la suite de la description, on utilisera le protocole de transport des données utiles comme identifiant de type de paquets (TCP, UDP, SCTP, DCCP, ...)

Dans l'étape 33, on détermine si le type de paquet déterminé à l'étape 32 est géré par l'étape 35 ci-après. Dans la négative, on passe à l'étape 37. Dans l'affirmative, on passe à l'étape 34.

Dans l'étape 34, on détermine le QoE (« Quality Of the Experiment », ou « qualité de l'essai »). Pour cela, on retrouve toutes les données concernant la qualité du réseau (congestion, PER, bande passante, RTT, taux de retransmission, etc.). Toutes ces données sont évaluées à chaque fois qu'un paquet est envoyé ou reçu par le tunnel (étapes 8, 12, 384 et 387).

Dans l'étape 35, on détermine un canal préféré (et donc un protocole de transport préféré) à utiliser pour transmettre dans les meilleures conditions les données utiles au réseau LAN distant. Un canal peut être caractérisé uniquement par son protocole de transport, mais d'autres caractéristiques peuvent être utilisées telles que le paramètre TOS (« Type Of Service », ou « type de service »).

Dans l'étape 36, on détermine un mode de transport pour le type du paquet en cours de traitement (appelé ci-après « paquet traité »). Le mode de transport correspond à la manière de gérer la transmission d'un type de paquet donné. Ce mode peut être stable (TCP ou UDP, par exemple) ou transitoire (TCP-vers-UDP ou UDP-vers-TCP, par exemple).

Une mise en œuvre possible de cette étape 36 est illustrée en figures 8, 9a et 9b. Dans cette mise en œuvre, on considère le cas de canaux caractérisés par leur protocole de transport. En fonction du protocole de transport préféré et du mode de transport courant pour le type du paquet traité, on gère à l'étape 36 l'évolution du mode de transport (parmi quatre modes possibles : deux modes stables, TCP et UDP, et deux modes transitoires, TCP-vers-UDP, et UDP-vers-TCP) pour chaque type de paquet (type déterminé à l'étape 32). Par exemple, si on détermine à l'étape 35 que pour un paquet de type UDP, aussi appelé paquet UDP (c'est-à-dire un paquet dont UDP est le

protocole des données utiles du protocole embarqué), le protocole de transport préféré est TCP (c'est-à-dire le canal préféré est le canal TCP), mais que le mode de transport courant pour les paquets UDP est le mode UDP, alors dans l'étape 36 on entre dans le mode transitoire UDP-vers-TCP, et le canal de transmission effectif pour le paquet traité peut être soit le canal préféré TCP, soit le canal UDP (voir ci-après la description détaillée des figures 8, 9a et 9b).

Dans l'étape 38, on sélectionne le canal effectif, en fonction du canal préféré (cf étape 35), du mode de transport courant et de paramètres de fenêtres de transmission (voir description ci-après) pour le type du paquet traité. C'est un mécanisme de sélection, permettant de passer progressivement d'un mode de transport à un autre, pour un type de paquet donné. Ce mécanisme est important pour éviter une congestion du tunnel qui serait liée au basculement de la transmission d'un flux de données d'un canal de transmission à un autre.

Par exemple, si le protocole de transport préféré pour un type de paquet commute de UDP à TCP, il n'est pas possible d'envoyer directement tous les paquets de ce type sur le canal TCP, car l'augmentation brutale du nombre de paquets à transmettre sur ce canal TCP va amener un dépassement de la fenêtre de congestion TCP. En conséquence, les paquets seront temporisés par la pile TCP même si la bande passante réellement disponible est assez grande. Ces paquets temporisés vont augmenter le RTT mesuré de la communication de bout en bout, et peuvent générer une retransmission inutile dans le cas de paquets TCP (expiration d'une temporisation de retransmission, appelée RTO (« Retransmission Time Out ») dans le cas TCP).

Dans le cas d'une commutation du mode de transport TCP au mode de transport UDP, il faut faire attention à une augmentation brutale de la taille du canal UDP, qui peut brutalement ralentir la transmission TCP, générant également des troubles.

La **figure 6** décrit une deuxième base d'information 600 sous la forme d'une table des canaux de transport d'un ou plusieurs tunnels 100. Chaque canal 610 est identifié de manière unique par son identifiant « PID » 650 et est notamment caractérisé par l'adresse IP de la tête de tunnel distante 601 « TDA », située à l'extrémité du tunnel (c'est-à-dire à la sortie du tunnel), et par son mode de transport 602 (« Carrier mode »).

En effet, une même tête de tunnel peut ouvrir plusieurs tunnels à destination de têtes de tunnel distinctes utilisant chacune un ou plusieurs canaux.

Dans une variante d'implémentation, il est possible d'envisager d'ajouter les numéros de ports source et destination du mode de transport 602. Ceci aura pour avantage de pouvoir gérer de manière distincte plusieurs canaux utilisant le même mode de transport 602, mais avec, par exemple, des types de services 310 différents pour les datagrammes transitant dans chacun de ces canaux.

Dans un mode de réalisation particulier, correspondant au cas où on gère des basculements de transmission entre des canaux TCP et UDP, le mode de transport 602 peut prendre les quatre valeurs suivantes :

- un mode TCP utilisant le canal réel TCP ;
- un mode UDP utilisant le canal réel UDP ;
- un mode TCP vers UDP (aussi appelé « canal virtuel TCP vers UDP »), utilisant les canaux réels TCP et UDP. Dans ce mode transitoire, pour, on sélectionne de façon dynamique un canal effectif parmi les canaux réels TCP et UDP au moyen d'un mécanisme de basculement progressif (décrit en relation avec la figure 9b) ;
- un mode UDP vers TCP (aussi appelé « canal virtuel UDP vers TCP »), utilisant les canaux réels UDP et TCP. Dans ce mode transitoire, pour, on sélectionne de façon dynamique un canal effectif parmi les canaux réels UDP et TCP au moyen d'un mécanisme de basculement progressif (décrit en relation avec la figure 9a) ;

Comme illustré sur la figure 6, pour chaque canal 610, on dispose d'informations complémentaires, à savoir : la taille maximale « MTU » 605 des datagrammes pouvant transiter dans le canal, et un identifiant « FIDs » 606 d'une liste 607 des identifiants 501 des flux de données 510 (de la table de flux de données 500).

On présente maintenant, en relation avec la **figure 7**, un algorithme de génération d'un deuxième message d'erreur ICMP par la tête de tunnel suite à la réception d'un premier message d'erreur ICMP 410 issu d'un tunnel tel que défini à la figure 4.

Dans une première étape 701, l'application tunnel 200 (cf. figure 2a) analyse et extrait des informations du premier message d'erreur ICMP 410.

Dans l'étape 702, on prépare un deuxième message d'erreur ICMP (du même type que le premier message d'erreur ICMP 410) dans lequel on recopie les valeurs des champs de type de service « TOS » 301, de temps de vie « TTL » 312, d'identification « Identification » 302, et la valeur du champ « Next Hop MTU » 404 contenues dans le premier message d'erreur ICMP 410 reçu.

Dans l'étape 703, on détermine l'identifiant 650 « PID » du canal 610 de transport (cf. figure 6) dont est issu le premier message d'erreur ICMP 410 à l'aide des informations des champs de destination 308 « Adresse Destination » et de protocole 306 « Protocol », contenues dans le champ d'en-tête IP fixe 350 retourné dans le message d'erreur ICMP 410, que l'on compare respectivement aux informations contenues dans les champs d'adresse IP de la tête de tunnel distante « TDA » 601 et de son mode de transport 602 de la base d'information des canaux de transport 600.

Dans une variante de réalisation, pour identifier de manière unique le canal il est possible d'envisager d'utiliser en outre les informations contenues dans les 8 premiers octets 411 du premier message d'erreur ICMP 410 reçu (lorsque plusieurs canaux utilisant un même type de protocole de transport sont présents dans le tunnel considéré). En effet, ces 8 premiers octets 411 peuvent contenir les ports source et destination du mode de transport 602, utilisés par le datagramme IP ayant provoqué le premier message d'erreur ICMP 410.

Dans l'étape 704, on détermine la valeur du « MTU » (taille limite des données utiles pouvant être transmises dans un paquet) du canal 610 en lisant la valeur contenue dans le champ « Next Hop MTU » 404 du message d'erreur ICMP 410 en cours d'analyse, décrémenté de : (en référence à la figure 2b)

- la taille d'un champ d'en-tête IP,
- la taille du champ d'en-tête du mode de transport 251,
- la taille du champ d'en-tête du protocole d'encapsulation 252, et
- la taille du champ d'en-tête du protocole embarqué 253.

Dans l'étape 705, on met à jour la valeur du « MTU » 605 du canal 610 (de la base d'information des canaux 600), dont l'identifiant 650 « PID » a été déterminé à l'étape 703, avec la valeur du « MTU » du canal déterminée à l'étape 704.

Dans l'étape 706, on extrait l'identifiant du flux de données 606 « FID » contenu dans le champ d'en-tête optionnel 309 du datagramme IP (ayant provoqué le premier message d'erreur ICMP 410) en cours d'analyse.

Dans l'étape 707, on achève la préparation du deuxième message d'erreur ICMP à l'aide des champs 502 à 506 (cf. figure 5) préalablement enregistrés pour le flux de données « FID » déterminé à l'étape 706. En d'autres termes, les champs 502 à 506 sont lus dans la table de flux de données 500, grâce à l'identifiant du flux de données « FID » obtenu à l'étape 706. Ce deuxième message d'erreur ICMP peut alors être envoyé à destination de l'équipement dont l'adresse est contenue dans le champ d'adresse source « SA » 502 du flux de données 510.

Après l'envoi du deuxième message d'erreur ICMP, on passe aux étapes 710 et 711 au cours desquelles on compare (étape 710) et met à jour (étape 711), si nécessaire, la valeur courante du MTU de chaque canal 610 dont le mode de transport 602 « Carrier » indique un mode de transport virtuel (par exemple protocole UDP vers protocole TCP, ou réciproquement), dans l'hypothèse ou celui-ci met en œuvre temporairement le même mode de transport 602 et utilise la même adresse « TDA » 601 (adresse IP de la tête de tunnel distante) que le canal identifié par l'identifiant 650 « PID » déterminé à l'étape 703. Dans cette affirmative, on met à jour la ou les valeurs du « MTU » 605, comme à l'étape 705.

Dans une variante de réalisation, à l'émission d'un seul message d'erreur ICMP à destination de l'équipement identifié par son identifiant 606 « FID » à l'étape 706, il est possible d'envisager d'émettre un message d'erreur ICMP à destination de chacun des équipements source qui émettent les flux de la liste 607, identifiée à l'aide de l'identifiant « FIDs » 606 qui est associé à l'identifiant 650 « PID » du canal 610 déterminé à l'étape 703.

On présente maintenant, en relation avec la **figure 8**, un algorithme de gestion du mode de transport courant et de fenêtres de transmission, pour un type de paquet, selon un mode de réalisation particulier du procédé selon l'invention.

Comme on le verra, l'invention permet de notifier à un équipement source un changement de « MTU » provoqué par un changement de canal de transport 610 lors du démarrage d'une phase transitoire (aussi appelée phase de transport virtuel) de basculement d'un canal de transport à un autre.

Pour chacun des types de paquet, on gère donc un mode de transport et deux fenêtres de transmission (une par canal de transmission du tunnel). Une fenêtre de transmission associée à un canal donné est définie par un jeu de paramètres qui permettent de définir une quantité maximale de paquets pouvant être transmis sur ce canal donné pendant une durée déterminée (qui correspond au RTT). Pour un type de paquet donné, en fonction du mode de transport, les deux fenêtres augmenteront ou diminueront afin de commuter progressivement d'un canal à un autre.

On rappelle que le type d'un paquet en cours de traitement (ou paquet traité) est déterminé lors de l'étape de classification 32 de la figure 3.

Dans le présent exemple, on considère deux types de paquets : paquet TCP ou paquet UDP. Pour chacun de ces deux types de paquet, on gère un mode de transport et deux fenêtres de transmission, appelées ci-après « fenêtre de transmission TCP » et « fenêtre de transmission UDP ».

Il est important de noter que la description ci-après de la figure 8 est faite pour un paquet ayant un type donné, et donc qu'à chaque fois que le mode de transport ou un paramètre de fenêtre (Wtcp, Stcp, Wtcp_max, Wudp, Sudp, Wudp_max) est mentionné, il convient de comprendre qu'il s'agit du mode de transport ou d'un paramètre de fenêtre appartenant à un ensemble de variables propres audit type de paquet donné. La même remarque s'applique pour les figures 9a et 9b décrites ci-après.

Pour un paquet traité ayant un type donné, on arrive à l'étape 860 après avoir déterminé le canal de transmission préféré à l'étape 35 (cf. figure 3).

Dans l'étape 860, on commute en fonction du canal préféré : on passe à l'étape 861 si le canal préféré est le canal UDP (c'est-à-dire si le protocole de transport préféré est UDP), ou à l'étape 862 si le canal préféré est le canal TCP (c'est-à-dire si le protocole de transport préféré est TCP).

Dans l'étape 862 (cas où le canal préféré est le canal TCP), on commute en fonction du mode de transport courant (pour le type du paquet traité).

