

Groupe de travail Réseau
Request for Comments : 3977
 RFC rendue obsolète : 977
 RFCmise à jour : 2980
 Catégorie : En cours de normalisation

C. Feather, THUS plc

octobre 2006

Traduction Claude Brière de L'Isle

Protocole de transfert des nouvelles du réseau (NNTP)

Statut de ce mémoire

Le présent document spécifie un protocole Internet en cours de normalisation pour la communauté de l'Internet, et appelle à des discussions et des suggestions pour son amélioration. Prière de se reporter à l'édition actuelle du STD 1 "Normes des protocoles officiels de l'Internet" pour connaître l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

(Cette traduction incorpore les errata approuvés à la date du 29 mars 2018)

Notice de copyright

Copyright (C) The Internet Society (2006).

Résumé

Le protocole de transfert des nouvelles du réseau (NNTP, *Network News Transfer Protocol*) est utilisé dans l'Internet depuis dix ans, et reste (en volume) un des protocoles les plus populaires utilisés aujourd'hui. Le présent document remplace la RFC 977, et met à jour officiellement la spécification du protocole. Il donne des éclaircissements sur certaines imprécisions de la RFC 977, inclut de nouvelles fonctionnalités de base, et fournit un mécanisme spécifique pour ajouter des extensions normalisées à NNTP.

Table des Matières

1. Introduction.....	2
1.1 Note de l'auteur.....	3
2. Notation.....	3
3. Concepts de base.....	4
3.1 Commandes et réponses.....	4
3.2 Codes de réponse.....	5
3.3 Capacités et extensions.....	8
3.4 Commandes obligatoires et facultatives.....	11
3.5 Traitement en parallèle.....	12
3.6 Articles.....	13
4. Format WILDMAT.....	14
4.1 Syntaxe de wildmat.....	14
4.2 Sémantique de wildmat.....	14
4.3 Extensions.....	15
4.4 Exemples.....	15
5. Commandes d'administration de session.....	15
5.1 Connexion initiale.....	15
5.2 CAPABILITIES.....	16
5.3 MODE READER.....	18
5.4 QUIT.....	19
6. Envoi et restitution d'article.....	20
6.1 Choix de groupe et d'article.....	20
6.2 Restitution d'articles et de paragraphes d'articles.....	26
6.3 Envoi d'article.....	33
7. Commandes d'information.....	36
7.1 DATE.....	36
7.2 HELP.....	37
7.3 NEWGROUPS.....	37
7.4 NEWNEWS.....	38
7.5 Heure.....	39
7.6 Commandes LIST.....	40
8. Commandes d'accès aux champs d'articles.....	44
8.1 Métadonnées d'article.....	44

8.2 Cohérence de la base de données.....	45
8.3 OVER.....	45
8.4 LIST OVERVIEW.FMT.....	48
8.5 HDR.....	50
8.6 LIST HEADERS.....	52
9. Syntaxe en BNF augmenté pour NNTP.....	54
9.1 Introduction.....	54
9.2 Commandes.....	55
9.3 Continuation de commande.....	56
9.4 Réponses.....	56
9.5 Lignes de capacités.....	57
9.6 Variantes de LIST.....	57
9.7 Articles.....	58
9.8 Non terminaux généraux.....	58
9.9 Extensions et validation.....	60
10. Considérations d'internationalisation.....	60
10.1 Introduction et situation historique.....	60
10.2 Cette spécification.....	61
10.3 Problèmes en instance.....	61
11. Considérations relatives à l'IANA.....	62
12. Considérations sur la sécurité.....	62
12.1 Informations personnelles et propriétaires.....	62
12.2 Abus des informations du serveur d'enregistrement.....	62
12.3 Authentification faible et contrôle d'accès.....	62
12.4 Usurpation de DNS.....	63
12.5 Problèmes de UTF-8.....	63
12.6 Mise en antémémoire des listes de capacités.....	63
13. Remerciements.....	64
14. Références.....	65
14.1 Références normatives.....	65
14.2 Références pour information.....	66
Appendice A. Interaction avec d'autres spécifications.....	66
A.1 Saut à la ligne d'en-tête.....	66
A.2 Identifiants de message.....	67
A.3 Envoi d'article.....	68
Appendice B. Sommaire des commande.....	68
Appendice C. Sommaire des codes de réponse.....	69
Appendice D. Changements depuis la RFC 977.....	72
Adresse de l'auteur.....	73
Déclaration complète de droits de reproduction.....	73

1. Introduction

Le présent document spécifie le protocole de transfert des nouvelles du réseau (NNTP, *Network News Transfer Protocol*) qui est utilisé pour la distribution, la recherche, la restitution, et l'envoi des articles de nouvelles du réseau en utilisant un mécanisme fiable fondé sur le flux. Pour les clients lecteurs de nouvelles, NNTP permet la restitution d'articles de nouvelles qui sont mémorisés dans une base de données centrale, donnant aux abonnés la capacité de ne choisir que les articles qu'ils veulent lire.

Le modèle Netnews assure l'indexation, le référencement croisé, et l'expiration des messages périmés. NNTP est conçu pour une transmission efficace des articles de nouvelles du réseau (*Netnews*) sur un canal fiable de communication bidirectionnel.

