Groupe de travail Réseau
Request for Comments : 5444
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

T. Clausen, LIX, Ecole Polytechnique
C. Dearlove, BAE Systems ATC
J. Dean, Naval Research Laboratory
C. Adjih, INRIA Rocquencourt
février 2009

Format de paquet/message de réseau ad hoc mobile généralisé (MANET)

Statut du présent mémoire
Le présent document spécifie un protocole de l'Internet sur la voie de la normalisation pour la communauté de l'Internet, et appelle à des discussions et suggestions pour son amélioration. Prière de se référer à l'édition en cours des "Protocoles officiels de l'Internet" (STD 1) pour voir l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Notice de droits de reproduction
Copyright (c) 2009 IETF Trust et les personnes identifiées comme auteurs du document. Tous droits réservés.

Le présent document est soumis au BCP 78 et aux dispositions légales de l'IETF Trust qui se rapportent aux documents de l'IETF (http://trustee.ietf.org/license-info) en vigueur à la date de publication de ce document. Prière de revoir ces documents avec attention, car ils décrivent vos droits et obligations par rapport à ce document.

Résumé
Le présent document spécifie un format de paquet capable de porter plusieurs messages qui peuvent être utilisés par des protocoles d'acheminement de réseau ad hoc mobile.

Table des matières

1. Introduction....2
2. Notation et terminologie....3
2.1 Notation....3
2.2 Terminologie....3
3. Déclaration d'applicabilité....4
4. Vue d'ensemble et fonctionnement du protocole....4
5. Syntaxe....4
5.1 Paquets....5
5.2 Messages....5
5.3 Blocs d'adresses....7
5.4 TLV et blocs de TLV....8
5.5 Éléments mal formés....11
6. Considérations relatives à l'IANA....11
6.1 Revue d'expert : lignes directrices d'évaluation....11
6.2 Types de message....12
6.3 Types de TLV Paquet....13
6.4 Types de TLV Message....13
6.5 Types de TLV Bloc d'adresses....13
7. Considérations sur la sécurité....14
7.1 Suggestions d'authentification et d'intégrité....14
7.2 Suggestions de confidentialité....14
8. Contributeurs....15
9. Remerciements....15
10. Références....16
10.1 Références normatives....16
10.2 Références pour information....16
Appendice A. Multiplexage et démultiplexage....16
Appendice B. Usage prévu....17
Appendice C. Exemples....17
C.1 Exemples de bloc d'adresses....17
C.2 Exemples de TLV....19

Appendice D. Illustrations....20
D.1 Paquet....20
D.2 Message....22
D.3 Corps de message....25
D.4 Bloc d'adresses....26
D.5 Bloc de TLV....31
D.6 TLV....32
Appendice E. Exemple complet....35
Adresse des auteurs....36

1. Introduction

Le présent document spécifie la syntaxe d'un format de paquet conçu pour porter plusieurs messages de protocole d'acheminement pour un échange d'informations entre des routeurs de réseau ad hoc mobile (MANET, *Mobile Ad hoc NETwork*). Les messages consistent en un en-tête de message, qui est destiné au contrôle de la dissémination de messages, et un corps de message, qui contient des informations de protocole. Seule la syntaxe de paquet et des messages est spécifiée.

Le présent document spécifie :

- Un format de paquet, permettant que zéro, un ou plusieurs messages soient contenus dans une seule transmission. Un paquet avec zéro message peut être envoyé dans le cas où les seules informations à échanger sont contenues dans l'en-tête de paquet.

- Un format de message, où le message est composé d'un en-tête de message et d'un corps de message.

- Un en-tête de format de message, qui contient des informations qui peuvent être suffisantes pour permettre à un protocole qui utilise la présente spécification de prendre des décisions de traitement et de transmission.

- Un format de corps de message, contenant des attributs associés au message ou l'origine du message, ainsi que des blocs d'adresses, ou des préfixes d'adresse, avec les attributs associés.

- Un format de bloc d'adresse, où un bloc d'adresse représente des ensembles d'adresses, ou des préfixes d'adresses, sous une forme compacte avec des adresses agrégées.

- Un format généralisé de type-longueur-valeur (TLV) représentant des attributs. Chaque TLV peut être associé à un paquet, un message, ou une ou plusieurs adresses ou préfixes d'adresse dans un seul bloc d'adresses. Plusieurs TLV peuvent être inclus, chacun associé à un paquet, un message, et le même ensemble, différent, ou en chevauchement d'ensembles d'adresses ou préfixes d'adresses.

La spécification a été explicitement conçue avec les propriétés suivantes, sans ordre particulier :

Logique d'analyse - la notation utilisée dans la présente spécification facilite une logique d'analyse générique, indépendante du protocole.

Extensibilité - les paquets et messages définis par un protocole qui utilise la présente spécification sont extensibles en définissant de nouveaux messages et de nouveaux TLV. Les protocoles qui utilisent la présente spécification vont être capables d'identifier correctement et sauter ces nouveaux messages et TLV, tout en analysant correctement le reste du paquet et message.

Efficacité - quand les adresses rapportées partagent des séquences de bit communes (par exemple, préfixes d'adresse ou identifiants d'adresse IPv6) la représentation de bloc d'adresse permet une représentation compacte. Les en-têtes de message compacts sont assurés en permettant l'inclusion des seuls éléments d'en-tête de message requis. La structure de paquet multi-messages permet une réduction du nombre d'octets transmis et du nombre de paquets transmis. La structure de paquet et de codage de message permet l'analyse, la vérification, et l'identification des éléments individuels en une seule passe.

Séparation de la transmission et du traitement - un protocole qui utilise la présente spécification peut être conçu de façon à ce que la détection de dupliqués et les décisions de transmission de message à portée contrôlée puissent être prises en

utilisant les informations contenues dans l'en-tête de message, sans traitement du corps de message.

2. Notation et terminologie

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

De plus, le présent document utilise la notation du paragraphe 2.1 et la terminologie du paragraphe 2.2.

2.1 Notation

Le présent document utilise les notations suivantes pour les éléments et variables.

Ce format utilise l'ordre des octets du réseau (bit de poids fort en premier) pour tous les champs. Le bit de poids fort dans un octet est numéroté comme bit 0, et le bit de moindre poids d'un octet est numéroté comme bit 7 [Stevens].

2.1.1 Éléments

La présente spécification définit des éléments. Un élément est un groupe de tout nombre de bits consécutifs qui ensemble forment une entité syntaxique représentée en utilisant la notation <élément>. Chaque élément dans le présent document est défini comme :
- un champ de bits d'une taille spécifique OU
- un élément composite, composé d'autres <élément>.

Un élément composite est défini comme suit : <élément> := spécification

où, sur le côté droit suivant :=, spécification est représenté en utilisant la syntaxe d'expression régulière définie dans [SingleUNIX]. Seule la notation suivante est utilisée :

<élément1><élément2> - indique que <élément1> est immédiatement suivi par <élément2>.
(<élément1><élément2>) - indique un groupement des éléments inclus entre les parenthèses.

? - Zéro ou une occurrence de l'élément ou groupe précédent.

* - Zéro, une ou plusieurs occurrences de l'élément ou groupe précédent.

2.1.2 Variables

Des variables sont introduites dans la spécification seulement comme moyen de préciser la description. Les deux notations suivantes sont utilisées :

<foo> - si <foo> est un champ d'entier non signé, alors <foo> est aussi utilisé pour représenter la valeur de ce champ.

bar - variable, généralement obtenue par des calculs fondés sur la ou les valeurs du ou des éléments.

2.2 Terminologie

Le présent document utilise la terminologie suivante :

Paquet : entité de niveau supérieur de la présente spécification. Un paquet contient un en-tête de paquet et zéro, un ou plusieurs messages.

Message : entité fondamentale qui porte les informations de protocole, sous forme d'objets d'adresse et de TLV.

Adresse : un nombre d'octets qui constituent une adresse de la longueur indiquée par l'en-tête de message encapsulant. La signification d'une adresse est définie par le protocole qui utilise la présente spécification.

Préfixe d'adresse : une adresse plus une longueur de préfixe, la longueur de préfixe étant un nombre de bits d'adresse mesuré depuis l'extrémité gauche/de plus fort poids de l'adresse.

Objet Adresse : soit une adresse, soit un préfixe d'adresse, comme spécifié dans un bloc d'adresse dans cette spécification.

TLV : structure de type-longueur-valeur. C'est une façon générique par laquelle un attribut peut être représenté et correctement analysé sans que l'analyseur ait à comprendre l'attribut.

3. Déclaration d'applicabilité

La présente spécification décrit un format de paquet générique, conçu pour être utilisé par les protocoles d'acheminement MANET. La spécification a été inspirée et étendue à partir de celle utilisée par le protocole d'acheminement d'état de liaison optimisé (OLSR, *Optimized Link State Routing*) [RFC3626].

Les MANET sont, couramment mais pas exclusivement, caractérisés comme étant capables de fonctionner sur des interfaces de réseau sans fil de capacité limitée à modérée. Les MANET sont donc moins tolérants au gaspillage d'octets transmis que le sont la plupart des réseaux filaires. La présente spécification représente donc un compromis entre des attributs parfois en compétition, spécifiquement d'efficacité, d'extensibilité, et la facilité d'utilisation.

L'efficacité est prise en charge en réduisant la taille de paquet et en permettant plusieurs messages disjoints dans un seul paquet. La taille réduite de paquet est principalement prise en charge par l'agrégation d'adresse, des champs facultatifs d'en-tête de paquet et d'en-tête de message, et des champs facultatifs dans les blocs d'adresse et les TLV. La prise en charge de paquets multi-messages permet une réduction du nombre de paquets, dont chacun peut supporter des coûts de bande passante significatifs provenant des couches de transport, de réseau, et inférieures.

La présente spécification assure l'extensibilité externe et interne. L'extensibilité externe est prise en charge par la capacité d'ajouter des TLV Paquet et de définir de nouveaux types de messages. L'extensibilité interne est prise en charge par la capacité d'ajouter des TLV Message et des TLV Bloc d'adresses aux messages existants. Les protocoles peuvent définir de nouveaux types de TLV, et donc le contenu de leurs champs Valeur, et de nouveaux types de message (voir au paragraphe 6.1). Les protocoles peuvent aussi réutiliser les définitions de type de TLV provenant d'autres protocoles qui utilisent aussi la présente spécification.

La présente spécification vise à être suffisamment expressive et souple pour être capable de s'accommoder de différentes classes de protocole d'acheminement MANET (par exemple, protocoles d'acheminement proactif, réactif, et hybride) ainsi que de leurs extensions. Avoir un format commun de paquet et de message, et une façon commune de représenter les adresses IP et les attributs associés, permet à un code d'analyse générique d'être développé, sans considération de l'algorithme utilisé par le protocole d'acheminement.

Toutes les adresses dans un message sont supposées être de la même taille, spécifiée dans l'en-tête de message. Dans le cas d'adresses mixtes IPv6 et IPv4, les adresses IPv4 peuvent être représentées comme des adresses IPv6 transposées en IPv4, comme spécifié dans la [RFC4291].

Les messages définis par la présente spécification sont destinés à porter la signalisation de protocole d'acheminement MANET entre routeurs MANET. La présente spécification inclut des éléments qui peuvent prendre en charge l'arrosage de portée limitée, ainsi que d'être utilisables pour la livraison en point à point de signalisation de protocole d'acheminement MANET dans un réseau multi bonds. Les paquets peuvent être en envoi individuel ou en diffusion groupée et peuvent utiliser tout protocole de transport approprié ou aucun.

Un protocole d'acheminement MANET qui utilise le format de message défini dans la présente spécification peut contraindre la syntaxe (par exemple, en exigeant un ensemble spécifique de champs d'en-tête de message) que le protocole va employer. Les protocoles qui ont de telles restrictions n'ont pas besoin d'être capables d'analyser toutes les structures possibles de message définies par le présent document mais doivent être cohérents dans la génération et la réception des messages qu'ils définissent. Si un protocole spécifie quels éléments sont inclus, l'indexage direct des champs appropriés est possible, selon les restrictions de syntaxe imposées par le protocole. De tels protocoles peuvent avoir une extensibilité plus limitée.

4. Vue d'ensemble et fonctionnement du protocole

La présente spécification ne décrit pas de protocole. Elle décrit un format de paquet, qui peut être utilisé pzr tout protocole d'acheminement de réseau ad hoc mobile.

5. Syntaxe

Cette Section fournit la spécification syntaxique normative d'un paquet, représenté par l'élément <paquet> et les éléments d'après lesquels il est composé. La spécification est donnée en utilisant la notation du paragraphe 2.1.

Des illustrations graphiques de la disposition des éléments spécifiés sont données à l'Appendice D, une illustration graphique d'un exemple complet (un paquet incluant un message avec des blocs d'adresses et des TLV) est donnée à l'Appendice E.

Ce format utilise l'ordre des octets du réseau, comme indiqué au paragraphe 2.1.

5.1 Paquets

<paquet> est défini par :

<paquet> := <pkt-en-tête> <message>*

où <message> est comme défini au paragraphe 5.2. Une analyse réussie est terminée quand tous les octets du paquet (comme défini par le datagramme contenant le paquet) sont utilisés.

<pkt-en-tête> est défini par :

<pkt-en-tête> := <version> <pkt-fanions> <pkt-seq-num>? <tlv-bloc>?

où :

<version> est un champ d'entier non signé de 4 bits et spécifie la version de la spécification sur laquelle le paquet et les messages contenus sont construits. Le présent document spécifie la version 0.

<pkt-fanions> est un champ de 4 bits, qui spécifie l'interprétation du reste de l'en-tête de paquet:

bit 0 (phasseqnum) : si il est à zéro ("0"), alors <pkt-seq-num> n'est pas inclus dans le <pkt-en-tête>. Si il est établi ("1"), alors <pkt-seq-num> est inclus dans le <pkt-en-tête>.

bit 1 (phastlv) : si il est à zéro ("0"), alors <tlv-bloc> n'est pas inclus dans le <pkt-en-tête>. Si il est établi ("1"), alors <tlv-bloc> est inclus dans le <pkt-en-tête>.

bits 2-3 : réservés et DEVRAIENT chaque être à zéro ("0") à l'émission et DEVRAIENT être ignorés à réception.

<pkt-seq-num> est omis si le fanion phasseqnum est à zéro ("0") ; autrement, est un champ d'entier non signé de 16 bits, qui spécifie un numéro de séquence de paquet.

<tlv-bloc> est omis si le fanion phastlv est à zéro ("0") et est autrement comme défini au paragraphe 5.4.

On suppose que la couche réseau est capable de livrer la longueur exacte de charge utile, évitant donc d'avoir à porter la longueur de paquet dans le paquet.

5.2 Messages

Les paquets peuvent, en plus de l'en-tête de paquet, contenir un ou plusieurs messages. Les messages contiennent :
- Un en-tête de message.
- Un bloc TLV Message qui contient zéro, un ou plusieurs TLV, associés au message complet.

- Zéro, un ou plusieurs blocs Adresse, chacun contenant un ou plusieurs objets Adresse.
- Un bloc TLV Bloc d'adresse, contenant zéro, un ou plusieurs TLV et suivant chaque bloc d'adresse, par lequel les adresses peuvent être associées à des attributs supplémentaires.

<message> est défini par :

<message> := <msg-en-tête> <tlv-bloc> (<adr-bloc><tlv-bloc>)*

<msg-en-tête> := <msg-type>
<msg-fanions>
<msg-adr-longueur>
<msg-taille>
<msg-orig-adr>?
<msg-bond-limite>?
<msg-bond-compte>?
<msg-seq-num>?

où :

<tlv-bloc> est comme défini au paragraphe 5.4.

<addr-bloc> est comme défini au paragraphe 5.3.

<msg-type> est un champ d'entier non signé de 8 bits, qui spécifie le type du message.

<msg-fanions> est un champ de 4 bits, qui spécifie l'interprétation du reste de l'en-tête de message:

bit 0 (mhasorig) : si il est à zéro ("0"), alors <msg-orig-adr> n'est pas inclus dans le <msg-en-tête>. Si il est établi ("1") alors <msg-orig-adr> est inclus dans le <msg-en-tête>.

bit 1 (mhashoplimit) : si il est à zéro ("0"), alors <msg-bond-limite> n'est pas inclus dans le <msg-en-tête>. Si il est établi ("1") alors <msg-bond-limit> est inclus dans le <msg-en-tête>.

bit 2 (mhashopcount) : si il est à zéro ("0"), alors <msg-bond-compte> n'est pas inclus dans le <msg-en-tête>. Si il est établi ("1"), alors <msg-bond-compte> est inclus dans le <msg-en-tête>.

bit 3 (mhasseqnum) : si il est à zéro ("0"), alors <msg-seq-num> n'est pas inclus dans le <msg-en-tête>. Si il est établi ("1") alors <msg-seq-num> est inclus dans le <msg-en-tête>.

<msg-adr-longueur> est un champ d'entier non signé de 4 bits, codant la longueur de toutes les adresses incluses dans ce message (<msg-orig-adr> ainsi que de chaque adresse incluse dans les blocs d'adresses comme défini au paragraphe 5.3), comme suit :

<msg-adr-longueur> = longueur d'une adresse en octets - 1

<msg-adr-longueur> est donc 3 pour les adresses IPv4, ou 15 pour les adresses IPv6.

adresse-longueur est une variable dont la valeur est la longueur d'une adresse en octets et est calculée comme suit :

adresse-longueur = <msg-adr-longueur> + 1

<msg-taille> est un champ d'entier non signé de 16 bits, qui spécifie le nombre d'octets qui constituent le <message>, incluant le <msg-en-tête>.

<msg-orig-adr> est omis si le fanion mhasorig est à zéro ("0") ; autrement, c'est un identifiant de longueur égale à la longueur d'adresse qui peut servir à identifier de façon univoque le routeur MANET qui a généré le message.

<msg-bond-limite> est omis si le fanion mhashoplimit est à zéro ("0") ; autrement, c'est un champ d'entier non signé de 8 bits qui peut contenir le nombre maximum de bonds sur lesquels le message devrait être encore transmis.

<msg-bond-compte> est omis si le fanion mhashopcount est à zéro ("0") ; autrement, c'est un champ d'entier non signé de

8 bits qui peut contenir le nombre de bonds que le message a effectués.

<msg-seq-num> est omis si le fanion mhasseqnum est à zéro ("0") ; autrement, c'est un champ d'entier non signé de16 bits qui peut contenir un numéro de séquence, généré par le routeur MANET générateur du message.

5.3 Blocs d'adresses

Un bloc d'adresse peut spécifier une ou plusieurs adresses, qui vont toutes être longues de adresse-longueur octets, comme spécifié en utilisant le <msg-adr-longueur> dans le <msg-en-tête> du message qui contient le bloc d'adresse. Un bloc d'adresse peut aussi spécifier des longueurs de préfixes qui peuvent être appliquées à toutes les adresses dans le bloc d'adresses, si c'est approprié. Cela permet à un bloc d'adresses de spécifier des adresses ou des préfixes d'adresse. Un protocole peut spécifier qu'une adresse avec une longueur de préfixe maximum (égale à la longueur d'adresse en bits, c'est-à-dire, 8 * adresse-longueur) est considérée comme étant une adresse, plutôt qu'un préfixe d'adresse, permettant donc qu'un bloc d'adresses contienne un mélange d'adresses et de préfixes d'adresse. Le terme commun "objet d'adresse" est utilisé dans cette spécification pour couvrir les deux ; noter qu'un objet d'adresse dans un bloc d'adresse inclut toujours la longueur de préfixe, si il est présent.