Si à l'étape 862 le mode courant est le mode stable TCP (associé à un précédent canal préféré qui est le canal TCP), le système doit entrer dans le mode transitoire TCP-vers-UDP, qui est établi à l'étape 863. Au préalable, on passe à l'étape 869, dans laquelle on initialise les paramètres de fenêtre pour le type du paquet traité :

- la quantité de données déjà transmises sur le canal UDP pour le type du paquet traité (Sudp) est mise à 0 (Sudp=0) ;
- la taille (Wudp) de la fenêtre UDP (quantité maximale de paquets pouvant être transmis sur le canal UDP) est mise à la taille de segment maximale (MSS, pour « Maximum Segment Size » en anglais) de la connexion TCP sur le canal TCP (Wudp=MSS) ;
- la condition d'arrêt (Wudp_max), qui correspond à la taille maximale de la fenêtre UDP, est mise à la valeur courante (Cwnd) de la fenêtre de congestion TCP (Wudp_max=Cwnd). Cette condition d'arrêt déterminera la sortie du mode transitoire TCP-vers-UDP.

Si à l'étape 862 le mode courant est le mode transitoire UDP-vers-TCP (associé à un précédent canal préféré qui est le canal TCP), le système doit là aussi entrer dans le mode transitoire TCP-vers-UDP, qui est établi à l'étape 863. On peut conserver les paramètres de fenêtre, pour le type du paquet traité, qui ont déjà été initialisés (cf figure 9a).

Si à l'étape 862 le mode courant est le mode stable UDP (associé à un précédent canal préféré qui est le canal UDP) ou le mode transitoire TCP-vers-UDP (associé à un précédent canal préféré qui est le canal UDP), aucune action n'est nécessaire.

Après l'étape 863 (dans laquelle on entre dans la phase transitoire où l'on va utiliser le mode de transport virtuel TCP vers UDP), on passe à l'étape 801. Pour le type du paquet traité, son canal de transport (identifié par son identifiant « PID » 550) est modifié pour référencer le mode de transport 602, de type TCP vers UDP, dans la table des canaux de transport 600. Ensuite, on ajoute l'identifiant « FID » 501 du type du paquet traité dans la liste des flux 607 transitant dans le canal 610 (de la base d'information des canaux 600). Enfin, on vérifie si la valeur du « MTU » du canal de transport nouvellement sélectionné est inférieure à la valeur du « MTU » du canal

précédemment utilisé par le type du paquet traité. En cas de vérification positive, on génère un message d'erreur ICMP (type=3, code=4) à partir des informations contenues dans le datagramme IP en cours de traitement, sans pour autant éliminer ce datagramme. Ensuite, comme en cas de vérification négative, l'étape 801 est terminée.

Après l'étape 801, on passe à l'étape 866 dans laquelle on lance l'exécution de l'algorithme de gestion du mode transitoire TCP-vers-UDP (décrit ci-après en relation avec la figure 9b). Après ce lancement, on passe à l'étape 38 de la figure 13.

Dans l'étape 861 (cas où le canal préféré est le canal UDP), on commute en fonction du mode de transport courant (pour le type du paquet traité).

Si à l'étape 861 le mode courant est le mode stable UDP (associé à un précédent canal préféré qui est le canal UDP), le système doit entrer dans le mode transitoire UDP-vers-TCP, qui est établi à l'étape 864. Au préalable, on passe à l'étape 868, dans laquelle on initialise les paramètres de fenêtre pour le type du paquet traité :

- la quantité de données déjà transmises sur le canal TCP pour le type du paquet traité (Stcp) est mise à 0 (Stcp=0) ;
- la taille (Wtcp) de la fenêtre TCP (quantité maximale de paquets pouvant être transmis sur le canal TCP) est mise à la taille de la valeur courante (Cwnd) de la fenêtre de congestion TCP (Wtcp=Cwnd). Ainsi, initialement, les deux fenêtres (Wtcp et Cwnd) ont la même taille ;
- la condition d'arrêt (Wtcp_max), qui correspond à la taille maximale de la fenêtre TCP, est mise à la valeur courante de la fenêtre UDP (Wudp). Cette condition d'arrêt déterminera la sortie du mode transitoire UDP-vers-TCP.

Si à l'étape 861 le mode courant est le mode transitoire TCP-vers-UDP (associé à un précédent canal préféré qui est le canal UDP), le système doit là aussi entrer dans le mode transitoire UDP-vers-TCP, qui est établi à l'étape 864. On peut conserver les paramètres de fenêtre, pour le type du paquet traité, qui ont déjà été initialisés (cf figure 9a).

Si à l'étape 861 le mode courant est le mode stable TCP (associé à un précédent canal préféré qui est le canal TCP) ou le mode transitoire UDP-vers-TCP (associé à un précédent canal préféré qui est le canal TCP), aucune action n'est nécessaire.

Après l'étape 864 (dans laquelle on entre dans une phase transitoire où l'on va utiliser le mode de transport virtuel UDP vers TCP), on passe à l'étape 802. Pour le type du paquet traité, son canal de transport (identifié par son identifiant « PID » 550) est modifié pour référencer le mode de transport 602, de type UDP vers TCP, dans la table des canaux de transport 600. Ensuite, on ajoute l'identifiant « FID » 501 du type du paquet traité dans la liste des flux 607 transitant dans le canal 610 (de la base d'information des canaux 600). Enfin, on vérifie si la valeur du « MTU » du canal de transport nouvellement sélectionné est inférieure à la valeur du « MTU » du canal précédemment utilisé par le type du paquet traité. En cas de vérification positive, on génère un message d'erreur ICMP (type=3, code=4) à partir des informations contenues dans le datagramme IP en cours de traitement, sans pour autant éliminer ce datagramme. Ensuite, comme en cas de vérification négative, l'étape 802 est terminée.

Après l'étape 802, on passe à l'étape 865 dans laquelle on lance l'exécution de l'algorithme de gestion du mode transitoire UDP-vers-TCP (décrit ci-après en relation avec la figure 9a). Après ce lancement, on passe à l'étape 38 de la figure 13.

Dans une variante de réalisation des étapes 801 et 802 précitées, pour chacun des flux (de la liste des flux 607) transitant dans le canal de transport (identifié par l'identifiant « PID » 650) du flux traité, il est possible d'envisager d'appliquer la méthode consistant à générer un message d'erreur ICMP en utilisant les informations contenues dans la table des flux de données 500 pour construire le message d'erreur ICMP 410 et la valeur du « MTU » du nouveau canal de transport considéré.

La **figure 9a** présente un algorithme de gestion du mode transitoire UDP-vers-TCP, selon un mode de réalisation particulier du procédé selon l'invention.

Comme on le verra, l'invention permet de notifier à un équipement source un changement de « MTU » (taille limite des données utiles pouvant être transmises dans un paquet) provoqué par un changement de canal de transport 610 lors de l'échéance d'une phase transitoire.

Cette figure décrit comment, pour un flux donné, les paramètres des fenêtres de transmission TCP et UDP (pour ce flux donné) évoluent pour permettre une commutation progressive depuis le mode de transport stable UDP vers le mode de transport stable TCP.

Afin d'éviter des problèmes dus à l'envoi d'un trop grand nombre de paquets sur le canal TCP comparé à la taille (Cwnd) de sa fenêtre de congestion, on prend en compte le mécanisme prévu dans le canal TCP pour éviter les congestions et on augmente la taille (Wtcp) de la fenêtre (virtuelle) de transmission TCP, pour être en accord avec l'évolution naturelle de la taille (Cwnd) de la fenêtre de congestion.

Dans une étape 959, on incrémente d'une unité le nombre (Nudp_tcp) de types de paquet dans le mode UDP-vers-TCP.

Les basculements de canal de transmission dans un tunnel faisant suite à des variations de conditions de transmission dans le tunnel ayant un impact sur les flux d'un type donné transitant dans le tunnel, il est opéré un basculement de canal pour l'ensemble des flux du type donné. C'est pourquoi on stocke le nombre de types de paquet dans un mode et non le nombre de flux dans ce mode.

Puis, pour gérer la taille (Wtcp) de la fenêtre de transmission TCP, une temporisation est lancée dans l'étape 951 (correspondant au RTT du canal TCP, mis à jour à chaque fois qu'un paquet est reçu, cf étape 12), et on attend l'expiration de cette temporisation, à l'étape 952.

Pendant ce temps (entre les étapes 951 et 952), des données sont envoyés conformément à l'étape 38 (cette étape 38 est effectuée une fois pour chaque flux).

Après que la temporisation a expiré, on teste dans l'étape 953 si le mode de transport a changé (suite à une modification du canal préféré à l'étape 35, on peut décider à l'étape 36 de changer le mode de transport depuis le mode transitoire UDP-vers-TCP vers le mode transitoire TCP-vers-UDP).

Si le mode de transport a changé, on passe avant de terminer à l'étape 957 dans laquelle on décrémente le nombre (Nudp_tcp) de types de paquet dans le mode UDP-vers-TCP.

Si le mode de transport n'a pas changé, l'algorithme se poursuit pour continuer à gérer l'évolution de la fenêtre (virtuelle) de transmission TCP. On passe à l'étape 954

dans laquelle on augmente la taille (Wtcp) de la fenêtre de transmission TCP de MSS/Nudp_tcp. Ainsi, on suit l'évolution maximale de la taille (Cwnd) de la fenêtre de congestion TCP en évitant des congestions, et en tenant compte de ce que Nudp_tcp types de paquets sont simultanément dans le mode transitoire UDP-vers-TCP. En outre, dans l'étape 954, la quantité de données déjà transmises sur le canal TCP pour le type de paquet traité (Stcp) pendant le dernier RTT est mise à 0 (Stcp=0).

Dans l'étape 955, on décide si la phase transitoire dans le mode transitoire UDP-vers-TCP est terminée. Pour cela on vérifie si la taille (Wtcp) de la fenêtre de transmission TCP a été suffisamment augmentée (Wtcp > Wtcp_max ?), et s'il n'y a plus de données destinées à être envoyées sur le canal UDP pour le type de paquet traité (Sudp=0 ?).

Si la phase transitoire est terminée, on passe à l'étape 956 dans laquelle on établit le mode stable TCP.

Ensuite, on passe à l'étape 960 dans laquelle on met à jour les paramètres du canal de transport (identifié par son identifiant « PID » 550) du flux traité. On va ainsi supprimer le flux traité de la liste des flux 607 du canal de transport précédemment utilisé (par le flux traité) et l'ajouter à la liste des flux du canal de transport nouvellement sélectionné à l'étape 956 (mode TCP). Dans cette même étape 960, on teste si la valeur du « MTU » 605 du canal de transport nouvellement sélectionné est supérieure à la valeur du « MTU » du canal de transport précédent. En cas de vérification positive, on génère un message d'erreur ICMP (type=3, code=4), puis on le transmet à l'équipement source (c'est-à-dire à l'équipement qui a émis le datagramme IP en cours de traitement), pour lui notifier une augmentation du MTU.

Enfin, on passe à l'étape 957 dans laquelle on décrémente le nombre (Nudp_tcp) de types de paquet dans le mode UDP-vers-TCP.

Si la phase transitoire n'est pas terminée, on passe à l'étape 958 dans laquelle on met à 0 la quantité de données déjà transmises sur le canal UDP pour le type du paquet traité (Sudp=0), puis on revient à l'étape 951 et une nouvelle temporisation est lancée.

La **figure 9b** présente un algorithme de gestion du mode transitoire TCP-vers-UDP, selon un mode de réalisation particulier du procédé selon l'invention.

Cette figure décrit comment, pour un flux donné, les paramètres des fenêtres de transmission TCP et UDP (pour ce flux donné) évoluent pour permettre une commutation progressive depuis le mode de transport stable TCP vers le mode de transport stable UDP.

Ce mécanisme de gestion de la fenêtre de transmission UDP est similaire à celui décrit ci-dessus en relation avec la figure 9a pour la fenêtre de transmission TCP. La taille (Wudp) de la fenêtre de transmission UDP indique la quantité maximale de flux pouvant être transmis sur le canal UDP pendant une durée RTT. Cette durée RTTu est une valeur calculée par le système, et correspond à un temps d'aller-retour comme décrit pour TCP (cette valeur peut être calculée en envoyant périodiquement une requête de contrôle spécifique, depuis une tête de tunnel vers une autre, l'autre tête de tunnel répondant immédiatement à cette requête).

Pour éviter une congestion due à l'envoi d'un trop grand nombre de flux sur le canal UDP (avant que le canal TCP ait fini de vider sa mémoire tampon d'émission), on ne peut pas brutalement commuter depuis une transmission entièrement TCP vers une transmission entièrement UDP. On utilise donc le mécanisme de la figure 9b (similaire à celui de la figure 9a).

Dans une étape 909, on incrémente d'une unité le nombre (Ntcp_udp) de types de paquet dans le mode TCP-vers-UDP.

Puis, pour gérer la taille (Wudp) de la fenêtre de transmission UDP, une temporisation est lancée dans l'étape 901 (correspondant au RTTu précité), et on attend l'expiration de cette temporisation, à l'étape 902.

Pendant ce temps (entre les étapes 901 et 902), des flux sont envoyés conformément à l'étape 38 (cette étape 38 est effectuée une fois pour chaque flux).

Après que la temporisation a expiré, on teste dans l'étape 903 si le mode de transport a changé (suite à une modification du canal préféré à l'étape 35, on peut décider à l'étape 36 de changer le mode de transport depuis le mode transitoire TCP-vers-UDP vers le mode transitoire UDP-vers-TCP).

Si le mode de transport a changé, on passe avant de terminer à l'étape 907 dans laquelle on décrémente le nombre (Ntcp_udp) de types de paquet dans le mode TCP-vers-UDP.