Bien que la spécification du protocole dans le présent document soit largement compatible avec la version spécifiée dans la [RFC977], un certain nombre de changements sont résumés dans l'Appendice D. En particulier :

- o le jeu de caractères par défaut est changé de l'US-ASCII [ANSI1986] en UTF-8 [RFC3629] (noter que US-ASCII est un sous ensemble de l'UTF-8) ;
- o un certain nombre de commandes qui étaient facultatives dans la RFC 977 ou qui avaient été tirées de la [RFC2980] sont maintenant obligatoires ; et
- o une commande CAPABILITIES a été ajoutée pour permettre aux clients de déterminer quelle fonctionnalité est disponible sur un serveur.

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].

Une mise en œuvre n'est pas conforme si elle échoue à satisfaire à une ou plusieurs des exigences marquées par DOIT pour ce protocole. Une mise en œuvre qui satisfait à toutes les exigences marquées DOIT et toutes celles marquées DEVRAIT pour ces protocoles est dite "inconditionnellement conforme" ; celle qui satisfait à toutes les exigences marquées DOIT mais pas toutes les exigences marquées DEVRAIT pour NNTP est dite être "conditionnellement conforme".

Dans le reste de ce document, les termes "client" et "hôte client" se réfèrent à un hôte qui utilise le service NNTP, tandis que les termes "serveur" et "hôte serveur" se réfèrent à un hôte qui offre le service NNTP.

1.1 Note de l'auteur

Le présent document est écrit en XML en utilisant un DTD spécifique de NNTP. Un logiciel personnalisé est utilisé pour convertir cela en format de la [RFC2629] et ensuite au paquetage public "xml2rfc" pour réduire cela à du texte, nroff source, et de l'HTML. Aucun perl n'a été utilisé pour produire ce document.

2. Notation

Les conventions de notation suivantes sont utilisées dans le présent document.

Des MAJUSCULES indiquent du texte littéral à inclure dans la commande.

Des minuscules indiquent un jeton décrit ailleurs.

Des crochets "[" "]" indiquent que le matériel inclus est facultatif.

Des marques elliptiques "..." indiquent que l'argument peut être répété un nombre quelconque de fois (il doit se produire au moins une fois).

Une barre verticale "|" indique un choix entre deux arguments mutuellement exclusifs (exactement un doit être fourni).

Le nom "message-id" pour un argument de commande ou réponse indique que c'est l'identifiant de message d'un article comme décrit au paragraphe 3.6, incluant les crochets angulaires.

Le nom "wildmat" pour un argument indique que c'est du wildmat comme défini à la Section 4. Si l'argument ne satisfait pas aux exigences de cette section (par exemple, il ne répond pas à la grammaire du paragraphe 4.1) le serveur NNTP PEUT en faire une interprétation (non spécifiée par ce document) ou autrement DOIT le traiter comme une erreur de syntaxe.

Les réponses à chaque commande seront décrites dans des tableaux qui précisent le format requis d'une réponse, suivi par la signification qui devrait être attribuée à cette réponse.

Les termes "NUL", "TAB", "LF", "CR, et "espace" se réfèrent respectivement aux octets %x00, %x09, %x0A, %x0D, et %x20 (c'est-à-dire, les octets qui ont ces codes en US-ASCII [ANSI1986] et donc en UTF-8 [RFC3629]). Le terme "CRLF" ou "paire CRLF" signifie la séquence CR immédiatement suivie par LF (c'est-à-dire, %x0D.0A). Un "caractère US-ASCII imprimable" est un octet dans la gamme %x21-7E. Les caractères entre guillemets (*Quoted characters*) se réfèrent aux octets avec ces codes en US-ASCII (donc "." et "<" se réfèrent à %x2E et %x3C) et seront toujours des caractères US-ASCII imprimables ; de même, "chiffre" se réfère aux octets %x30-39.

Un "mot-clé" DOIT consister seulement en lettres US-ASCII, chiffres, et les caractères point (".") et tiret ("-") et DOIT commencer par une lettre. Les mots-clés DOIVENT faire au moins trois caractères.

Les exemples dans ce document ne sont pas normatifs mais servent à illustrer les usages, les arguments, et les réponses. Dans les exemples, un "[C]" sera utilisé pour représenter l'hôte client et un "[S]" sera utilisé pour représenter l'hôte serveur. La plupart des exemples ne s'appuient pas sur un état de serveur particulier. Dans certains cas, cependant, ils supposent que le groupe de nouvelles actuellement choisi (voir la commande GROUP au paragraphe 6.1.1) est invalide ; lorsque c'est le cas, c'est indiqué au début de l'exemple. Les exemples peuvent utiliser des commandes ou autres mots-clés qui ne sont pas définis dans la présente spécification (comme une commande XENCRYPT). Ceci est utilisé pour illustrer un point et

n'implique pas qu'une telle commande soit définie ailleurs ou doive exister dans une mise en œuvre particulière.

Des termes qui peuvent être lus comme spécifiant des détails d'une mise en œuvre de client ou serveur, comme "base de données", sont utilisés simplement pour faciliter la description. Pourvu que les mises en œuvre se conforment aux spécifications de protocole et de format du présent document, aucune technique spécifique n'est rendue obligatoire.

3. Concepts de base

3.1 Commandes et réponses

NNTP fonctionne sur tout canal fiable de flux de données bidirectionnel de 8 bits. Lorsque la connexion est établie, l'hôte serveur NNTP DOIT envoyer un message d'accueil. L'hôte client et l'hôte serveur échangent alors (respectivement) des commandes et des réponses jusqu'à ce que la connexion soit close ou interrompue. Si la connexion utilisée est TCP, l'hôte serveur commence alors le service NNTP en écoutant sur un accès TCP. Lorsque un hôte client souhaite utiliser le service, il DOIT établir une connexion TCP avec l'hôte serveur en se connectant à cet hôte sur le même accès que celui sur lequel écoute le serveur.