Une adresse est spécifiée comme une séquence de adresse-longueur octets de la forme Tête:Milieu:Queue. Il n'y a pas de signification associée à Tête, Milieu, ou Queue ; cette représentation est seulement pour permettre l'agrégation des adresses, qui ont souvent des parties communes (par exemple, des préfixes communs ou plusieurs adresses IPv6 sur la même interface). Un bloc d'adresses contient un ensemble ordonné d'adresses qui partagent toutes la même Tête et la même Queue, mais ont des Milieux individuels. Indépendamment, Tête et Queue peuvent être vides, permettant la représentation de objets d'adresse qui n'ont pas de Têtes ou Queues communes. Des exemples détaillés de bloc d'adresses sont inclus dans l'Appendice C.1.

Un bloc d'adresse peut spécifier des préfixes d'adresse :
- o avec une seule longueur de préfixe pour tous les préfixes d'adresse OU
- o avec une longueur de préfixe pour chaque préfixe d'adresse.

<adresse-bloc> est défini par :

```
<adresse-bloc> := <nomb-adr>
                  <adr-fanions>
                  (<tête-longueur><tête>?)?
                  (<longueur de queue><queue>?)?
                  <milieu>*
                  <prefixe-longueur>*
```

où :

<nomb-adr> est un champ d'entier non signé de 8 bits contenant le nombre d'adresses représenté dans le bloc d'adresses, qui NE DOIT PAS être zéro.

<adr-fanions> est un champ de 8 bits qui spécifie l'interprétation du reste du bloc d'adresses :

bit 0 (ahashead) : si il est à zéro ("0"), alors <tête-longueur> et <tête> ne sont pas inclus dans le <adresse-bloc>. Si il est établi ("1"), alors <tête-longueur> est inclus dans le <adresse-bloc>, et <tête> est inclus dans le <adresse-bloc> sauf si <tête-longueur> est zéro.

bit 1 (ahasfulltail) et bit 2 (ahaszerotail) : sont interprétés conformément au Tableau 1. Une combinaison non montrée dans ce tableau NE DOIT PAS être utilisée.

bit 3 (ahassingleprelen) et bit 4 (ahasmultiprelen) : sont interprétés conformément au Tableau 2. Une combinaison non montrée dans ce tableau NE DOIT PAS être utilisée.

bits 5-7 : sont RÉSERVÉS et DEVRAIENT être chacun à zéro ("0") à l'émission et DEVRAIENT être ignorés à réception.

ahasfulltail	ahaszerotail	<long. queue>	<queue>
0	0	non inclus	non inclus
1	0	inclus	inclus sauf si <longueur de queue> est zéro
0	1	inclus	non inclus

Tableau 1 : Interprétation des fanions ahasfulltail et ahaszerotail

ahassingle prelen	ahasmulti prelen	nombre de champs <prefix-longueur>	longueur de préfixe du nième préfixe d'adresse en bits
0	0	0	8 * adresse-longueur
1	0	1	<prefixe-longueur>
0	1	<nomb-adr>	nième <prefixe-longueur>

Tableau 2 : Interprétation des fanions ahassingleprelen et ahasmultiprelen

<tête-longueur> si il est présent, est un champ d'entier non signé de 8 bits qui contient le nombre d'octets dans la tête de toutes les adresses dans le bloc d'adresse, c'est-à-dire, chaque élément <tête> inclus est long de <tête-longueur> octets.

"tête-longueur" est une variable, définie comme égale à <tête-longueur>, si il est présent, ou 0 autrement.

<tête> est omis si tête-longueur est égal à 0 ; autrement, c'est un champ des tête-longueur octets les plus à gauche communs à toutes les adresses dans le bloc d'adresse.

<longueur-de-queue> si il est présent, est un champ d'entier non signé de 8 bits qui contient le nombre d'octets dans la queue de toutes les adresses dans le bloc d'adresse, c'est-à-dire, chaque élément <queue> inclus est long de <longueur-de-queue> octets.

"longueur-de-queue" est une variable, définie comme égale à <longueur-de-queue>, si il est présent, ou 0 autrement.

<queue> est omis si "longueur-de-queue" est égal à 0, ou si le fanion ahaszerotail est établi ("1") ; autrement, c'est un champ de "longueur-de-queue" octets les plus à droite commun à toutes les adresses dans le bloc d'adresses. Si le fanion ahaszerotail est établi ("1") alors les "longueur-de-queue" octets les plus à droite de toutes les adresses dans le bloc d'adresse sont à 0.

"mid-longueur" est une variable qui DOIT être non négative, définie par :

mid-longueur := adresse-longueur - tête-longueur - longueur-de-queue

c'est-à-dire, chaque élément <mid> inclus est long de mid-longueur octets.

<mid> est omis si mid-longueur est égal à 0 ; autrement, chaque <mid> est un champ de longueur mid-longueur octets, représentant le milieu de l'adresse correspondante dans le bloc d'adresse. Quand il n'est pas omis, un bloc d'adresse contient exactement <nomb-adr> champs <mid>.

<prefix-longueur> est un champ d'entier non signé de 8 bits contenant la longueur, en bits, d'un préfixe d'adresse. Si le fanion ahassingleprelen est établi ("1") alors un seul champ <prefix-longueur> est inclus contenant la longueur de préfixe de toutes les adresses dans le bloc d'adresses. Si le fanion ahasmultiprelen est établi ("1") alors <num-addr> champs <prefix-longueur> sont inclus, dont chacun contient la longueur de préfixe du préfixe d'adresse correspondant dans le bloc d'adresse (dans le même ordre). Autrement, aucun champ <prefix-longueur> n'est présent ; chaque objet Adresse peut être considéré comme ayant une longueur de préfixe égale à 8 * adresse-longueur bits. Le bloc d'adresses est mal formé si un élément <prefix-longueur> a une valeur supérieure à 8 * adresse-longueur.

5.4 TLV et blocs de TLV

Un TLV permet l'association d'un attribut arbitraire à un message ou paquet, ou à une seule adresse ou ensemble contigu d'adresses dans un bloc d'adresses. L'attribut (valeur) est constitué d'un nombre entier d'octets consécutifs. Des attributs différents ont des types différents ; les attributs inconnus à l'analyse peuvent être sautés.

Les TLV sont groupés en blocs de TLV, avec tous les TLV au sein d'un bloc de TLV qui associent des attributs soit au paquet (pour le bloc de TLV dans l'en-tête de paquet) au message (pour le bloc de TLV qui suit immédiatement l'en-tête de message) ou aux adresses dans le bloc d'adresse immédiatement précédant. Les TLV individuels dans un bloc de TLV qui suit immédiatement un bloc d'adresse peuvent associer des attributs à une seule adresse, à une gamme d'adresses, ou à toutes les adresses dans ce bloc d'adresses. Quand il associe un attribut à plus d'une adresse, un TLV peut inclure une valeur pour toutes les adresses ou une valeur par adresse. Des exemples détaillés de TLV sont inclus dans l'Appendice C.2.

Un bloc de TLV est défini par :

<tlv-bloc> := <tlvs-longueur> <tlv>*

où :

<tlvs-longueur> est un champ d'entier non signé de 16 bits qui contient le nombre total d'octets dans tous les éléments <tlv> suivant immédiatement (<tlvs-longueur> non inclus).

<tlv> est comme défini au paragraphe 5.4.1.

5.4.1 TLV

Il y a trois sortes de TLV, chacune représentée par un élément <tlv> :
- o un TLV Paquet, inclus dans le bloc de TLV Paquet dans un en-tête de paquet,
- o un TLV Message, inclus dans le bloc de TLV Message dans un message, avant tout bloc Adresses.
- o un TLV Bloc d'adresses, inclus dans un bloc de TLV Bloc d'adresses suivant un bloc d'adresses. Un TLV Bloc d'adresse s'applique à :
  - * tous les objets Adresse dans le bloc d'adresse, OU
  - * toute séquence continue d'objets Adresse dans le bloc d'adresse, OU
  - * un seul objet Adresse dans le bloc d'adresses.

<tlv> est défini par :

```
<tlv> := <tlv-type>
         <tlv-fanions>
         <tlv-type-ext>?
         (<index-start><index-stop>?)?
         (<longueur><valeur>?)?
```

où :

<tlv-type> est un champ d'entier non signé de 8 bits qui spécifie le type du TLV, spécifique de la sorte de TLV (c'est-à-dire, TLV Paquet, TLV Message, ou TLV Bloc d'adresses).

<tlv-fanions> est un champ de 8 bits qui spécifie l'interprétation du reste du TLV :

bit 0 (thastypeext) : si il est à zéro ("0"), alors <tlv-type-ext> n'est pas inclus dans le <tlv>. Si il est établi ("1") alors <tlv-type-ext> est inclus dans le <tlv>.

bit 1 (thassingleindex) et bit 2 (thasmultiindex) : sont interprétés conformément au Tableau 3. Une combinaison non montrée dans ce tableau NE DOIT PAS être utilisée. Ces deux fanions DOIVENT être à zéro ("0") dans les TLV Paquet et les TLV Message.

bit 3 (thasvalue) et bit 4 (thasextlen) : sont interprétés conformément au Tableau 4. Une combinaison non montrée dans ce tableau NE DOIT PAS être utilisée.

bit 5 (tismultivalue) : ce fanion sert à spécifier comment le champ <valeur> est interprété, comme spécifié ci-dessous.

Ce fanion DOIT être à zéro ("0") dans les TLV Paquet et les TLV Message, si le fanion thasmultiindex est à zéro ("0") ou si le fanion thasvalue est à zéro ("0").

bits 6-7 : sont RÉSERVÉS et DEVRAIENT chacun être à zéro ("0") à l'émission et ignorés à réception.

thassingleindex	thasmultiindex	<index-start>	<index-stop>
0	0	non inclus	non inclus
1	0	inclus	non inclus
0	1	inclus	inclus

Tableau 3 : Interprétation des fanions thassingleindex et thasmultiindex

thasvalue	thasextlen	<longueur>	<valeur>
0	0	non inclus	non inclus
1	0	8 bits	inclus sauf si <longueur> est zéro
1	1	16 bits	inclus sauf si <longueur> est zéro

Tableau 4 : Interprétation des fanions thasvalue et thasextlen

<tlv-type-ext> est un champ d'entier non signé de 8 bits, qui spécifie une extension au TLV Type, spécifique du type et de la sorte de TLV (c'est-à-dire, TLV Paquet, TLV Message, ou TLV Bloc d'adresses).

tlv-type-ext est une variable, définie comme égale à <tlv-type-ext>, si il est présent, ou 0 autrement.

tlv-fulltype est une variable, définie par : tlv-fulltype = 256 * <tlv-type> + tlv-type-ext

<index-start> et <index-stop> quand ils sont présents, seulement dans un TLV Bloc d'adresses, sont chacun un champ d'entier non signé de 8 bits.

index-start et index-stop sont des variables, définies en accord avec le Tableau 5.
La variable end-index est définie comme suit :
* Pour les TLV Message et Paquet : end-index := 0
* Pour les TLV Bloc d'adresses : end-index := <num-addr> - 1

Un TLV Bloc d'adresses s'applique aux objets Adresse depuis la position index-start jusqu'à la position index-stop (incluses) dans le bloc d'adresses, où le premier objet Adresse a la position zéro.

thassingleindex	thasmultiindex	index-start :=	index-stop :=
0	0	0	end-index
1	0	<index-start>	<index-start>
0	1	<index-start>	<index-stop>

Tableau 5 : Interprétation des fanions thassingleindex et thasmultiindex

nombre-valeurs est une variable, définie par : nombre-valeurs = index-stop - index-start + 1

<longueur> est omis ou est un champ d'entier non signé de 8 bits ou de 16 bits conformément au Tableau 4. Si le fanion tismultivalue est établi ("1") alors <longueur> DOIT être un multiple entier de nombre-valeurs, et la variable seule-longueur est définie par : seule-longueur := <longueur> / nombre-valeurs

Si le fanion tismultivalue est à zéro ("0") alors la variable seule-longueur est définie par :

seule-longueur := <longueur>

<valeur> si il est présent (voir le Tableau 4) est un champ de longueur <longueur> octets. Dans un TLV Bloc d'adresses, <valeur> est associé aux objets Adresse de la position index-start à la position index-stop, incluses. Si le fanion tismultivalue est à zéro ("0") alors ce champ entier est associé à chacun des objets Adresse indiqués. Si le fanion tismultivalue est établi ("1") alors ce champ est divisé également en champs nombre-valeurs, chacun de longueur seule-longueur octets, et ceux-ci sont associés, dans l'ordre, aux objets Adresse indiqués.

5.4.2 Usage des TLV

Un TLV associe un attribut à un paquet, un message, ou un ou plusieurs objets Adresse consécutifs dans un bloc d'adresses. L'interprétation et le traitement de cet attribut, et les relations (incluant l'ordre de traitement) entre les différents attributs associés à la même entité DOIVENT être définies par tout protocole qui utilise la présente spécification.

Tout protocole qui utilise la présente spécification DOIT définir les comportements appropriés si ces informations associées ne sont pas cohérentes, en particulier si deux TLV de même type avec des valeurs différentes s'appliquent à la même entité (paquet, message, ou adresse) mais ceci n'est pas significatif. Le protocole DOIT aussi spécifier un ordre de traitement approprié pour les TLV associés à une entité donnée.

5.5 Éléments mal formés

Un élément est mal formé si il ne peut pas être analysé conformément à sa spécification syntaxique (y compris si les octets disponibles sont insuffisants). Si l'élément mal formé est dans l'en-tête de paquet, alors le paquet DOIT être éliminé en silence, et les messages contenus NE DOIVENT PAS être traités et NE DOIVENT PAS être transmis. Si l'élément mal formé est contenu dans un message (c'est-à-dire, est dans le bloc de TLV Message, Bloc d'adresse, ou un bloc de TLV Bloc d'adresses) alors ce message DOIT être éliminé en silence ; il NE DOIT PAS être traité et NE DOIT PAS être transmis.

6. Considérations relatives à l'IANA

Le présent document introduit quatre espaces de noms qui ont été enregistrés : types de message, types de TLV Paquet, types de TLV Message, et types de TLV Bloc d'adresses. Cette section spécifie les registres de l'IANA pour ces espaces de noms et donne des lignes directrices à l'autorité d'allocation des numéros de l'Internet concernant les enregistrements dans ces espaces de noms.

Les termes suivants sont utilisés avec les significations définies dans la [RFC5226] : "espace de noms", "valeur allouée", "enregistrement", "non alloué", "réservé", "allocation hiérarchique", et "expert désigné".

Les politiques suivantes sont utilisées avec les significations définies dans la [RFC5226] : "utilisation privée", "revue d'expert", et "action de normalisation".

6.1 Revue d'expert : lignes directrices d'évaluation

Pour les demandes d'enregistrement où une revue d'expert est requise, l'expert désigné DEVRAIT prendre en compte les recommandations générales suivantes :

- o L'objet de ces registres est de prendre en charge les protocoles standard et expérimentaux d'acheminement de MANET et les extensions à ces protocoles.
- o L'intention est que tous les enregistrements soient accompagnés d'une RFC publiée.
- o Afin de permettre l'enregistrement avant que la RFC soit approuvée pour publication, l'expert désigné peut approuver l'enregistrement une fois qu'il semble clair qu'on s'attend à la publication d'une RFC.
- o L'expert désigné va envoyer une demande à la liste de diffusion du groupe de travail MANET, ou à son successeur comme désigné par le Directeur de zone, pour des commentaires et relectures. Cette demande va inclure une référence au projet Internet qui demande l'enregistrement.
- o Avant l'expiration d'une période de 30 jours, l'expert désigné va soit approuver, soit refuser la demande d'enregistrement et publier une note de la décision à la liste de diffusion du groupe de travail MANET ou son successeur, ainsi que informer l'IANA et l'IESG. Une note de refus DOIT être justifiée par une explication et, dans les

cas où c'est possible, des suggestions sur la façon dont la demande peut être modifiée pour devenir acceptable, DEVRAIENT être fournies.

Pour le registre des types de message, les lignes directrices suivantes s'appliquent :

- L'enregistrement d'un type de message implique la création de deux registres pour les TLV Message spécifiques du type de message et les TLV Bloc d'adresses spécifiques du type de message. Le document qui demande l'enregistrement du type de message DOIT indiquer comment ces types de TLV spécifiques du type de message vont être alloués, à partir de toutes options de la [RFC5226], et toutes allocations initiales. L'expert désigné DEVRAIT prendre en compte les politiques d'allocation spécifiées pour ces registres lors de la revue de la demande d'allocation de type de message.

Pour les registres des types de TLV Paquet, Message, et Bloc d'adresses, les lignes directrices suivantes s'appliquent :

- Ces allocations sont hiérarchiques, c'est-à-dire, l'allocation d'un type crée un registre pour les types étendus correspondant à ce type. Le document qui demande l'enregistrement du type DOIT indiquer comment ces types étendus sont alloués, à partir de toutes les options de la [RFC5226], et toutes les allocations initiales. Normalement cette allocation devrait aussi subir une revue d'expert, mais avec la possible allocation de certaines extensions de type comme réservé, expérimental, et/ou privé.
- La demande pour un type de TLV DOIT inclure la spécification de la taille permise, de la syntaxe de toute structure interne, et de la signification, du champ Valeur (si il y en a un) du TLV.

Pour les registres de types de TLV Message et Bloc d'adresses, les lignes directrices supplémentaires suivantes s'appliquent :

- Les valeurs de type de TLV 0 à 127 sont communes pour tous les types de message. Les TLV qui reçoivent des enregistrements dans l'intervalle de 0 à 127 DEVRAIENT être de conception modulaire pour permettre la réutilisation entre les protocoles.
- Les valeurs de type de TLV 128 à 223 sont spécifiques du type de TLV Message, pertinentes seulement dans le contexte du type de message contenant. L'enregistrement de valeurs de type de TLV dans l'intervalle de 128 à 223 exige qu'un registre existe dans l'intervalle 128 à 223 pour une valeur spécifique de type de message (voir le paragraphe 6.2.1) et les enregistrements sont faits en accord avec les politiques d'allocation spécifiées pour ces registres spécifiques du type de message. Les types de TLV spécifiques du type de message DEVRAIENT être enregistrés pour les TLV que l'expert désigné estime trop spécifiques du type de message pour l'enregistrement d'une valeur de 0 à 127. Plusieurs différentes définitions de TLV PEUVENT être allouées à la même valeur de type de TLV dans l'intervalle de 128 à 223, étant donné qu'elles sont associées à des registres différents de type de TLV spécifique du type de message. Lorsque possible, les définitions existantes de TLV globaux et les définitions de TLV modulaires globales pour l'enregistrement dans la gamme de 0 à 127 DEVRAIENT être utilisées.

6.2 Types de message

Un nouveau registre a été créé pour les types de message, avec des allocations initiales et les politiques d'allocation spécifiées au Tableau 6.

Type	Description	Politique d'allocation
0-223	Non alloué	Revue d'expert
224-255	Non alloué	Utilisation expérimentale

Tableau 6 :Types de message

6.2.1 Création d'un registre de TLV spécifique du type de message

Quand un type de message est enregistré, les registres DOIVENT alors être spécifiés pour les TLV Message spécifiques du type de message (Tableau 8) et les TLV Bloc d'adresses spécifiques du type de message (Tableau 10). Un document qui crée un registre de TLV spécifique du type de message DOIT aussi spécifier le mécanisme par lequel les types de TLV spécifiques du type de message sont alloués, parmi ceux de la [RFC5226].