Si le mode de transport n'a pas changé, l'algorithme se poursuit pour continuer à gérer l'évolution de la fenêtre (virtuelle) de transmission UDP. On passe à l'étape 904 dans laquelle on augmente la taille (Wudp) de la fenêtre de transmission UDP de MSS/Ntcp_udp. En outre, dans l'étape 904, la quantité de flux déjà transmis sur le canal UDP pour le flux traité (Sudp) pendant le dernier RTT (RTTu) est mise à 0 (Sudp=0).

Dans l'étape 905, on décide si la phase transitoire dans le mode transitoire TCP-vers-UDP est terminée. Pour cela on vérifie si la taille (Wudp) de la fenêtre de transmission UDP a été suffisamment augmentée (Wudp > Wudp_max ?), et s'il n'y a plus de données destinées à être envoyées sur le canal TCP pour le type de paquet traité (Stcp=0 ?).

Si la phase transitoire est terminée, on passe à l'étape 906 dans laquelle on établit le mode stable UDP.

Ensuite, on passe à l'étape 910 dans laquelle on met à jour les paramètres du canal de transport (identifié par son identifiant « PID » 550) des flux du type de paquet traité. On va ainsi supprimer chacun de ces flux de la liste des flux 607 du canal de transport précédemment utilisé (par le flux traité) et l'ajouter à la liste des flux du canal de transport nouvellement sélectionné à l'étape 906 (mode UDP). Dans cette même étape 960, on teste si la valeur du « MTU » 605 du canal de transport nouvellement sélectionné est supérieure à la valeur du « MTU » du canal de transport précédent. En cas de vérification positive (augmentation ou diminution du MTU), on génère un message d'erreur ICMP (type=3, code=4), puis on le transmet à l'équipement source (c'est-à-dire à l'équipement qui a émis le datagramme IP en cours de traitement), pour lui notifier une augmentation du MTU.

Enfin, on passe à l'étape 907 dans laquelle on décrémente le nombre (Ntcp_udp) de types de paquet dans le mode TCP-vers-UDP.

Si la phase transitoire n'est pas terminée, on passe à l'étape 908 dans laquelle on met à 0 la quantité de données déjà transmises sur le canal TCP pour le type de paquet traité (Stcp=0), puis on revient à l'étape 901 et une nouvelle temporisation est lancée.

Dans une variante de réalisation des étapes 910 et 960 précitées, pour chacun des flux (de la liste des flux 607) transitant dans le canal de transport (identifié par l'identifiant « PID » 650) du flux traité, il est possible d'envisager d'appliquer la

méthode consistant à générer un message d'erreur ICMP en utilisant les informations contenues dans la table des flux de données 500 pour construire le message d'erreur ICMP 410 et la valeur du « MTU » du nouveau canal de transport considéré.

La **figure 10** illustre une configuration schématique d'un appareil de communication 1000 (tête de tunnel 101 ou 102 de la figure 1a) adapté pour mettre en œuvre la technique de l'invention. Il comprend une mémoire de type RAM 1002 qui fonctionne comme une mémoire principale, une zone de travail, etc., de l'unité centrale (CPU) 1001. L'unité centrale 1001 est capable, à la mise sous tension de l'appareil de communication, d'exécuter des instructions à partir de la mémoire de type ROM 1003. Après la mise sous tension, l'unité centrale 1001 est capable d'exécuter des instructions provenant de la mémoire RAM 1002 en relation avec une application logicielle après que ces instructions ont été chargées à partir de la ROM 1003 ou du disque dur (HD) 1006, par exemple. Une telle application logicielle, lorsqu'elle est exécutée par l'unité centrale 1001, provoque la réalisation de tout ou partie des étapes des organigrammes illustrés sur les figures 7, 8, 9a, 9b, 11, 12 et 13.

REVENDICATIONS

1. Procédé de notification à un dispositif source (109, 110, 111, 112) d'une taille limite de paquets de données destinées à être transmises dans un tunnel (100) comprenant une pluralité de canaux de transmission, ledit dispositif source étant relié à une tête de tunnel (101) via un réseau de communication (107), ledit procédé étant mis en oeuvre par ladite tête de tunnel formant un point d'entrée dudit tunnel, caractérisé en ce qu'il comprend les étapes suivantes, pour chaque flux de données destinées à être transmises dans ledit tunnel et provenant dudit dispositif source :

a) détection d'un événement déclencheur lié à un nouveau canal associé audit flux pour le transport dudit flux dans ledit tunnel ;

b) obtention d'une première taille limite de paquets associée à un canal précédent associé audit flux pour le transport dudit flux dans ledit tunnel ;

c) obtention d'une deuxième taille limite de paquets associée audit nouveau canal ;

d) détection d'un changement de taille limite de paquets (701), par comparaison de ladite première taille limite de paquets avec ladite deuxième taille limite de paquets ;

e) en cas de détection positive, transmission (707) d'un message de changement de taille limite de paquets vers ledit dispositif source.

2. Procédé selon la revendication 1, caractérisé en ce qu'un premier événement déclencheur est un basculement de transmission, pour ledit flux, dudit canal précédent vers ledit nouveau canal, qui est un canal distinct dudit canal précédent.

3. Procédé selon la revendication 2, caractérisé en ce que le tunnel comprend des canaux de transmission réels et des canaux de transmission virtuels, un canal réel étant un canal dans lequel les flux transportés sont encapsulés par un protocole de transport unique, en ce que ledit premier événement déclencheur est un basculement appartenant au groupe comprenant :

- un basculement depuis un premier canal réel vers un second canal réel ;
- un basculement depuis un canal réel vers un canal virtuel ;
- un basculement depuis un canal virtuel vers un canal réel ;
- un basculement depuis un premier canal virtuel vers un second canal virtuel ;

et en ce que un canal virtuel associé à un flux donné pour le transport dudit flux dans ledit tunnel est défini par :

- deux canaux réels dudit tunnel, et

- un mécanisme de basculement progressif de l'un vers l'autre desdits canaux réels, permettant pour chaque paquet dudit flux donné de sélectionner de façon dynamique un canal effectif parmi lesdits canaux réels.

4. Procédé selon la revendication 3, caractérisé en ce que la taille limite de paquets associée à un canal virtuel est égale à la plus petite taille limite de paquets parmi les deux tailles limites de paquets associées aux deux canaux réels.

5. Procédé selon la revendication 1, caractérisé en ce qu'un second événement déclencheur est une réception d'un message d'erreur indiquant un changement de taille limite de paquets pour un canal de transport du flux qui constitue à la fois le canal précédent et le nouveau canal, lesdites première et deuxième tailles limites de paquets étant respectivement une précédente et une nouvelle taille limite de paquets dudit canal de transmission.

6. Procédé selon l'une quelconque des revendications 1 à 5, caractérisé en ce qu'il comprend :

- une phase de configuration comprenant une première étape d'association d'une information de type de paquet, à une information de canal de transmission, à une information (605) de taille limite de paquets, et à une liste (607) d'identifiants de flux (501) transmis par ledit canal de transmission et composés de paquets dudit type de paquet ;
- une phase de mise à jour, quand un changement de taille limite de paquets est détecté pour la transmission d'un type de paquet donné, de l'information de taille limite de paquets associée audit type de paquet.

7. Procédé selon la revendication 6, caractérisé en ce que, si ledit nouveau canal est un canal réel, la phase de mise à jour comprend une étape de mise à jour de l''information de taille limite de paquets associée à des types de paquets transmis par un canal virtuel défini par un couple de canaux réels comprenant ledit nouveau canal.

8. Procédé selon l'une quelconque des revendications 6 et 7, caractérisé en ce que la phase de configuration comprend en outre une deuxième étape d'association d'un identifiant de flux (501) à une adresse de dispositif source (502),

et en ce que la phase de mise à jour comprend en outre les étapes suivantes, en cas de détection du second événement déclencheur :

- récupération d'un identifiant de flux (501) dans le message d'erreur, ledit identifiant de flux étant inséré par la tête de tunnel dans les paquets dudit flux pendant une étape de transmission desdits paquets dans le tunnel ;
- récupération de l'adresse de dispositif source (502) associée à l'identifiant de flux récupéré ;
- construction d'un message de changement de taille limite de paquets à partir de l'information de taille limite de paquets mise à jour ;
- transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source (502) récupérée.

9. Procédé selon la revendication 8, caractérisé en ce qu'il comprend les étapes suivantes :

- récupération d'un ensemble (607) d'identifiants de flux (501) à partir d'un identifiant dudit nouveau canal ;

pour chaque identifiant de flux de l'ensemble récupérée :

- récupération de l'adresse de dispositif source (502) associée audit identifiant de flux;
- construction d'un message de changement de taille limite de paquets à partir de ladite information de taille limite de paquets mise à jour ;
- transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source (502) récupérée.

10. Produit programme d'ordinateur téléchargeable depuis un réseau de communication et/ou enregistré sur un support lisible par ordinateur et/ou exécutable par un processeur, caractérisé en ce qu'il comprend des instructions de code de programme pour la mise en oeuvre du procédé selon au moins une des revendications 1 à 9, lorsque ledit programme est exécuté sur un ordinateur.

11. Moyen de stockage, éventuellement totalement ou partiellement amovible, lisible par un ordinateur, stockant un jeu d'instructions exécutables par ledit ordinateur pour mettre en œuvre le procédé selon au moins une des revendications 1 à 9.

12. Tête de tunnel destinée à mettre en œuvre un procédé de notification à un dispositif source d'une taille limite de paquets de données destinées à être transmises dans un tunnel comprenant une pluralité de canaux de transmission, ledit dispositif source étant relié à ladite tête de tunnel via un réseau de communication, ladite tête de tunnel comprenant des moyens de transmission dans ledit tunnel de flux de données provenant dudit dispositif source,

caractérisée en ce que la tête de tunnel comprend :

- des moyens de détection d'un événement déclencheur lié à un nouveau canal associé à un flux donné pour le transport dudit flux donné dans ledit tunnel ;
- des moyens d'obtention d'une première taille limite de paquets associée à un canal précédent associé audit flux donné pour le transport dudit flux donné dans ledit tunnel ;
- des moyens d'obtention d'une deuxième taille limite de paquets associée audit nouveau canal ;
- des moyens de détection d'un changement de taille limite de paquets, par comparaison de ladite première taille limite de paquets avec ladite deuxième taille limite de paquets ;
- des moyens de transmission d'un message de changement de taille limite de paquets vers ledit dispositif source.

13. Tête de tunnel selon la revendication 12, caractérisée en ce qu'un premier événement déclencheur est un basculement de transmission, pour ledit flux donné, dudit canal précédent vers ledit nouveau canal, qui est un canal distinct dudit canal précédent.

14. Tête de tunnel selon la revendication 13, caractérisée en ce que le tunnel comprend des canaux de transmission réels et des canaux de transmission virtuels, un canal réel étant un canal dans lequel les flux transportés sont encapsulés par un protocole de transport unique, en ce que ledit premier événement déclencheur est un basculement appartenant au groupe comprenant :

- un basculement depuis un premier canal réel vers un second canal réel ;

- un basculement depuis un canal réel vers un canal virtuel ;
- un basculement depuis un canal virtuel vers un canal réel ;
- un basculement depuis un premier canal virtuel vers un second canal virtuel ;

et en ce que un canal virtuel associé à un flux donné pour le transport dudit flux donné dans ledit tunnel est défini par :

- deux canaux réels dudit tunnel, et
- un mécanisme de basculement progressif de l'un vers l'autre desdits canaux réels, permettant pour chaque paquet dudit flux donné de sélectionner de façon dynamique un canal effectif parmi lesdits canaux réels.

15. Tête de tunnel selon la revendication 14, caractérisée en ce que la taille limite de paquets associée à un canal virtuel est égale à la plus petite taille limite de paquets parmi les deux tailles limites de paquets associées aux deux canaux réels.

16. Tête de tunnel selon la revendication 12, caractérisée en ce qu'un second événement déclencheur est une réception d'un message d'erreur indiquant un changement de taille limite de paquets pour un canal de transport du flux donné qui constitue à la fois le canal précédent et le nouveau canal, lesdites première et deuxième tailles limites de paquets étant respectivement une précédente et une nouvelle taille limite de paquets dudit canal de transmission.

17. Tête de tunnel selon l'une quelconque des revendications 12 à 16, caractérisée en ce qu'elle comprend :

- des moyens de configuration comprenant des premiers moyens d'association d'une information de type de paquet, à une information de canal de transmission, à une information de taille limite de paquets, et à une liste d'identifiants de flux transmis par ledit canal de transmission et composés de paquets dudit type de paquet ;
- des moyens de mise à jour de l'information de taille limite de paquets associée audit type de paquet.

18. Tête de tunnel selon la revendication 17, caractérisée en ce qu'elle comprend des moyens de mise à jour de l'information de taille limite de paquets associée à des types de paquets transmis par un canal virtuel défini par un couple de canaux réels comprenant ledit nouveau canal.

19. Tête de tunnel selon l'une quelconque des revendications 17 et 18, caractérisée en ce que les moyens de configuration comprennent en outre des deuxièmes moyens d'association d'un identifiant de flux à une adresse de dispositif source,

et en ce que les moyens de mise à jour comprennent :

- des moyens de récupération d'un identifiant de flux dans le message d'erreur, ledit identifiant de flux étant inséré par la tête de tunnel dans les paquets dudit flux pendant une étape de transmission desdits paquets dans le tunnel ;
- des moyens de récupération de l'adresse de dispositif source associée à l'identifiant de flux récupéré ;
- des moyens de construction d'un message de changement de taille limite de paquets à partir de l'information de taille limite de paquets mise à jour ;
- des moyens de transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

20. Tête de tunnel selon la revendication 19, caractérisée en ce qu'elle comprend :

- des moyens de récupération d'un ensemble d'identifiants de flux à partir d'un identifiant dudit nouveau canal ;
- des moyens de récupération de l'adresse de dispositif source associée audit identifiant de flux;
- des moyens de construction d'un message de changement de taille limite de paquets à partir de ladite information de taille limite de paquets mise à jour ;
- des moyens de transmission dudit message de changement de taille limite de paquets au dispositif source correspondant à l'adresse de dispositif source récupérée.

