SOMMAIRE
INTERNET PROTOCOL
(IP) SPECIFICATION

1. Introduction 1

1.1. Motivation 1
1.2. Cadre 1
1.3. Interfaces 1
1.4. Fonctionnement 1

2. Vue d'ensemble 3

2.1. Relations avec les autres protocoles 3
2.2. Modèle de fonctionnement 3
2.3. Description fonctionnelle 4
2.4. Routeurs 6

3. Spécifications 6

1.1. Format d'en-tête Internet 6
3.1. Discussion 15
3.1.1. Adressage 15
3.1.2. Fragmentation et Réassemblage 16
3.1.3. Exemple de procédure de fragmentation 17
3.1.4. Exemple de procédure de réassemblage 18
3.1.5. Identification 19
3.1.6. Type de Service 20
3.1.7. Durée de vie 20
3.1.8. Options 21
3.1.9. Checksum 21
3.1.10. Erreurs 21
3.2. Interfaces 21
3.2.1. Un exemple d'interface supérieure 22

4. Appendice A: Exemples & Scénarios 23

4.1. Exemple 1: 23
4.2. Exemple 2: 24

4.3.Exemple 3: 25

5. Appendice B: Ordre de transmission des données.....26

6. Glossaire.....26

7. Références.....28

RFC: 791
Remplace: RFC 760

INTERNET PROTOCOL (IP) SPECIFICATION

1. Introduction

1.1. Motivation

Le Protocole Internet est conçu pour supporter l'intercommunication de systèmes informatiques sur une base de réseau par commutation de paquets. Un tel système est appelé "catenet" [1]. Le rôle du protocole Internet est la transmission de blocs de données, appelés datagrammes, d'une source vers une destination, la source et la destination étant des ordinateurs hôtes identifiés par une adresse de longueur fixe. Le protocole Internet dispose des mécanismes permettant la fragmentation de longs datagrammes et leur réassemblage, lors de leur transmission à travers des réseaux de "dimension" inférieure.

1.2. Cadre

Le protocole Internet est limité aux fonctions nécessaires à l'acheminement d'un paquet de bits (un datagramme Internet) depuis une source vers une destination via un ensemble de réseaux interconnectés. Aucun mécanisme particulier destiné à augmenter la fiabilité des données de "bout en bout" n'y est implémenté, ni mécanisme de contrôle de flux, de séquencement, ni aucun autre service communément fourni par d'autres protocoles "hôte vers hôte". Le protocole Internet capitalisera les services des réseaux qui le supportent pour offrir divers types et qualités de service.

1.3. Interfaces

Ce protocole est appelé par d'autres protocoles "hôte-vers-hôte" de l'environnement Internet. Ce protocole appelle à son tour un protocole de réseau local pour transporter le datagramme vers le routeur le plus proche ou directement vers l'hôte destinataire.
Par exemple, un module TCP s'appuiera sur le module Internet pour transporter un segment TCP (comprenant un en-tête TCP plus les données utilisateur) considéré lui-même comme le segment de données du datagramme Internet. Le module TCP renseignera les adresse et les autres paramètres de l'en-tête Internet par passage de paramètres lors de l'appel. Le module Internet constituera alors le datagramme Internet et appellera à son tour l'interface réseau local pour transmettre le datagramme.
Dans le cas d'ARPAnet, par exemple, le module Internet appellera le module réseau local qui ajoutera l'en-tête 1822 [2] en début de datagramme, constituant ainsi un message ARPANET à transmettre à l'IMP. L'adresse ARPANET sera déduite de l'adresse Internet par l'interface réseau local et sera l'adresse d'un hôte raccordé à l'ARPAnet, celui-ci pouvant être un routeur vers d'autres réseaux.

1.4. Fonctionnement

Le protocole Internet implémente deux fonctions de base : l'adressage et la fragmentation. Les modules Internet exploiteront les adresses inscrites dans l'en-tête Internet pour acheminer le datagramme vers sa destination. La sélection d'un chemin partant de la source vers la destination est appelée le **routage**.

Les modules Internet exploitent des champs de l'en-tête Internet pour fractionner et réassembler les datagrammes Internet lorsque le réseau à traverser n'accepte que des paquets de taille plus réduite.

Le modèle de fonctionnement est qu'il existera un module Internet dans chaque hôte concerné par la communication Internet ainsi que dans chaque routeur situé sur le chemin du datagramme. Ces modules partagent un certain nombre de règles communes pour l'interprétation des champs d'adresse et pour la fragmentation et le réassemblage des datagrammes Internet. De plus, ces modules (surtout dans les routeurs) disposeront de fonctions permettant de prendre des décisions de routage ainsi que quelques autres fonctions.

Le protocole Internet considère chaque datagramme Internet comme une entité indépendante et sans relation aucune avec d'autres datagrammes. Il n'y a dans ce concept aucune notion de circuit ou de communication (ni virtuelle ni d'aucun ordre).

Le protocole Internet utilise quatre mécanismes clefs pour procurer le service promis : Type de Service, Durée de Vie, Options, et Checksum d'en-tête.

Le **Type de Service** indique la qualité du service désiré. Le type de service est un ensemble générique de paramètres qui caractérisent les choix de service disponibles sur le réseau qui supporte la communication Internet. Cette indication de type de service sera utilisée par les routeurs pour commuter les paramètres de transmission actuels d'un réseau particulier, le réseau à utiliser sur le segment suivant, ou pour spécifier le routeur suivant lors du routage d'un datagramme.

La **Durée de Vie** indique une limite haute pour la durée d'existence d'un datagramme dans le réseau. Elle est initialisée par l'émetteur du datagramme et décrémentée par les éléments actifs situés sur le chemin que parcourt le datagramme. Si cette durée de vie atteint la valeur zéro avant que le datagramme Internet n'atteigne sa destination, ce dernier sera détruit. Cette durée de vie peut être vue comme une temporisation d'autodestruction du datagramme.

Les **Options** permettent le transport de signaux de contrôle utiles voire nécessaires dans certaines situations particulières mais secondaire pour la fonction essentielle de la communication. Les options permettent par exemple le marquage temporel, le codage de sécurité, et des commandes de routage spéciales.

Le Checksum d'en-tête permet de vérifier que les informations ajoutées par un module Internet ont été correctement transmises. Les données peuvent néanmoins contenir des erreurs. Si le Checksum échoue, le datagramme Internet est immédiatement rejeté par l'entité qui l'a reçu.

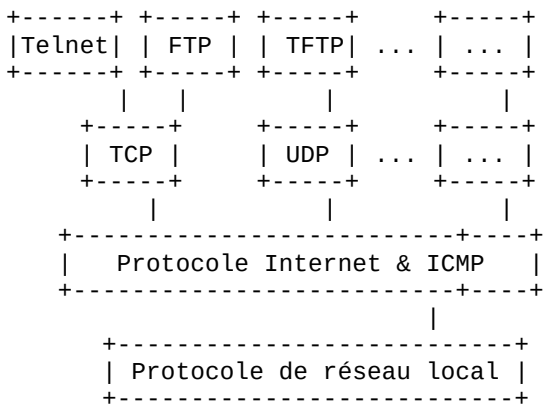
Le protocole Internet ne prend absolument pas en charge le contrôle d'intégrité des données transportées. Il n'intègre aucun mécanisme d'acquittement ni "bout en bout" ni "segment par segment". Aucun contrôle d'erreur n'est effectué sur les données, lequel sera à la charge des modules supérieurs, seule l'en-tête Internet est contrôlée. Il n'y a aucun mécanisme de retransmission de paquet. Il n'y a aucun mécanisme de contrôle de flux.

Les erreurs détectées pourront être signalées via l'Internet Control Message Protocol (ICMP) [3] implémenté dans le module Internet.

2. Vue d'ensemble

2.1. Relations avec les autres protocoles

Le diagramme suivant montre la position du protocole Internet dans la "pile" de protocoles :



Relations entre les protocoles
Figure 1.

Le protocole Internet s'interface d'un côté avec un protocole hôte-vers-hôte de niveau supérieur et de l'autre côté avec un protocole de réseau local. Dans ce contexte, un "réseau local" peut être un petit réseau d'entreprise comme un réseau beaucoup plus étendu comme ARPAnet.

2.2. Modèle de fonctionnement

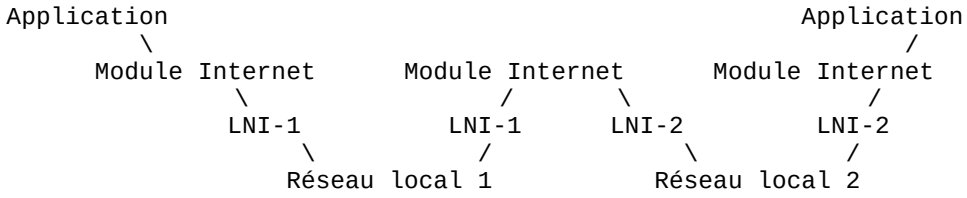
Le modèle de fonctionnement de la transmission d'un datagramme d'un programme d'application vers un autre est illustré par le scénario suivant :

Nous supposons ici que la transmission traverse un routeur intermédiaire. L'application émettrice prépare les données à envoyer et appelle son module Internet local pour envoyer un datagramme en lui passant l'adresse de destination et quelques autres paramètres comme arguments.
Le module Internet prépare une en-tête de datagramme et lui ajoute les données. Le module Internet détermine une adresse réseau locale correspondant à cette adresse Internet, dans notre cas, il s'agit de l'adresse d'un routeur. Il envoie ensuite ce datagramme ainsi que l'adresse réseau locale à l'interface réseau local.

L'interface réseau local crée sa propre en-tête réseau local, et ajoute à son tour le datagramme, puis émet physiquement le paquet ainsi constitué sur le réseau.
Le datagramme arrive sur un routeur hôte encapsulé dans son en-tête réseau local, l'interface réseau local décapsule cette en-tête, et transfère le datagramme vers le module Internet routeur. Le module Internet routeur détermine en fonction de l'adresse Internet vers quel hôte et sur quel nouveau segment de réseau le datagramme doit être transmis. Le module Internet détermine une nouvelle adresse réseau local visant à ce moment l'ordinateur cible. Il appelle l'interface réseau local traitant ce segment pour y reporter le datagramme.

Cette interface réseau local crée une nouvel en-tête réseau local et y attache le datagramme puis transmet le tout sur le nouveau segment de réseau (lequel en l'occurrence supporte l'hôte cible).

Arrivé à destination, le datagramme est extrait de son enrobage réseau local par l'interface réseau local destinataire, puis est transmis au module Internet destinataire.
Le module Internet détermine à quel programme applicatif le datagramme est destiné. Il passe alors les données au programme applicatif en réponse à un appel système, accompagné de l'adresse de la source et de quelques autres paramètres.