6.3 Types de TLV Paquet

Un nouveau registre pour les types de TLV Paquet a été créé, avec les allocations initiales et les politiques d'allocation spécifiées dans le Tableau 7.

Type	Description	Politique d'allocation
0-223	Non alloué	Revue d'expert
224-255	Non alloué	Utilisation expérimentale |

Tableau 7 : Types de TLV Paquet

6.3.1 Création d'un registre d'extension de type de TLV Paquet

Quand un type de TLV Paquet est enregistré, un nouveau registre pour les extensions de type de ce type doit être créé. Un document qui définit un type de TLV Paquet DOIT aussi spécifier le mécanisme par lequel ces extensions de type sont allouées, parmi ceux de la [RFC5226].

6.4 Types de TLV Message

Un nouveau registre pour les types de TLV Message indépendant du type de message a été créé, avec les allocations initiales et les politiques d'allocation spécifiées au Tableau 8.

Type	Description	Politique d'allocation
0-127	Non alloué	Revue d'expert
128-223	Spécifique du type de message	Réservé, voir le Tableau 9
224-255	Non alloué	Utilisation expérimentale

Tableau 8 : Types de TLV Message

Les types de TLV Message de 128 à 223 sont réservés pour les TLV Message spécifiques du type de message, pour lesquels un nouveau registre est créé avec l'enregistrement d'un type de message, et avec les allocations initiales et les politiques d'allocation spécifiées au Tableau 9.

Type	Description	Politique d'allocation
0-127	Commun à tous les types de message	Réservé
128-223	Spécifique du type de message	Voir ci-dessous
224-255	Commun à tous les types de message	Réservé

Tableau 9 : Types de TLV Message spécifiques du type de message

Les politiques d'allocation pour les types de TLV Message spécifique du type de message DOIVENT être spécifiés quand est créé le registre associé au type de message qui les contient, voir le paragraphe 6.2.1.

6.4.1 Création d'un registre d'extension de type de TLV Message

Si un type de TLV Message est enregistré, alors un nouveau registre pour les extensions de type de ce type doit être créé. Un document qui définit un type de TLV Message DOIT aussi spécifier le mécanisme par lequel ces extensions de type sont allouées, parmi ceux de la [RFC5226].

6.5 Types de TLV Bloc d'adresses

Un nouveau registre pour les types de TLV Bloc d'adresses indépendant du type de message a été créé, avec les allocations initiales et les politiques d'allocation spécifiées au Tableau 10.

Type	Description	Politique d'allocation
0-127	Non alloué	Revue d'expert
128-223	Spécifique du type de message	Réservé, voir le Tableau 11
224-255	Non alloué	Utilisation expérimentale

Tableau 10 : Types de TLV Bloc d'adresses

Les types de TLV Bloc d'adresses 128 à 223 sont réservés pour les TLV Bloc d'adresses spécifiques du type de message, pour lesquels un nouveau registre est créé avec l'enregistrement d'un type de message, et avec les allocations initiales et les

politiques d'allocation spécifiées au Tableau 11.

Type	Description	Politique d'allocation
0-127	Commun à tous les types de message	Réservé
128-223	Spécifique du type de message	voir ci-dessous
224-255	Commun à tous les types de message	Réservé

Tableau 11 : Types de TLV Bloc d'adresses spécifiques du type de message

Les politiques d'allocation pour les types de TLV Bloc d'adresses spécifiques du type de message DOIVENT être spécifiées à la création du registre associé au type de message contenant, voir le paragraphe 6.2.1.

6.5.1 Création d'un registre d'extension de type de TLV Bloc d'adresses

Quand un type de TLV Bloc d'adresses est enregistré, alors un nouveau registre pour les extensions de type de ce type doit être créé. Un document qui définit un type de TLV Message DOIT aussi spécifier le mécanisme par lequel ses extensions de type sont allouées, parmi ceux de la [RFC5226].

7. Considérations sur la sécurité

La présente spécification ne décrit pas de protocole ; elle décrit un format de paquet. À ce titre, elle ne spécifie aucune considération de sécurité ; elles sont l'affaire des protocoles qui utilisent cette spécification. Cependant, certains mécanismes de sécurité sont activés par cette spécification et peuvent faire partie d'un protocole qui utilise cette spécification. Les mécanismes qui peuvent faire partie d'une approche d'authentification et d'intégrité dans un protocole qui utilise cette spécification sont décrits au paragraphe 7.1. Les mécanismes qui peuvent faire partie d'une approche de la confidentialité dans un protocole qui utilise cette spécification sont décrits au paragraphe 7.2. Il n'est cependant pas exigé qu'un protocole qui utilise la présente spécification devrait utiliser l'un ou l'autre.

7.1 Suggestions d'authentification et d'intégrité

Les suggestions d'authentification et d'intégrité faites ici se fondent sur l'usage prévu dans l'Appendice B, spécifiquement que :

- Les messages sont conçus pour être des porteurs d'informations de protocole et PEUVENT, à chaque bond, être transmis et/ou traités par le protocole qui utilise la présente spécification.
- Les paquets sont destinés à porter un certain nombre de messages entre routeurs MANET voisins dans une seule transmission et sur un seul bond logique.

Par conséquent :

- Pour les messages transmis lorsque le message est inchangé par les routeurs MANET transmetteurs, l'authentification et l'intégrité de bout en bout PEUVENT être mises en œuvre, entre routeurs MANET avec une association de sécurité existante, en incluant un TLV Message convenable contenant une signature numérique dans le message. Comme <msg-bond-compte> et <msg-bond-limite> sont les seuls champs qui devraient être modifiés quand un tel message est transmis de cette manière, cette signature peut être calculée sur la base du message entier, incluant l'en-tête de message, avec les champs <msg-bond-compte> et <msg-bond-limite> réglés à 0, si ils sont présents.

- L'authentification et l'intégrité bond par bond au niveau paquet PEUVENT être mises en œuvre, entre routeurs MANET avec une association de sécurité existante, en incluant un TLV Paquet convenable contenant une signature numérique du paquet. Comme les paquets sont reçus comme ils sont transmis, cette signature peut être calculée sur la base du paquet entier ou de parties du paquet comme approprié.

7.2 Suggestions de confidentialité

La présente spécification ne permet pas explicitement de protéger la confidentialité du paquet/message. Une telle confidentialité serait normalement, quand c'est exigé, fournie bond par bond, soit par des mécanismes de couche de liaison, soit à la couche IP en utilisant la [RFC4301], et s'appliquerait seulement à un paquet.

Il est cependant possible pour un protocole qui utilise la présente spécification de protéger la confidentialité des informations incluses dans un TLV Paquet, Message, ou Bloc d'adresses qui spécifie que le champ Valeur de ce type de TLV doit être chiffré, ainsi que le mécanisme de chiffrement.

Dans un cas extrême, toutes les informations peuvent être chiffrées en définissant soit :

- Un paquet, consistant seulement en un en-tête de paquet (sans message) et contenant un TLV Paquet, où le type de TLV Paquet indique que son champ Valeur contient un ou plusieurs messages chiffrés. À réception, et une fois que ce TLV Paquet est déchiffré avec succès, ces messages peuvent alors être analysés en accord avec la présente spécification et traités en accord avec le protocole qui utilise cette spécification.

- Un message, consistant seulement en un en-tête de message et un seul TLV Message, où le type de TLV Message indique que son champ Valeur contient une version chiffrée des TLV Message, Bloc d'adresses, et TLV Bloc d'adresses restants du message. À réception, et une fois que ce TLV Message est déchiffré avec succès, le message complet peut alors être analysé en accord avec cette spécification et traité en accord avec le protocole qui utilise cette spécification.

Dans l'un et l'autre cas, le protocole DOIT définir le type de TLV chiffré, ainsi que le format du bloc de données chiffrées contenu dans le champ Valeur du TLV.

8. Contributeurs

La présente spécification est le résultat des efforts conjoints des contributeurs suivants de l'équipe de conception OLSRv2 (par ordre alphabétique) :

- Cedric Adjih, INRIA, France, <Cedric.Adjih@inria.fr>
- Emmanuel Baccelli, INRIA, France, <Emmanuel.Baccelli@inria.fr>
- Thomas Heide Clausen, LIX, Ecole Polytechnique, France, <T.Clausen@computer.org>
- Justin W. Dean, NRL, USA, <jdean@itd.nrl.navy.mil>
- Christopher Dearlove, BAE Systems, UK, <chris.dearlove@baesystems.com>
- Satoh Hiroki, Hitachi SDL, Japon, <hiroki.satoh.yj@hitachi.com>
- Philippe Jacquet, INRIA, France, <Philippe.Jacquet@inria.fr>
- Monden Kazuya, Hitachi SDL, Japon, <kazuya.monden.vw@hitachi.com>

9. Remerciements

Les auteurs tiennent à remercier l'équipe OLSR [RFC3626], incluant Anis Laouiti (INT, France), Pascale Minet, Laurent Viennot (tous deux de INRIA, France) et Amir Qayyum (Center for Advanced Research in Engineering, Pakistan) de leurs contributions. Elwyn Davies (Folly Consulting, UK), Lars Eggert (Nokia, Finland), Chris Newman (Sun Microsystems, USA), Tim Polk (NIST, USA) et Magnus Westerlund (Ericsson, Sweden) qui ont tous fourni des relectures détaillées et des commentaires significatifs.

Les auteurs tiennent à remercier les personnes suivantes des intenses discussions techniques, des premières révisions, et des commentaires sur la spécification et ses composants (par ordre alphabétique) :

- Brian Adamson (NRL)
- Teco Boot (Infinity Networks)
- Florent Brunneau (LIX)
- Ian Chakeres (CenGen)
- Alan Cullen (BAE Systems)
- Ulrich Herberg (LIX)
- Joe Macker (NRL)
- Yasunori Owada (Niigata University)
- Charlie E. Perkins (WiChorus)
- Henning Rogge (FGAN)
- Andreas Schjonhaug (LIX)

et tout le groupe de travail IETF MANET

10. Références

10.1 Références normatives

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC4291] R. Hinden, S. Deering, "Architecture d'adressage IP version 6", février 2006. (*MàJ par* 5952 et 6052, 8064) *(D.S.)*

[RFC5226] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, mai 2008. (*Remplace* RFC2434 *; remplacée par* RFC8126)

[SingleUNIX] IEEE Std 1003.1, et ISO/CEI JTC 1/SC22/WG15, "Single UNIX Specification, version 3, édition 2004", avril 2004.

10.2 Références pour information

[RFC3626] T. Clausen et P. Jacquet, éd., "Protocole d'acheminement d'état de liaison optimisé (OLSR)", octobre 2003. *(Exp.)*

[RFC4301] S. Kent et K. Seo, "Architecture de sécurité pour le protocole Internet", DOI 10.17487/RFC4301, décembre 2005. *(P.S.) (Remplace la* RFC2401)

[Stevens] Stevens, W., "TCP/IP Illustrated Volume 1 - The Protocols", 1994.

Appendice A. Multiplexage et démultiplexage

Le format de paquet et de message spécifié dans ce document est conçu pour permettre que zéro, un ou plusieurs messages soient contenus dans un seul paquet. Ces messages peuvent être du même ou de différents protocoles. Donc, un processus de multiplexage et démultiplexage DOIT être présent.

Le multiplexage de messages sur un routeur MANET donné en un seul paquet, plutôt que d'avoir chaque message qui génère son propre paquet, réduit le nombre total d'octets et le nombre de paquets transmis par ce routeur MANET.

Le processus de multiplexage et démultiplexage fonctionnant sur un accès UDP ou numéro de protocole IP donné, et ses protocoles associés, DOIT :

- Pour chaque type de message, un protocole -- sauf mention contraire, celui qui fait la réservation IANA pour ce type de message -- DOIT être désigné comme "propriétaire" de ce type de message.

- Les champs d'en-tête de paquet, y compris le bloc de TLV Paquet, sont utilisés par le processus de multiplexage et démultiplexage, qui PEUT rendre de telles informations disponibles pour être utilisées dans ses instances de protocole.

- Le champ <pkt-seq-num>, si il est présent, contient un numéro de séquence qui est incrémenté de 1 pour chaque paquet généré par un nœud. Le numéro de séquence après 65535 est 0. En d'autres termes, le numéro de séquence "revient à zéro" de la façon habituelle.

- Les messages entrants DOIVENT être soit éliminés en silence, soit DOIVENT être livrés à l'instance de protocole qui possède le type de message associé. Les messages entrants NE DEVRAIENT PAS être livrés à une autre instance de protocole et NE DEVRAIENT PAS être livrés à plus d'une instance de protocole.

- Les messages sortants d'un type donné DOIVENT être générés seulement par l'instance de protocole qui possède ce type de message et être livrés au processus de multiplexage et démultiplexage.

- Si deux protocoles souhaitent utiliser le même type de message, alors cette interaction DEVRAIT être spécifiée par le protocole qui est le propriétaire désigné de ce type de message.

Appendice B. Usage prévu

Cet appendice décrit l'usage prévu des champs d'en-tête de message, y compris de leur contenu et utilisation. D'autres utilisations de cette spécification sont permises.

Le format de message spécifié dans ce document est conçu pour porter la signalisation de protocole d'acheminement MANET entre routeurs MANET et pour prendre en charge l'arrosage de portée limitée ainsi que la livraison en point à point.

Les messages sont destinés à être capables d'être transmis sur un ou plusieurs bonds logiques, dans un nouveau paquet pour chaque bond logique. Chaque bond logique peut consister en un ou plusieurs bonds logiques.

Spécifiquement, l'arrosage de portée limitée est pris en charge pour les messages quand :

- Le champ <msg-orig-addr>, si il est présent, contient l'identifiant univoque du routeur MANET qui a généré le message.

- Le champ <msg-seq-num>, si il est présent, contient un numéro de séquence qui commence à 0 quand le premier message d'un certain type est généré par le nœud d'origine, et qui est incrémenté de 1 pour chaque message généré de ce type. Le numéro de séquence après 65535 est 0. En d'autres termes, le numéro de séquence "revient à zéro" de la façon habituelle.

- Si les champs <msg-orig-addr> et <msg-seq-num> sont tous deux présents, alors l'en-tête de message assure la suppression des dupliqués en utilisant l'identifiant qui consiste en les <msg-orig-adr>, <msg-seq-num>, et <msg-type> du message. Ils servent à identifier de façon univoque le message dans le MANET dans la période jusqu'à ce que <msg-seq-num> soit répété, c'est-à-dire, revienne à une valeur correspondante.

- Le champ <msg-bond-limite>, si il est présent, contient le nombre de bonds sur lequel il est permis au paquet de voyager avant d'être éliminé par un routeur MANET. La <msg-bond-limite> est réglée par le générateur du message et est utilisée pour empêcher les messages de circuler sans fin dans un MANET. Quand il transmet un message, un routeur MANET devrait diminuer la <msg-bond-limite> de 1, et le message devrait être éliminé quand <msg-bond-limite> atteint 0.

- Le champ <msg-bond-compte>, si il est présent, contient le nombre de bonds sur lesquels le paquet a voyagé à travers le MANET. Le <msg-bond-compte> est réglé à 0 par le générateur du message et est utilisé pour empêcher les messages de circuler sans fin dans un MANET. Quand il transmet un message, un routeur MANET devrait augmenter <msg-bond-compte> de 1 et devrait éliminer le message quand <msg-bond-compte> atteint 255.

- Si les champs <msg-bond-limite> et <msg-bond-compte> sont tous deux présents, alors l'en-tête de message fournit les informations pour prendre les décisions de transmission pour l'arrosage de portée limitée. Ce peut être par tout mécanisme d'arrosage approprié spécifié par un protocole qui utilise la présente spécification.

Appendice C. Exemples

Le présent appendice contient des exemples de parties de cette spécification.

C.1 Exemples de bloc d'adresses

Les exemples suivants illustrent comment certaines combinaisons d'adresses peuvent être efficacement incluses dans les blocs d'adresses. Ces exemples sont pour IPv4, avec des longueurs d'adresse égales à 4.
a, b, c, etc. représentent des valeurs d'octet distinctes, non zéro.

Noter qu'il est permis d'utiliser une représentation moins efficace, en particulier une dans laquelle les fanions ahashead et ahasfulltail sont à zéro ("0"), et donc tête-longueur = 0, longueur de queue = 0, mid-longueur = adresse-longueur, et (sans préfixe d'adresse) le bloc d'adresse consiste en le nombre d'adresses, <adr-fanions> de valeur 0, et une liste non agrégée d'adresses. C'est la façon la plus efficace de représenter une seule adresse, et la seule façon de représenter, par exemple, a.b.c.d et e.f.g.h dans un bloc d'adresse.

Exemples:

o Pour inclure a.b.c.d, a.b.e.f, et a.b.g.h :
* tête-longueur = 2 ;
* longueur-de-queue = 0 ;
* mid-longueur = 2 ;
* <adr-fanions> a ahashead établi (valeur 128) ;
* <longueur-de-queue> et <queue> sont omis.
Le bloc d'adresse est alors 3 128 2 a b c d e f g h (11 octets).

o Pour inclure a.b.c.g et d.e.f.g :
* tête-longueur = 0 ;
* longueur-de-queue = 1 ;
* mid-longueur = 3 ;
* <adr-fanions> a ahasfulltail établi (valeur 64) ;
* <tête-longueur> et <tête> sont omis.
Le bloc d'adresse est alors 2 64 1 g a b c d e f (10 octets).

o Pour inclure a.b.d.e et a.c.d.e :
* tête-longueur = 1 ;
* longueur-de-queue = 2 ;
* mid-longueur = 1 ;
* <adr-fanions> a ahashead et ahasfulltail établis (valeur 192).
Le bloc d'adresse est alors 2 192 1 a 2 d e b c (9 octets).

o Pour inclure a.b.0.0, a.c.0.0, et a.d.0.0 :
* tête-longueur = 1 ;
* longueur-de-queue = 2 ;
* mid-longueur = 1 ;
* <adr-fanions> a ahashead et ahaszerotail établis (valeur 160);
* <queue> est omis.
Le bloc d'adresse est alors 3 160 1 a 2 b c d (8 octets).

o Pour inclure a.b.0.0 et c.d.0.0 :
* tête-longueur = 0 ;
* longueur-de-queue = 2 ;
* mid-longueur = 2 ;
* <adr-fanions> a ahaszerotail établi (valeur 32) ;
* <tête> et <queue> sont omis.
Le bloc d'adresse est alors 2 32 2 a b c d (7 octets).

o Pour inclure a.b.0.0/n et c.d.0.0/n :
* tête-longueur = 0 ;
* longueur-de-queue = 2 ;
* mid-longueur = 2 ;
* <adr-fanions> a ahaszerotail et ahassingleprelen établis (valeur 48);
* <tête> et <queue> sont omis.
Le bloc d'adresse est alors 2 48 2 a b c d n (8 octets).

o Pour inclure a.b.0.0/n et c.d.0.0/m :
* tête-longueur = 0 ;
* longueur-de-queue = 2 ;
* mid-longueur = 2 ;
* <adr-fanions> a ahaszerotail et ahasmultiprelen établis (valeur 40) ;
* <tête> et <queur> sont omis.
Le bloc d'adresse est alors 2 40 2 a b c d n m (9 octets).

C.2 Exemples de TLV

Supposons que la définition d'un TLV Bloc d'adresse de type EXAMPLE1 (et sans extension de type) ait des valeurs d'un

seul octet par adresse. Les valeurs a, a, b, et c peuvent être associées aux quatre adresses dans le précédent bloc d'adresses d'un certain nombre de façons, où c est une valeur par défaut qui peut être omise.