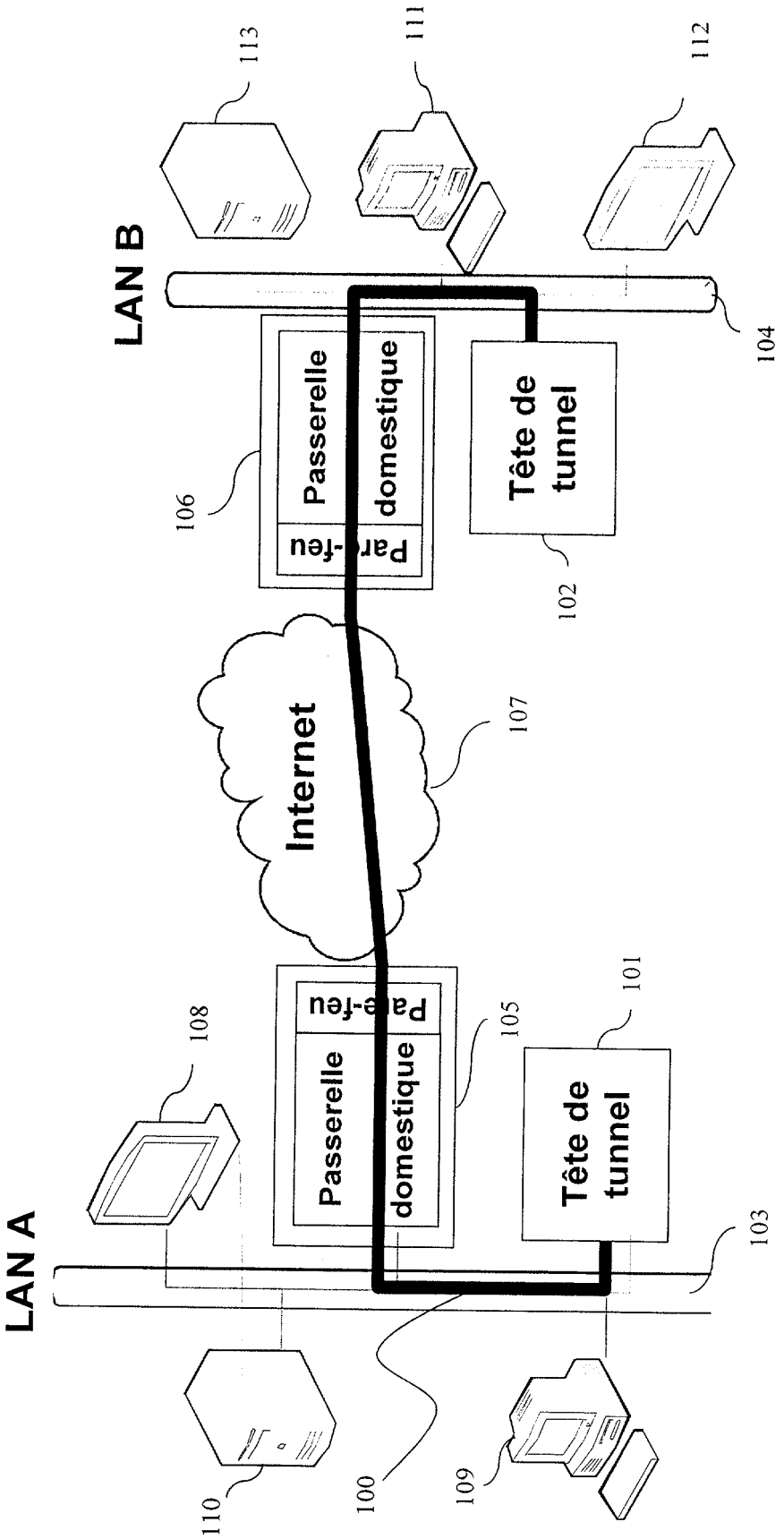

LAN A
LAN B
Internet
Passerelle domestique
Pare-feu
Tête de tunnel
100
101
102
103
104
105
106
107
108
109
110
111
112
113

Fig. 1a

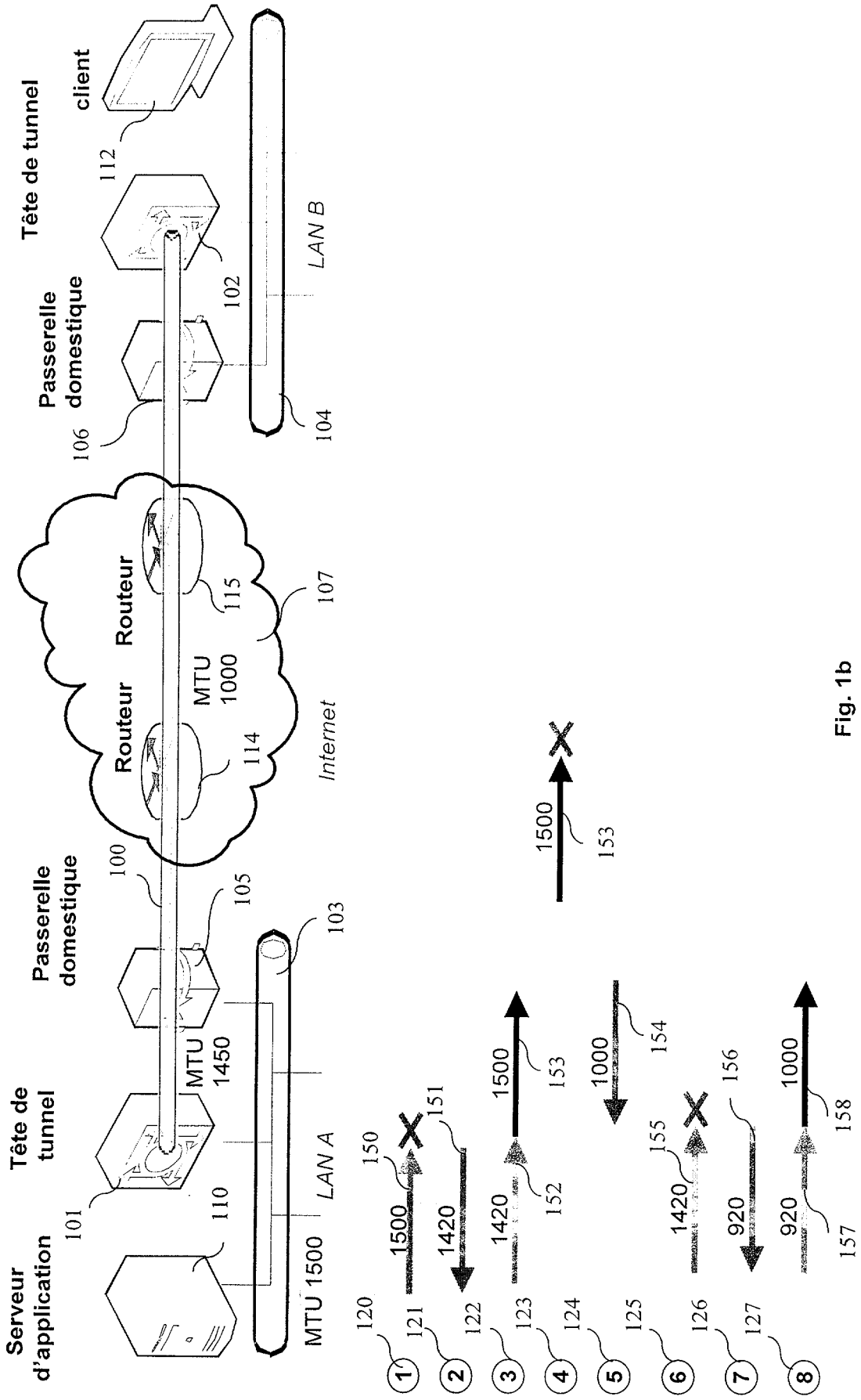

Serveur d'application
Tête de tunnel
Passerelle domestique
Passerelle domestique
Tête de tunnel
client
Routeur
Routeur
Routeur
Internet
MTU 1500
LAN A
LAN B
MTU 1450
MTU 1000
100
101
102
103
104
105
106
107
110
112
114
115
120
121
122
123
124
125
126
127
150
151
152
153
154
155
156
157
158
1500
1420
1500
1500
1000
1420
920
920
1000

Fig. 1b

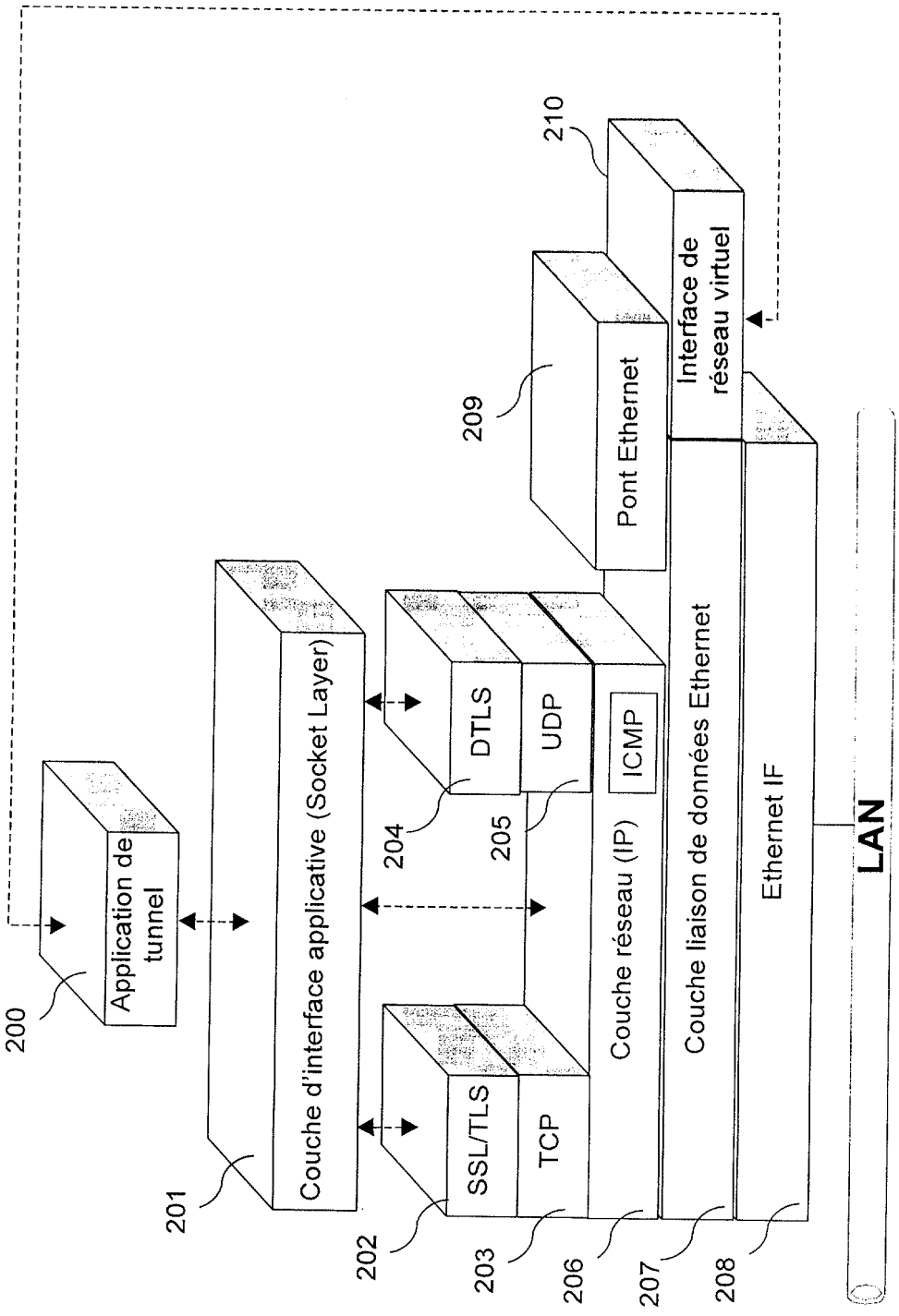

200
Application de tunnel
201
Couche d'interface applicative (Socket Layer)
202
SSL/TLS
203
TCP
204
DTLS
205
UDP
206
Couche réseau (IP)
ICMP
207
Couche liaison de données Ethernet
208
Ethernet IF
LAN
209
Pont Ethernet
210
Interface de réseau virtuel