Le jeu de caractères pour toutes les commandes NNTP est UTF-8 [RFC3629]. Les commandes dans NNTP DOIVENT consister en un mot-clé qui PEUT être suivi par un ou plusieurs arguments. Une paire CRLF DOIT terminer toutes les commandes. Plusieurs commandes NE DOIVENT PAS être sur la même ligne. Sauf mention contraire ailleurs dans le présent document, les arguments DEVRAIENT consister en caractères US-ASCII imprimables. Les mots-clés et arguments DOIVENT chacun être séparés par un ou plusieurs caractères espaces ou tabulation. Les lignes de commandes NE DOIVENT PAS excéder 512 octets, ce qui inclut la paire CRLF de terminaison. Les arguments NE DOIVENT PAS excéder 497 octets. Un serveur PEUT assouplir ces limites pour des commandes définies dans une extension.

Lorsque la présente spécification permet des caractères UTF-8 en dehors de la gamme de U+0000 à U+007F, les mises en œuvre NE DOIVENT PAS utiliser la marque d'ordre des octets (U+FEFF, codage %xEF.BB.BF) et DOIVENT utiliser le joncteur de mot (U+2060, codage %xE2.91.A0) pour la signification d'espace insécable de largeur zéro dans les lignes de commandes et les lignes initiales des réponses. Les mises en œuvre DEVRAIENT appliquer partout ces mêmes principes.

Le terme "caractère" signifie un seul codet Unicode. Les mises en œuvre ne sont pas obligées d'exécuter la normalisation Unicode. Donc, U+0084 (A tréma) est un caractère, tandis que U+0041 U+0308 (A composé avec tréma) en fait deux ; les deux ne sont pas nécessairement traités comme équivalents.

Les commandes peuvent avoir des variantes ; si c'est le cas, elles utilisent un second mot-clé immédiatement après le premier pour indiquer quelle variante est requise. Les seules commandes de cette sorte dans la présente spécification sont LIST et MODE. Noter que de telles variantes sont parfois désignées comme si elles étaient des commandes de plein droit : la commande "LIST ACTIVE" devrait être lue comme l'abrégié de "variante ACTIVE de la commande LIST".

Les mots clés sont insensibles à la casse ; la casse des mots clés pour les commandes DOIT être ignorée par le serveur. Les arguments de commande et réponse ne sont spécifiques de la casse ou du langage que lorsque ils sont déclarés, soit dans le présent document, soit dans d'autres spécifications pertinentes.

Dans certains cas, une commande implique plus de données que n'en contient une seule ligne. Le reste des données peut être envoyé soit immédiatement après la ligne de commande (il n'y a aucun exemple de cela dans la présente spécification, mais il y en a dans des extensions comme la [RFC4644]) ou suivant une demande du serveur (indiquée par une réponse 3xx).

Chaque réponse DOIT commencer par un code de réponse à trois chiffres qui est suffisant pour distinguer toutes les réponses. Certaines réponses valides sont définies comme multi lignes ; pour toutes les autres, la réponse est contenue dans une seule ligne. La ligne initiale de la réponse NE DOIT PAS excéder 512 octets, ce qui inclut le code de réponse et la paire CRLF de terminaison ; une extension PEUT spécifier un maximum supérieur pour les commandes qu'elle définit, mais pas pour une autre commande. Les réponses à une seule ligne consistent seulement en une ligne initiale. Les réponses multi lignes consistent en une ligne initiale suivie par un bloc de données multi lignes.

Un serveur NNTP PEUT avoir un temporisateur d'inactivité auto déclencheur. Un tel temporisateur DEVRAIT être au moins de trois minutes, mais il PEUT y avoir une limite plus courte au temps pendant lequel le serveur accepte d'attendre la première commande du client. La réception de toute commande du client durant l'intervalle de temporisation DEVRAIT suffire pour remettre à zéro le temporisateur auto déclencheur. De même, la réception de toute quantité significative de données de la part d'un client qui envoie un bloc de données multi lignes (comme durant une commande POST ou IHAVE) DEVRAIT suffire pour remettre à zéro le temporisateur auto déclencheur. Lorsque le temporisateur arrive à expiration, le serveur DEVRAIT clore la connexion sans envoyer de réponse au client.

3.1.1 Blocs de données multi ligne

Un bloc de données multi lignes est utilisé dans certaines commandes et réponses. Il DOIT respecter les règles suivantes :