Exemples :

o En utilisant un TLV multi valeurs pour couvrir toutes les adresses :
* <tlv-fanions> a thasvalue et tismultivalue établis (valeur 20) ;
* <index-start> et <index-stop> sont omis ;
* <longueur> = 4 (seule-longueur = 1).
* Le TLV est alors EXAMPLE1 20 4 a a b c (7 octets).

o En utilisant un TLV multi valeurs et omettant la dernière adresse :
* <tlv-fanions> a thasmultiindex, thasvalue, et tismultivalue établis (valeur 52) ;
* <index-start> = 0 ;
* <index-stop> = 2 ;
* <longueur> = 3 (seule-longueur = 1).
* Le TLV est alors EXAMPLE1 52 0 2 3 a a b (8 octets).

o En utilisant deux TLV d'une seule valeur et en omettant la dernière adresse. D'abord :
* <tlv-fanions> a thasmultiindex et thasvalue établis (valeur 48) ;
* <index-start> = 0 ;
* <index-stop> = 1 ;
* <longueur> = 1 ;
* <valeur> = a.
* Le TLV est alors EXAMPLE1 48 0 1 1 a (6 octets).
Ensuite :
* <tlv-fanions> a thassingleindex et thasvalue établis (valeur 80) ;
* <index-start> = 2 ;
* <index-stop> est omis ;
* <longueur> = 1 ;
* <valeur> = b.
* Le TLV est alors EXAMPLE1 80 2 1 b (5 octets).
La longueur totale des TLV est 11 octets.

Dans ce cas, le premier est le plus efficace. Dans d'autres cas, des schémas comme les autres peuvent être préférés. Sans considération de l'efficacité, n'importe lequel peut être utilisé.

On suppose la définition d'un TLV Bloc d'adresses de type EXAMPLE2 (et pas d'extension de typ) qui n'a pas de valeur et qui est à associer à la seconde et la troisième adresse dans un bloc d'adresses. Cela peut être indiqué avec un seul TLV :
o <tlv-fanions> a thasmultiindex établi (valeur 32) ;
o <index-start> = 1 ;
o <index-stop> = 2 ;
o <longueur> et <valeur> sont omis.
o Le TLV est alors EXAMPLE2 32 1 2 (4 octets).

On suppose la définition d'un TLV Message de type EXAMPLE3 (et pas d'extension de type) qui peut prendre un champ Valeur de n'importe quelle longueur. Pour un tel TLV avec 8 octets de données (a à h) :
o <tlv-fanions> a thasvalue établi (valeur 16) ;
o <index-start> et <index-stop> sont omis ;
o <longueur> = 8.
o Le TLV est alors EXAMPLE3 16 8 a b c d e f g h (11 octets).

Si, dans cet exemple, le nombre d'octets de données était 256 ou plus, alors <tlv-fanions> aurait aussi thasextlen établi et aurait la valeur 24. La longueur exigerait deux octets (celui de plus fort poids en premier). La longueur de TLV serait 4 + N octets, où N est le nombre d'octets de données (il peut être 3 + N octets si N est 255 ou moins).

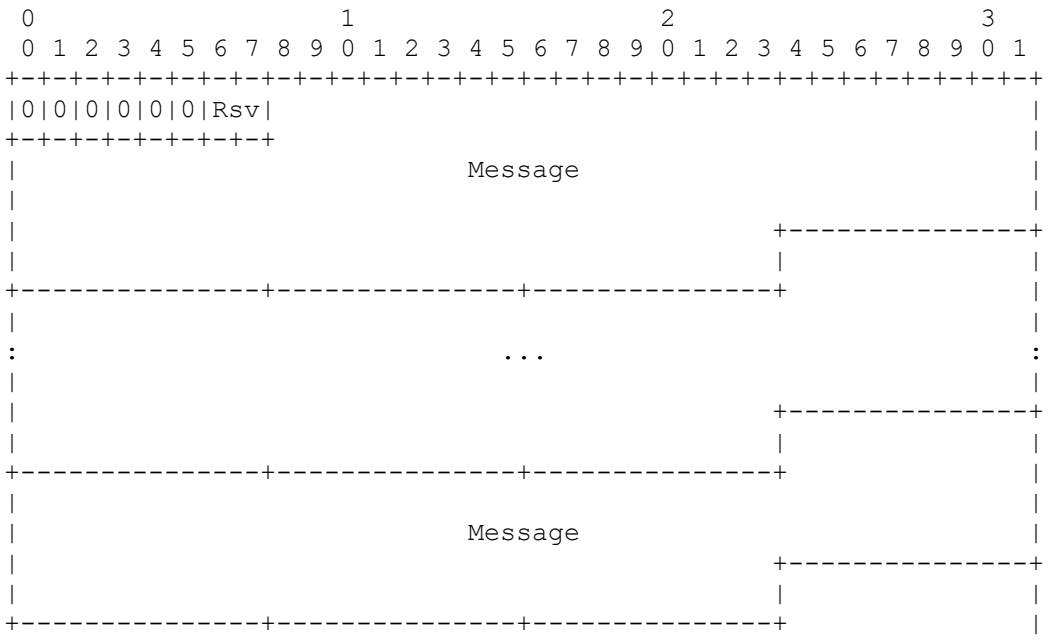
Appendice D. Illustrations

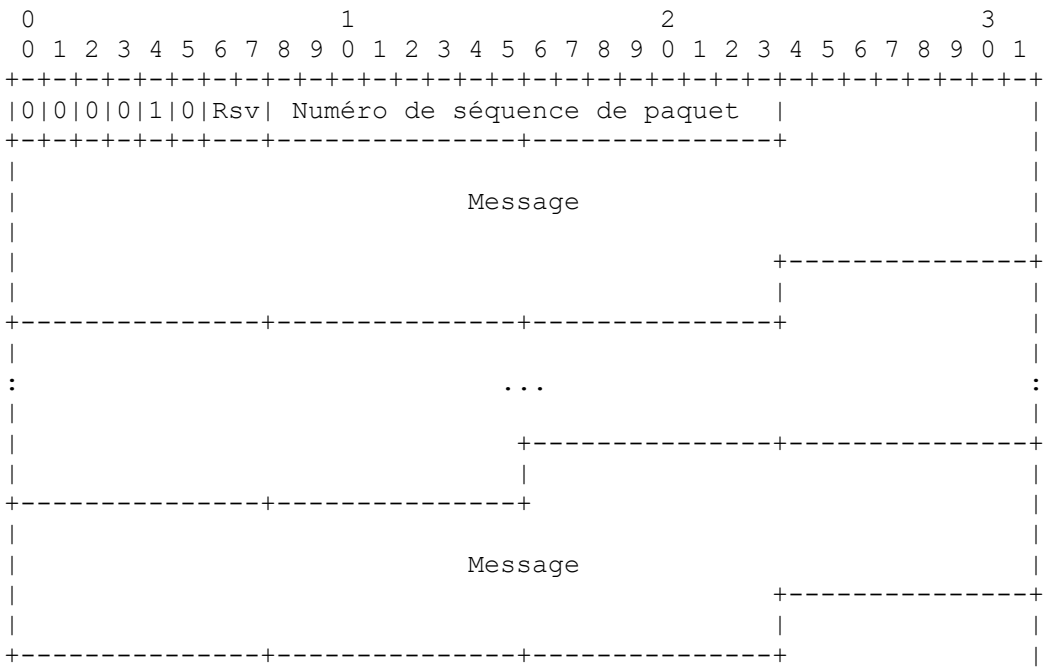
Cet appendice pour information illustre les éléments qui sont normativement spécifiés à la Section 5.

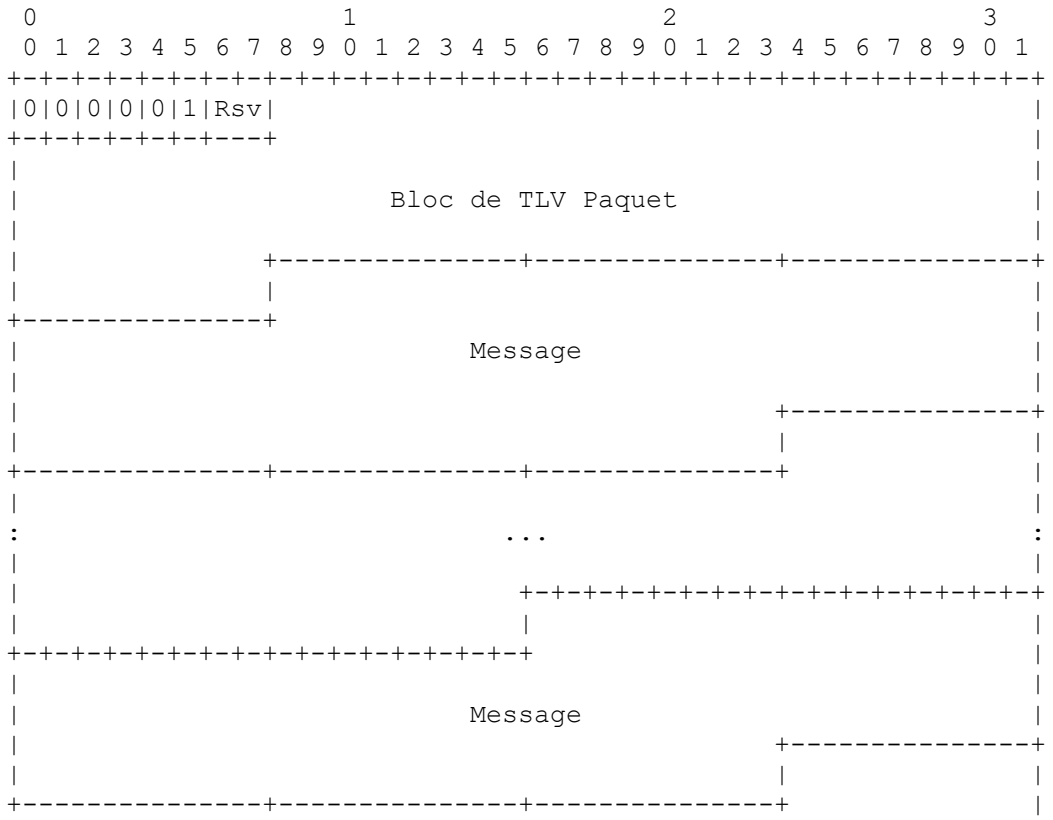
Les bits marqués Rsv devraient être à zéro ("0"). Les bits marqués M peuvent être à zéro ("0") ou établis ("1").

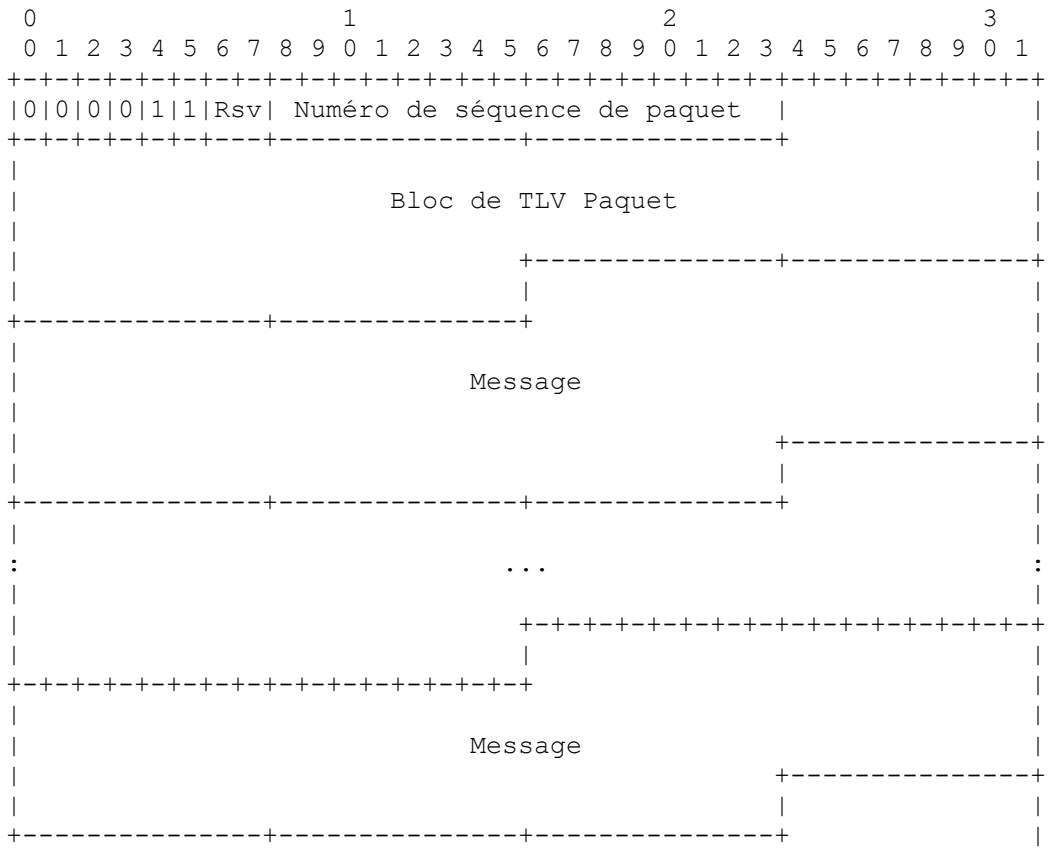
D.1 Paquet

Cette section illustre les options possibles pour l'élément <paquet>. Elles sont différenciées par le champ de fanions dans le premier octet. Le paquet peut inclure tout nombre (zéro, un ou plusieurs) de messages. Le nombre de messages est déterminé à partir de quand le paquet est épuisé, étant donnée la longueur de paquet provenant de la couche réseau.





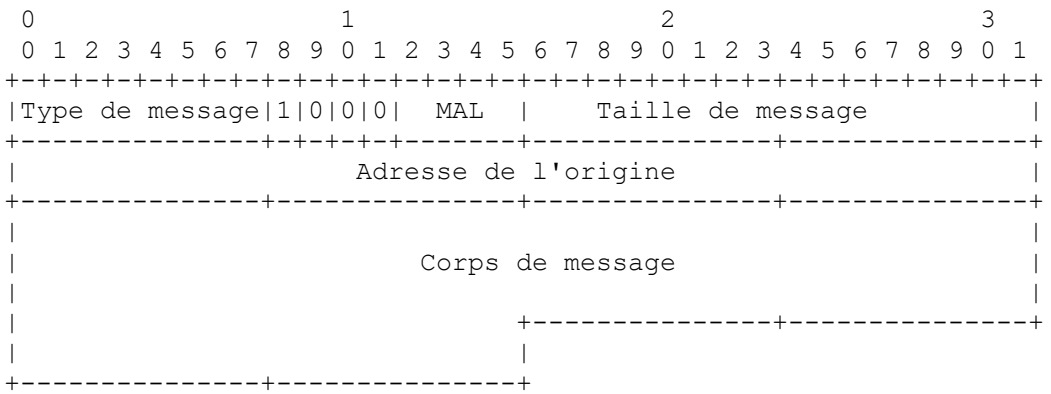


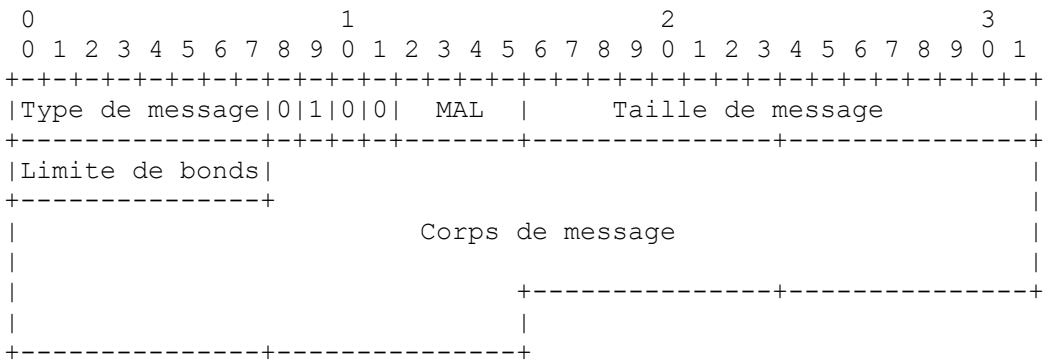


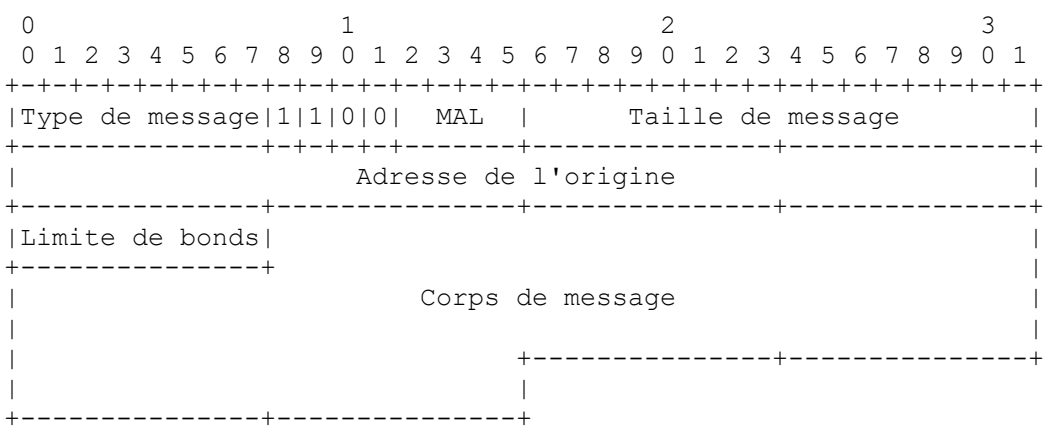
D.2 Message

Cette section illustre les options possibles pour l'élément <message>. Elles sont différenciées par leur second octet (fanions). La longueur du corps de message est déterminée en utilisant le champ de taille de message, qui est la longueur

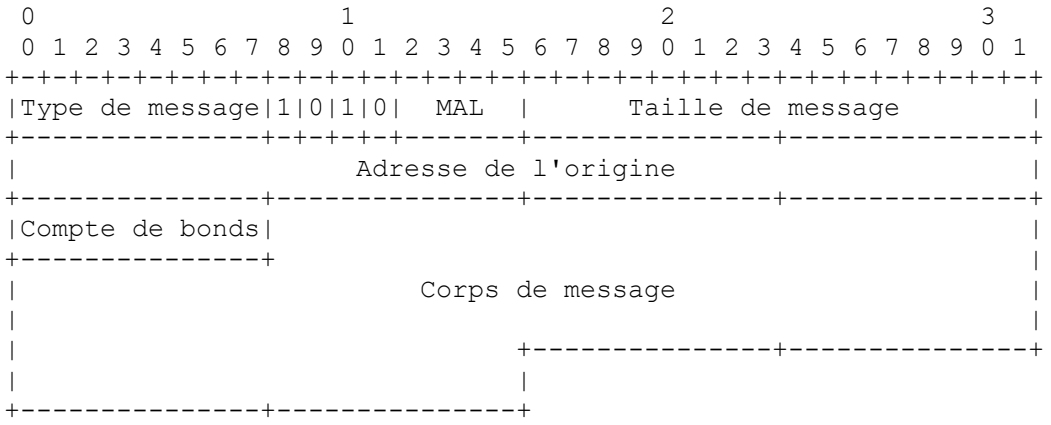
combinée de tous les champs montrés. Le champ marqué MAL définit la longueur de toutes les adresses (y compris l'adresse de l'origine, si elle est présente, et toutes les adresses dans les blocs d'adresses) en octets, comme un de plus que la valeur dans le champ.