Fig. 2a

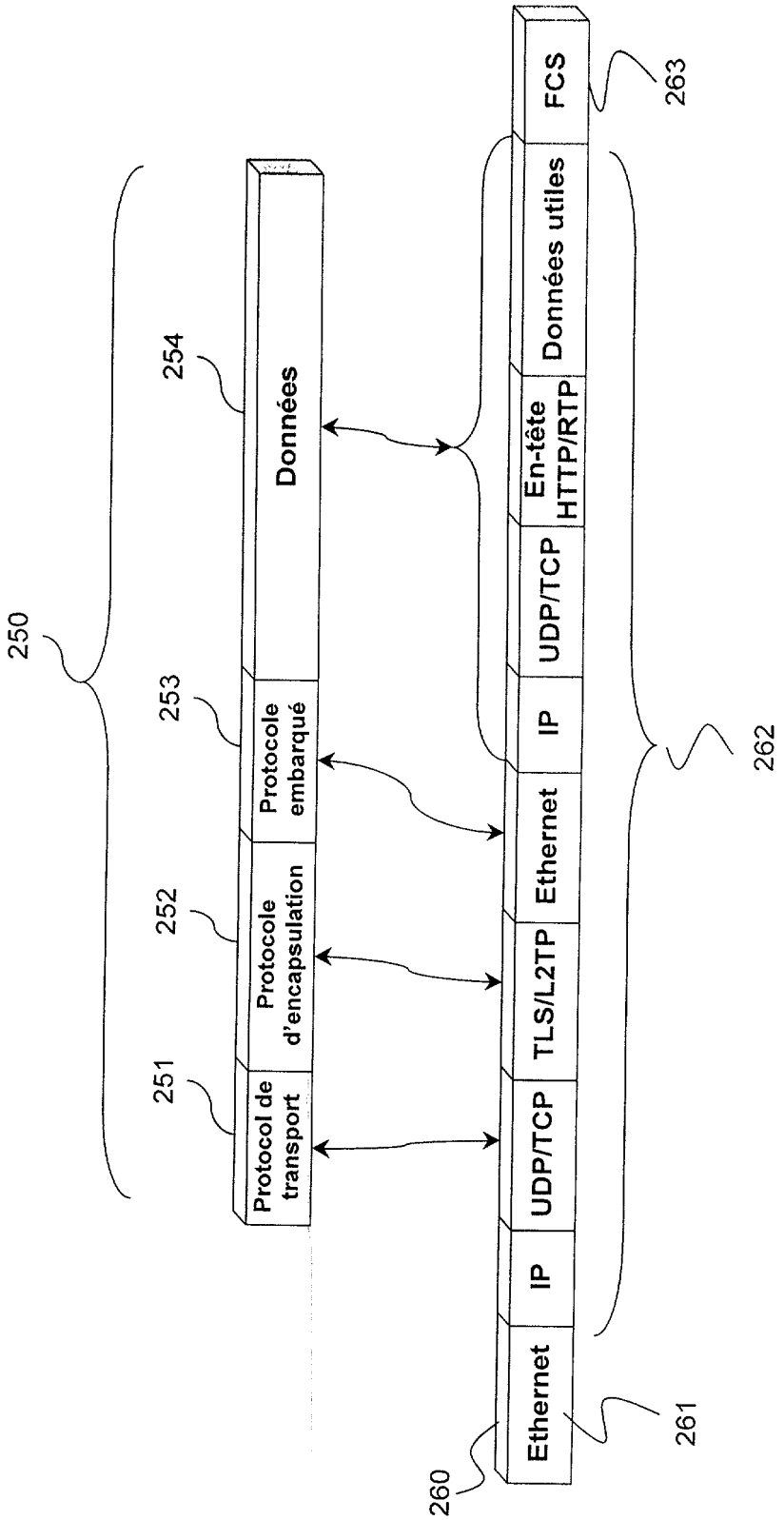

250
251
252
253
254
Protocol de transport
Protocole d'encapsulation
Protocole embarqué
Données
260
Ethernet
IP
UDP/TCP
TLS/L2TP
Ethernet
IP
UDP/TCP
En-tête HTTP/RTP
Données utiles
FCS
261
262
263

Fig. 2b

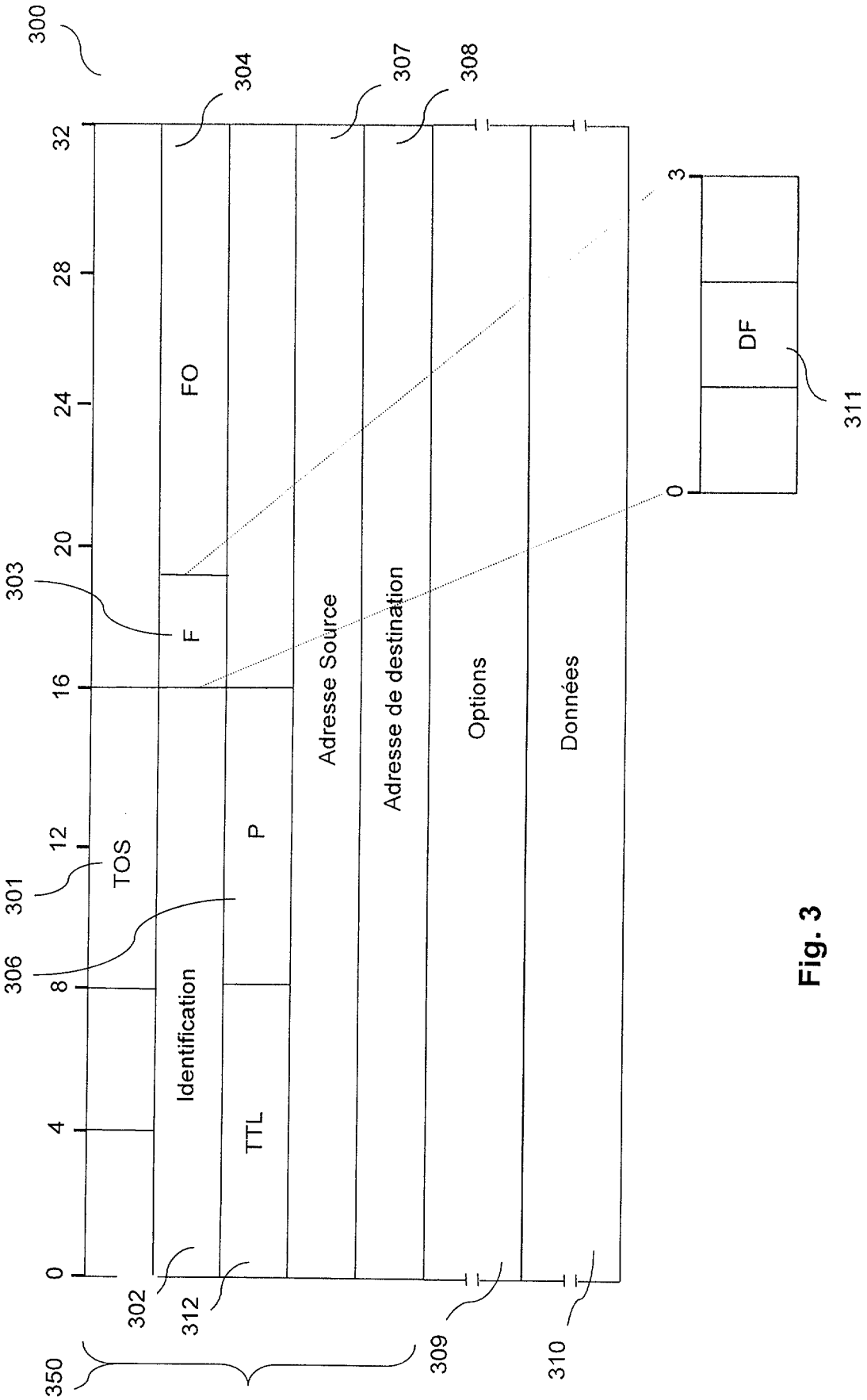

300
304
307
308
32
28
24
20
16
12
8
4
0
303
301
306
302
312
350
309
310
FO
F
TOS
P
Identification
TTL
Adresse Source
Adresse de destination
Options
Données
3
0
DF
311

Fig. 3

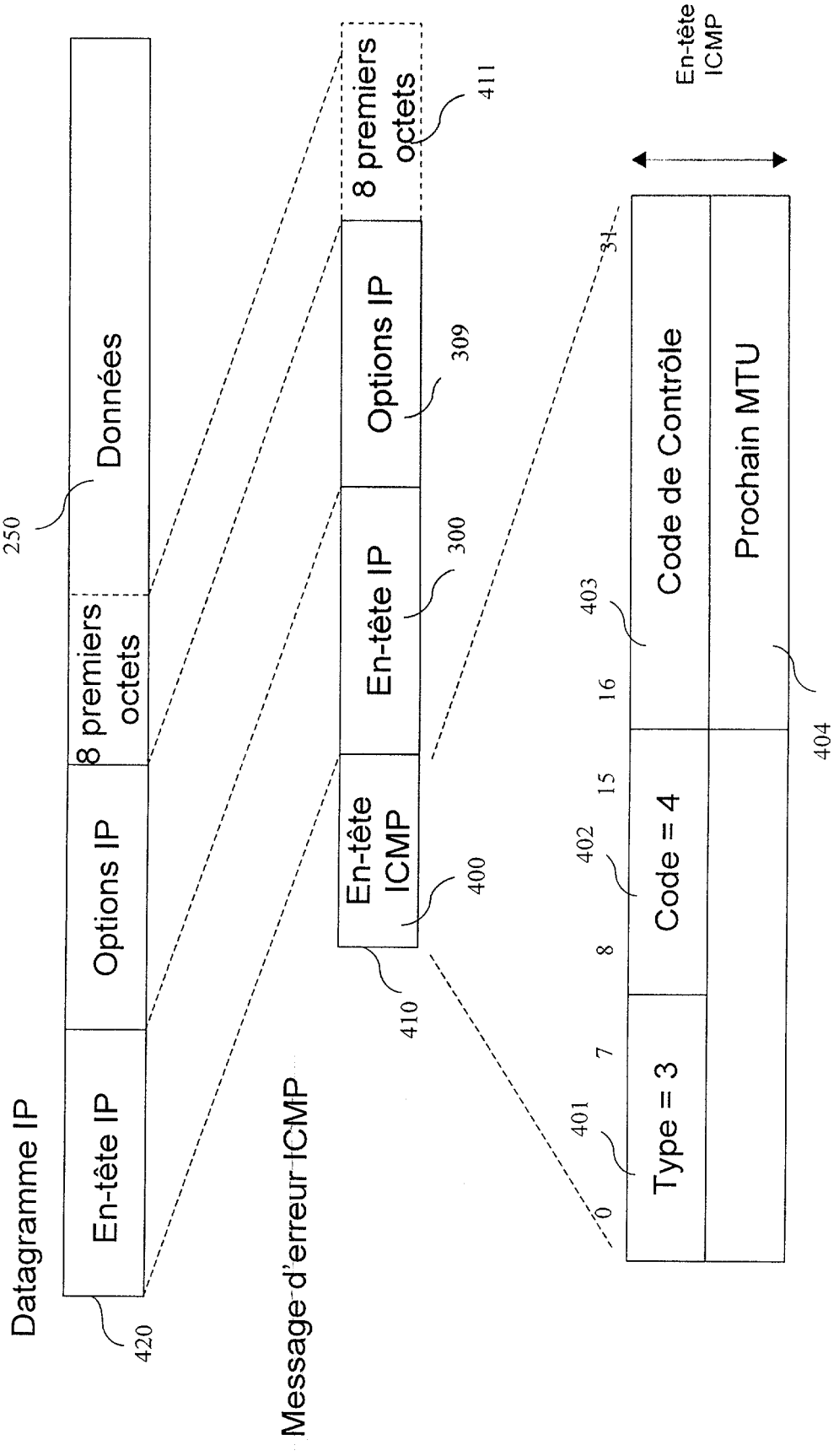

Datagramme IP
En-tête IP
Options IP
8 premiers octets
Données
420
250
Message d'erreur ICMP
En-tête ICMP
En-tête IP
Options IP
8 premiers octets
400
410
300
309
411
En-tête ICMP
Type = 3
Code = 4
Code de Contrôle
Prochain MTU
401
402
403
404
0
7
8
15
16
31

Fig. 4

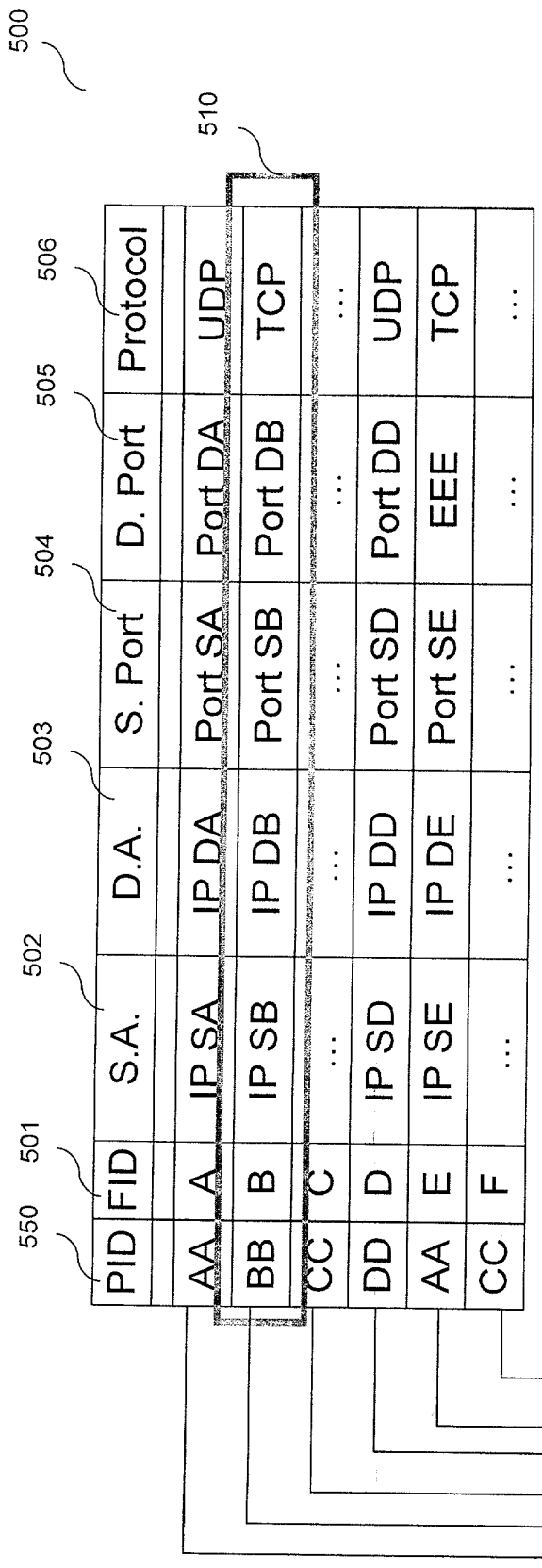

500
510
550
501
502
503
504
505
506
PID
FID
S.A.
D.A.
S. Port
D. Port
Protocol
AA A IP SA IP DA Port SA Port DA UDP
BB B IP SB IP DB Port SB Port DB TCP
CC C ... ...
DD D IP SD IP DD Port SD Port DD UDP
AA E IP SE IP DE Port SE EEE TCP
CC F ... ...