Chemin de transmission
Figure 2

2.3. Description fonctionnelle

La fonction ou rôle du Protocole Internet est d'acheminer les datagrammes à travers un ensemble de réseaux interconnectés. Ceci est réalisé en transférant les datagrammes d'un module Internet à l'autre jusqu'à atteindre la destination. Les modules Internet sont des programmes exécutés dans des hôtes et des routeurs du réseau Internet. Les datagrammes sont transférés d'un module Internet à l'autre sur un segment particulier de réseau selon l'interprétation d'une adresse Internet. De ce fait, un des plus importants mécanismes du protocole Internet est la gestion de cette adresse Internet.
Lors de l'acheminement d'un datagramme d'un module Internet vers un autre, les datagrammes peuvent avoir éventuellement à traverser une section de réseau qui admet une taille maximale de paquet inférieure à celle du datagramme. Pour surmonter ce problème, un mécanisme de fragmentation est géré par le protocole Internet.

Adressage

Une distinction doit être faite entre *noms, adresses,* et *chemins* [4]. Un nom indique ce que nous cherchons. Une adresse indique où cela se trouve. Un chemin indique comment y aboutir. Le protocole Internet s'occupe essentiellement des adresses. C'est à des protocoles de niveau plus élevé (ex., hôte-vers-hôte ou application) que revient la tâche de lier des noms à des adresses. Le module Internet déduit de l'adresse Internet une adresse réseau local. La tâche qui consiste à transcrire l'adresse de réseau local en termes de chemin (ex., sur un réseau local ou dans un routeur) revient au protocole de bas niveau.

Les adresses ont une longueur fixe de 4 octets (32 bits). Une adresse commence toujours par un numéro de réseau, suivi d'une adresse locale (appelée le champ "reste") codant l'adresse de l'hôte sur ce réseau. Il existe trois formats ou classes d'adresses Internet : pour la classe A, le bit de poids fort vaut zéro, les 7 bits suivants désignent le réseau, les derniers 24 bits désignent l'adresse locale de la machine; pour la classe B, les deux bits de poids fort valent 1 et 0, les 14 bits suivants désignent le réseau et les 16 derniers bits l'adresse locale de machine ; pour la classe C, les trois bits de poids fort forment le schème 110, les 21 bits suivants forment l'adresse réseau et les 8 derniers bits l'adresse locale.

La transcription d'adresse Internet en adresses de réseau local doit être sujette à quelques précautions ; un hôte physique unique peut abriter plusieurs adresses Internet distinctes comme s'il s'agissait de

plusieurs hôtes indépendants. Certains hôtes peuvent disposer de plusieurs interfaces physiques (multi-homing).
De ce fait, il faudra pouvoir considérer le cas d'un hôte à plusieurs interfaces physiques chacune abritant plusieurs adresses Internet distinctes.
Des exemples de répartition d'adresses peuvent être trouvés dans "Address Mappings" [5].

Fragmentation

La fragmentation du datagramme Internet devient nécessaire dès lors qu'un datagramme de grande taille arrive sur une portion de réseau qui n'accepte la transmission que de paquets plus courts.
Un datagramme Internet peut être spécifié "non fractionnable" Un tel datagramme Internet ne doit jamais être fragmenté quelques soient les circonstances. Si un datagramme Internet non fractionnable ne peut être acheminé jusqu'à sa destination sans être fragmenté, alors il devra être rejeté.
La fragmentation, la transmission et le réassemblage à travers un réseau local hors de vue d'un module de protocole Internet est appelée fragmentation Intranet [6].

Les procédures de fragmentation et réassemblage Internet doivent pouvoir "casser" un datagramme Internet en un nombre de "fragments" arbitraire et quelconque pourvu que le réassemblage soit possible. Le récepteur des fragments utilise le champ d'identification pour s'assurer que des fragments de plusieurs datagrammes ne puissent être mélangés. Le champ "Fragment Offset" indique au récepteur la position du fragment reçu dans le datagramme original. Les champs "Fragment Offset" et "Longueur Totale" déterminent la portion du datagramme original que représente le fragment. L'indicateur bit "Dernier Fragment" indique (lors de sa remise à zéro) au récepteur qu'il s'agit du dernier fragment. Ces champs véhiculent suffisamment d'information pour réassembler les datagrammes.

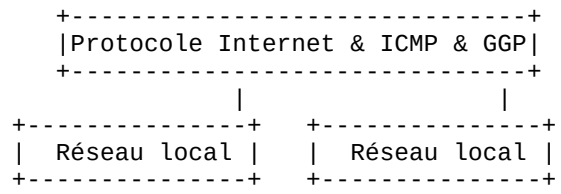
Le champ d'identification sert à distinguer les fragments d'un datagramme de ceux d'un autre datagramme. Le module Internet émetteur d'un datagramme Internet initialise le champ d'identification à une valeur qui doit être unique pour cette paire source-destination et pour ce protocole pendant toute la durée de transmission de ce datagramme. Le module Internet terminant l'émission d'un datagramme met le bit "Dernier Fragment" et le champ "Fragment Offset" à zéro.
Pour fragmenter un long datagramme, un module Internet (par exemple, dans un routeur), crée deux nouveaux datagrammes et copie le contenu des champs d'en-tête Internet originaux dans les deux nouvel en-têtes. Les données du datagramme original sont divisées en deux portions, la première d'une taille multiple de 8 octets (64 bit) (la taille de la seconde portion n'est donc pas nécessairement un multiple de 8 octets). Nous appellerons le nombre de blocs de 8 octets dans la première portion NBF (ou Nombre de Blocs du Fragment). La première portion de données est placée dans le premier des deux nouveaux datagramme, et le champ "Longueur Totale" est renseigné avec la taille de ce datagramme. Le bit "Dernier Fragment" est basculé à 1. La seconde portion de données est placée dans le second des deux nouveaux datagrammes, et le champ "longueur totale" est renseigné avec la taille du second datagramme. Le bit "Dernier Fragment" est placé à la même valeur que celui du datagramme original. Le champ "Fragment Offset" du second datagramme constitué est renseigné avec la valeur du même champ du datagramme original plus NFB.

Cette procédure peut être généralisée à une fragmentation en n fragments, plutôt que les deux décrits ci-dessus.
Pour réassembler les fragments d'un datagramme Internet, un module Internet (par exemple dans un hôte destinataire) recombine les datagrammes dont les valeurs des quatre champs suivants sont identiques : identification, source, destination, et protocole. La recombinaison est réalisée en replaçant la portion de donnée contenue dans chaque fragment dans un tampon à la position relative indiquée par

le champ "Fragment Offset" lu dans l'en-tête correspondant. Le premier fragment sera donc placé en début de tampon, et le dernier fragment récupéré aura le bit "Dernier Fragment" à zéro.

2.4. Routeurs

Les routeurs implémentent le protocole Internet pour transférer les datagrammes entre réseaux différents. Les routeurs implémenteront de plus le protocole routeur vers routeur ou Gateway to Gateway Protocol (GGP) [7] leur servant à coordonner le routage et à s'échanger d'autres informations de gestion du réseau.
Dans un routeur, les protocoles de niveau supérieur n'ont pas à être reconnus. Les fonctions GGP sont ajoutées dans l'implémentation du module IP.

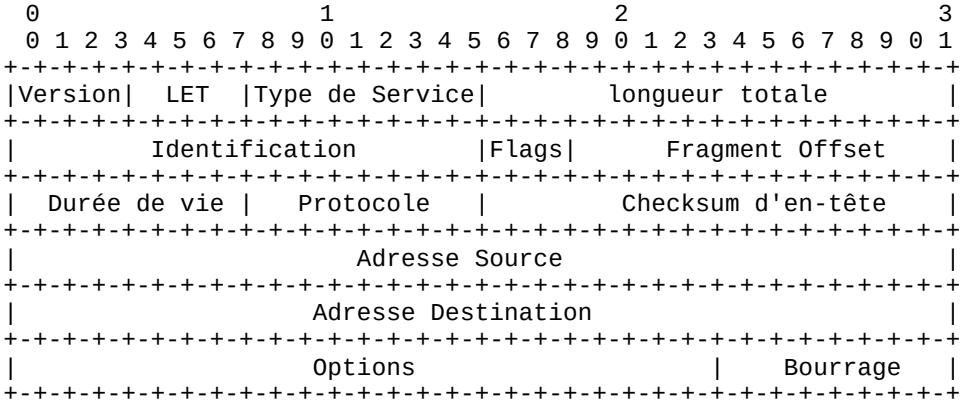


Protocoles dans les routeurs
Figure 3.

3. Spécifications

1.1. Format d'en-tête Internet

Un résumé du contenu de l'en-tête Internet suit :



Exemple d'en-tête de Datagramme Internet
Figure 4.

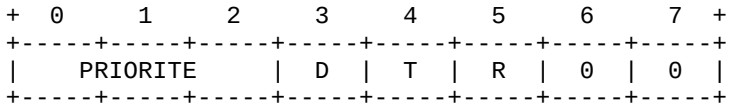
Notez que chaque marque indique une position bit.

Version : 4 bits
Le champ Version renseigne sur le format de l'en-tête Internet. Ce document décrit le format de la version 4 du protocole.

Longueur d'En-Tête : 4 bits
Le champ Longueur d'En-Tête (LET) code la longueur de l'en-tête Internet, l'unité étant le mots de 32 bits, et de ce fait, marque le début des données. Notez que ce champ ne peut prendre une valeur en dessous de 5 pour être valide.

Type de Service : 8 bits
Le Type de Service donne une indication sur la qualité de service souhaitée, qui reste cependant un paramètre "abstrait". Ce paramètre est utilisé pour "guider" le choix des paramètres des services actuels lorsqu'un datagramme transite dans un réseau particulier. Certains réseaux offrent un mécanisme de priorité, traitant préférentiellement un tel trafic par rapport à un trafic moins prioritaire (en général en acceptant seulement de véhiculer des paquets d'un niveau de priorité au dessus d'un certain seuil lors d'une surcharge momentanée). Principalement, le choix offert est une négociation entre les trois contraintes suivantes : faible retard, faible taux d'erreur, et haut débit.

```
Bits 0-2 :      Priorité.
Bit    3 :      0 = Retard standard,          1 = Retard faible.
Bits   4 :      0 = Débit standard,           1 = Haut débit.
Bits   5 :      0 = Taux d'erreur standard    1 = Taux d'erreur
                                              faible.
Bit  6-7 :      Réservé.
```



Priorité

```
111 - Network Control
110 - Internetwork Control
101 - CRITIC/ECP
100 - Flash Override
011 - Flash
010 - Immediate
001 - Priority
000 - Routine
```

L'utilisation des indications en termes de retard, débit, et qualité de transmission peut augmenter le "coût" (d'un certain point de vue) du service. Dans la plupart des réseaux, de meilleures performances pour l'un de ces paramètres s'obtient au prix d'une dégradation des performances pour un autre. A moins d'une situation exceptionnelle, il sera préférable de ne pas activer plus de deux optimisations sur les trois.