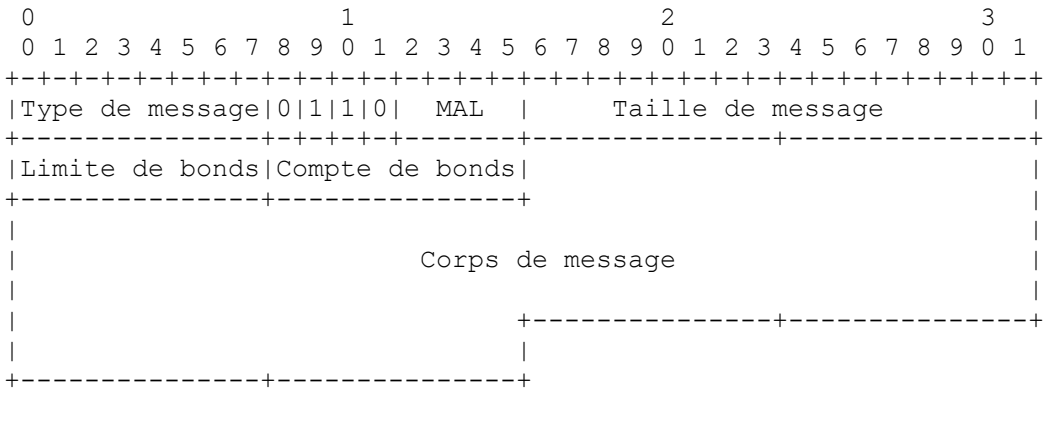


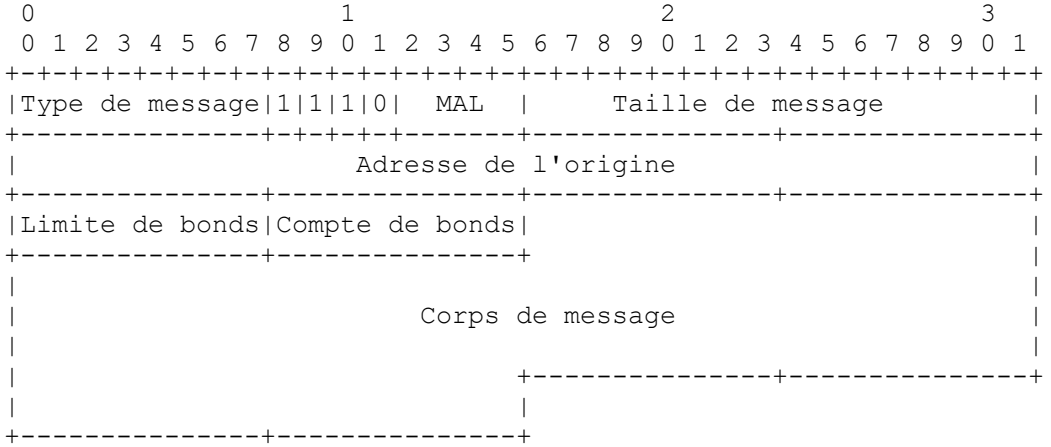




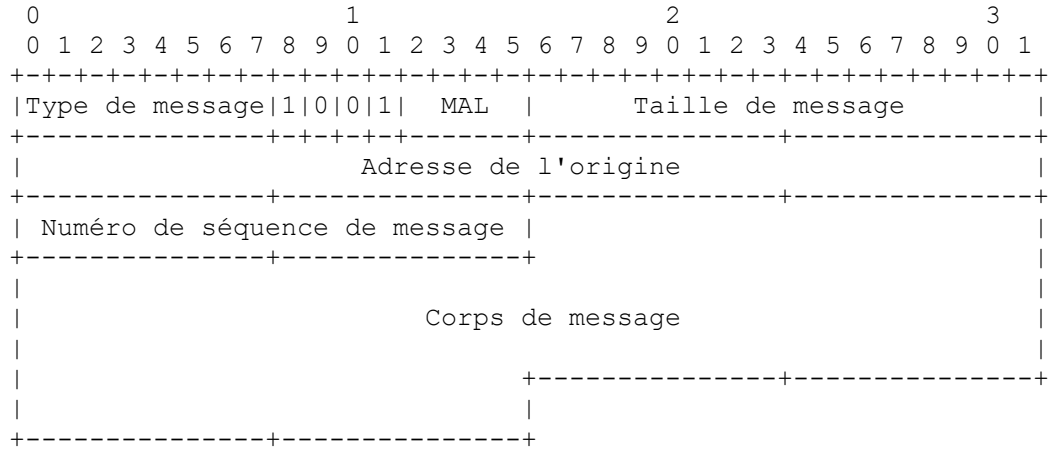
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0
0
1
0
MAL
Taille de message
Compte de bonds
Corps de message

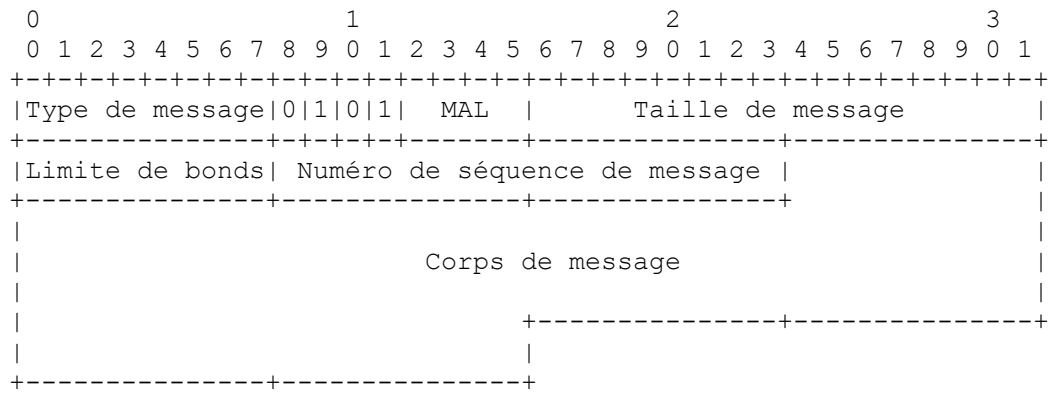

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1
0
1
0
MAL
Taille de message
Adresse de l'origine
Compte de bonds
Corps de message

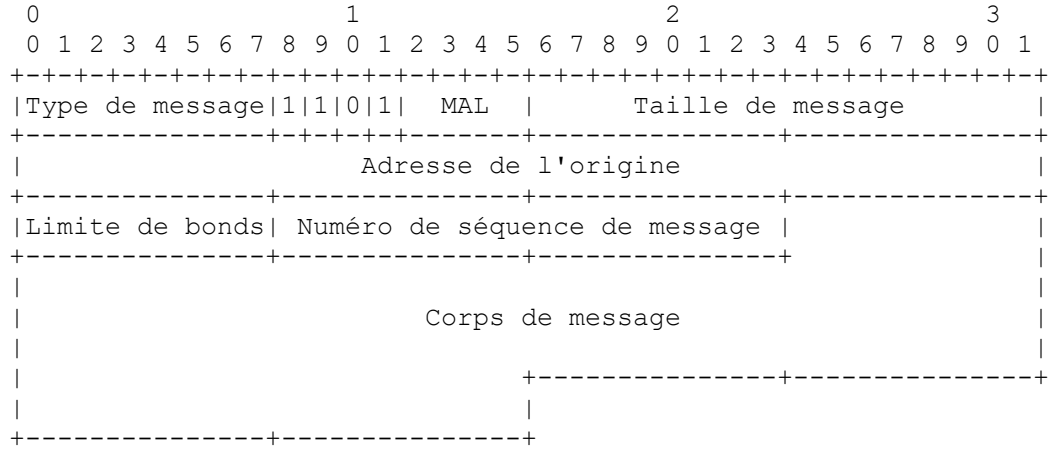

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0
1
1
0
MAL
Taille de message
Limite de bonds
Compte de bonds
Corps de message


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1
1
1
0
MAL
Taille de message
Adresse de l'origine
Limite de bonds
Compte de bonds
Corps de message

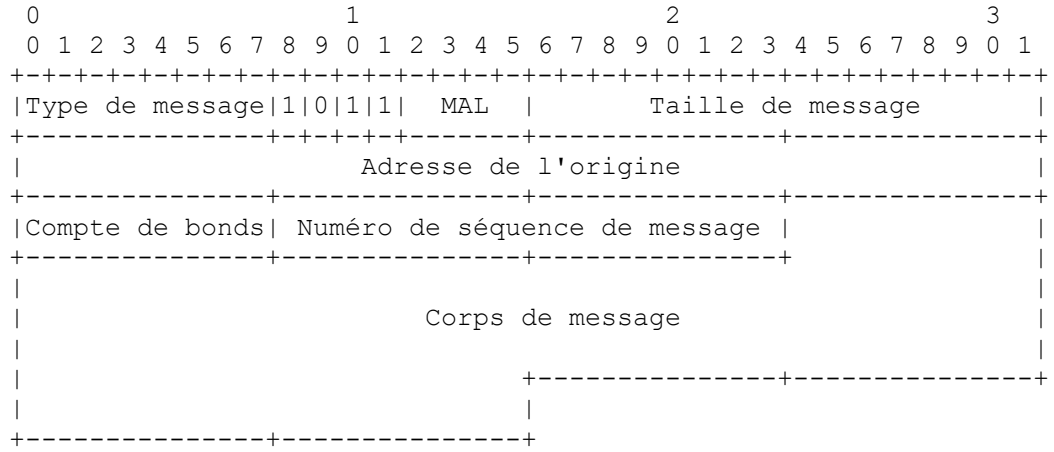
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0 0 0 1
MAL
Taille de message
Numéro de séquence de message
Corps de message

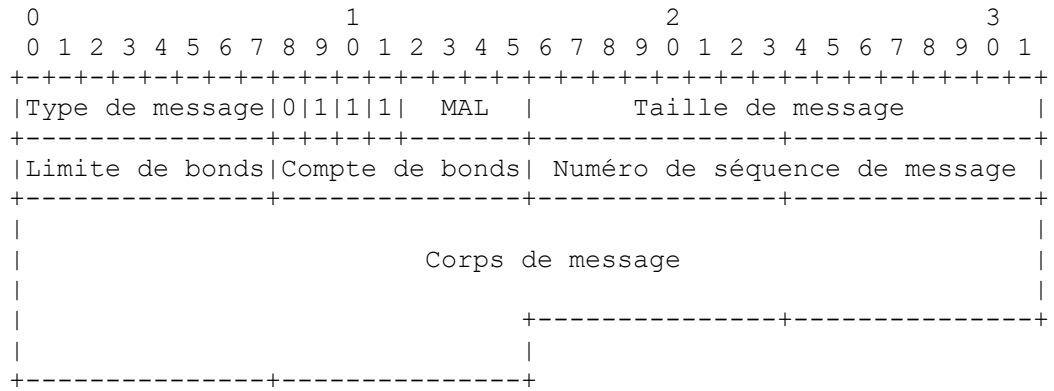

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1 0 0 1
MAL
Taille de message
Adresse de l'origine
Numéro de séquence de message
Corps de message

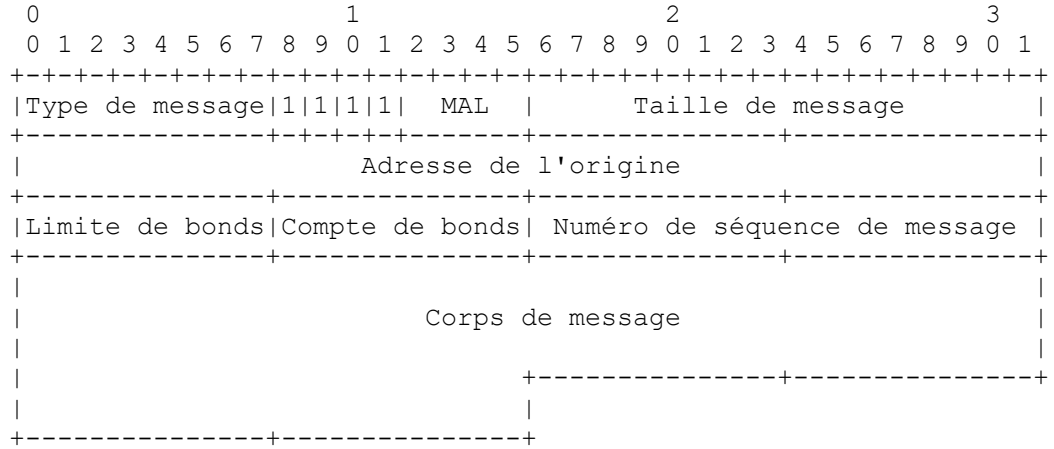

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0 1 0 1
MAL
Taille de message
Limite de bonds
Numéro de séquence de message
Corps de message


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1 1 0 1
MAL
Taille de message
Adresse de l'origine
Limite de bonds
Numéro de séquence de message
Corps de message

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0
0
1
1
MAL
Taille de message
Compte de bonds
Numéro de séquence de message
Corps de message

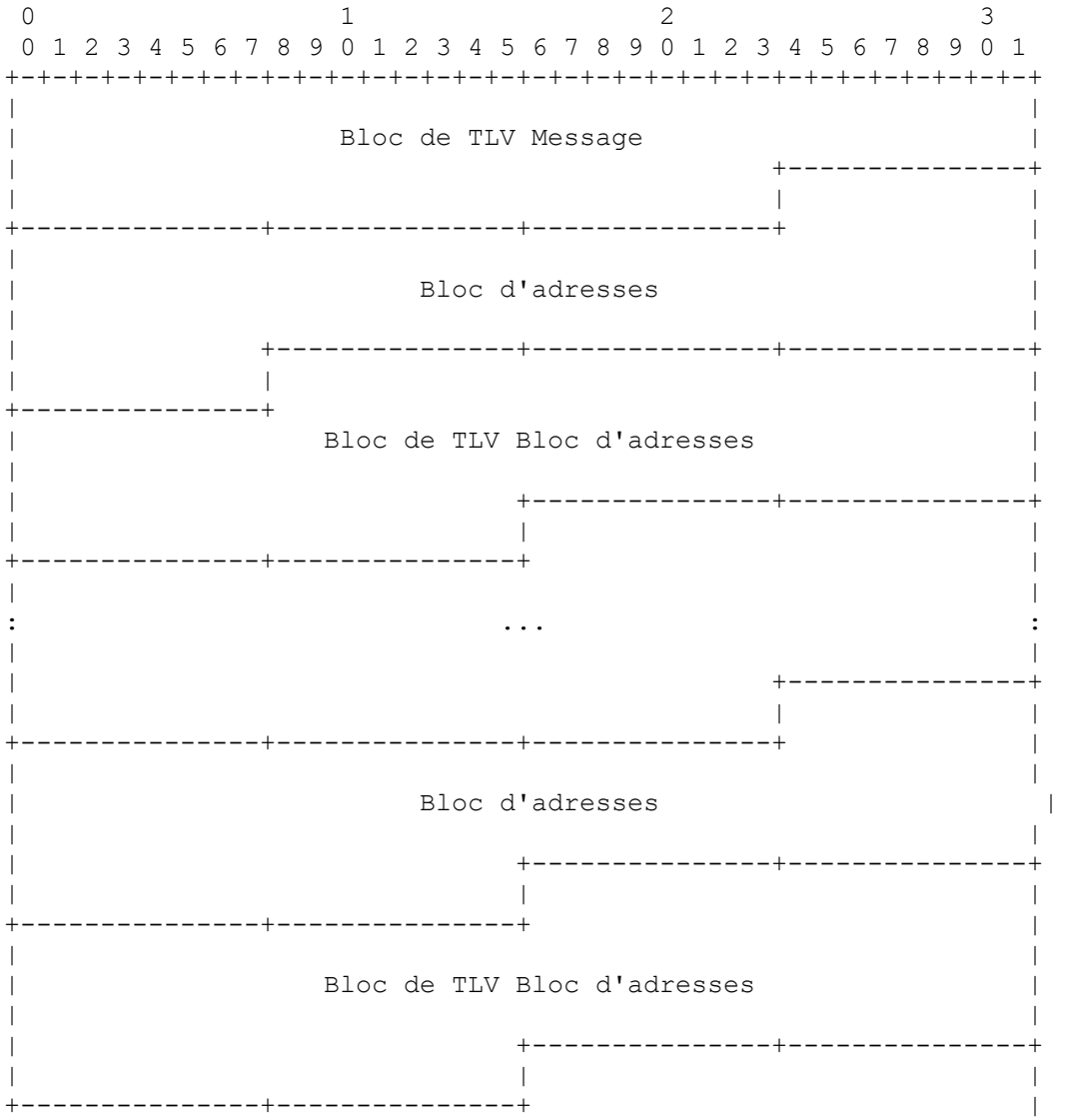

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1
0
1
1
MAL
Taille de message
Adresse de l'origine
Compte de bonds
Numéro de séquence de message
Corps de message


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
0
1
1
1
MAL
Taille de message
Limite de bonds
Compte de bonds
Numéro de séquence de message
Corps de message


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type de message
1
1
1
1
MAL
Taille de message
Adresse de l'origine
Limite de bonds
Compte de bonds
Numéro de séquence de message
Corps de message