Fig. 5

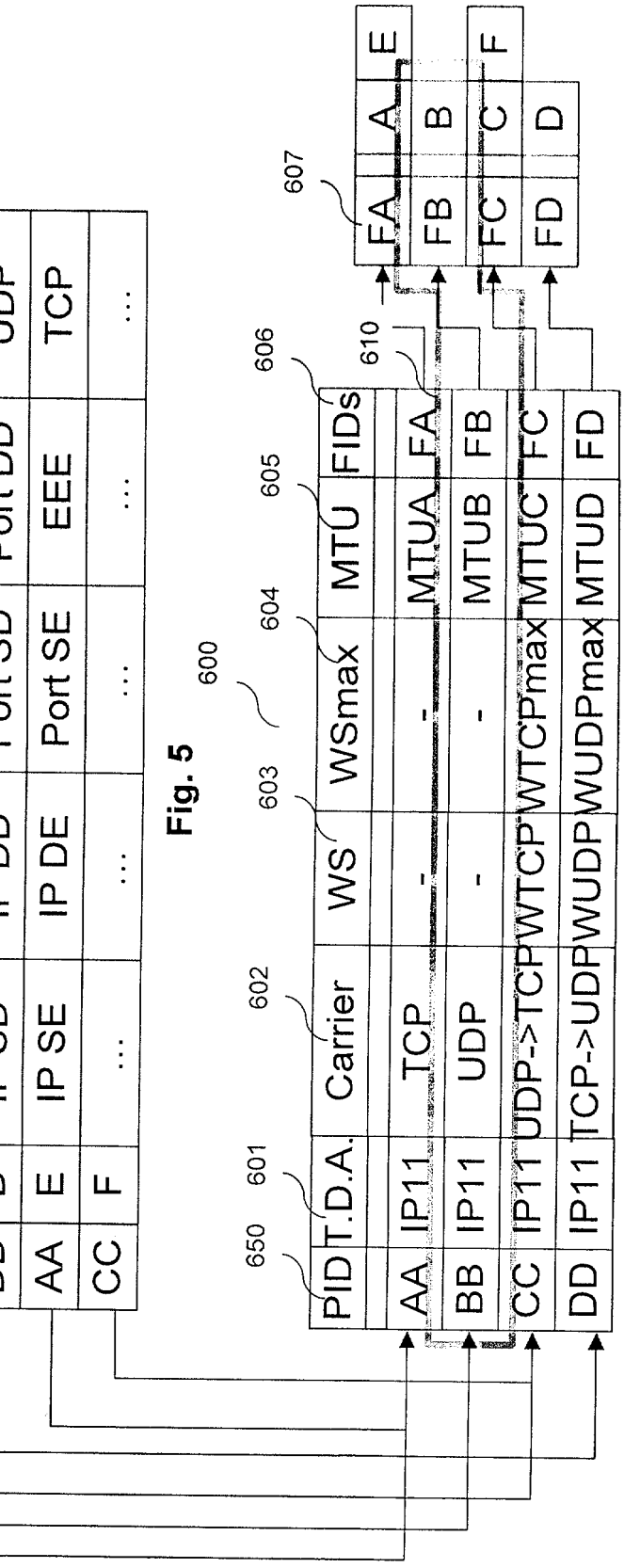

600
650
601
602
603
604
605
606
607
610
PID
T.D.A.
Carrier
WS
WSmax
MTU
FIDs
AA IP11 TCP - - MTUA FA
BB IP11 UDP - - MTUB FB
CC IP11 UDP->TCP WTCP WTCPmax MTUC FC
DD IP11 TCP->UDP WUDP WUDPmax MTUD FD
FA A E
FB B
FC C F
FD D

Fig. 6

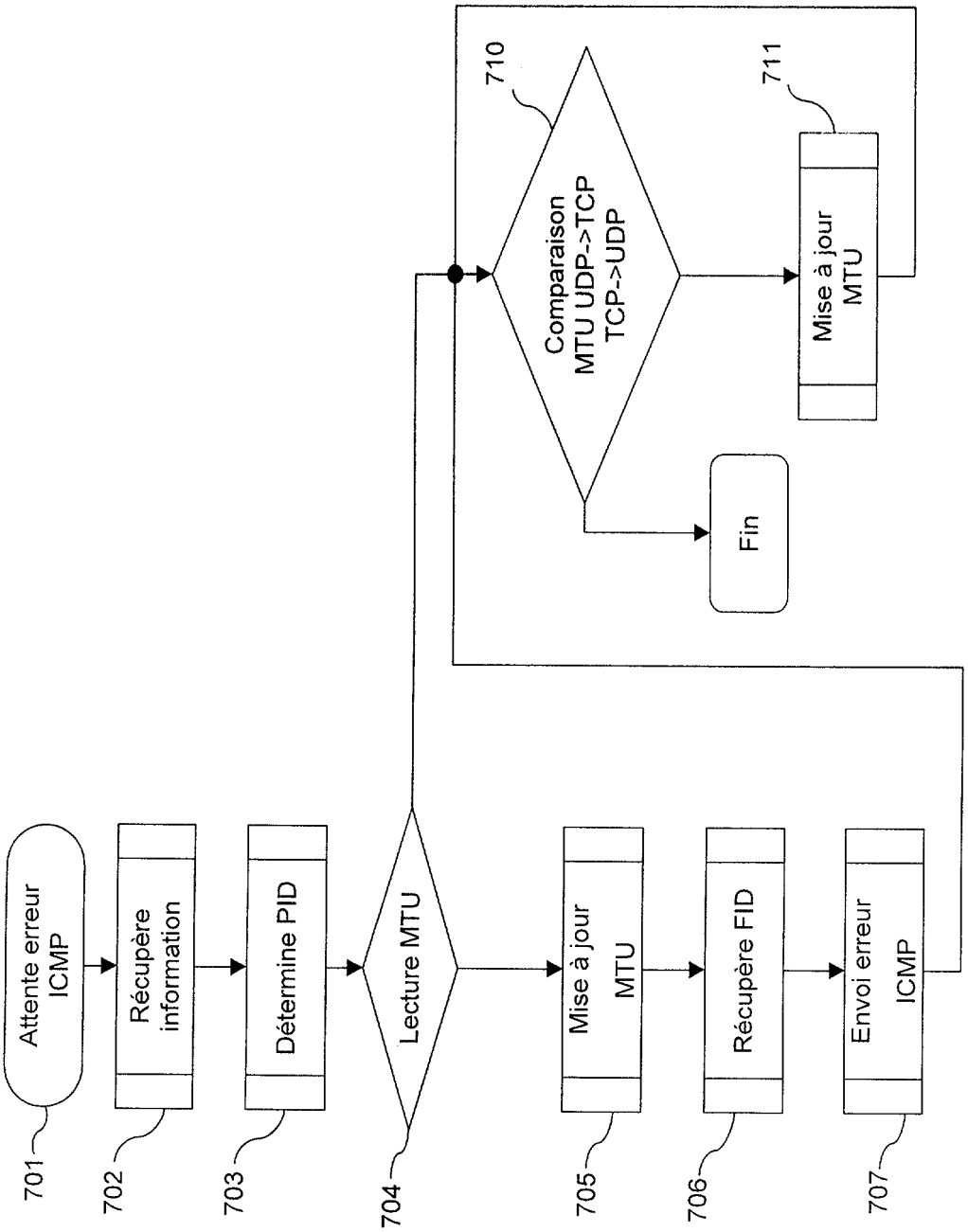

Attente erreur ICMP
Récupère information
Détermine PID
Lecture MTU
Mise à jour MTU
Récupère FID
Envoi erreur ICMP
Comparaison MTU UDP->TCP TCP->UDP
Mise à jour MTU
Fin
701
702
703
704
705
706
707
710
711

Fig. 7

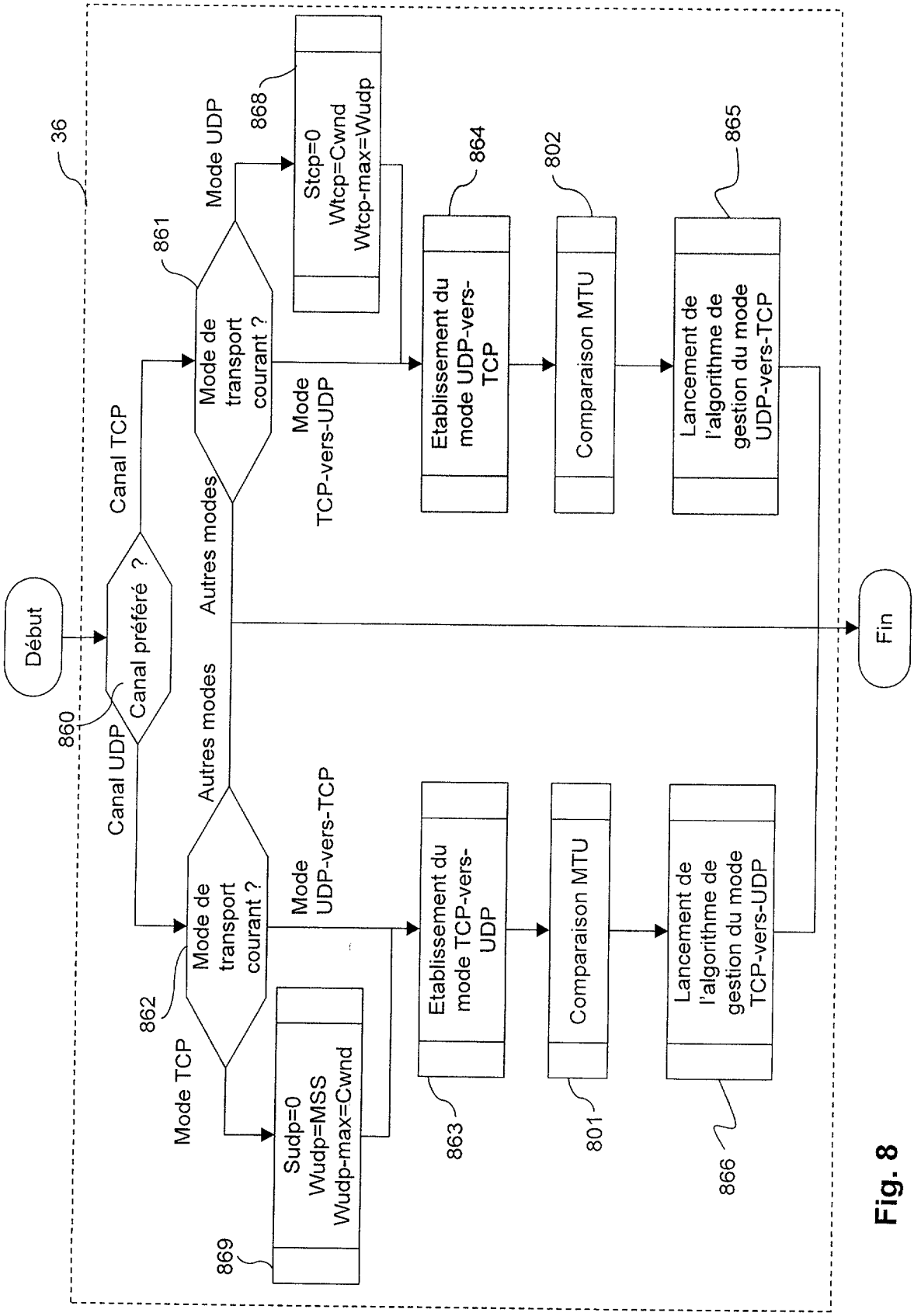

36
Début
Canal préféré ?
860
Canal TCP
Canal UDP
Mode de transport courant ?
861
Mode UDP
Autres modes
Stcp=0
Wtcp=Cwnd
Wtcp-max=Wudp
868
Mode TCP-vers-UDP
Etablissement du mode UDP-vers-TCP
864
Comparaison MTU
802
Lancement de l'algorithme de gestion du mode UDP-vers-TCP
865
Mode de transport courant ?
862
Mode TCP
Autres modes
Mode UDP-vers-TCP
Sudp=0
Wudp=MSS
Wudp-max=Cwnd
869
Etablissement du mode TCP-vers-UDP
863
Comparaison MTU
801
Lancement de l'algorithme de gestion du mode TCP-vers-UDP
866
Fin