Le "Type de Service" sert à préciser le traitement effectué sur le datagramme pendant sa transmission à travers Internet. Des exemples d'association de ce code aux améliorations de service proposées par des réseaux existants comme AUTODIN II, ARPANET, SATNET, et PRNET sont données dans la RFC 795 "Service Mappings" [8].

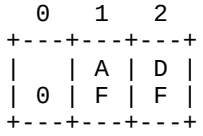
La priorité dite "Network Control" est stipulée comme étant une priorité à l'intérieur d'un seul réseau. Le fait d'utiliser cette option instaure une priorité pour chaque section traversée. La priorité "Internetwork Control" n'est gérée que par les routeurs. Si l'utilisation de ces priorités ont une signification particulière ou supplémentaire pour l'un des réseaux, il est de la responsabilité de ce dernier de lire et d'interpréter les présentes informations.

Longueur Totale : 16 bits
Le champ "Longueur Totale" est la longueur du datagramme entier y compris en-tête et données, mesurée en octets. Ce champ ne permet de coder qu'une longueur de datagramme d'au plus 65,535 octets. Une telle longueur rendrait de toutes façon les datagrammes impossible à gérer pour la plus grande partie des réseaux. Les hôtes devront au moins pouvoir accepter des datagrammes d'une longueur jusqu'à 576 octets (qu'il s'agisse d'un datagramme unique ou d'un fragment). Il est de même recommandé que des hôtes ne décident d'envoyer des datagrammes de plus de 576 octets que dans la mesure où ils sont sûrs que la destination est capable de les accepter.
Le nombre 576 a été choisi pour permettre à un bloc de données de taille raisonnable d'être transmis dans un datagramme, tenant compte des données à ajouter pour constituer les en-têtes de protocole. Par exemple, cette taille permet la transmission d'un bloc de 512 octets, plus 64 octets d'en-tête dans un datagramme unique. (*NdT : je rappelle ici que la taille de 512 octets correspond à un secteur sur la plupart des supports de stockage)* La taille maximale d'un en-tête Internet étant de 60 octets, et sa taille typique étant de 20 octets, ce nombre permet de conserver une bonne marge pour les données protocolaires de plus haut niveau.

Identification : 16 bits
Une valeur d'identification assignée par l'émetteur pour identifier les fragments d'un même datagramme.

Flags : 3 bits
Divers commutateurs de contrôle.

```
Bit 0: réservé, doit être laissé à zéro
Bit 1: (AF) 0 = Fragmentation possible,  1 = Non fractionnable.
Bit 2: (DF) 0 = Dernier fragment, 1 = Fragment intermédiaire.
```



Position relative : 13 bits
Ce champ indique le décalage du premier octet du fragment par rapport au datagramme complet. Cette position relative est mesurée en blocs de 8 octets (64 bits). Le décalage du premier fragment vaut zéro.

Durée de vie : 8 bits
Ce champ permet de limiter le temps pendant lequel un datagramme reste dans le réseau. Si ce champ prend la valeur zéro, le datagramme doit être détruit. Ce champ est modifié pendant le traitement de l'en-tête Internet. La durée de vie est mesurée en secondes. Chaque module Internet doit retirer au moins une unité de temps à ce champ, même si le traitement complet du datagramme par le module est effectué en moins d'une seconde. De ce fait, cette durée de vie doit être interprétée comme la limite absolue maximale de temps pendant lequel un datagramme peut exister. Ce mécanisme est motivé par la nécessité de détruire les datagrammes qui n'ont pu être acheminés, en limitant la durée de vie même du datagramme.

Protocole : 8 bits
Ce champ indique quel protocole de niveau supérieur est utilisé dans la section données du datagramme Internet. Les différentes valeurs admises pour divers protocoles sont listée dans la RFC "Assigned Numbers" [9].

Checksum d'en-tête : 16 bits
Un Checksum calculé sur l'en-tête uniquement. Comme certains champs de l'en-tête sont modifiés (ex., durée de vie) pendant leur transit à travers le réseau, ce Checksum doit être recalculé et vérifié en chaque point du réseau où l'en-tête est réinterprétée.

L'algorithme utilisé pour le Checksum est le suivant :
On calcule le complément à un sur 16 bits de la somme des compléments à un de tous les octets de l'en-tête pris par paires (mots de 16 bits). Lorsque l'on calcule le Checksum, on considère une en-tête dont le champ réservé pour ce même Checksum vaut zéro.

L'algorithme de Checksum peut paraître élémentaire mais l'expérimentation a montré que cette technique était suffisante. Il se peut que cet algorithme soit plus tard remplacé par un calcul de type CRC, suivant la nécessité future.

Adresse source : 32 bits
L'adresse Internet de la source. Cf. section 3.2.

Adresse destination : 32 bits
L'adresse Internet du destinataire. Cf. section 3.2.

Options : variable
Les datagrammes peuvent contenir des options. Celles-ci doivent être implémentées par tous les modules IP (hôtes et routeurs). Le caractère "optionnel" concerne leur transmission, et non leur implémentation.
Dans certains environnements, l'option de sécurité peut être obligatoire dans tous les datagrammes.
Le champ d'option est de longueur variable. Un datagramme peut comporter zéro ou plus options. Voici les deux formats possibles d'une option :

Cas 1: Une option codée sur un seul octet.

Cas 2: Un octet codant le type d'option, un octet donnant la taille de l'option, les octets de données d'option.

La taille de l'option compte tous les octets de l'option y compris le type, son propre octet et tous les octets de donnée d'option.

L'octet de type d'option est composé de trois champs de bits :

```
1 bit   indicateur de recopie,
2 bits  classe d'option,
5 bits  numéro d'option.
```

L'indicateur de recopie marque le fait que l'option est recopiée dans tous les segments d'un datagramme fragmenté.

```
0 = non recopiée
1 = recopiée
```

Les classe d'option sont :

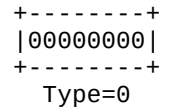
```
0 = contrôle
1 = réservé pour usage futur
2 = débogage et mesure
3 = réservé pour usage futur
```

Les options suivantes sont actuellement définies :

CLASSE	NUMERO	LONGUEUR	DESCRIPTION
0	0	-	Fin de liste d'option. Sur un seul octet pas d'octet de taille.
0	1	-	Pas d'opération. Sur un seul octet pas d'octet de taille.
0	2	11	Sécurité. Transporte les informations de sécurité, compartiment, Groupe utilisateur (TCC), et Codes de Restriction compatibles DOD (application militaire).
0	3	var.	Routage lâche. Utilisé pour acheminer le datagramme selon des informations données par la source.
0	9	var.	Routage strict. Utilisé pour acheminer le datagramme selon des informations données par la source.
0	7	var.	Traceur. Utilisé pour mémoriser le chemin pris par un datagramme Internet.
0	8	4	ID de flux. Transporte l'identificateur du flux.
2	4	var.	Marqueur temporel.

Définition des options spécifiques

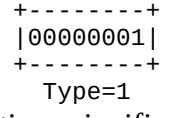
Fin de liste d'option



Cette option indique la fin de la liste d'options qui ne coïncide pas nécessairement avec la fin de l'en -tête, selon la définition de la longueur de celle-ci. Cette option est utilisable une fois à la fin du bloc d'options, et non pas après chaque option, et peut n'être utilisée que dans le cas où la fin de liste d'options ne peut coïncider avec la fin de l'en -tête Internet. (*NdT : Rappel, une en-tête IP comporte toujours un multiple de 4 octets*).

Cet octet peut être recopié, introduit ou supprimé lors d'opérations de fragmentation, ou pour toute autre raison.

Pas d'opération

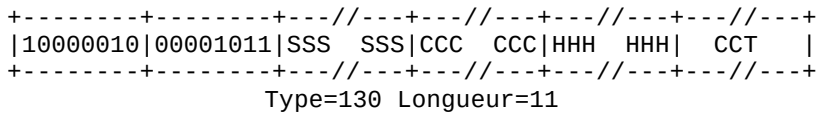


Cette option peut être utilisée entre deux options significatives, par exemple, pour aligner le début de l'option suivante sur le début d'un mot de 32 bits.

Peut être recopié, introduit, ou supprimé lors d'opérations de fragmentation, ou pour toute autre raison.

Sécurité

Cette option permet à un hôte d'envoyer des informations de sécurité, compartimentation, restrictions d'usage, et CCT (groupe fermé). Le format de cette option est le suivant :



Sécurité (Champ S) : 16 bits
Définit un niveau de sécurité parmi 16 (dont 8 sont réservés pour usage futur).

```
00000000 00000000 - Non classé
11110001 00110101 - Confidentiel
01111000 10011010 - EFTO
10111100 01001101 - MMMM
01011110 00100110 - PROG
10101111 00010011 - Restreint
11010111 10001000 - Secret
01101011 11000101 - Top Secret
00110101 11100010 - (Réservé pour usage futur)
10011010 11110001 - (Réservé pour usage futur)
01001101 01111000 - (Réservé pour usage futur)
00100100 10111101 - (Réservé pour usage futur)
00010011 01011110 - (Réservé pour usage futur)
10001001 10101111 - (Réservé pour usage futur)
11000100 11010110 - (Réservé pour usage futur)
11100010 01101011 - (Réservé pour usage futur)
```

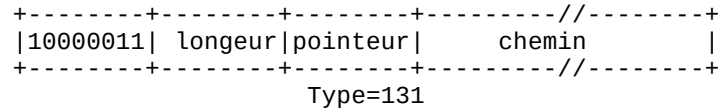
Compartiments (Champ C): 16 bits
Une valeur nulle de ce champ indique que l'information n'est pas compartimentée. Les autres valeurs admissibles sont attribuées par la "Defense Intelligence Agency" américaine.

Restrictions d'usage (Champ H) : 16 bits
Les valeurs pour marquer la prise de contrôle et la levée de restrictions sont des digraphes alphanumériques définis dans le "Defense Intelligence Agency Manual" DIAM 65-19, "Standard Security Markings".

Code de Contrôle de Transmission (Champ CCT) : 24 bits
Procure un moyen de différentier le trafic et de définir des groupes contingentés d'utilisateurs partageant un même centre d'intérêt. Les valeurs de CCT sont des trigraphes, et sont attribués par le HQ DCA Code 530.
Cette option est à recopier impérativement lors d'une fragmentation. Elle doit apparaître au plus une fois dans un datagramme.