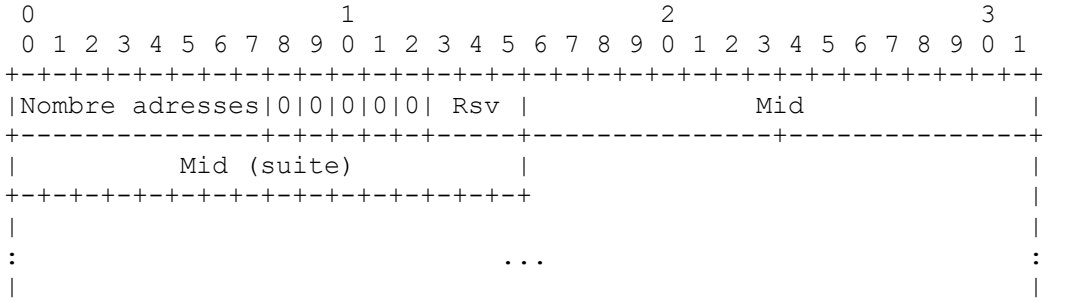
D.3 Corps de message

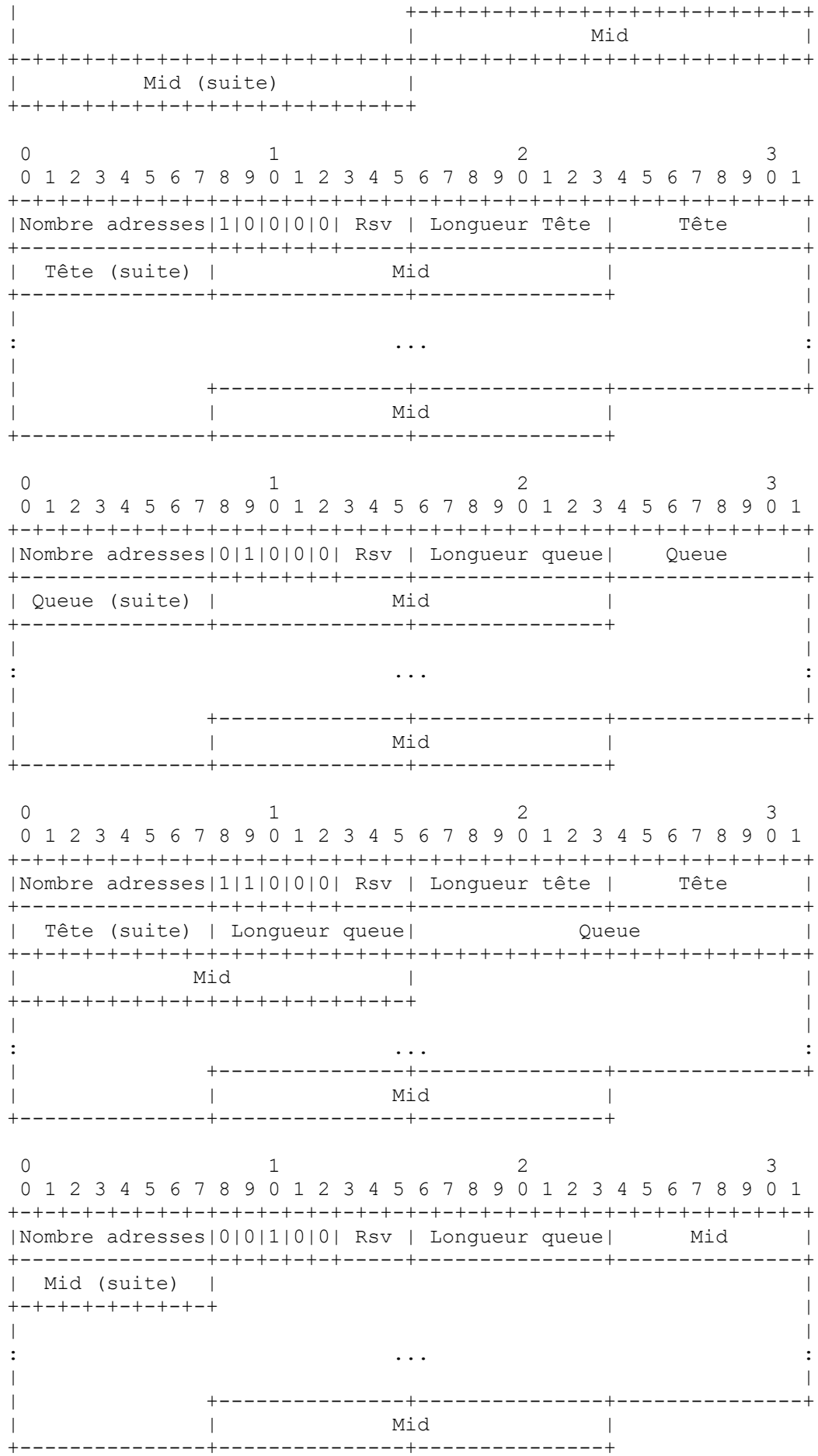
Cette section illustre le format d'un corps de message (l'élément <message> en excluant son élément initial <msg-en-tête>). Le corps de message inclut un bloc de TLV Message (contenant zéro, un ou plusieurs TLV) et peut inclure tout nombre (zéro, un ou plusieurs) de paires de blocs d'adresse et de TLV Bloc d'adresses.

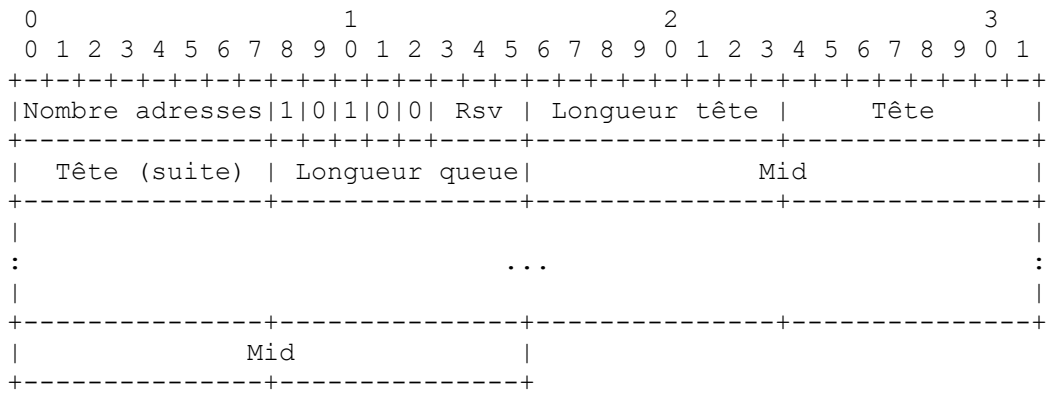


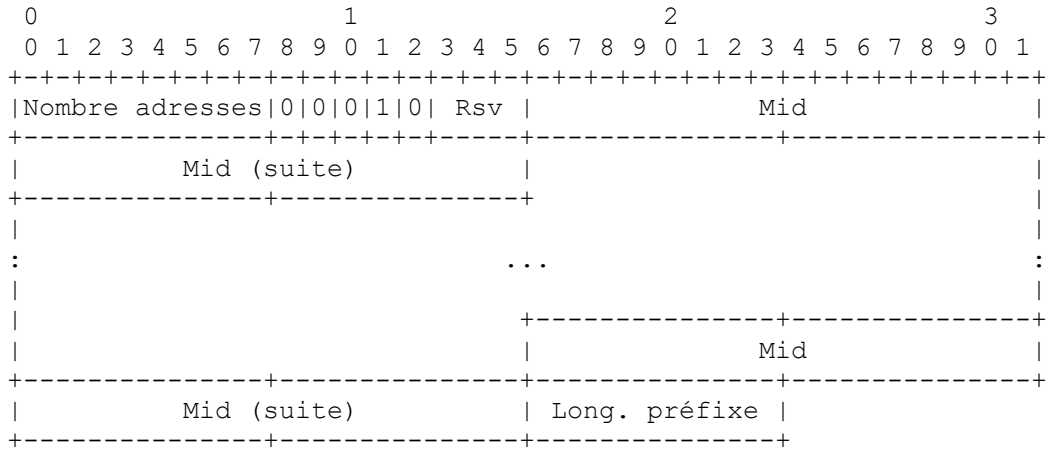
D.4 Bloc d'adresses

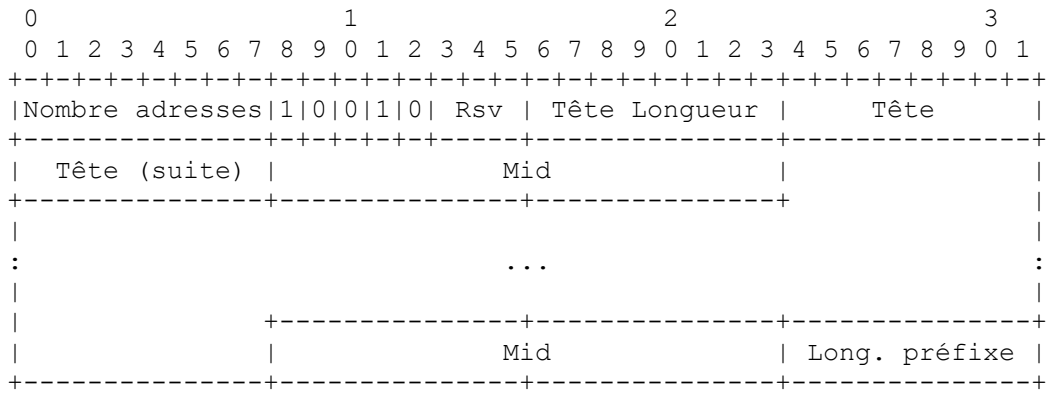
Cette section illustre les options possibles pour l'élément <addr-bloc>. Elles sont différenciées par leur second octet (fanions). Le nombre d'éléments Mid est égal au nombre des adresses (sauf quand mid-longueur est zéro, quand il n'y a pas d'élément Mid). Lorsque un nombre variable de champs de longueur de préfixe est montré, leur nombre est égal au nombre d'adresses.

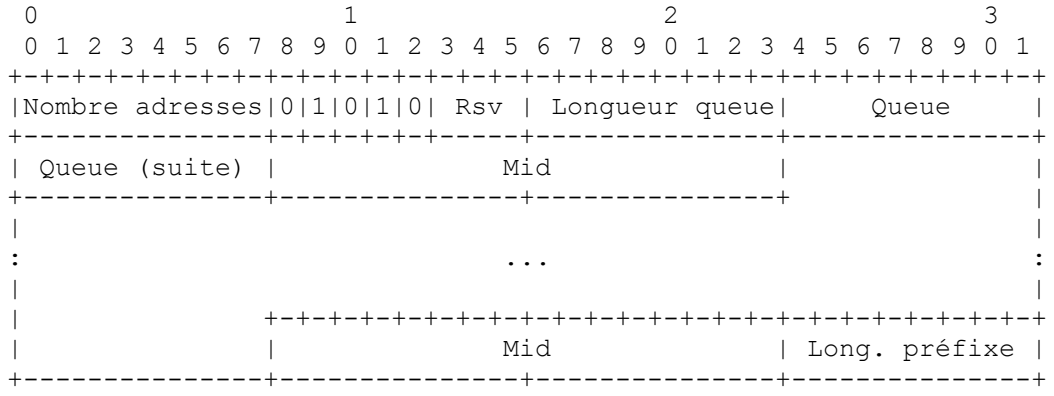


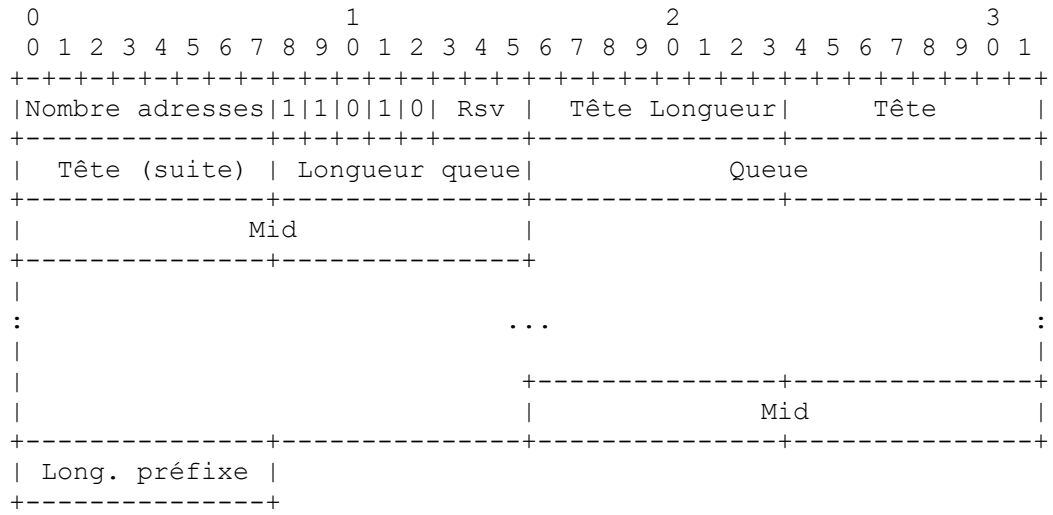

Mid
Mid (suite)
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 0 0 0 0
Rsv
Longueur Tête
Tête
Tête (suite)
Mid
...
Mid
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 1 0 0 0
Rsv
Longueur queue
Queue
Queue (suite)
Mid
...
Mid
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 1 0 0 0
Rsv
Longueur tête
Tête
Tête (suite)
Longueur queue
Queue
Mid
...
Mid
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 0 1 0 0
Rsv
Longueur queue
Mid
Mid (suite)
...
Mid

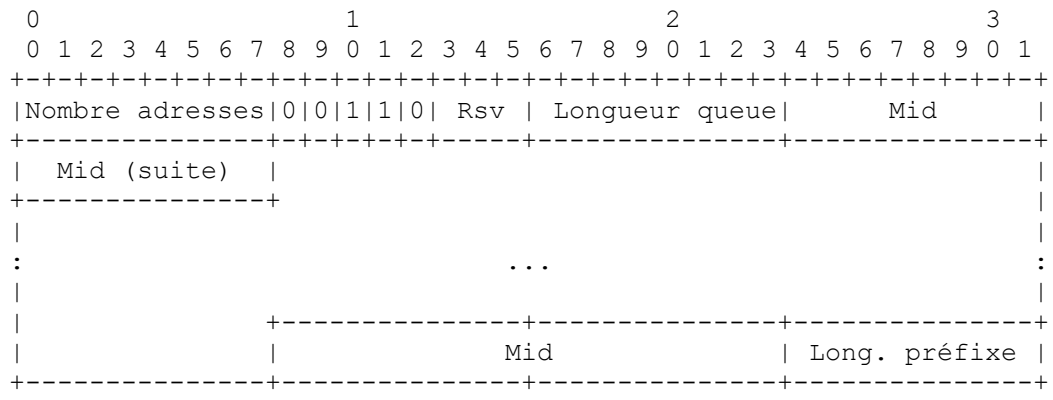

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 0 1 0 0
Rsv
Longueur tête
Tête
Tête (suite)
Longueur queue
Mid
...
Mid

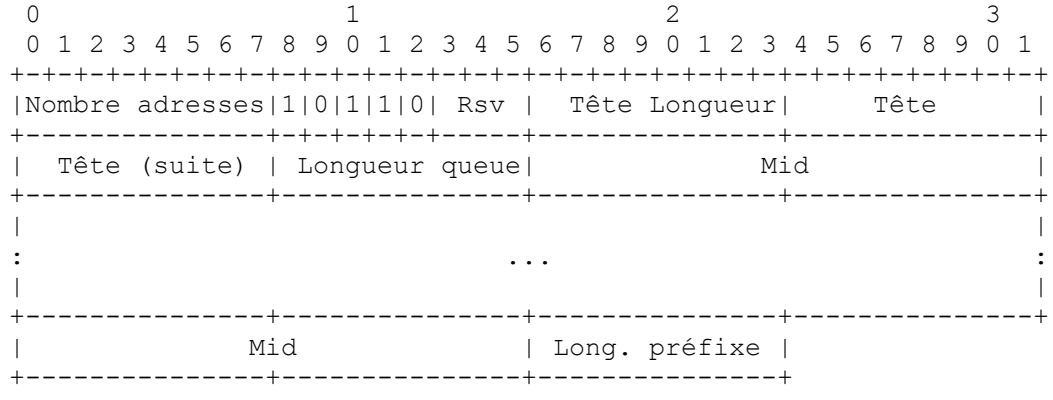

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 0 0 1 0
Rsv
Mid
Mid (suite)
...
Mid
Mid (suite)
Long. préfixe

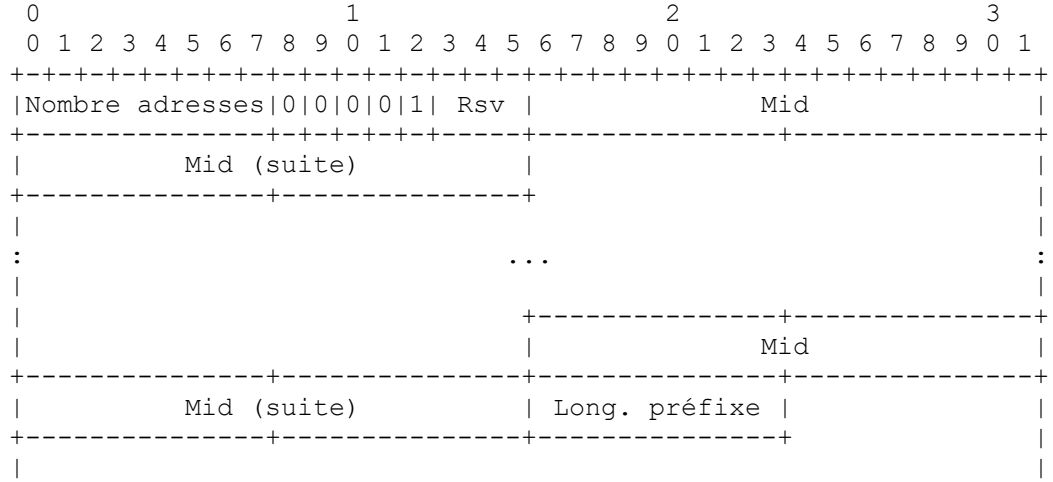

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 0 0 1 0
Rsv
Tête Longueur
Tête
Tête (suite)
Mid
...
Mid
Long. préfixe

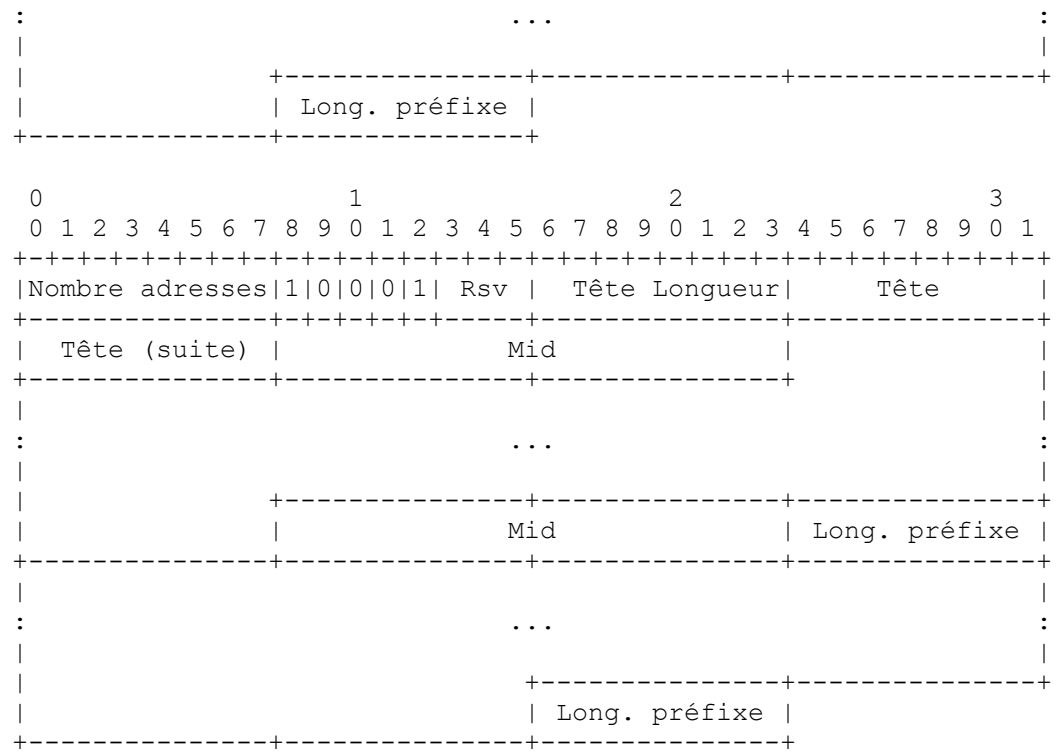

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 1 0 1 0
Rsv
Longueur queue
Queue
Queue (suite)
Mid
...
Mid
Long. préfixe