Fig. 8

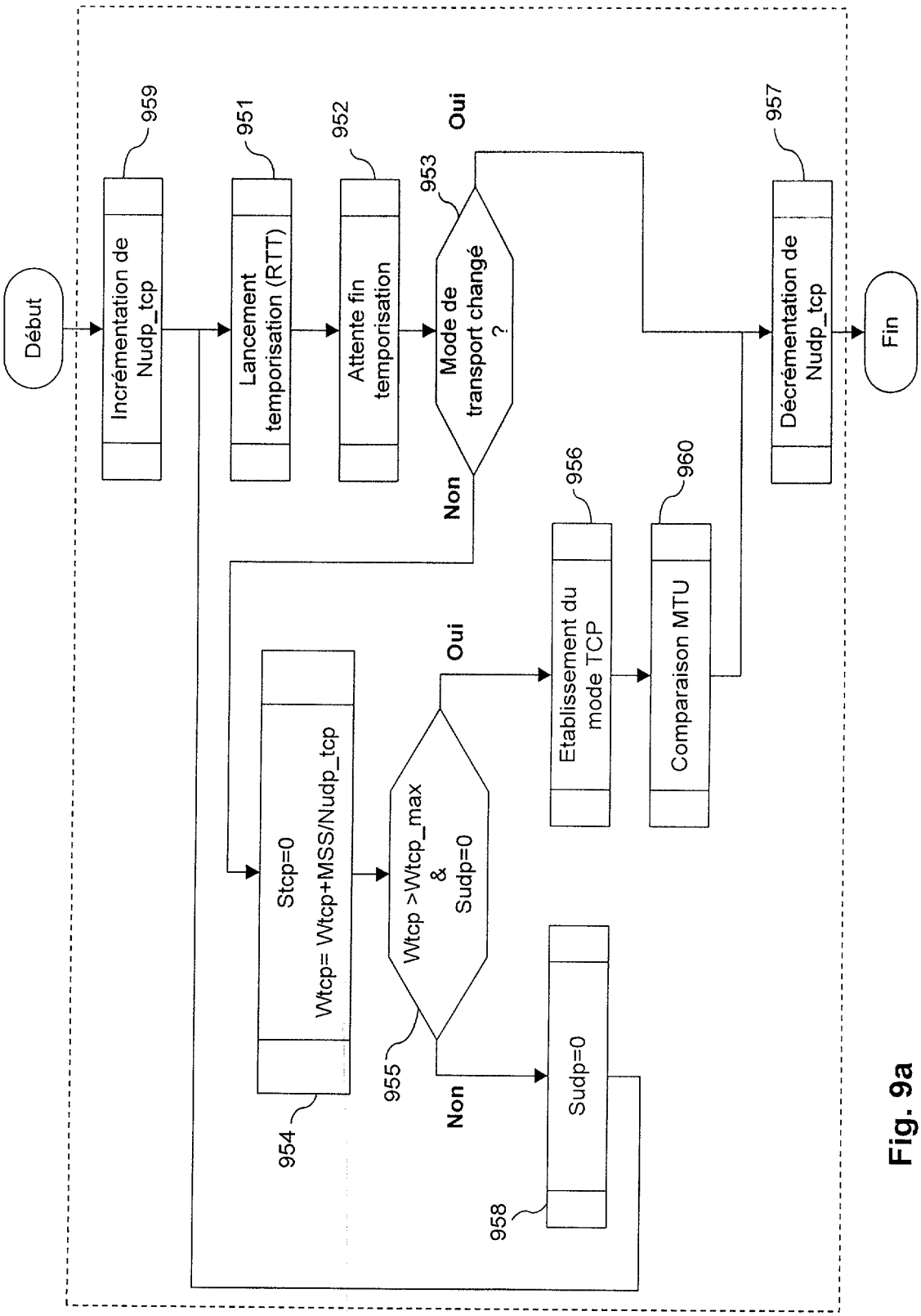

Début
Incrémentation de Nudp_tcp
959
Lancement temporisation (RTT)
951
Attente fin temporisation
952
Mode de transport changé ?
953
Oui
Non
Décrémentation de Nudp_tcp
957
Fin
Etablissement du mode TCP
956
Comparaison MTU
960
Stcp=0
Wtcp= Wtcp+MSS/Nudp_tcp
954
Wtcp >Wtcp_max & Sudp=0
955
Oui
Non
Sudp=0
958

Fig. 9a

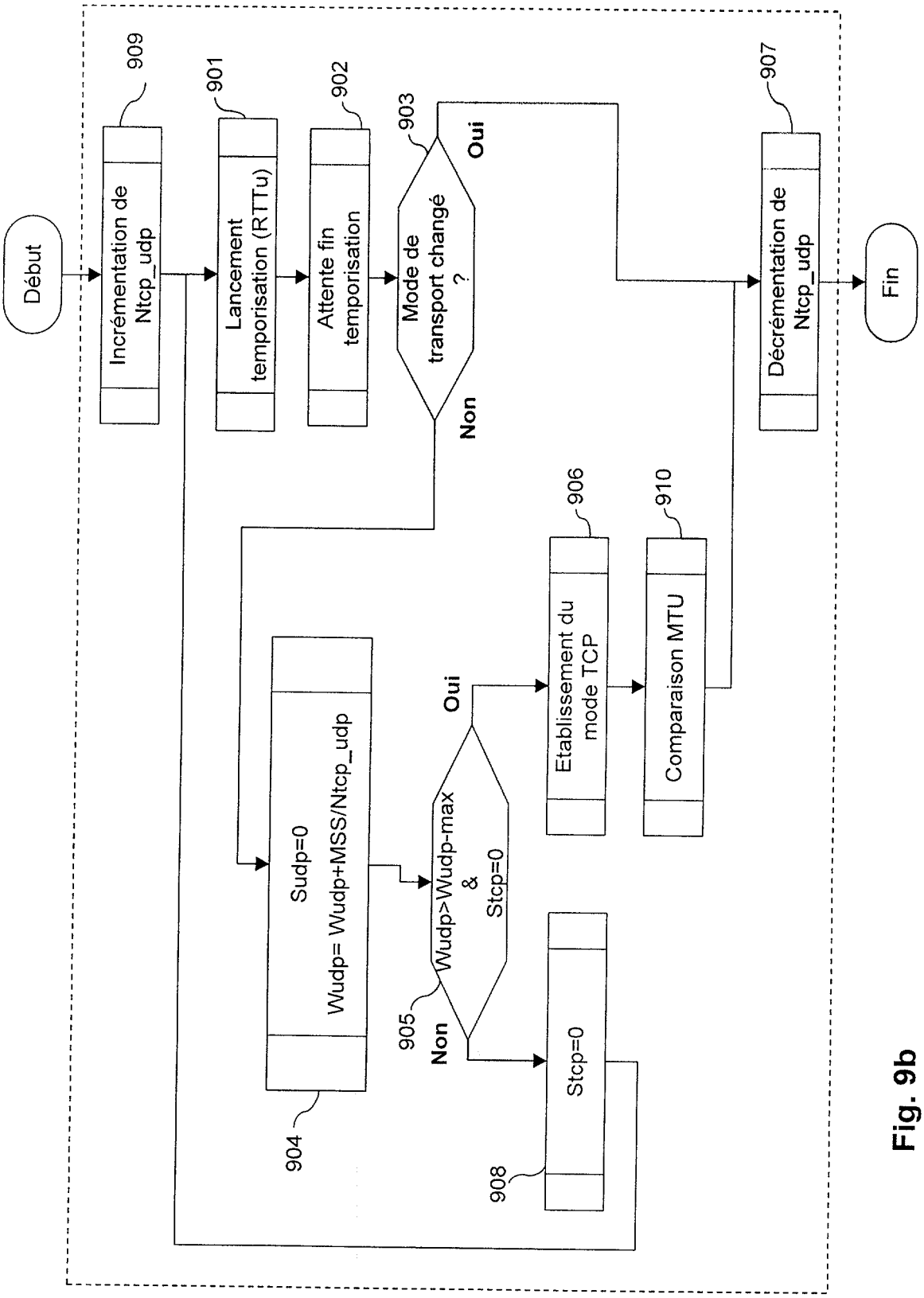

Début
Incrémentation de Ntcp_udp
909
Lancement temporisation (RTTu)
901
Attente fin temporisation
902
Mode de transport changé ?
903
Oui
Non
Décrémentation de Ntcp_udp
907
Fin
Sudp=0
Wudp= Wudp+MSS/Ntcp_udp
904
Wudp>Wudp-max & Stcp=0
905
Oui
Non
Etablissement du mode TCP
906
Comparaison MTU
910
Stcp=0
908