1. Le bloc consiste en une séquence de zéro, une ou plusieurs "lignes", chacune étant un flux d'octets se terminant par une paire CRLF. À part ces terminaisons de ligne, le flux NE DOIT PAS inclure les octets NUL, LF, ou CR.
2. Dans une réponse multi lignes, le bloc suit immédiatement le CRLF à la fin de la ligne initiale de la réponse. Lorsque elle est utilisée dans tout autre contexte, la commande spécifique va définir quand le bloc est envoyé.
3. Si une ligne du bloc de données commence par un "octet de terminaison" ("." ou %x2E), cette ligne DOIT être "bourrée de points" par l'ajout sur le devant d'un octet de terminaison supplémentaire à cette ligne du bloc.
4. Les lignes du bloc DOIVENT être suivies par une ligne de terminaison consistant en un seul octet de terminaison suivi par une paire CRLF de façon normale. Donc, sauf si il est vide, un bloc multi lignes est toujours terminé par les cinq octets CRLF "." CRLF (%x0D.0A.2E.0D.0A).
5. Lorsque un bloc multi lignes est interprété, le "point de bourrage" DOIT être retiré ;c'est-à-dire que le receveur DOIT s'assurer que, dans toute ligne qui commence par l'octet de terminaison suivi par des octets autres qu'une paire CRLF, cet octet de terminaison initial n'est pas pris en compte.
6. De même, la ligne de terminaison ("." CRLF ou %x2E.0D.0A) NE DOIT PAS être considérée comme faisant partie du bloc multi lignes ; c'est-à-dire, le receveur DOIT s'assurer que toute ligne qui commence par l'octet de terminaison suivie immédiatement par une paire CRLF n'est pas prise en compte. (La première paire CRLF du CRLF "." CRLF qui termine un bloc non vide fait, bien sûr, partie de la dernière ligne du bloc.)

Noter que les textes qui utilisent un codage (comme UTF-16 ou UTF-32) qui peuvent contenir les octets NUL, LF, ou CR autres qu'une paire CRLF ne peuvent pas être transportés de façon fiable dans le format ci-dessus (c'est-à-dire qu'ils violent l'exigence marquée DOIT ci-dessus). Cependant, sauf mention contraire, la présente spécification n'exige pas que le contenu soit en UTF-8, et donc (sous réserve de la même exigence) il PEUT inclure des octets au-dessus et en dessous de 128 mélangés arbitrairement.

Le présent document ne fixe pas de limite à la longueur d'une ligne dans un bloc multi lignes. Cependant, les normes qui définissent le format des articles peuvent le faire.

3.2 Codes de réponse

Chaque réponse DOIT commencer par un indicateur d'état de trois chiffres. Ce sont les rapports d'état qui proviennent du serveur et indiquent la réponse à la dernière commande reçue du client.

Le premier chiffre de la réponse indique en gros le succès, l'échec, ou les progrès de la commande précédente :

- 1xx – message d'information
- 2xx – commande bien terminée
- 3xx – commande bien jusqu'ici ; envoyer le reste
- 4xx – commande syntaxiquement correcte mais a échoué pour certaines raisons
- 5xx – commande inconnue, non prise en charge, indisponible, ou erreur de syntaxe

Le chiffre suivant du code indique la catégorie de fonction de la réponse :

- x0x – message de connexion, d'établissement, et divers
- x1x – choix de groupe de nouvelles
- x2x – choix d'article
- x3x – fonctions de distribution
- x4x - envoi
- x8x – réservé pour les extensions d'authentification et de confidentialité
- x9x – réservé pour utilisation privée (extensions non standard)

Certaines réponses contiennent des arguments comme des numéros et noms en plus de l'indicateur d'état. Dans ce cas, pour simplifier l'interprétation par le client, le numéro et le type de tels arguments sont fixés pour chaque code de réponse, ainsi que si le code est sur une seule ligne ou multi lignes. Toute extension DOIT aussi suivre ce principe. Noter que, pour des raisons historiques, le code de réponse 211 est une exception en ce que la réponse peut être sur une seule ligne ou multi lignes selon la commande (GROUP ou LISTGROUP) qui l'a générée. Dans tous les autres cas, le client DOIT seulement

utiliser l'indicateur d'état lui-même pour déterminer la nature de la réponse. Les codes de réponse exacts qui peuvent être retournés par une certaine commande sont précisés dans la description de cette commande.

Les arguments DOIVENT être séparés de l'indicateur d'état numérique et chacun les uns des autres par une seule espace. Tous les arguments numériques DOIVENT être en format base 10 (décimal) et PEUVENT avoir des zéros en tête. Les arguments de chaîne DOIVENT contenir au moins un caractère et NE DOIVENT PAS contenir de TAB, LF, CR, ou espace. Le serveur PEUT ajouter tout texte après le code de réponse ou le dernier argument, comme approprié, et le client NE DOIT PAS prendre de décisions sur la base de ce texte. Un tel texte DOIT être séparé de l'indicateur d'état numérique ou du dernier argument par au moins une espace.

Le serveur DOIT répondre à toute commande par la réponse générique appropriée (donnée au paragraphe 3.2.1) si il se représente la situation. Autrement, chaque commande reconnue DOIT retourner un des codes de réponse spécifiquement énumérés dans sa description ou dans une extension. Un serveur PEUT fournir des extensions à la présente spécification, incluant de nouvelles commandes, de nouvelles variantes ou caractéristiques de commandes existantes, et autres façons de changer l'état interne du serveur. Cependant, le serveur NE DOIT PAS produire d'autres réponses à un client qui n'invoque aucune des caractéristiques supplémentaires. (Donc, un client qui se restreint à la présente spécification va seulement recevoir les réponses qui sont mentionnées.)

Si un client reçoit une réponse inattendue, il DEVRAIT utiliser le premier chiffre de la réponse pour déterminer le résultat. Par exemple, un 2xx inattendu devrait être interprété comme un succès, et un 4xx ou 5xx inattendu comme un échec.

Les codes de réponse non spécifiés dans le présent document PEUVENT être utilisés pour toute commande supplémentaire spécifique de l'installation non spécifiée. Ils DEVRAIENT être choisis pour tenir dans le schéma de x9x spécifié ci-dessus.