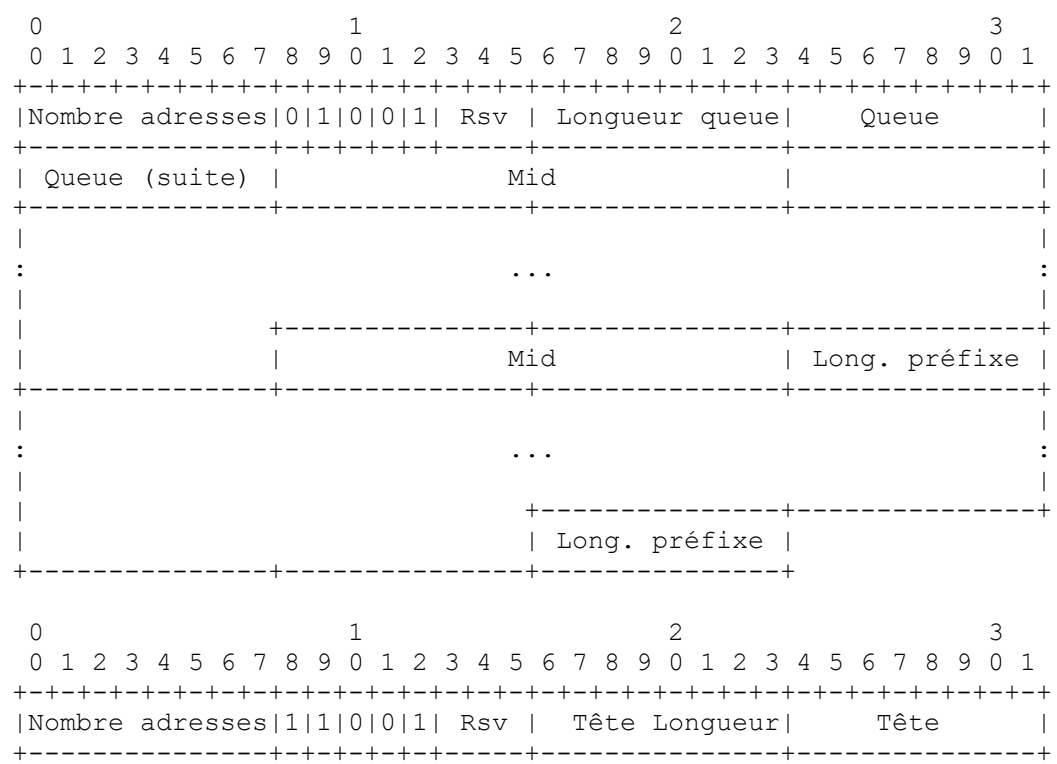

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 1 0 1 0
Rsv
Tête Longueur
Tête
Tête (suite)
Longueur queue
Queue
Mid
...
Mid
Long. préfixe

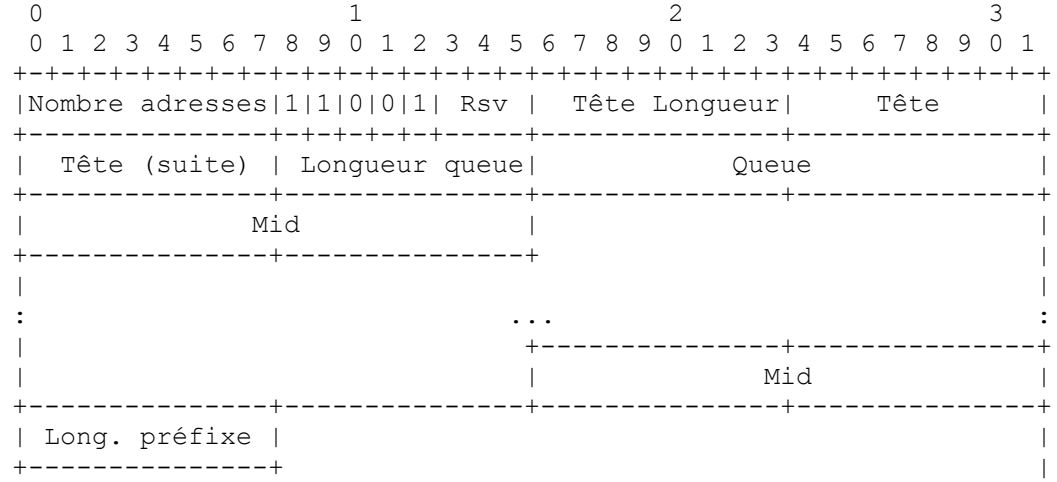

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 0 1 1 0
Rsv
Longueur queue
Mid
Mid (suite)
...
Mid
Long. préfixe

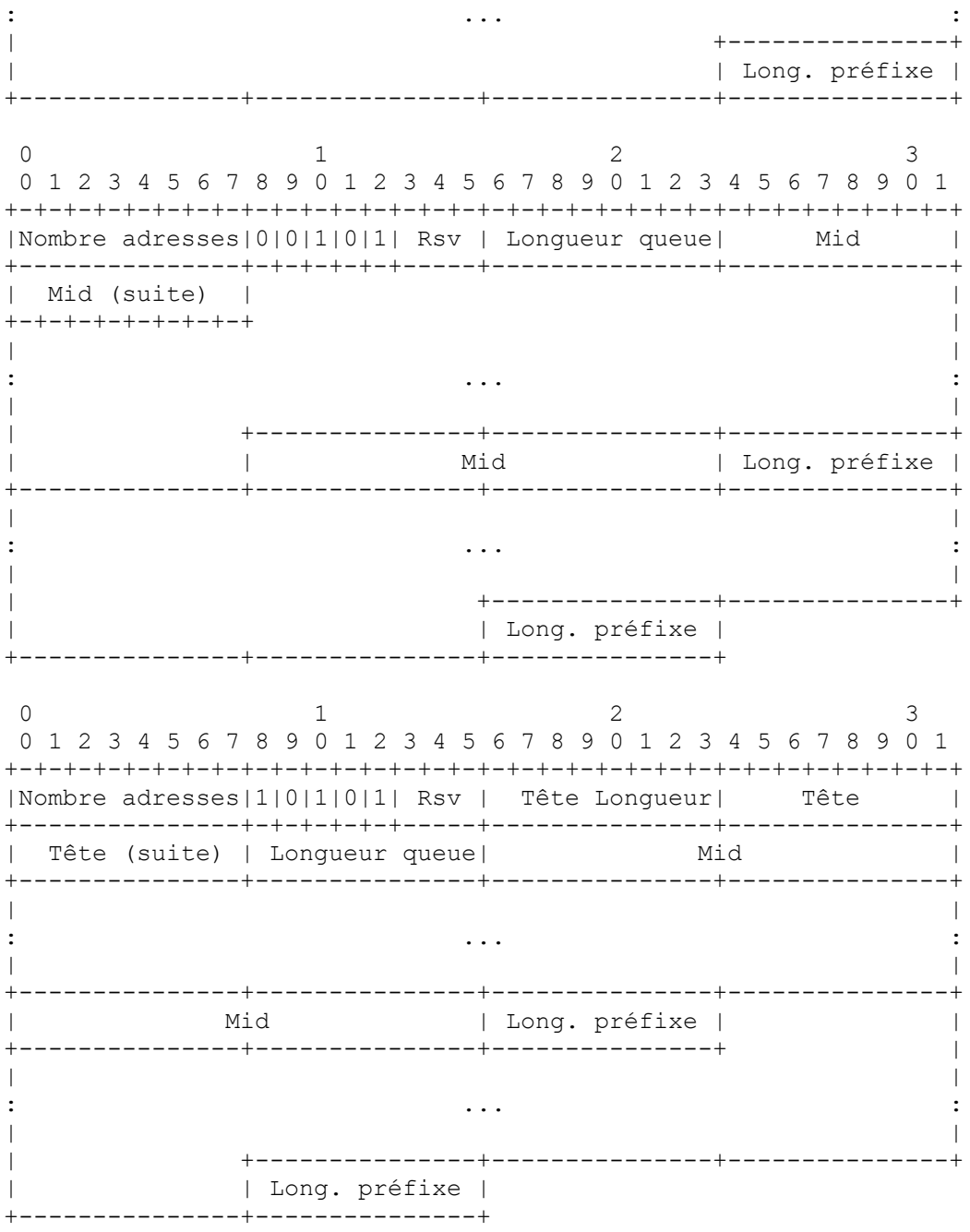

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 0 1 1 0
Rsv
Tête Longueur
Tête
Tête (suite)
Longueur queue
Mid
...
Mid
Long. préfixe


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 0 0 0 1
Rsv
Mid
Mid (suite)
...
Mid
Mid (suite)
Long. préfixe


...
Long. préfixe
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 0 0 0 1 Rsv
Tête Longueur
Tête
Tête (suite)
Mid
...
Mid
Long. préfixe
...
Long. préfixe

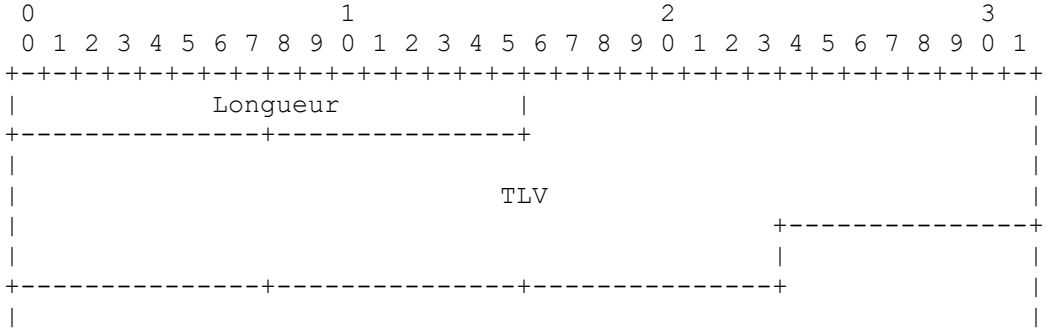

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
0 1 0 0 1 Rsv
Longueur queue
Queue
Queue (suite)
Mid
...
Mid
Long. préfixe
...
Long. préfixe

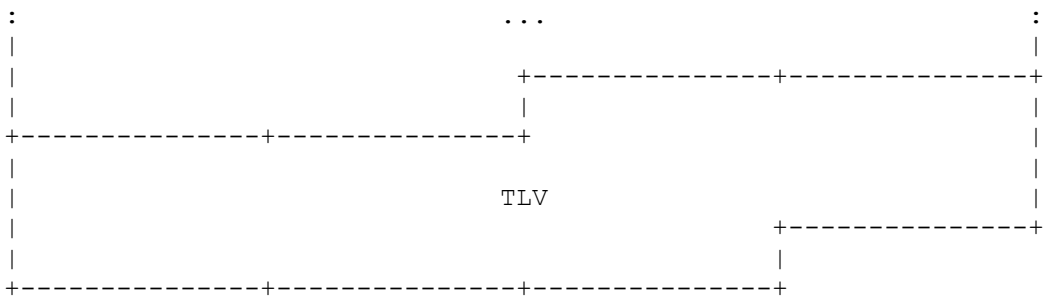

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Nombre adresses
1 1 0 0 1 Rsv
Tête Longueur
Tête
Tête (suite)
Longueur queue
Queue
Mid
...
Mid
Long. préfixe



D.5 Bloc de TLV

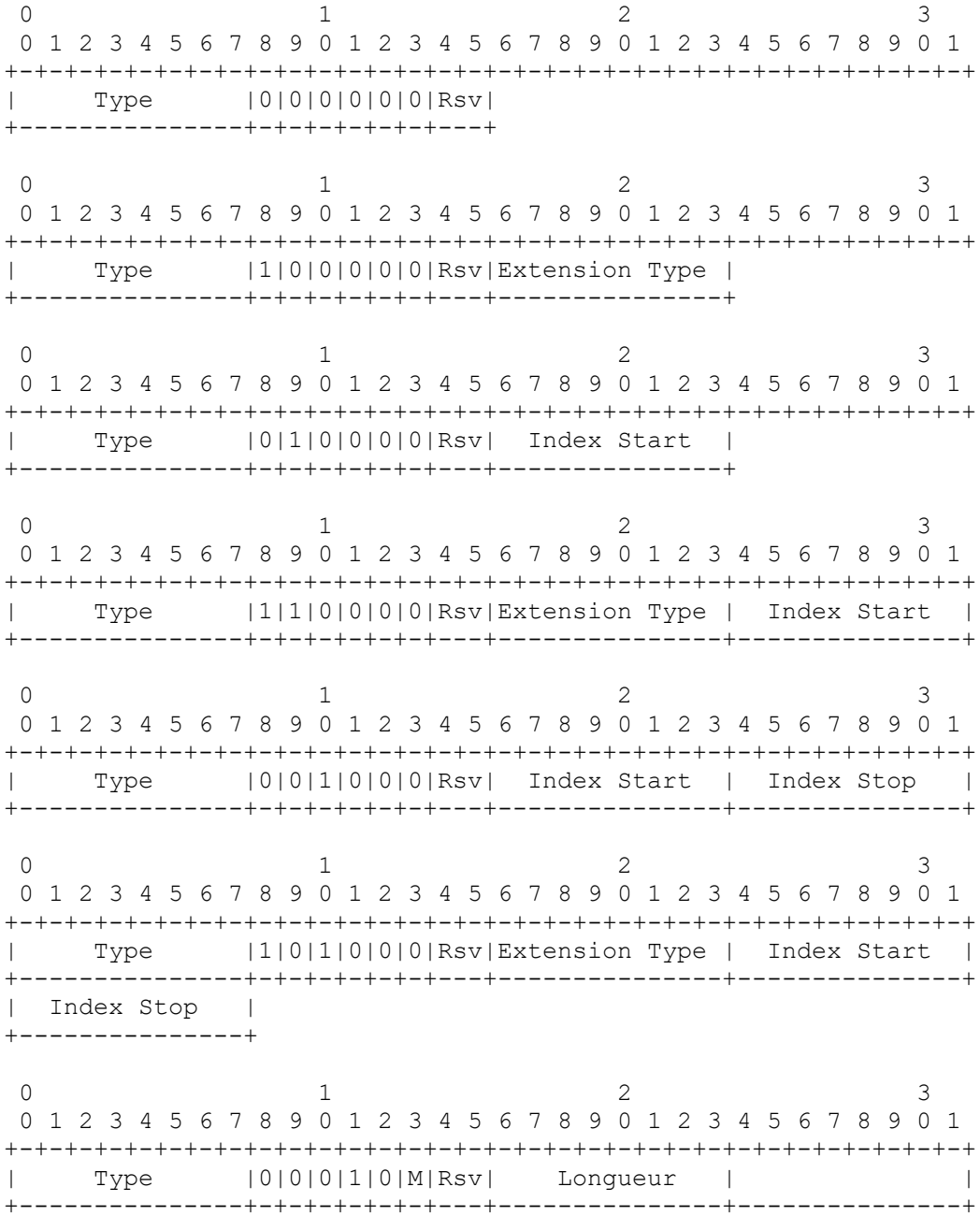
Cette section illustre le format d'un élément <tlv-bloc>. Il peut y avoir un nombre quelconque (zéro, un ou plusieurs) de TLV, avec une longueur totale de TLV (en octets) égale au champ Longueur.

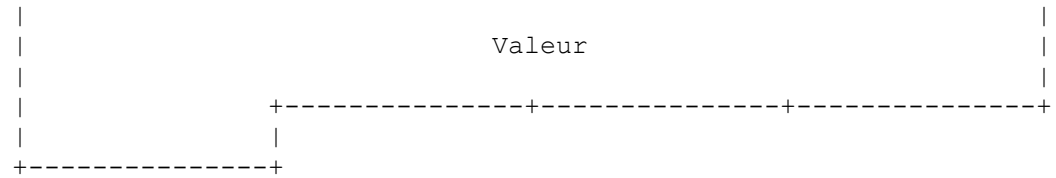


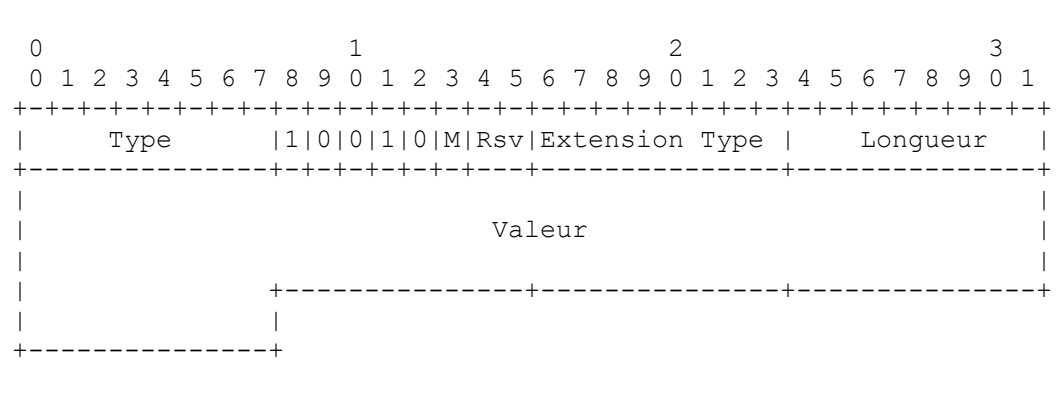


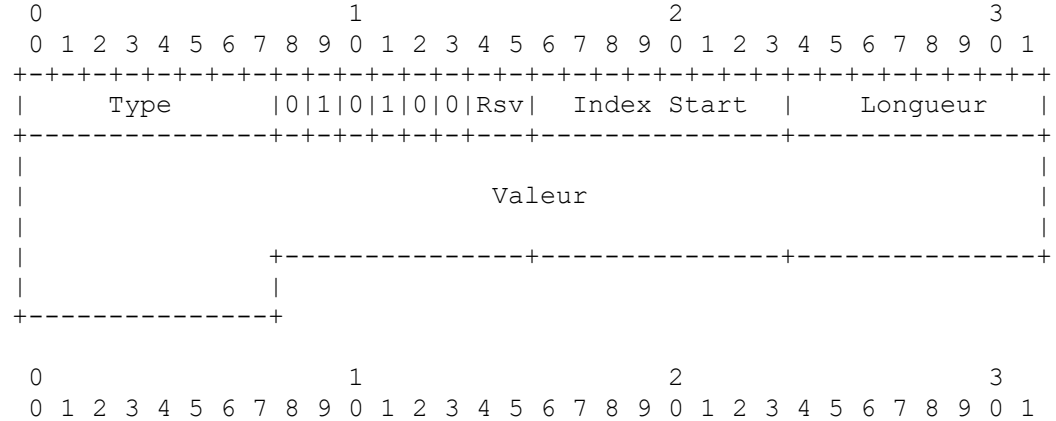
D.6 TLV

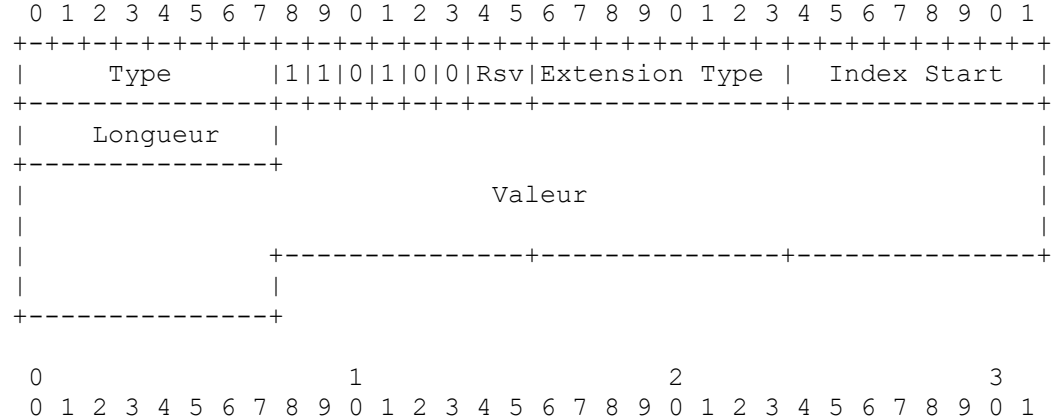
Cette section illustre les options possibles pour l'élément <tlv>. Elles sont différenciées par leur second octet (fanions). Si il n'y a pas de champ Index, alors ce peut être un TLV Paquet, un TLV Message, ou un TLV Bloc d'adresses ; si il y a un ou deux champs Index, alors ce doit être un TLV Bloc d'adresses. Le champ Longueur donne la longueur du champ Valeur (en octets).