Routage lâche et enregistrement du chemin

NdT : le paragraphe ci-dessous est la traduction stricte de la norme. La rédaction originale pouvant apparaître comme quelque peu obscure, vous trouverez en fin de paragraphe un commentaire explicatif du principe de cette option.



L'option de routage lâche et d'enregistrement de chemin (LSRR) permet à la source d'un datagramme Internet de transmettre des informations de routage à destination des routeurs qui acheminent le datagramme vers la destination, et d'enregistrer les indications de chemin parcouru.
Cette option débute avec l'octet de type de l'option. Le second octet donne la longueur de cette option en comptant les deux premiers octets, l'octet pointeur, et longueur-3 octets de données de chemin. Le troisième octet contient une valeur de décalage relatif pointant, dans le champ de chemin, le premier octet de l'adresse Internet de routage suivante à traiter. Ce décalage se calcule relativement au premier octet de l'option, et accepte comme valeur minimale la valeur 4.

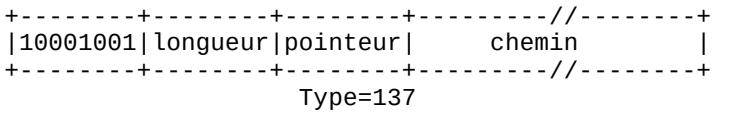
Un chemin est composé d'une liste d'adresses Internet. Chaque adresse étant codée sur 32 bits, et donc 4 octets. Si la valeur du pointeur est plus grande que la longueur d'option, le chemin source est vide (et le chemin enregistré plein) et le routage doit prendre comme référence le champ d'adresse destinataire.
Si l'adresse contenue dans le champ d'adresse destinataire a été atteinte et le pointeur est supérieur à la longueur, l'adresse suivante de source remplace le contenu du champ d'adresse, et l'adresse enregistrée remplace l'adresse source utilisée, le pointeur est augmenté de quatre unités.

L'adresse enregistrée correspond à l'adresse du module Internet qui est en train de traiter l'en-tête pour réaliser l'acheminement.
Cette procédure qui consiste à remplacer l'adresse source par l'adresse enregistrée (bien que le chemin soit inscrit dans l'ordre inverse que ce qui serait nécessaire pour répondre au datagramme en utilisant le chemin inverse) permet de conserver à cette option (ainsi qu'à l'adresse IP en général) une longueur constante tout au long du "voyage" du datagramme à travers Internet.

Cette option spécifie un routage "lâche" en ce sens qu'un routeur ou un hôte IP est autorisé à choisir n'importe quel autre routeur ou hôte intermédiaire qui se situe entre lui même et le destinataire final.
Doit impérativement être reporté lors d'une fragmentation. Ne peut apparaître qu'une fois au plus dans un datagramme.

Note : Il faut comprendre le champ "chemin" comme une liste des adresses Internet de chaque module intermédiaire entre la source et le destinataire, constituant un chemin "préférentiel" tel que le connaît l'émetteur du datagramme. Au fur et à mesure que le datagramme progresse dans le réseau, chaque adresse est effectivement remplacée par celle du module réellement traversé par le datagramme. Le routage est dit "lâche" car le chemin suivi effectivement par le datagramme n'est pas obligatoirement celui qui est préconisé par la liste initiale fournie par la source.

Routage strict et enregistrement de chemin



L'option de routage lâche et d'enregistrement de chemin (LSRR) permet à la source d'un datagramme Internet de transmettre des informations de routage à destination des routeurs qui acheminent le datagramme vers la destination, et d'enregistrer les indications de chemin parcouru.

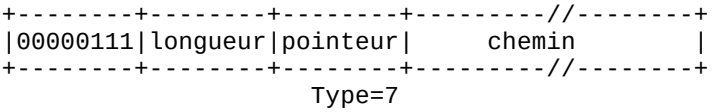
Cette option débute avec l'octet de type de l'option. Le second octet donne la longueur de cette option en comptant les deux premiers octets, l'octet pointeur, et longueur-3 octets de données de chemin. Le troisième octet contient une valeur de décalage relatif pointant, dans le champ de chemin, le premier octet de l'adresse Internet de routage suivante à traiter. Ce décalage se calcule relativement au premier octet de l'option, et accepte comme valeur minimale la valeur 4.

Un chemin est composé d'une liste d'adresses Internet. Chaque adresse étant codée sur 32 bits, et donc 4 octets. Si la valeur du pointeur est plus grande que la longueur d'option, le chemin source est vide (et le chemin enregistré plein) et le routage doit prendre comme référence le champ d'adresse destinataire.
Si l'adresse contenue dans le champ d'adresse destinataire a été atteinte et le pointeur est supérieur à la longueur, l'adresse suivante de source remplace le contenu du champ d'adresse, et l'adresse enregistrée remplace l'adresse source utilisée, le pointeur est augmenté de quatre unités.
L'adresse enregistrée correspond à l'adresse du module Internet qui est en train de traiter l'en-tête pour réaliser l'acheminement.

Cette procédure qui consiste à remplacer l'adresse source par l'adresse enregistrée (bien que le chemin soit inscrit dans l'ordre inverse que ce qui serait nécessaire pour répondre au datagramme en utilisant le chemin inverse) permet de conserver à cette option (ainsi qu'à l'adresse IP en général) une longueur constante tout au long du "voyage" du datagramme à travers Internet.

Cette option spécifie un routage "strict" en ce sens qu'un routeur ou un hôte IP doit obligatoirement choisir le routeur ou hôte intermédiaire suivant tel que préconisé par la route source.
Doit impérativement être recopié lors d'une fragmentation. Doit apparaître au plus une fois dans un datagramme.

Traceur



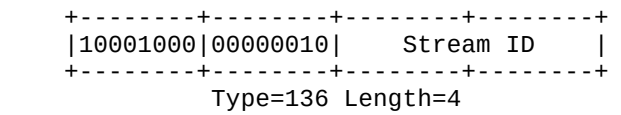
L'option traceur permet d'enregistrer le chemin parcouru par un datagramme Internet.
Cette option débute avec l'octet de type de l'option. Le second octet donne la longueur de cette option en comptant les deux premiers octets, l'octet pointeur, et longueur-3 octets de données de chemin. Le troisième octet contient une valeur de décalage relatif pointant, dans le champ de chemin, le premier octet ou doit être enregistrée l'adresse Internet suivante. Ce décalage se calcule relativement au premier octet de l'option, et accepte comme valeur minimale la valeur 4.

Un chemin est composé d'une liste d'adresses Internet. Chaque adresse étant codée sur 32 bits, et donc 4 octets. Si la valeur du pointeur est plus grande que la longueur d'option, le chemin enregistré est plein. L'émetteur du datagramme devra composer cette option en prévoyant une taille de liste initiale suffisamment longue pour pouvoir enregistrer autant d'adresses de modules intermédiaires que le datagramme est supposé traverser. La taille de l'option ne doit effectivement plus changer lors de l'enregistrement effectif du chemin. Le chemin, au départ du datagramme est initialisé avec des zéros par l'émetteur.

Lorsqu'un module Internet traite un datagramme, il doit vérifier si celui-ci comporte un traceur. Si c'est le cas, il insère sa propre adresse Internet à la position de chemin indiquée par le pointeur, puis incrémente le pointeur de quatre unités.
Si cette liste d'adresse est entièrement remplie (le pointeur excède la longueur de l'option), le datagramme est retransmis sans enregistrer la nouvelle adresse du module Internet actuel. S'il reste de la place dans la liste, mais que cette place est trop petite pour insérer une adresse complète, alors cela indique une erreur et le datagramme doit être détruit. Dans ces deux cas, un message d'erreur ICMP doit être envoyé au hôte source [3].

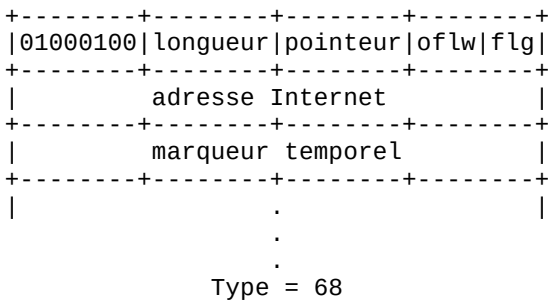
Ne doit pas être recopié lors d'une fragmentation, mais apparaître seulement dans le premier fragment. Ne peut apparaître qu'une fois au plus dans un datagramme.

Identificateur de flux



Permet la transmission d'un identificateur de flux 16-bits SATNET à travers des réseaux qui ne supportent pas la notion de flux.
Doit être recopié lors de fragmentation. Ne peut apparaître au plus une fois dans un datagramme.

Marqueur temporel



La longueur compte le nombre d'octets de l'option y compris le type, la longueur (lui-même), le pointeur, et l'octet de dépassement de capacité/commutateurs (longueur maximale 40).
Le Pointeur contient une valeur qui pointe sur le premier octet de la première place libre pour un nouveau marqueur temporel. L'origine de ce décalage relatif est pris au début de l'option, et donc le décalage minimum acceptable est 5. La liste de marqueurs est pleine lorsque la valeur du pointeur dépasse la longueur de l'option.
Le champ de dépassement de capacité (oflw) [4 bits] compte le nombre de modules IP qui n'ont pas pu enregistrer de marqueur temporel faute de place dans la liste.

Les commutateurs ou bits de contrôle [4 bits] ont les significations suivantes :

```
0 -- Etiquettes temporelles seules, enregistrées sous forme de mots
     consécutifs de 32 bits.

1 -- chaque étiquette est précédée de l'adresse Internet de l'entité
     qui l'a enregistrée.

3 -- les adresses Internet sont spécifiés dès le départ.
     Un module IP n'enregistre l'étiquette que si son adresse Internet
     propre correspond à l'adresse suivante spécifiée dans la liste.
```

L'étiquette temporelle compte sur 32-bits le temps écoulé depuis 0 heures UT en millisecondes, et est justifiée à droite. Si cette valeur n'est pas disponible en millisecondes ou ne peut être calculé à partir de la référence 0 heures UT, alors la valeur disponible sera marquée dans l'étiquette et le bit de poids fort sera marqué à un pour prévenir de l'utilisation d'un format non standard.

L'émetteur du datagramme devra composer cette option en prévoyant une taille de liste initiale suffisamment longue pour pouvoir enregistrer autant d'adresses de modules intermédiaires que le datagramme est supposé traverser. La taille de l'option ne doit effectivement plus changer lors de l'enregistrement effectif des étiquettes. Dans la liste initiale, les adresses auront été marquées par l'émetteur, et les étiquettes initialisées à zéro.
Si la liste d'étiquettes temporelles est pleine (le pointeur pointe au delà de l'en-tête), le datagramme est retransmis sans ajout de nouvelle étiquette, mais le compteur de dépassement de capacité est incrémenté.