Ni le présent document, ni aucune extension enregistrée (voir le paragraphe 3.3.3) ne spécifiera de codes de réponse du schéma x9x. (Les mises en œuvre d'extensions sont invitées en conséquence à ne pas utiliser de telles réponses pour des extensions qui peuvent ultérieurement être proposées à l'enregistrement.)

3.2.1 Codes de réponse génériques

Le serveur DOIT répondre à toute commande par une des réponses génériques appropriées suivantes si il se représente la situation.

Si la commande n'est pas reconnue, ou si c'est une commande facultative qui n'est pas mise en œuvre par le serveur, le code de réponse 500 DOIT être retourné.

Si il y a une erreur de syntaxe dans les arguments d'une commande reconnue, incluant le cas où plus d'arguments sont fournis que ce que la commande spécifie, ou si la ligne de commande est plus longue que ce que le serveur accepte, le code de réponse 501 DOIT être retourné. La ligne NE DOIT PAS être tronquée ou partagée et ensuite interprétée. Noter que lorsque une commande a des variantes qui dépendent d'un second mot-clé (par exemple, LIST ACTIVE et LIST NEWSGROUPS), 501 DOIT être utilisé lorsque la commande de base est mise en œuvre mais que la variante demandée ne l'est pas, et 500 DOIT être utilisé seulement lorsque la commande de base elle-même n'est pas mise en œuvre.

Si un argument est exigé comme chaîne codée en base64 [RFC4648] (il n'y a pas de tels arguments dans la présente spécification, mais il peut y en avoir dans les extensions) et n'est pas d'un codage valide, le code de réponse 504 DOIT être retourné.

Si le serveur rencontre une faute ou problème interne et s'il est incapable d'exécuter la commande (par exemple, un fichier nécessaire manque ou un service nécessaire n'a pas pu être contacté) le code de réponse 403 DOIT être retourné. Si le serveur reconnaît la commande mais ne fournit pas une caractéristique facultative (par exemple, parce qu'il n'a pas mémorisé les informations requises) ou si il traite seulement un sous ensemble des cas légitimes (voir la commande HDR, au paragraphe 8.5, pour un exemple) le code de réponse 503 DOIT être retourné.

Si le client n'est pas autorisé à utiliser la facilité spécifiée lorsque le serveur est dans son état actuel, le code de réponse approprié parmi les suivants DOIT être utilisé.

502 : il est nécessaire de terminer la connexion et d'en commencer une nouvelle avec l'autorité appropriée avant que la commande puisse être utilisée. Historiquement, des serveurs en mode commutation (voir le paragraphe 3.4.1) utilisaient cette réponse pour indiquer que cette commande va devenir disponible après qu'est utilisée la commande MODE READER (paragraphe 5.3) mais cet usage ne se conforme pas à la présente spécification et NE DOIT PAS être utilisé. Noter que le serveur NE DOIT PAS clore la connexion immédiatement après une réponse 502 sauf à la connexion initiale (paragraphe 5.1) et avec la commande MODE READER.

480 : le client doit s'authentifier auprès du serveur (c'est-à-dire, il doit fournir des informations sur l'identité du client) avant que la facilité puisse être utilisée sur cette connexion. Cela va impliquer l'utilisation d'une extension d'authentification telle que [RFC4643].

483 : le client doit négocier la protection appropriée de la confidentialité sur la connexion. Cela va impliquer l'utilisation d'une extension de confidentialité telle que celle de la [RFC4642].

401 : le client doit changer l'état de la connexion de quelque autre manière. Le premier argument de la réponse DOIT être l'étiquette de capacité (voir le paragraphe 5.2) de la facilité qui fournit le mécanisme nécessaire (usuellement une extension, qui peut être une extension privée). Le serveur NE DOIT PAS utiliser ce code de réponse sauf comme spécifié par la définition de la capacité en question.

Si le serveur doit terminer la connexion pour une raison quelconque, il DOIT donner le code de réponse 400 à la prochaine commande et ensuite clore immédiatement la connexion. À la suite d'une réponse 400, les clients NE DEVRAIENT PAS simplement se reconnecter immédiatement et réessayer les mêmes actions. Un client DEVRAIT plutôt utiliser un retard exponentiellement croissant entre les essais (par exemple, doubler le temps d'attente après chaque réponse 400) ou présenter un texte associé à l'utilisateur pour qu'il décide si et quand réessayer.

Le client DOIT être prêt à recevoir toutes ces réponses de toute commande (excepté, bien sûr, que le serveur NE DOIT PAS générer un code de réponse 500 pour les commandes obligatoires).