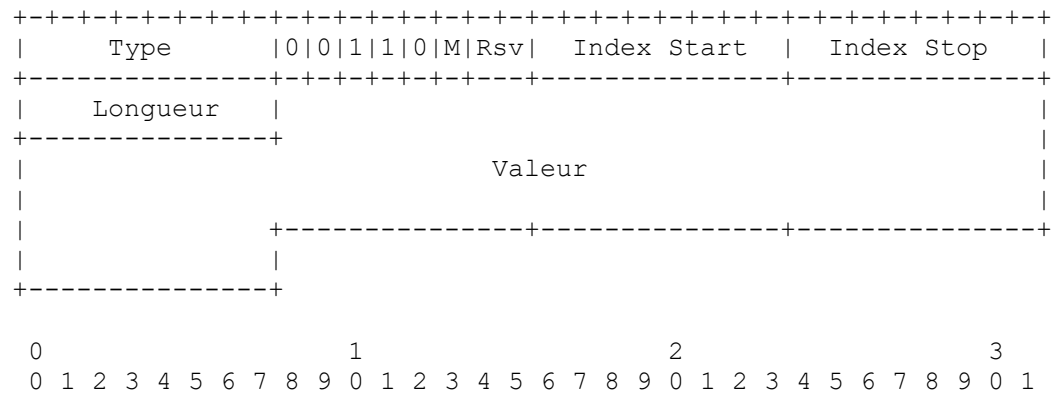


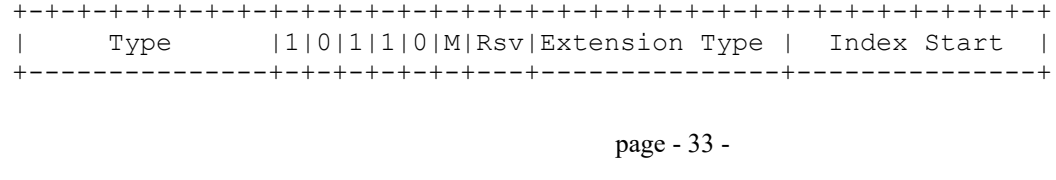

Valeur

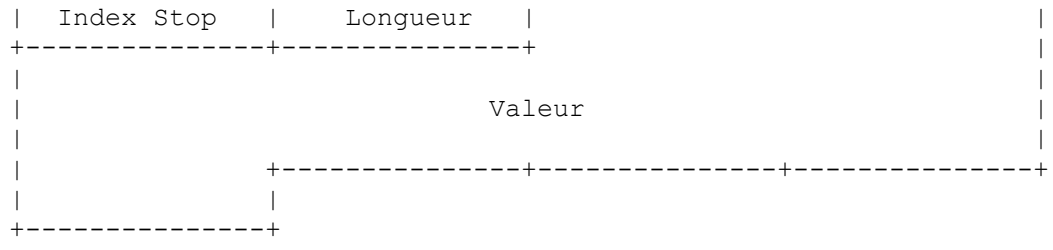

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type
1|0|0|1|0|M|Rsv
Extension Type
Longueur
Valeur

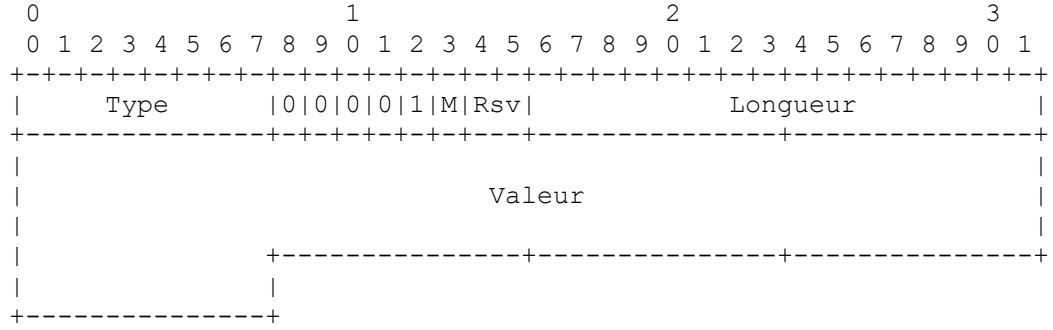

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type
0|1|0|1|0|0|Rsv
Index Start
Longueur
Valeur

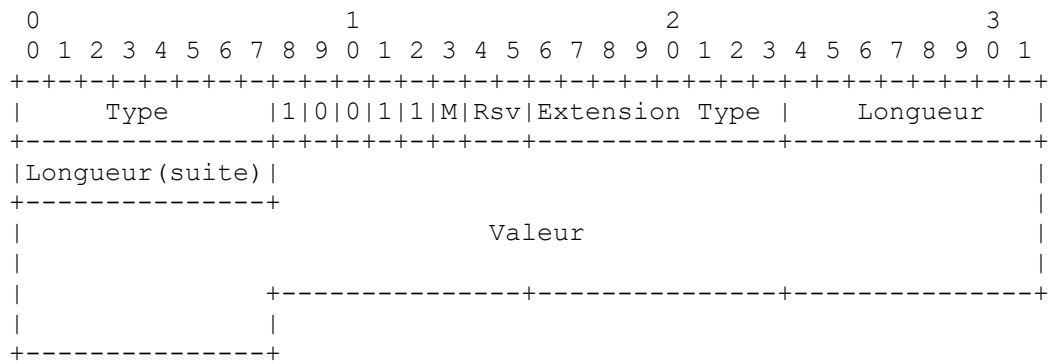

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type
1|1|0|1|0|0|Rsv
Extension Type
Index Start
Longueur
Valeur

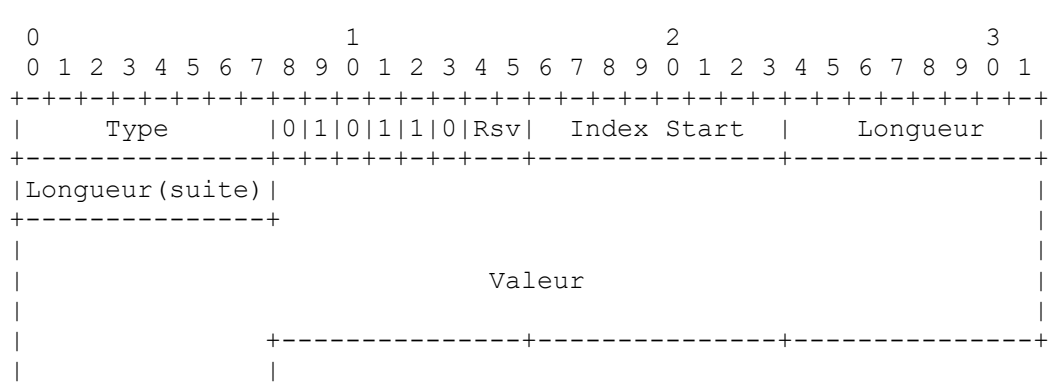

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type
0|0|1|1|0|M|Rsv
Index Start
Index Stop
Longueur
Valeur

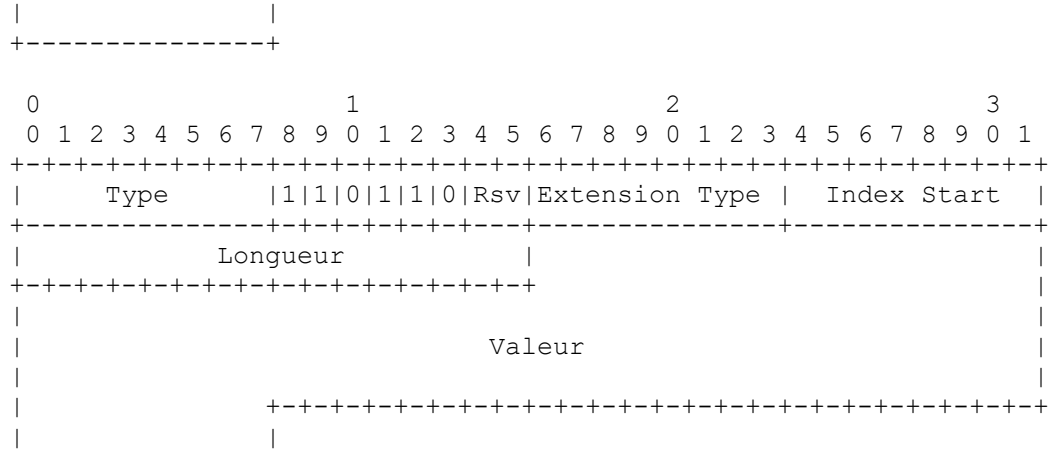

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type
1|0|1|1|0|M|Rsv
Extension Type
Index Start

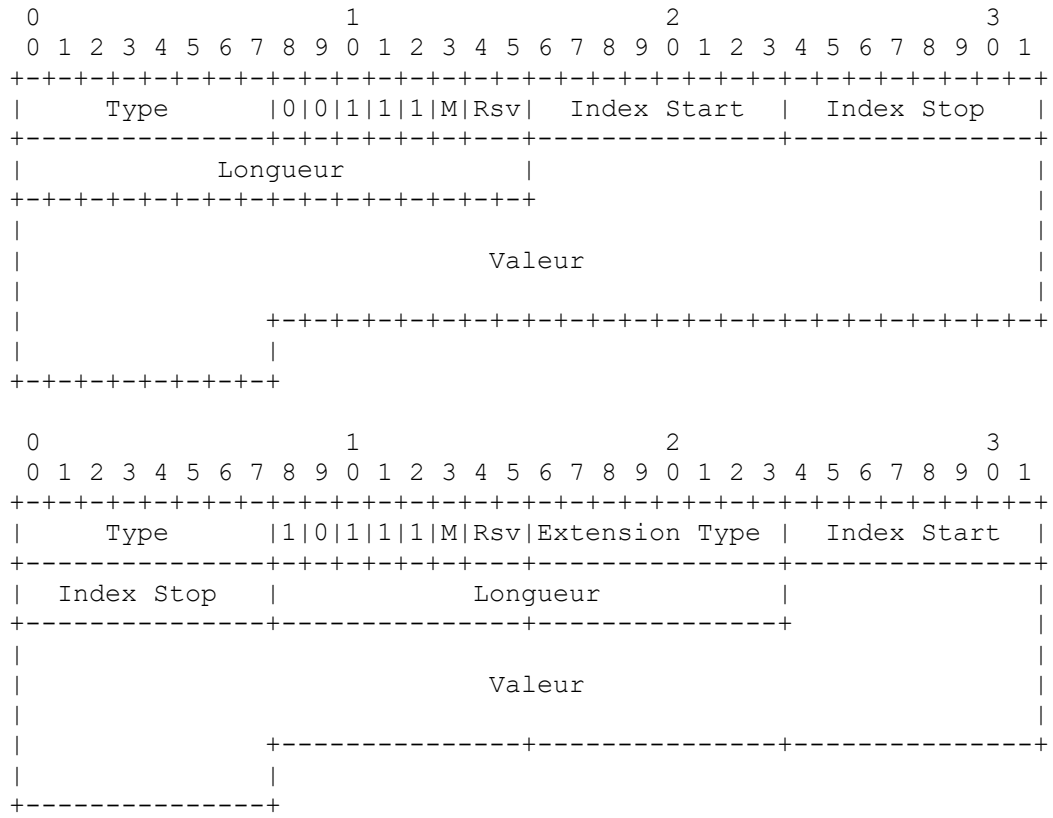

| Index Stop | Longueur |
Valeur


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type |0|0|0|0|1|M|Rsv| Longueur
Valeur


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type |1|0|0|1|1|M|Rsv|Extension Type | Longueur
Longueur(suite)|
Valeur


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type |0|1|0|1|1|0|Rsv| Index Start | Longueur
Longueur(suite)|
Valeur


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Type |1|1|0|1|1|0|Rsv|Extension Type | Index Start
Longueur |
Valeur



Appendice E. Exemple complet

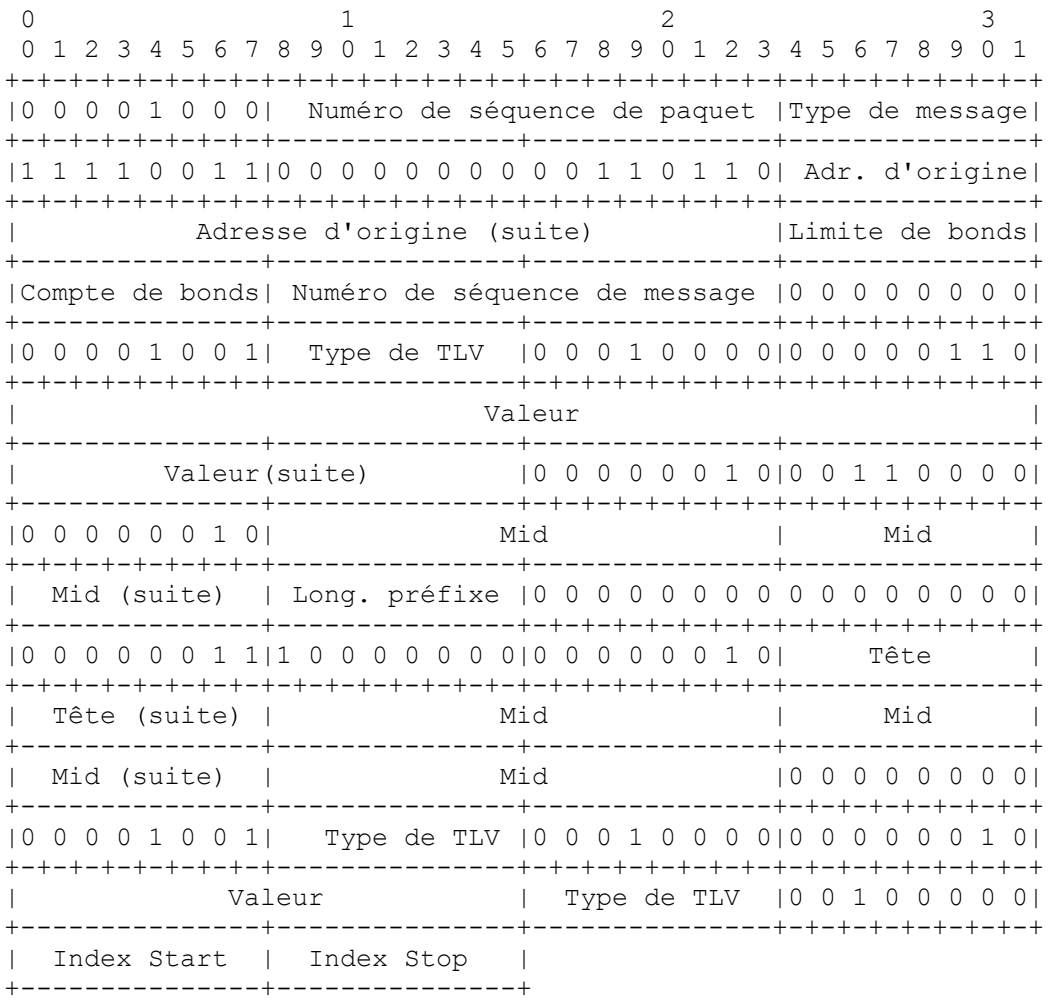
L'exemple de paquet suivant est inclus dans l'intention de montrer comment les en-têtes de paquet et en-têtes de message sont construits, et comment les adresses et attributs sont codés en utilisant les blocs d'adresses et les blocs de TLV. Cet exemple est spécifiquement construit pour ne pas montrer un message maximum ou une réduction de taille de paquet.

L'en-tête de paquet a le fanion phasseqnum de son champ de fanions établi (valeur 8) et donc il a un numéro de séquence de paquet, mais pas de bloc de TLV Paquet.

Le paquet contient un seul message d'une longueur de 54 octets. Ce message a les fanions mhasorig, mhashoplimit, mhashopcount, et mhasseqnum de son champ de fanions de quatre bits établi (valeur 15) et donc il inclut une adresse d'origine, une limite de bonds, un compte de bonds, et un numéro de séquence de message (qui est indépendant du type). Son champ de quatre bits Longueur d'adresse de message a la valeur 3, et donc les adresses dans le message ont une longueur de quatre octets, étant ici des adresses IPv4. Le message a un bloc de TLV Message d'une longueur de contenu de 9 octets, contenant un seul TLV Message. Ce TLV a le fanion thasvalue de son octet de fanions établi (valeur 16) et donc contient un champ Valeur, avec la longueur de valeur précédente de 6 octets. Le message a ensuite deux blocs d'adresses, chacun avec un bloc de TLV Bloc d'adresse suivant.

Le premier bloc d'adresses contient 2 préfixes d'adresse. Il a les fanions ahaszerotail et ahassingleprelen de son octet de fanions établi (valeur 48) et donc n'a pas de Tête (tête-longueur est de zéro octet). Il a une longueur de queue de 2 octets, donc mid-longueur est deux octets. Les deux octets de queue de chaque adresse ne sont pas inclus (car ahaszerotail est établi) et ont une valeur de zéro. Le bloc d'adresse a une seule longueur de préfixe. Le bloc de TLV Bloc d'adresses suivant est vide (longueur de contenu 0 octet).

Le second bloc d'adresses contient 3 adresses. Il a le fanion ahashead de son octet de fanions établi (valeur 128) a une tête-longueur de 2 octets, et pas de queue (longueur de queue est de zéro octet) ; donc, mid-longueur est de deux octets. Il est suivi par un bloc de TLV Bloc d'adresses, d'une longueur de contenu de 9 octets, contenant deux TLV Bloc d'adresses. Le premier de ces TLV a le fanion thasvalue de son octet de fanions établi (valeur 16) et a une seule valeur de longueur de 2 octets, qui s'applique à toutes les adresses du bloc d'adresses. Le second de ces TLV a le fanion thasmultiindex de son octet de fanions établi (valeur 32) et donc n'a pas de champ Longueur de valeur ou Valeur. Il a deux champs Index (Index Start et Index Stop) qui indiquent les adresses auxquelles ce TLV s'applique (gamme inclusive, compté à partir de zéro).



Adresse des auteurs

Thomas Heide Clausen
LIX, Ecole Polytechnique
téléphone : +33 6 6058 9349
mél : T.Clausen@computer.org
URI : http://www.thomasclausen.org/

Christopher M. Dearlove
BAE Systems ATC
téléphone : +44 1245 242194
mél : chris.dearlove@baesystems.com
URI : http://www.baesystems.com/

Justin W. Dean
Naval Research Laboratory
téléphone : +1 202 767 3397
mél : jdean@itd.nrl.navy.mil
URI : http://pf.itd.nrl.navy.mil/

Cedric Adjih
INRIA Rocquencourt
téléphone : +33 1 3963 5215
mél : Cedric.Adjih@inria.fr
URI: http://menetou.inria.fr/~adjih/