S'il reste de la place dans la liste, mais insuffisamment pour enregistrer une étiquette temporelle complète, ou si le champ de dépassement de capacité lui-même est au maximum de sa valeur, le datagramme est considéré en erreur et sera détruit. Dans ces deux cas, un message d'erreur ICMP doit être retourné à l'émetteur [3].
L'option de marquage temporel ne doit pas être recopiée lors d'une fragmentation. Elle est transportée dans le premier fragment. Doit apparaître au plus une fois dans un datagramme.

Bourrage : variable
Le champ de bourrage n'existe que pour assurer à l'en-tête une taille totale multiple de 4 octets. Le bourrage se fait par des octets à zéro.

3.1. Discussion

L'implémentation d'un protocole doit répondre au principe de robustesse. Chaque implémentation doit s'attendre à pouvoir opérer face à une autre implémentation programmée par quelqu'un d'autre. Bien que la fonction de cette spécification soit de décrire explicitement ce protocole, il reste néanmoins la possibilité de voir apparaître des interprétations divergentes. On adopte comme principe général qu'implémentation doit être stricte quant à ce qu'elle émet, et libérale par rapport à ce qu'elle reçoit. C'est à dire qu'elle doit faire attention à émettre des datagrammes conformes et correctement constitués, mais doit accepter tout datagramme qu'elle est en mesure d'interpréter (ex., exempt d'erreurs d'ordre technique et tant que sa signification reste déchiffrable).

Les services de base d'Internet s'appuient sur le concept datagramme qui prévoit une possibilité de fragmentation par les routeurs, avec une fonction de réassemblage exécutée par le module Internet du destinataire. Bien sûr, la fragmentation et le réassemblage des datagrammes, localement à un segment de réseau ou suite à un accord particulier entre deux routeurs situés sur un même réseau sont permis, dans la mesure où cette technique est totalement transparente pour les protocoles Internet et à fortiori pour les protocoles de niveau supérieur. Ce type de fragmentation-réassemblage transparent est appelé "dépendant du réseau" (ou encore Intranet) et ne sera plus évoqué dans la suite.

Les adresses Internet distinguent les sources et les destinations en termes de "hôtes" et comportent de plus un champ "protocole". Il est supposé ici que chaque protocole de niveau supérieur disposera de toutes les fonctions de routage nécessaires à l'intérieur même du hôte.

3.1.1. Adressage

Pour conserver toute la souplesse d'assignation d'adresse à des réseaux et pouvoir prendre en compte un grand nombre de réseaux de petite taille ou de taille moyenne, la structure des champs d'adresse est codée de sorte à désigner un petit nombre de réseaux accueillant un très grand nombre d'hôtes, un nombre modéré de réseaux accueillant un nombre modéré d'hôtes, et un grand nombre de réseaux accueillant un nombre restreint d'hôtes. De plus, un encodage spécial permet de prévoir un mode d'adressage étendu futur.
Formats d'adresse :

Poids forts	Format	Class
0	7 bits réseau, 24 bits hôte	A
10	14 bits réseau, 16 bits hôte	B
110	21 bits réseau, 8 bits hôte	C
111	basculement en mode adressage étendu	

Une valeur zéro dans le champ réseau signifie "ce réseau". Ceci n'est utilisé que dans certains messages ICMP. Le mode d'adressage étendu est à ce jour non défini. Ces deux interprétations sont réservées pour un usage futur.

Les valeurs assignées actuellement pour les adresses de réseau sont données dans le document "Assigned Numbers" [9].
L'adresse locale, définie par rapport au réseau local, doit permettre à un hôte "physique" de pouvoir être considéré comme plusieurs hôtes Internet. Ceci veut dire qu'une table de transcription doit exister entre les adresses Internet d'hôte et les adresses d'interfaces réseau permettant à plusieurs adresses Internet d'être accessible par la même interface. Un hôte doit à l'inverse pouvoir dispose de plusieurs interfaces physiques au réseau et traiter les datagrammes y parvenant comme s'ils avaient été adressés à un hôte unique.

Les transcriptions d'adresses Internet en adresses ARPANET, SATNET, PRNET, ou d'autre réseaux sont définies dans le document "Address Mappings" [5].

3.1.2. Fragmentation et Réassemblage.

Le champ d'Identification (ID) permet, en combinaison avec les adresses source et destination et le champ de protocole, d'identifier les segments appartement au même datagramme en vue d'un réassemblage.

Le bit Dernier Fragment (DF) est marqué et si le datagramme ne porte pas le dernier fragment du datagramme original. Le champ Fragment Offset identifie la position relative du fragment transporté, par rapport au début du datagramme original non fragmenté. Les fragments sont mesurés par blocs de 8 octets. La stratégie de fragmentation est ainsi faite qu'un datagramme non fragmenté porte tous les champs de contrôle de fragmentation à zéro (DF = 0, fragment offset = 0). Si un datagramme Internet est fragmenté, alors le découpage devra être fait par blocs multiples de 8 octets excepté le dernier fragment.

Le format choisi pour Fragment Offset permet la numérotation de 2**13 = 8192 positions de blocs de 8 octets chacun pour un total de 65536 octets. Notez que ceci est cohérent avec le format du champ longueur totale (bien sûr, l'en-tête est comptée pour le calcul de la longueur totale, et pas pour la position relative des segments).

Lors d'une fragmentation, certaines options sont recopiées dans chaque en-tête de fragment, d'autres ne sont transmises qu'une fois dans l'en-tête du premier segment.

Tout module Internet doit être capable de traiter un datagramme d'au moins 68 octets sans fragmentation supplémentaire. Ceci est dû au fait qu'une en-tête Internet comprend au plus 60 octets, et le fragment minimal fait 8 octets.

Tout destinataire Internet doit être capable de recevoir un datagramme d'au moins 576 octets soit d'un seul morceau soit en plusieurs fragments à réassembler.
Les champs qui peuvent être modifiés lors d'une fragmentation sont :

```
(1) les champs d'option
(2) le bit More Fragments
(3) le champ Fragment Offset
(4) le champ de longueur totale d'en-tête
(5) le champ de longueur totale
(6) le Checksum d'en-tête
```

Si le bit anti-fragmentation (DF) est marqué, alors toute fragmentation du datagramme Internet est rigoureusement INTERDITE, bien que le datagramme puisse être rejeté. Ceci peut être utilisé pour

prévenir le cas où les modules récepteurs ne disposent pas de ressources mémoires suffisantes pour réassembler correctement les fragments.
Un exemple d'utilisation de cette fonctionnalité est lorsque l'on veut diminuer la charge en ligne d'un module de type "embarqué". Un tel hôte peut travailler sous un système d'exploitation minimum (bootstrap) acceptant un datagramme en entrée, l'enregistrant en mémoire, puis l'exécutant.
Les procédures de fragmentation et de réassemblage sont bien mieux décrites par des exemples. La procédure suivante est un exemple d'implémentation de fragmentation.
Dans les pseudo-programmes suivants, les conventions ci-après sont utilisées : "=<" signifie "inférieur ou égal", "#" signifie "différent de", "=" signifie "égal à", "<-" signifie "est initialisé avec". De plus, "x to y" inclue x et exclue y; par exemple, "4 to 7" comprend 4, 5, et 6 (mais pas 7).

3.1.3. Exemple de procédure de fragmentation

Le datagramme de taille la plus grande pouvant être transmis dans la section de réseau suivante est appelée unité de transmission maximale (UTM).
Si la longueur totale est inférieure ou égale à la taille de l'UTM alors le datagramme doit être directement transmis à l'étape suivant la fragmentation; autrement, le datagramme est coupé en deux, le premier de taille égale à la taille de l'UTM, et le second fragment avec ce qui reste. Le premier fragment est transmis à l'étape suivante, tandis que le deuxième est "réentré" dans la présente procédure, au cas où sa taille dépasserait encore la taille de l'UTM.

Notation:

```
FO    -  Fragment Offset
LET   -  Longueur d'en-tête
AF    -  Bit anti-fragmentation
DF    -  Bit Dernier fragment
LT    -  Longueur totale
OFO   -  Fragment Offset (tampon)
OLET  -  Longueur d'en-tête (tampon)
ODF   -  Bit Dernier Fragment (tampon)
OLT   -  Longueur totale (tampon)
NBF   -  Nombre de blocs de fragments
UTM   -  Unité de transmission maximum
```

Procédure:

```
IF LT =< UTM THEN
    Soumettre le datagramme à l'étape suivante
ELSE IF AF = 1 THEN
    détruire le datagramme
ELSE
// Pour produire le premier fragment :
(1)  Copier l'en-tête originale ;
(2)  OLET <- LET; OLT <- LT; OFO <- FO; ODF <- DF;
(3)  NBF <- (UTM-LET*4)/8;
(4)  Attacher les NBF*8 premiers octets de donnée;
(5)  Corriger l'en-tête:
     DF <- 1;  TL <- (LET*4)+(NBF*8);
     Recalculer le Checksum;
(6)  Soumettre le fragment à l'étape suivante ;

// pour produire le deuxième fragment :
(7)  Copier sélectivement l'en-tête internet (seulement certaines options
     cf. définitions);
(8)  attacher le reste des données;
(9)  Corriger l'en-tête:
     LET <- (((OLET*4)-(longueur des options non copiées))+3)/4;
```

```
        LT <- OLT - NBF*8 - (OLET-LET)*4);
        FO <- OFO + NBF;  DF <- ODF;  Recalculer Checksum;
    (10) Soumettre ce fragment au test de fragmentation; DONE.
```

Dans la procédure ci-dessus, tous les fragments (sauf le dernier) ont la taille maximale qu'admet le réseau en sortie. Une autre implémentation pourrait produire des fragments d'une taille inférieure. Par exemple, une solution consisterait à diviser récursivement un datagramme en deux (en respectant la règle des blocs de 8 octets) tant que les datagrammes restent supérieurs à la taille de l'UTM.

3.1.4. Exemple de procédure de réassemblage

Pour chaque datagramme, le tampon d'identification est constitué en concaténant les adresses de source, de destination, le champ protocole, et d'identification. Si le fragment reçu complète un datagramme en cours de réception (c'est à dire que son fragment offset et le bit Dernier Fragment sont tous deux à zéro), alors toutes les ressources allouées à la fonction de réassemblage pour ce tampon d'identification sont libérées et le datagramme achevé est passé à l'étape suivante du traitement.

Si aucun autre fragment n'est actuellement en mémoire pour ce tampon d'identification, alors des nouvelles ressources sont allouées pour démarrer un réassemblage. Les ressources pour le réassemblage consistent en un tampon de données, un autre pour l'en-tête, une table bit des blocs de fragments, un champ de longueur totale, et un temporisateur. Les données du fragment sont copiées dans le tampon de données à leur position relative indiquée par le fragment offset et l'indication de longueur, et les bits correspondants de la table bit des blocs de fragments sont marqués pour les blocs traités.