Fig. 9b

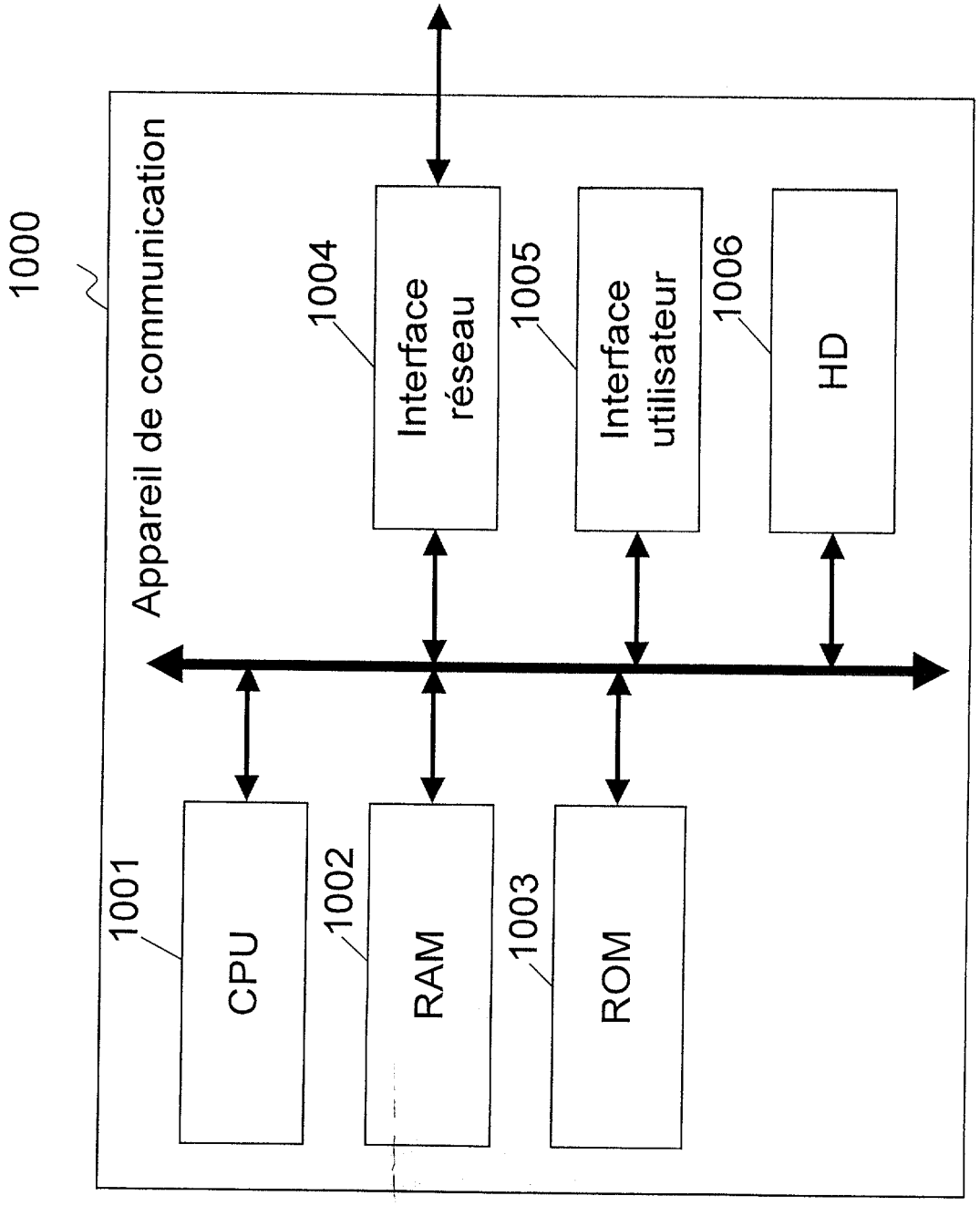

1000
Appareil de communication
1004
Interface réseau
1005
Interface utilisateur
1006
HD
1001
CPU
1002
RAM
1003
ROM

Fig. 10

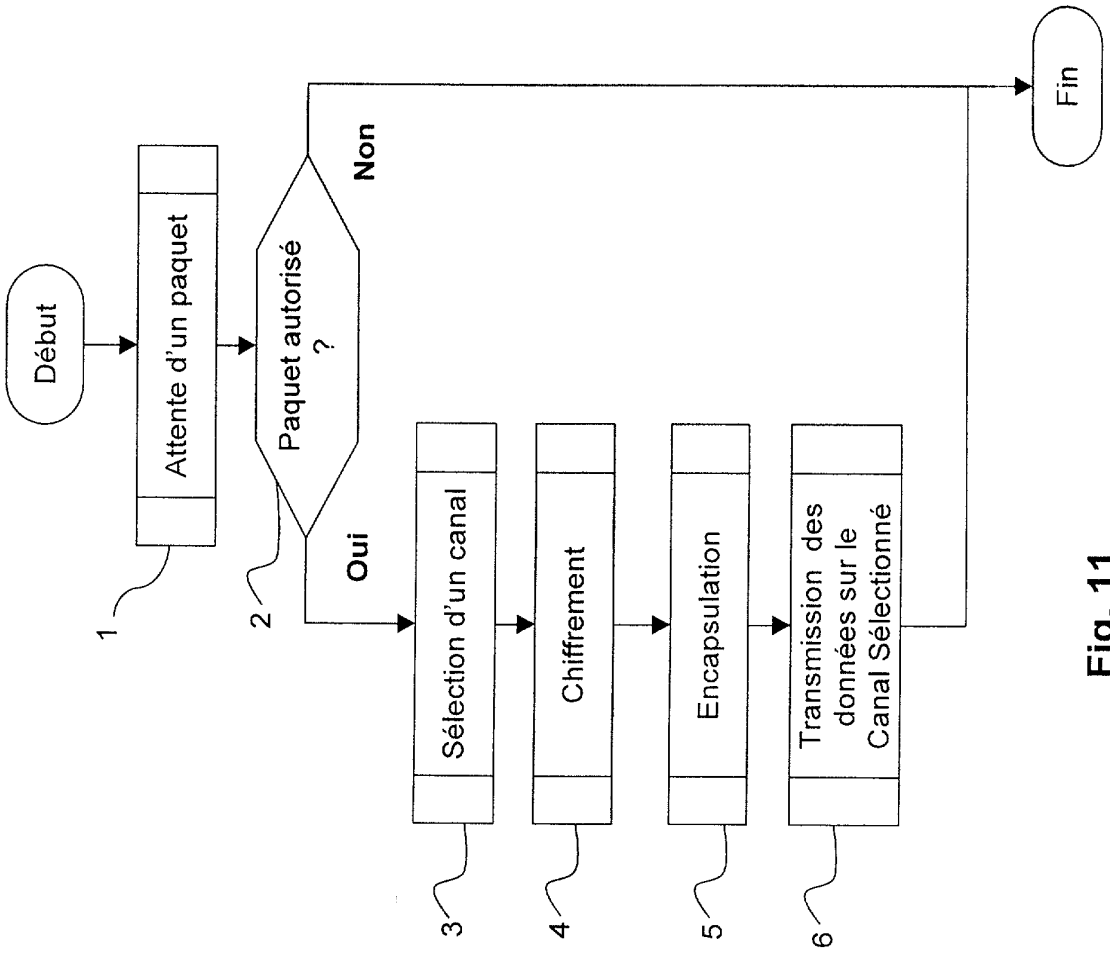

Début
Attente d'un paquet
Paquet autorisé ?
Non
Oui
Fin
Sélection d'un canal
Chiffrement
Encapsulation
Transmission des données sur le Canal Sélectionné
1
2
3
4
5
6

Fig. 11

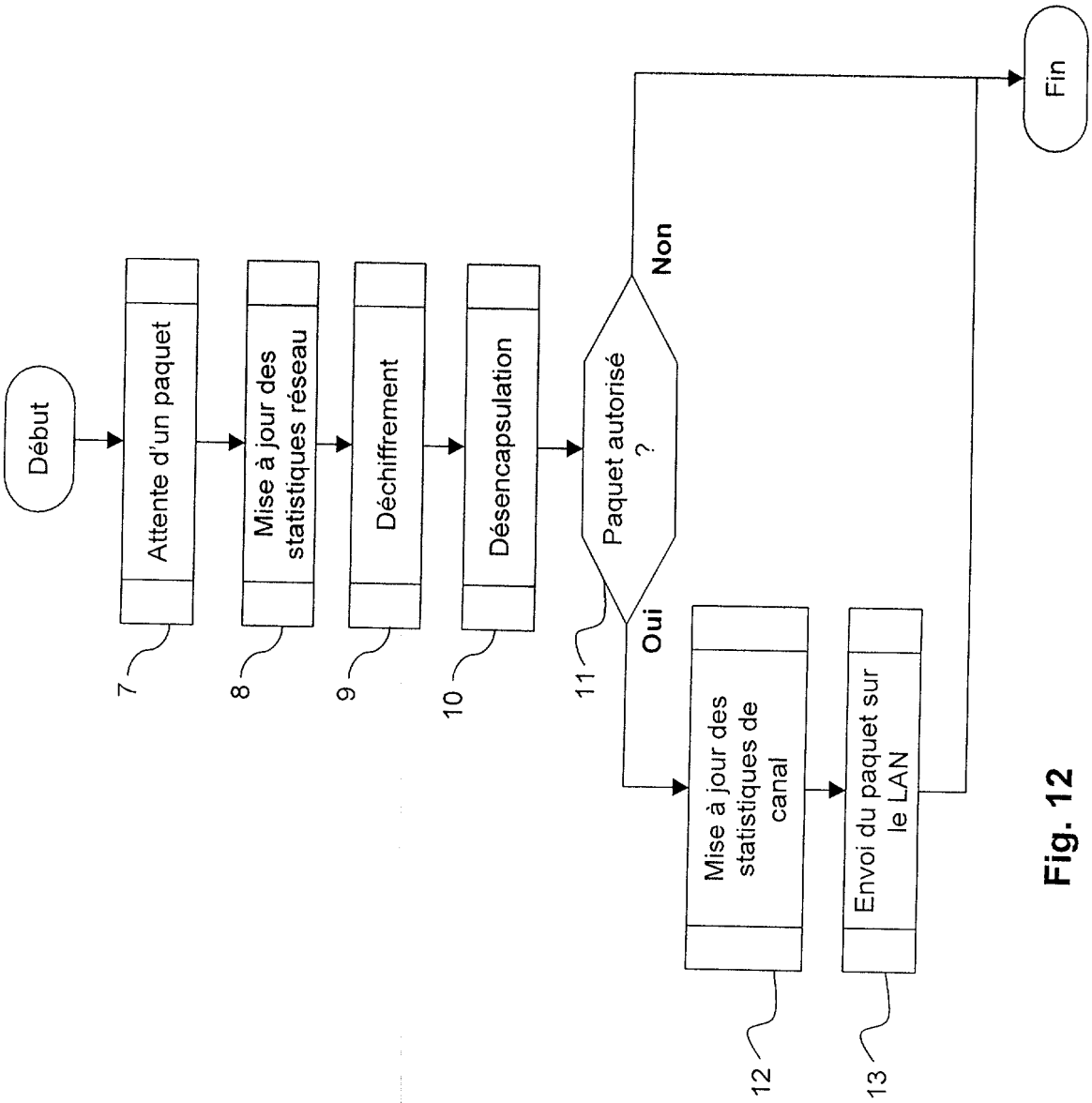

Début
Attente d'un paquet
Mise à jour des statistiques réseau
Déchiffrement
Désencapsulation
Paquet autorisé ?
Non
Oui
Mise à jour des statistiques de canal
Envoi du paquet sur le LAN
Fin
7
8
9
10
11
12
13

Fig. 12

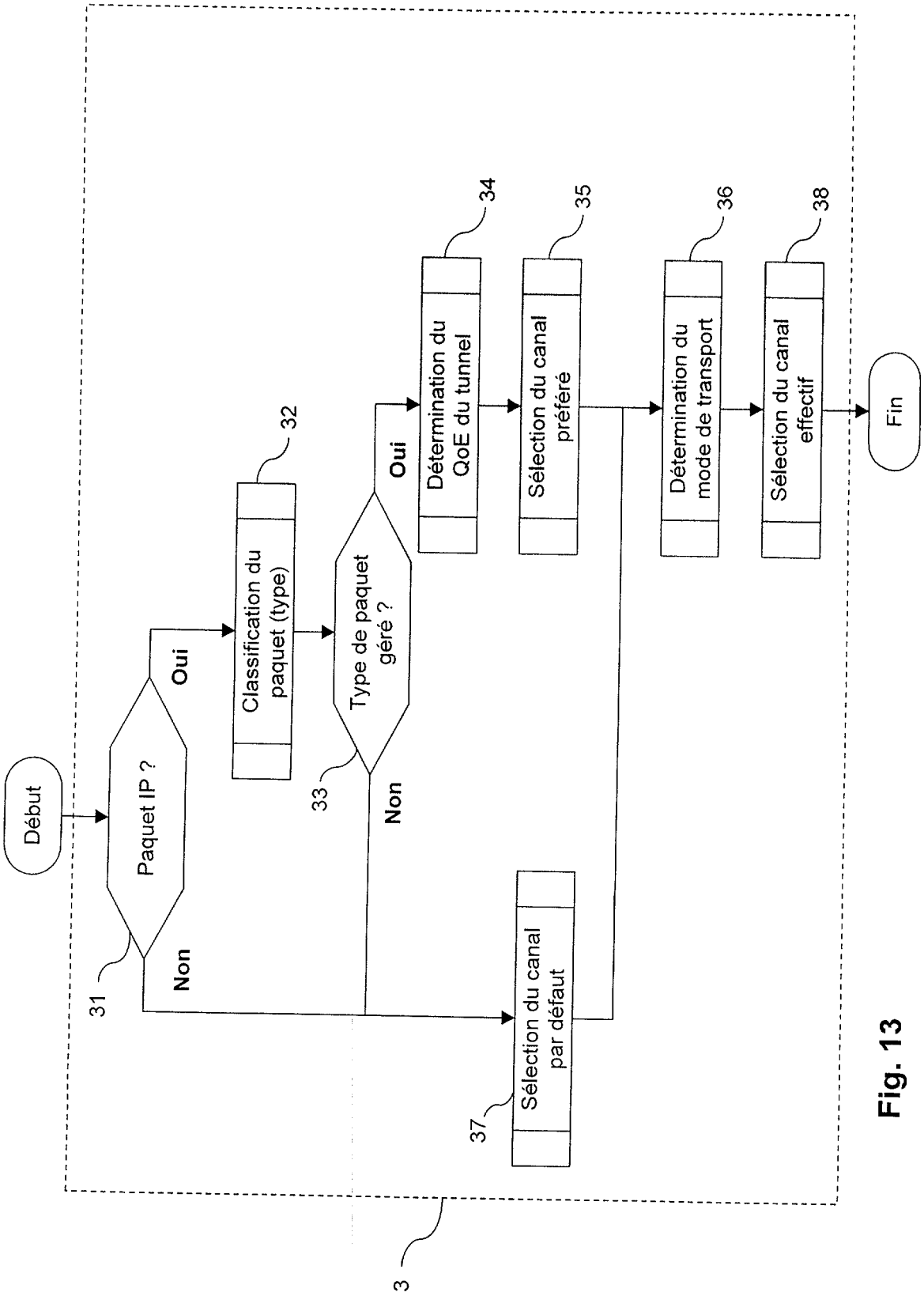

Début
Paquet IP ?
Oui
Non
31
Classification du paquet (type)
32
Type de paquet géré ?
33
Oui
Non
Détermination du QoE du tunnel
34
Sélection du canal préféré
35
Détermination du mode de transport
36
Sélection du canal effectif
38
Fin
Sélection du canal par défaut
37
3

Fig. 13

RÉPUBLIQUE FRANÇAISE

2922068

INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

RAPPORT DE RECHERCHE PRÉLIMINAIRE

établi sur la base des dernières revendications déposées avant le commencement de la recherche

N° d'enregistrement national

FA 701646
FR 0758149

DOCUMENTS CONSIDÉRÉS COMME PERTINENTS		Revendication(s) concernée(s)	Classement attribué à l'invention par l'INPI
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes		
A	CISCO SYSTEMS, INC.: "Resolve IP Fragmentation, MTU, MSS, and PMTUD Issues with GRE and IPSEC" DOCUMENT ID: 25885, [Online] 2 octobre 2006 (2006-10-02), pages 1-27, XP002479899 Extrait de l'Internet: URL:http://www.cisco.com/warp/public/105/pmtud_ipfrag.pdf> [extrait le 2008-05-13] * pages 18,21,24 * -----	1-20	H04L12/56 H04L12/46
A	CHRISTIAN NORTEL NETWORKS P: "Generic Routing Encapsulation over CLNS Networks; rfc3147.txt" IETF STANDARD, INTERNET ENGINEERING TASK FORCE, IETF, CH, 1 juillet 2001 (2001-07-01), XP015008928 ISSN: 0000-0003 Chapitre 6 -----	1-20	
A	US 2007/094723 A1 (SHORT TODD M [US] ET AL) 26 avril 2007 (2007-04-26) * alinéas [0020], [0021] * * alinéas [0026], [0027] * * alinéa [0030] - alinéa [0032] * -----	1-20	DOMAINES TECHNIQUES RECHERCHÉS (IPC) H04L

Date d'achèvement de la recherche	Examinateur
15 mai 2008	Engmann, Steffen

CATÉGORIE DES DOCUMENTS CITÉS

X : particulièrement pertinent à lui seul
Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie
A : arrière-plan technologique
O : divulgation non-écrite
P : document intercalaire

T : théorie ou principe à la base de l'invention
E : document de brevet bénéficiant d'une date antérieure à la date de dépôt et qui n'a été publié qu'à cette date de dépôt ou qu'à une date postérieure.
D : cité dans la demande
L : cité pour d'autres raisons

.....

& : membre de la même famille, document correspondant

EPO FORM 1503 12.99 (P04C14)

ANNEXE AU RAPPORT DE RECHERCHE PRÉLIMINAIRE RELATIF A LA DEMANDE DE BREVET FRANÇAIS NO. FR 0758149 FA 701646

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche préliminaire visé ci-dessus.
Les dits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du 15-05-2008
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets, ni de l'Administration française

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
US 2007094723 A1	26-04-2007	AUCUN	

EPO FORM P0465