3.2.1.1 Exemples

Exemple d'une commande inconnue :

```
[C] MAIL
[S] 500 Commande inconnue
```

Exemple d'une commande non prise en charge :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] NEWNEWS
[S] LIST ACTIVE NEWSGROUPS
[S] .
[C] OVER
[S] 500 Commande inconnue
```

Exemple d'une variante non prise en charge :

```
[C] MODE POSTER
[S] 501 Option MODE inconnue
```

Exemple d'une erreur de syntaxe :

```
[C] ARTICLE a.message.id@no.angle.brackets
[S] 501 Erreur de syntaxe
```

Exemple d'une ligne de commande trop longue :

```
[C] HEAD 53 54 55
[S] 501 Trop d'arguments
```

Exemple d'un mauvais wildmat :

```
[C] LIST ACTIVE u[ks].*
[S] 501 Erreur de syntaxe
```

Exemple d'une erreur de codage base64 (le second argument est censé être codé en base64) :

[C] XENCRYPT RSA abcd=efg
 [S] 504 Erreur de codage Base64

Exemple de tentative d'accès à une facilité non disponible pour cette connexion :

[C] MODE READER
 [S] 200 mode Reader, envoi permis
 [C] IHAVE <je.suis.un.article.que.tu.voudras@exemple.com>
 [S] 502 Permission refusée

Exemple de tentative d'accès à une facilité qui exige l'authentification :

[C] GROUP secret.group
 [S] 480 Permission refusée

Exemple d'une tentative réussie de suivre une telle authentification :

[C] XSECRET fred flintstone
 [S] 290 Mot de passe pour fred accepté
 [C] GROUP secret.group
 [S] 211 5 1 20 secret.group retenu

Exemple d'une tentative d'accès à une facilité qui exige la confidentialité :

[C] GROUP secret.group
 [S] 483 Connexion sécurisée requise
 [C] XENCRYPT
 [Client et serveur négocient le chiffrement sur la liaison]
 [S] 283 Liaison chiffrée établie
 [C] GROUP secret.group
 [S] 211 5 1 20 secret.group retenu

Exemple d'un besoin de changer de mode avant qu'une facilité soit utilisée :

[C] GROUP binary.group
 [S] 401 XHOST N'est pas sur cet hôte virtuel
 [C] XHOST binary.news.Exemple.org
 [S] 290 Hôte virtuel binary.news.Exemple.org retenu
 [C] GROUP binary.group
 [S] 211 5 1 77 binary.group retenu

Exemple d'échec temporaire :

[C] GROUP archive.local
 [S] 403 Serveur d'archive temporairement hors ligne

Exemple de serveur qui a besoin de clore immédiatement :

[C] ARTICLE 123
 [S] 400 Défaillance de l'alimentation électrique,
 [Le serveur clôt la connexion.]

3.3 Capacités et extensions

Tous les serveurs NNTP ne fournissent pas exactement les mêmes facilités, à la fois parce que la présente spécification permet des variations et parce que les serveurs peuvent fournir des extensions. Un ensemble de facilités qui sont en rapport est appelé une "capacité". La présente spécification donne un moyen pour déterminer quelles capacités sont disponibles, incluant une liste de capacités standard, et inclut un mécanisme (les extensions) pour définir de nouvelles capacités.

3.3.1 Descriptions de capacités

Un client peut déterminer les capacités disponibles du serveur en utilisant la commande CAPABILITIES (paragraphe 5.2). Elle retourne une liste de capacités, qui est une liste de lignes de capacités. Chaque ligne décrit une capacité disponible.

Chaque ligne de capacité consiste en un ou plusieurs jetons, qui DOIVENT être séparés par un ou plusieurs caractères espace ou tabulation. Un jeton est une chaîne de un ou plusieurs caractères UTF-8 imprimables (c'est-à-dire, soit des caractères US-ASCII imprimables, soit toute séquence UTF-8 en dehors de la gamme US-ASCII, mais pas d'espace ni de TAB). Sauf mention contraire, les jetons sont insensibles à la casse. Chaque ligne de capacité consiste en ce qui suit :

- o L'étiquette de capacité, qui est un mot-clé indiquant la capacité. Une étiquette de capacité peut être définie par la présente spécification ou une qui lui succède, ou par une extension.
- o L'étiquette est alors suivie par zéro, un ou plusieurs jetons, qui sont les arguments de la capacité. La forme et la signification de ces jetons sont spécifiques de chaque capacité.

Le serveur DOIT s'assurer que la liste de capacités reflète précisément les capacités (incluant les extensions) actuellement disponibles. Si une capacité n'est disponible qu'avec le serveur dans un certain état (par exemple, seulement après authentification) la liste DOIT seulement inclure l'étiquette de capacité lorsque le serveur est dans cet état. De même, si seulement certaines des commandes dans une extension sont disponibles, ou si le comportement de l'extension change d'une certaine manière, selon l'état du serveur, ceci DOIT être indiqué par des arguments différents dans la ligne de capacité.

Noter qu'une ligne de capacité ne peut commencer que par une lettre. Les lignes qui commencent par d'autres caractères sont réservées pour de futures versions de la présente spécification. Afin d'interopérer avec de telles versions, les clients DOIVENT être prêts à recevoir des lignes commençant par d'autres caractères et DOIVENT ignorer toutes celles qu'elles ne comprennent pas.

3.3.2 Capacité standard

Les capacités suivantes sont définies par la présente spécification.

VERSION : cette capacité DOIT être annoncée par tous les serveurs et DOIT être la première capacité de la liste de capacités ; elle indique la ou les versions de NNTP que le serveur prend en charge. Elle doit avoir au moins un argument; chaque argument est un nombre décimal et NE DOIT PAS avoir de zéro en tête. Les numéros de version sont alloués seulement dans les RFC qui mettent à jour ou remplacent la présente spécification ; les serveurs NE DOIVENT PAS créer leurs propres numéros de version. Le numéro de version de la présente spécification est 2.