S'il s'agit du premier fragment (fragment offset vaut zéro) son en-tête est copiée dans le tampon d'en-tête. S'il s'agit du dernier fragment (Le bit Dernier fragment vaut zéro) le champ de longueur totale est calculé. Si ce fragment, qu'il soit le dernier ou non, complète le datagramme (c'est à dire que tous les bits de la table des blocs de fragments attendus se retrouvent marqués), alors le datagramme est transféré à l'étape suivante de traitement; sinon, on compare la valeur actuelle du temporisateur avec la durée de vie notifiée dans ce fragment et on initialise le temporisateur avec la plus grande valeur des deux; la routine de réassemblage rend alors la main.

Si le temporisateur arrive en fin de course, toutes les ressources consommées pour ce tampon d'identification sont libérées. La valeur initiale de temporisation est la limite inférieure théorique du temps d'attente pour réassemblage. Ce choix se justifie du fait que le temps effectif de réassemblage peut augmenter si le champ durée de vie du fragment reçu est supérieur à la valeur courante de temporisation, mais en aucun cas diminuer étant donné le mécanisme mis en place. La valeur maximale que ce temporisateur peut prendre est la durée de vie maximum (approximativement 4,25 minutes). La valeur de temporisation initiale recommandée aujourd'hui est environ 15 secondes. Cette valeur sera susceptible de changement par l'usage. Notez que le choix de la valeur de paramètre est lié à la capacité du tampon disponible ainsi qu'à la vitesse de transmission du médium; c'est-à-dire, le débit * temporisation = taille du tampon (ex., 10Kb/s * 15s = 150Kb).

Notation:

```
FO    -  Fragment Offset
LET   -  Longueur d'en-tête
DF    -  Bit Dernier Fragment
DdV   -  Durée de Vie
NBF   -  Nombre de Blocs de Fragments
LT    -  Longueur Totale
LTD   -  Longueur Totale des Données
```

```
BUFID -  Tampon d'identification
RCVBT -  Table bit des blocs reçus
LIT   -  Limite Inférieure de Temporisation
```

Procédure:

```
(1)  BUFID <- source|destination|protocole|identification;
(2)  IF FO = 0 AND DF = 0
(3)     THEN IF tampon alloué pour BUFID
(4)             THEN libérer toutes les ressources pour ce BUFID;
(5)          soumettre le datagramme à l'étape suivante; DONE.
(6)     ELSE IF aucun tampon alloué pour BUFID
(7)             THEN réserver les ressource de réassemblage pour BUFID;
                     TIMER <- LIT; LTD <- 0;
(8)          copier les données fragment dans le tampon associé à
             BUFID à partir de l'octet FO*8 jusqu'à
                                l'octet (LT-(LET*4))+FO*8;
(9)          marquer les bits RCVBT de FO à FO+((LT-(LET*4)+7)/8);
(10)         IF DF = 0 THEN LTD <- LT-(LET*4)+(FO*8)
(11)         IF FO = 0 THEN copier l'en-tête dans le tampon d'en-tête
(12)         IF LTD # 0
(13)          AND tous les bits de RCVBT de 0 à (LTD+7)/8 marqué
(14)           THEN LT <- LTD+(LET*4)
(15)                Soumettre le datagramme au pas suivant;
(16)                Libérer toutes les ressources pour ce BUFID; DONE.
(17)         TIMER <- MAX(TIMER,DdV);
(18) Retour jusqu'au fragment suivant ou expiration de
     temporisation;
(19) EXPIRATION: Libérer les ressources pour ce BUFID; DONE.
```

Dans le cas où deux fragments contiennent les mêmes données soit intégralement, soit suite à un recoupement partiel, cette procédure utilisera la dernière version de données arrivées pour compléter le datagramme.

3.1.5. Identification

Le choix d'un identificateur de datagramme est motivé par la nécessité de pouvoir distinguer de façon unique les fragments appartenant à un datagramme particulier. Le module rassemblant les fragments juge que des fragments appartiennent à un même datagramme si ils ont une source, une destination, un protocole, et un identificateur identiques. De ce fait, l'émetteur doit choisir un identificateur unique pour telle paire de source et de destinataire, et pour tel protocole durant toute la durée de transit des fragments des datagrammes.

Il semble que le module Internet doive garder en mémoire une table des identificateurs, dans laquelle on trouvera une entrée par destinataire et protocole, laquelle sera maintenue dans la table au moins jusqu'à la fin de durée de vie maximale (théorique) du dernier fragment du datagramme émis vers cette destination.

Cependant, comme le champ d'identification autorise 65536 valeurs d'identificateurs distinctes, certains hôte choisiront des identificateurs pour chaque datagramme émis, indépendamment des valeurs de paire destination/protocole, par simple "rotation" des identificateurs.

Dans certains cas, il sera approprié de laisser le choix de cet identificateur à charge du protocole de plus haut niveau. Par exemple, lorsqu'un module TCP retransmet un segment TCP identique suite à une erreur, la probabilité d'une réception correcte sera augmentée si la retransmission porte le même

identificateur que la transmission originale dans la mesure où les fragments des deux transmissions peuvent servir à reconstruire correctement le segment TCP entier.

3.1.6. Type de Service

Le type of service (TOS) sélectionne la qualité de service Internet délivrée. Ce type de service est exprimé en termes de priorité, retard, débit, et fiabilité. Ces paramètres abstraits doivent être associés aux paramètres actuellement utilisés par chaque protocole local, pour chaque section de réseau traversée.

Priorité. Une mesure subjective de l'importance de ce datagramme.
Retard. Des datagrammes marqués de ce bit doivent être transmis le plus rapidement possible.
Débit. Ces datagrammes font partie d'une communication demandant un gros débit de données.
Fiabilité ou taux d'erreur. Un effort particulier doit être fait pour assurer l'acheminement correct de ces datagrammes.

Par exemple, ARPANET marque le bit priorité, et un choix entre des messages "standard" (type 0) et messages "uncontrolled" (type 3), (le choix entre des messages de type paquet unique ou paquets multiples peut aussi être considéré comme un paramètre de type de service). Les messages "uncontrolled" ont tendance à être acheminés plus rapidement, mais au prix d'une certaine fiabilité. Supposons qu'un datagramme Internet doive transiter par ARPANET. Donnons un type de service défini selon :

```
Priorité  :    5
Retard    :    0
Débit     :    1
Fiabilité :    1
```

Dans cet exemple, l'interprétation de ces paramètres en termes de paramètres de service ARPANET provoquerait

un marquage du bit prioritaire ARPANET, dans la mesure où la priorité Internet est donnée dans la moitié supérieure de sa plage

la sélection du type de messages "standard", au vu des contraintes demandées pour le débit et la fiabilité et aucune contrainte sur le temps d'acheminement. Plus de détails concernant cette interprétation dans le document "Service Mappings" [8].

3.1.7. Durée de vie

La durée de vie est initialisée par l'émetteur du datagramme à la durée maximum pendant lequel le datagramme pourra exister dans le réseau. Si un routeur ou autre module Internet intercepte un datagramme plus "vieux" que cette durée de vie, alors ce dernier doit être détruit.

Ce champ doit être décrémenté à chaque point du réseau où l'en-tête Internet est interprétée, d'une valeur représentant à peu près le temps passé à traiter le datagramme. Même si le système local n'est pas en mesure de fournir une mesure de ce temps, ce champ doit être décrémenté au minimum d'une unité. Sinon, le temps doit être mesuré en secondes (c-à-d. qu'une unité correspond à une seconde). De ce fait, la durée de vie maximale codable est de 255 secondes soit 4,25 minutes. Comme chaque module Internet doit impérativement décrémenter ce champ d'au moins une unité (une seconde) même si le traitement du datagramme a demandé beaucoup moins de temps, la durée de vie initiale doit

toujours être interprétée comme la durée théorique maximale pendant laquelle le datagramme peut exister. La justification de ce mécanisme est d'écarter automatiquement des datagrammes qui n'ont pu trouver leur destinataire, ainsi qu'imposer une limite théorique à la charge globale du réseau.
De plus, certains protocoles de niveau supérieur s'appuient sur la supposition qu'aucun "doublon" de datagramme provenant d'une connexion précédente ne peut arriver au delà d'un certain temps (Cf. TCP). Ce mécanisme de durée de vie permet de garantir à ces protocoles la validité de cette supposition.

3.1.8. Options

Les options sont optionnelles dans les datagrammes, mais les implémentations doivent nécessairement prévoir leur présence. En d'autres termes, la présence ou l'absence d'options dans le datagramme reste un choix de l'émetteur, mais toute implémentation de module Internet doit disposer des routines permettant leur traitement. On pourra trouver diverses options dans le champ d'en-tête.
L'insertion d'options peut conduire à une taille d'en-tête distincte d'un multiple de 32 bits. Cette dernière doit être complétée par des octets nuls afin de respecter ce point. Le premier de ces octets nuls sera interprété comme l'option particulière "fin de liste d'options", les octets suivants étant appelés "octets de bourrage".
Tout module Internet doit savoir réagir à toute option "officielle". L'option de sécurité doit être utilisée en cas de transmission de trafic compartimenté, restreint ou confidentiel.

3.1.9. Checksum

Le Checksum d'en-tête doit être recalculé à chaque fois que l'en-tête Internet a subi une modification. Par exemple, lorsque la durée de vie a été décrémentée, des options Internet ajoutées, modifiées ou supprimées, ou suite à une fragmentation. Ce Checksum Internet protège l'en-tête contre les erreurs de transmission.
Il existes certaines applications pour lesquelles quelques erreurs "bit" restent acceptables alors qu'un retard dû à une retransmission ne l'est pas. Si le protocole Internet avait introduit la notion de contrôle de transmission sur les données, de telles applications n'auraient pu s'appuyer sur ce protocole.

3.1.10. Erreurs

Toutes les erreurs en rapport avec le protocole Internet devront être reportées à l'aide de messages ICMP [3].

3.2. Interfaces

La description fonctionnelle des interfaces entre IP et la couche supérieur ne peut être exposée que dans sa signification sémantique théorique, dans la mesure où chaque système d'exploitation proposera ses propres primitives. Par conséquent, nous nous devons d'avertir le lecteur que des implémentations distinctes d'IP pourront présenter des interfaces différentes. Cependant, tous les modules IP doivent fournir un ensemble minimum de fonctions d'accès pour garantir la cohérence de la hiérarchie des protocoles. Cette section spécifie les interfaces fonctionnelles requises pour toutes les implémentations d'IP.
Les deux interfaces du protocole Internet visent d'un côté le protocole réseau local, et de l'autre un protocole de niveau supérieur voire directement un programme applicatif. Dans ce qui suit, le protocole de niveau supérieur ou le programme applicatif (où même le logiciel d'un routeur) sera assimilé à "l'utilisateur" dans la mesure où c'est lui qui "utilise" le module Internet. Comme le

protocole Internet est basé sur le principe du datagramme, la mémoire ou les états maintenus entre deux transmissions de datagrammes sont réduits au minimum, et chaque appel au module Internet fournit à celui-ci toutes les informations nécessaires à l'émission correcte et complète des données.

3.2.1. Un exemple d'interface supérieure

Les deux exemple d'appels suivants satisfont les exigences minimales de communication entre l'utilisateur et le module de protocole Internet ("=>" signifie "retour"):

SEND (src, dst, prot, TdS, TTL, BufPTR, lon, Id, AF, opt => result)

où :

```
src = adresse source
dst = adresse destinataire
prot = protocole
TdS = type de service
DdV = durée de vie
BufPTR = pointeur sur tampon
lon = longueur de tampon
Id  = Identificateur
AF = Antifragmentation
opt = donnée d'option
result = réponse
  OK = datagramme émis
  Error = erreur dans les arguments ou erreur réseau local
```

Notez que la priorité est prise en compte dans le TdS et les données de sécurité/compartiment sont passés comme option.

```
RECV (BufPTR, prot, => result, src, dst, TdS, lon, opt)
```

dans laquelle :

```
BufPTR = pointeur sur tampon
prot = protocole
result = réponse
  OK = datagramme reçu
  Error = erreur dans les arguments
lon = longueur du tampon
src = adresse source
dst = adresse destination
TdS = type de service
opt = donnée d'option
```

Lorsque l'utilisateur envoie un datagramme, il exécute un appel SEND en fournissant tous les arguments. Le module Internet, sur réception de cet appel, vérifie les arguments, prépare et envoie le message. Si les arguments sont corrects et le datagramme est accepté par le module réseau local, alors l'appel se termine par un retour normal. Dans le cas où soit les arguments sont erronés, soit que le datagramme a été refusé par la couche réseau local, l'appel se termine par un retour d'erreur. Sur erreur, un rapport le plus explicite devra être donné pour indiquer la cause du problème, le degré de détail restant à la discrétion de l'implémenteur.

Lorsqu'un datagramme est remis au module Internet par le module réseau local, deux cas se présentent : soit un appel RECV émis par l'utilisateur est en attente, soit le module Internet n'a pas été sollicité. Dans le premier cas, il est répondu à l'appel en attente à l'aide des données contenues dans le

datagramme entrant. Dans le second cas, L'utilisateur est averti de la présence d'un datagramme lui étant destiné. Si l'utilisateur visé n'existe pas, un message d'erreur ICMP doit être renvoyé à l'émetteur, et le datagramme détruit.
La notification à l'utilisateur pourra être faite via une pseudo-interruption ou tout mécanisme similaire le plus approprié en fonction des ressources et de la structure du système d'exploitation utilisé.
Il sera ainsi possible de répondre immédiatement à un appel RECV et de n'envoyer le datagramme que lors de sa réception (interface asynchrone), ou au contraire de bloquer l'utilisateur en attendant que le datagramme soit parvenu au module Internet (interface synchrone).

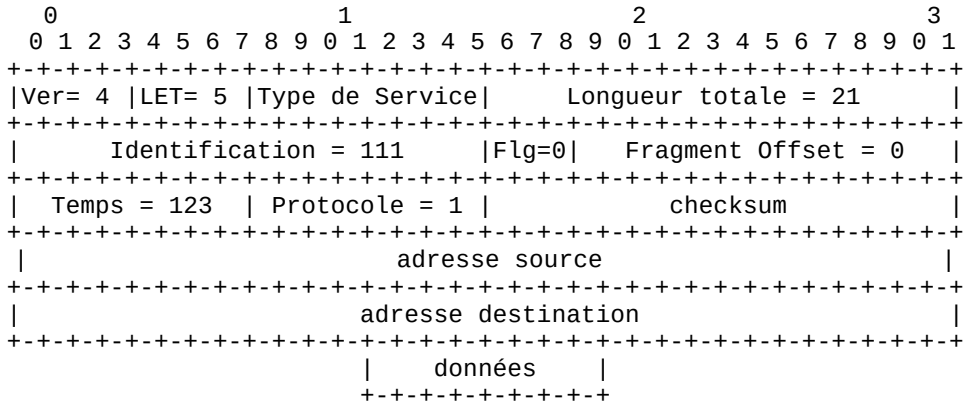
L'adresse source doit être indiquée dans l'appel SEND au cas où l'hôte émetteur disposerait de plusieurs adresses (raccordements physiques ou adresses logiques multiples). Le module Internet devra vérifier que l'adresse source donnée est une adresse valide pour cet hôte.

Une application pourra aussi permettre ou nécessiter un appel au module Internet pour indiquer son intérêt à ou encore se réserver l'usage exclusif d'une certaine classe de datagrammes (ex., tous ceux dont le champ protocole est égal à une certaine valeur).

4. Appendice A: Exemples & Scénarios

4.1. Exemple 1:

Voici l'exemple d'un datagramme transmettant le minimum de données possible :



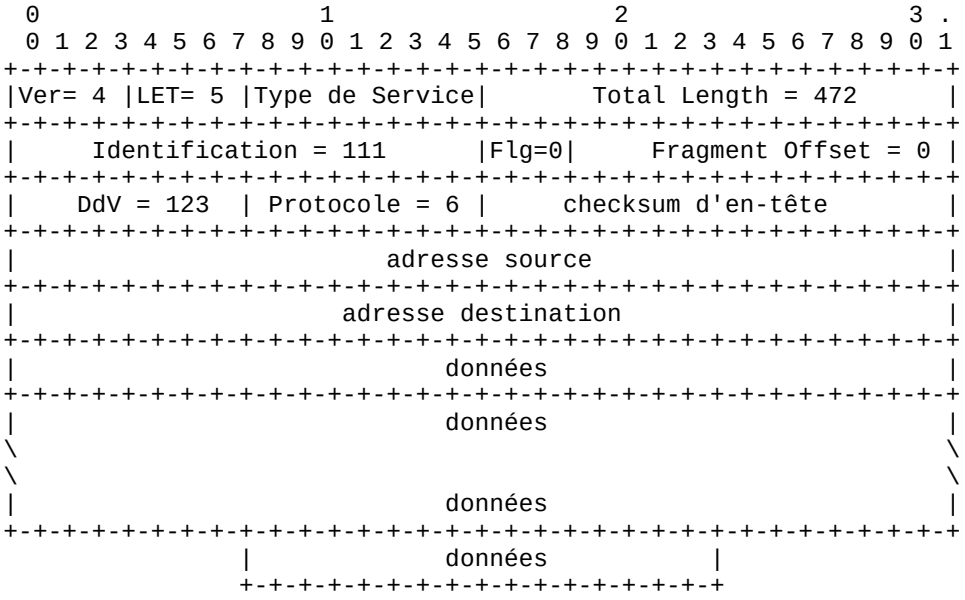
Exemple de Datagramme Internet
Figure 5.

Notez que chaque marque vaut pour une position bit.

Cet exemple donne le datagramme minimum dans le protocole Internet de version 4 ; l'en-tête Internet est formée de 5 mots de 32 bits, et la longueur totale du datagramme est de 21 octets. Ce datagramme est un datagramme complet (pas un fragment).

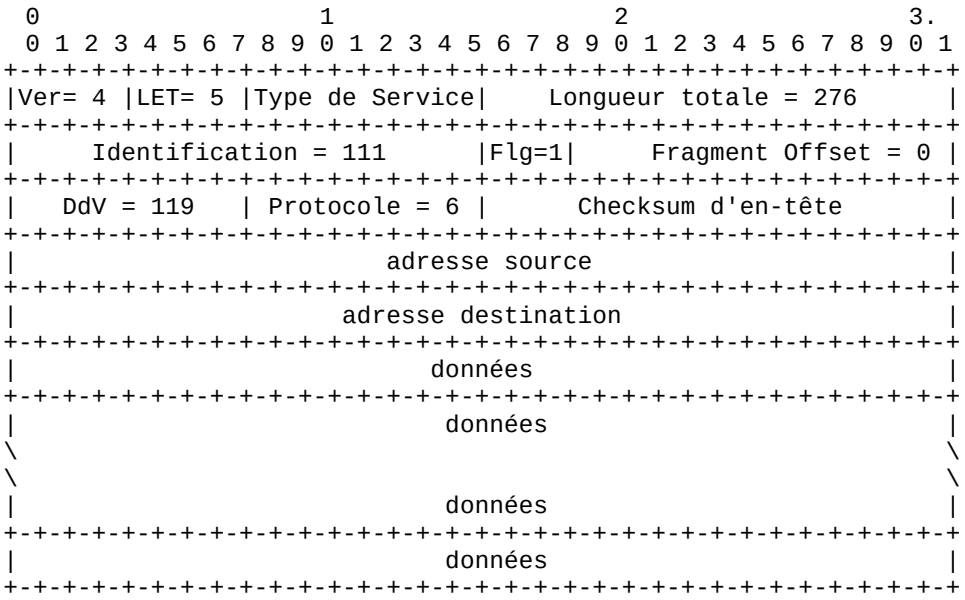
4.2. Exemple 2:

Dans cet exemple, nous exposons d'abord un datagramme Internet de taille moyenne (452 octets de données), puis deux datagrammes Internet portant les fragments de ce qui résulterait d'une fragmentation du premier datagramme pour un réseau dont la taille maximale de transmission vaut 280 octets.



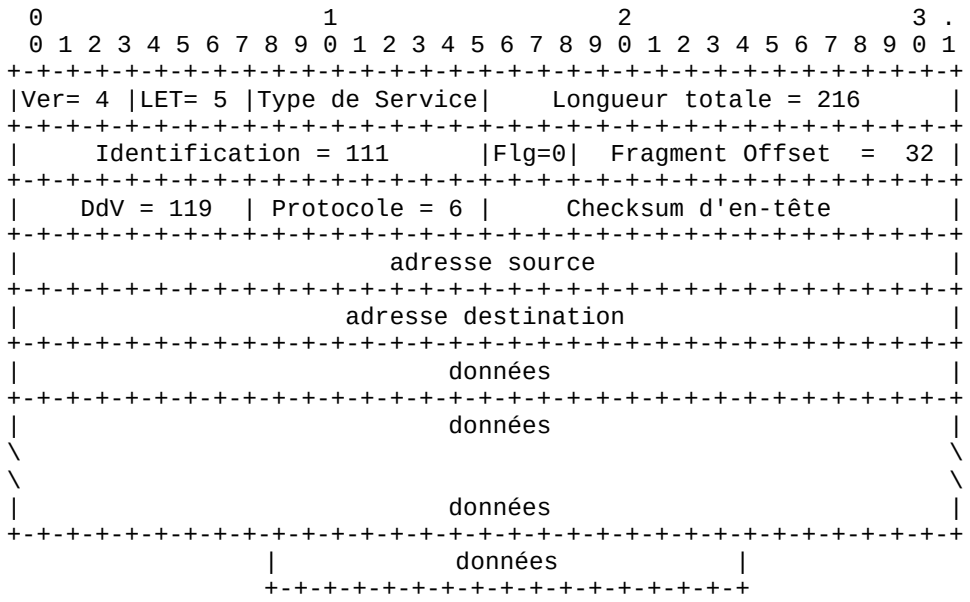
Exemple de Datagramme Internet
Figure 6.