READER : cette capacité indique que le serveur met en œuvre les diverses commandes utiles pour les clients lecteurs.

IHAVE : cette capacité indique que le serveur met en œuvre la commande IHAVE.

POST : cette capacité indique que le serveur met en œuvre la commande POST.

NEWNEWS : cette capacité indique que le serveur met en œuvre la commande NEWNEWS.

HDR : cette capacité indique que le serveur met en œuvre les commandes d'accès d'en-tête (HDR et LIST HEADERS).

OVER : cette capacité indique que le serveur met en œuvre les commandes d'accès de vue d'ensemble (OVER et LIST OVERVIEW.FMT). Si et seulement si le serveur prend en charge la forme identifiant de message de la commande OVER, il doit y avoir un seul argument MSGID.

LIST : cette capacité indique que le serveur met en œuvre au moins une variante de la commande LIST. Il DOIT y avoir un argument pour chaque variante de la commande LIST prise en charge par le serveur, donnant le mot-clé pour cette variante.

IMPLEMENTATION : cette capacité PEUT être fournie par un serveur. Si il le fait, les arguments DEVRAIENT être utilisés pour fournir des informations comme le nom du logiciel du serveur et le numéro de version. Le client NE DOIT PAS utiliser cette ligne pour déterminer les capacités du serveur. (Bien que les serveurs fournissent souvent ces informations dans l'accueil initial, les clients doivent deviner si c'est le cas ; cette capacité rend clair ce que sont les informations.)

MODE-READER : cette capacité indique que le serveur est à commutation de mode (paragraphe 3.4.2) et que la commande MODE READER doit être utilisée pour activer la capacité READER.

3.3.3 Extensions

Bien que NNTP soit largement et solidement déployé, certaines parties de la communauté de l'Internet pourraient souhaiter étendre le service NNTP. On doit souligner qu'aucune extension à NNTP ne devrait être considérée à la légère. La force de NNTP vient principalement de sa simplicité. L'expérience de nombreux protocoles a montré que les protocoles qui ont peu d'options tendent à l'ubiquité, tandis que les protocoles qui ont beaucoup d'options tendent à l'obscurité. Cela signifie que chaque extension, sans considération de ses avantages, doit être examinée avec soin à l'égard de ses coûts de mise en œuvre de déploiement, et d'interopérabilité. Dans de nombreux cas, le coût de l'extension du service NNTP va probablement dépasser le bénéfice.

Une extension est un paquetage de facilités associées, souvent, mais pas toujours incluant une ou plusieurs nouvelles commandes. Chaque extension DOIT définir au moins une nouvelle étiquette de capacité (cela sera souvent, mais ce n'est pas obligatoire, le nom d'une de ces nouvelles commandes). Bien que toute information de capacité supplémentaire puisse normalement être spécifiée en utilisant des arguments avec cette étiquette, une extension PEUT définir plus d'une étiquette de capacité. Cependant, ceci DEVRAIT être limité à des circonstances exceptionnelles.

Une extension est soit une extension privée, soit ses capacités sont incluses dans le registre IANA des capacités (voir au paragraphe 3.3.4) et est définie dans une RFC (auquel cas c'est une "extension enregistrée"). De telles RFC doivent soit être sur la voie de la normalisation, soit définir un protocole expérimental approuvé par l'IESG.

La définition d'une extension doit inclure ce qui suit :

- o un nom descriptif pour l'extension,
- o la ou les étiquettes de capacité définies par l'extension (l'étiquette de capacité d'une extension enregistrée NE DOIT PAS commencer par "X"),
- o la syntaxe, les valeurs, et la signification de tous les arguments pour chaque étiquette de capacité définie par l'extension,
- o toutes les nouvelles commandes NNTP associées à l'extension (les noms des commandes associées aux extensions enregistrées NE DOIVENT PAS commencer par "X"),
- o la syntaxe et les valeurs possibles des arguments associés aux nouvelles commandes NNTP,
- o les codes de réponse et les valeurs possibles des arguments pour les réponses aux nouvelles commandes NNTP,
- o tous nouveaux arguments que l'extension associe à toute autre commande NNTP préexistante,
- o toute augmentation de la longueur maximum des commandes et des lignes initiales de réponse au delà de la valeur spécifiée dans le présent document,
- o une déclaration spécifique sur l'effet de traitement en parallèle que cette extension peut avoir (si elle en a),
- o une déclaration spécifique sur les circonstances dans lesquelles utiliser cette extension peut altérer le contenu de la liste des capacités (autres que les nouvelles étiquettes de capacité qu'elle définit),
- o une déclaration spécifique sur les circonstances dans lesquelles l'extension peut causer la production par toute commande préexistante de réponses 401, 480, ou 483,
- o une description de la façon dont l'utilisation du MODE READER sur un serveur à commutation de mode interagit avec l'extension,
- o une description de la façon dont la prise en charge de l'extension affecte le comportement d'un serveur et client NNTP de toute manière non mentionnée ci-dessus,
- o la syntaxe formelle comme décrit au paragraphe 9.9.

Une extension privée PEUT ou NE PEUT PAS être incluse dans la liste des capacités. Si elle l'est, l'étiquette de capacité DOIT commencer par "X". Un serveur PEUT fournir des mots-clés supplémentaires (pour de nouvelles commandes et aussi pour de nouvelles variantes de commandes existantes) au titre d'une extension privée. Pour éviter le risque d'une collision avec de futures extensions enregistrées, ces mots-clés DEVRAIENT commencer par "X".