Et maintenant le premier fragment obtenu en coupant les données précédentes après le 256ème octet de données.



Exemple de Fragment Internet
Figure 7.

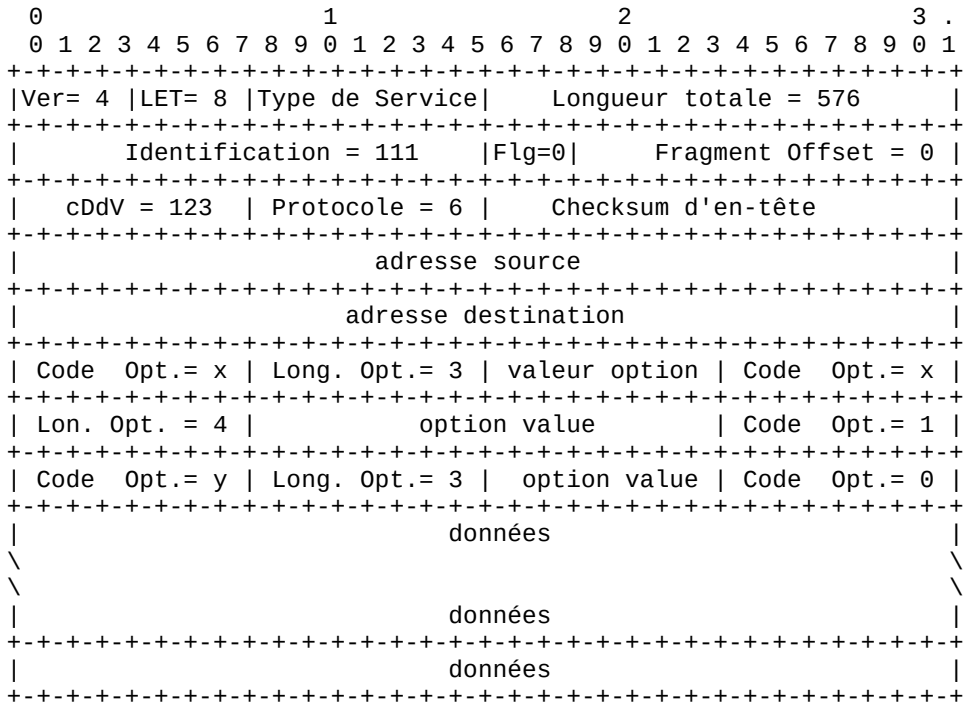
Et le second fragment.



Exemple de dernier fragment Internet
Figure 8.

4.3. Exemple 3:

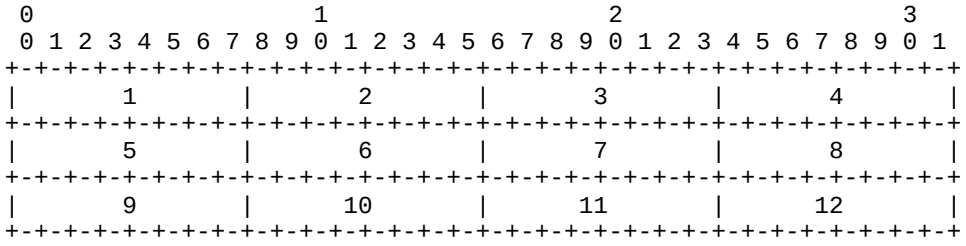
Voici un exemple de datagramme contenant des options :



Exemple Internet Datagramme
Figure 9.

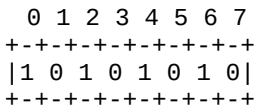
5. Appendice B: Ordre de transmission des données

L'ordre de transmission de l'en-tête et des données décrites dans ce document se résout au niveau octet. Lorsqu'un datagramme contient un groupe d'octets, l'ordre de transmission de ces octets est l'ordre "naturel" dans lequel nous allons les lire en Français. Par exemple, dans le schéma ci-dessous les octets sont transmis dans l'ordre de leur numérotation.



Ordre de transmission des octets
Figure 10.

Lorsqu'un octet représente une valeur numérique, le bit de gauche dans le schéma ci-dessous est celui de poids le plus fort. Ici le bit noté 0. L'exemple suivant montre le codage de la valeur 170 (décimale).



Conventions sur la signification des bits
Figure 11.

Par extension, lorsqu'une valeur numérique est codée sur plusieurs octets, le bit de gauche du champ complet est celui de poids le plus fort. De ce fait, lorsqu'un champ multi-octets est transmis, l'octet de poids le plus fort est toujours transmis en premier.

6. Glossaire

1822 : BBN Report 1822, "A Specification of the Interconnection of a Host and an IMP". La spécification de l'interface entre un hôte et ARPANET.

Adresse Internet : L'adresse sur 4 octets (32 bit) d'une source ou d'une destination composée d'une adresse Réseau et d'une adresse Locale.

Adresse Locale: L'adresse d'un hôte dans un réseau local. La transcription des adresses Internet en adresse physiques d'hôtes est assez libre, permettant des affectations non bijectives.

AF: Le bit AntiFragmentation dans les divers bits de contrôle.

Bit Dernier Fragment : Un bit indiquant si le fragment Internet contient les dernières données du datagramme qu'il transporte.

Bits de contrôle (flags) : Divers bits de signification booléenne dans l'en-tête Internet.

Bourrage : Ces octets sont ajoutés pour s'assurer que la section de données commence sur le début d'un mot de 32 bits. L'octet de bourrage vaut zéro.

Datagramme Internet : L'unité de transmission entre une paire de modules Internet (incluant l'en-tête Internet).

Destination : L'adresse du destinataire, un champ d'en-tête Internet.

DdV : Durée de Vie

DF : Le bit Dernier Fragment de l'en-tête Internet.

Durée de Vie : Champ d'en-tête Internet qui donne la durée de vie maximale pendant laquelle un datagramme peut exister dans le réseau.

En-tête : Information de contrôle au début d'un message, d'un segment, d'un datagramme, d'un paquet ou bloc de données.

Fragment Internet : Une portion de données d'un datagramme Internet associée à une en-tête.

Fragment Offset : Un champ permettant de coder la position relative du fragment par rapport au datagramme non fragmenté.

GGP : Protocole Routeur vers Routeur, le protocole utilisé entre routeurs pour s'échanger des informations de routage et autres fonctions.

ICMP : Internet Control Message Protocol. Implémenté dans le module Internet, le protocole ICMP est utilisé depuis les routeurs vers les hôtes et entre hôtes pour le report de fautes et des suggestions de routage.

Identification : Un champ d'en-tête Internet transportant une valeur d'identification temporaire destinée à aider au réassemblage des fragments d'un datagramme.

IHL : Un champ d'en-tête Internet codant la longueur de l'en-tête en mots de 32 bits.

IMP : L'Interface Message Processor, l'élément de commutation de paquet du réseau ARPANET.

Leader ARPANET : L'information de contrôle d'un message ARPANET au niveau de l'interface hôte-IMP.

Longueur Totale : Un champ d'en-tête Internet qui donne la longueur totale du datagramme en octets, y compris données et en-tête.

Message ARPANET : L'unité de transmission entre un hôte et un IMP dans ARPANET. La taille maximum est d'environ 1012 octets (8096 bits).

Module : Une implémentation, en général logicielle, d'un protocole ou d'une autre procédure.

NBF : Le Nombre de Blocs Fragment dans la portion données d'un fragment Internet. C'est à dire, le nombre de "mots" de 8 octets dans la section données d'un fragment.

Octet : Huit bits.

Options : Le champ d'en-tête Internet peut comporter plusieurs options, chaque option pouvant être constituée de plusieurs octets.

Paquet ARPANET : L'unité de transmission utilisé dans l'ARPANET entre deux IMPs. La taille maximum est d'environ 126 octets (1008 bits).

Protocole : Dans ce document, l'identificateur du protocole de niveau immédiatement supérieur, à qui doit être délivré le datagramme, champ d'en-tête Internet.

Reste : La partie d'une adresse Internet donnant l'adresse locale de la machine.

Source : L'adresse source, champ d'en-tête Internet.

TCP : Transmission Control Protocol : Un protocole sécurisé de transmission de données entre deux hôtes s'appuyant sur IP.

TFTP : Trivial File Transfer Protocol: Un protocole simple de transfert de fichiers basé sur UDP.

Segment TCP : L'unité de données échangé par un module TCP (avec une en-tête TCP).

TdS : Type de Service : Type de Service

Un champ d'en-tête Internet qui indique le type (ou qualité) du service pour ce datagramme Internet.

UDP : User Datagramme Protocol: Un protocole de la couche "transport" pour des communications transactionnelles.

Utilisateur : L'utilisateur du protocole Internet. Celui-ci peut être un module de protocole de niveau supérieur, un programme d'application, ou un programme routeur.

Version : Le champ de version indique le format de l'en-tête Internet.

7. Références

[1] Cerf, V., "Le Catenet Model for Internetworking," Information Processing Techniques Office, Defense Advanced Research Projects Agency, IEN 48, July 1978.

[2] Bolt Beranek and Newman, "Specification for the Interconnection of un Host and an IMP," BBN Technical Report 1822, Revised May 1978.

[3] Postel, J., "Internet Control Message Protocol - DARPA Internet Program Protocol Specification," RFC 792, USC/Information Sciences Institute, September 1981.

[4] Shoch, J., "Inter-Network Naming, Addressing, and Routing," COMPCON, IEEE Computer Society, Fall 1978.

[5] Postel, J., "Address Mappings," RFC 796, USC/Information Sciences Institute, September 1981.

[6] Shoch, J., "Packet Fragmentation in Inter-Network Protocols," Computer Networks, v. 3, n. 1, February 1979.

[7] Strazisar, V., "How to Build a Routeur", IEN 109, Bolt Beranek and Newman, August 1979.

[8] Postel, J., "Service Mappings," RFC 795, USC/Information Sciences Institute, September 1981.

[9] Postel, J., "Assigned Numbers," RFC 790, USC/Information Sciences Institute, September 1981.