Si le serveur annonce une capacité définie par une extension enregistrée, il DOIT mettre en œuvre l'extension de façon pleinement conforme à la spécification (par exemple, il DOIT mettre en œuvre toutes les commandes que l'extension décrit comme obligatoires). Si il ne met pas en œuvre l'extension comme spécifié, il NE DOIT PAS mentionner l'extension dans la liste des capacités sous son nom enregistré. Dans ce cas, il PEUT, mais NE DEVRAIT PAS, fournir une extension privée (pas sur la liste, ou mentionnée sous un nom différent) qui met en œuvre une partie de l'extension ou met en œuvre les commandes de l'extension avec une signification différente.

Un serveur NE DOIT PAS envoyer de codes de réponse différents aux commandes NNTP de base documentées ici ou aux commandes documentées dans les extensions enregistrées, dans les réponses à la disponibilité, ou utiliser une extension privée.

3.3.4 Registre initial de l'IANA

L'IANA tiendra un registre des étiquettes de capacité NNTP. Toutes les étiquettes de capacité du registre DOIVENT être des mots-clés et NE DOIVENT PAS commencer par X.

Le contenu initial du registre comporte les entrées suivantes :

Étiquette	Signification	Définition
AUTHINFO	Authentification	[RFC4643]
HDR	Restitution d'en-tête par lots	§ 3.3.2, § 8.5, et § 8.6
IHAVE	Commande IHAVE disponible	§ 3.3.2 et § 6.3.2
IMPLEMENTATION	Informations spécifiques de la mise en œuvre de serveur	§ 3.3.2
LIST	Variante de la commande LIST	§ 3.3.2 et § 7.6.1
MODE-READER	Serveur à commutation de mode et commande MODE READER disponible	§ 3.4.2
NEWNEWS	Commande NEWNEWS disponible	§ 3.3.2 et § 7.4
OVER	Prise en charge de la vue d'ensemble	§ 3.3.2, § 8.3, et § 8.4
POST	Commande POST disponible	§ 3.3.2 et § 6.3.1
READER	Commande READER disponible	§ 3.3.2
SASL	Mécanismes SASL pris en charge	[RFC4643]
STARTTLS	Sécurité de la couche transport	[RFC4642]
STREAMING	Alimentation en flux continu	[RFC4644]
VERSION	Versions NNTP prises en charge	§ 3.3.2

3.4 Commandes obligatoires et facultatives

Pour un certain nombre de raisons, toutes les commandes de la présente spécification ne sont pas obligatoires. Cependant, il est également indésirable que toutes les commandes soient facultatives, car cela signifierait qu'un client n'aurait aucune idée des facilités disponibles. Donc, le compromis est que certaines commandes de la présente spécification sont obligatoires (elles doivent être prises en charge par tous les serveurs) tandis que les autres ne le sont pas. Ces dernières sont subdivisées en bouquets, chacun indiqué par une seule étiquette de capacité.

- o Si l'étiquette est incluse dans la liste de capacités retournée par le serveur, le serveur DOIT prendre en charge toutes les commandes de ce bouquet.
- o Si l'étiquette n'est pas incluse, le serveur PEUT prendre en charge zéro, une ou plusieurs des commandes mais NE DEVRAIT PAS les prendre toutes en charge. En général, il n'y aura aucun moyen pour un client de déterminer quelles commandes sont prises en charge sans les essayer.

Les bouquets ont été choisis pour fournir des fonctionnalités utiles, et donc il est déconseillé aux auteurs de serveurs de ne mettre en œuvre qu'une partie d'un bouquet.

La description de chaque commande va indiquer qu'elle est obligatoire, ou va donner, en utilisant le terme "capacité indicative", l'étiquette de capacité indiquant si le bouquet qui inclut cette commande est disponible.

Lorsque un serveur ne met pas en œuvre une commande, il DOIT toujours générer un code de réponse générique 500 (ou un code de réponse générique de 501 dans le cas d'une variante d'une commande dépendant d'un second mot-clé lorsque la commande de base est reconnue). Autrement, la commande DOIT être pleinement mise en œuvre comme spécifié ; un serveur NE DOIT PAS mettre en œuvre seulement partiellement les commandes dans la présente spécification. (Les auteurs de clients devraient noter que certains serveurs non conformes à la présente spécification vont retourner un code de réponse générique de 502 à des commandes qui ne sont pas mises en œuvre.)

Note : certaines commandes ont des cas qui exigent que d'autres commandes soient utilisées d'abord. Si la première commande est mise en œuvre mais pas la seconde, la première DOIT quand même générer le code de réponse spécifique pertinent. Par exemple, si ARTICLE (paragraphe 6.2.1) est mis en œuvre mais pas GROUP (paragraphe 6.1.1) la réponse correcte à "ARTICLE 1234" reste 412.

3.4.1 Serveurs de lecture et serveurs de transit

NNTP est traditionnellement utilisé de deux façons différentes. La première utilisation est en "lecture", où le client va chercher des articles dans un grand répertoire géré par le serveur pour présentation immédiate ou différée à un utilisateur, et renvoie des articles créés par cet utilisateur au serveur (une action appelée "envoi" (*posting*)) pour être mémorisés et

distribués aux autres répertoires et utilisateurs. La seconde utilisation est pour le transfert en vrac d'articles d'un répertoire à un autre. Comme les hôtes qui font ce transfert tendent à être des homologues dans un réseau qui transmet les articles de l'un à l'autre, et pas des systèmes d'utilisateurs finaux, ce processus est appelé "d'homologue à homologue" (*peering*) ou "transit". (Même comme cela, un hôte est quand même le client et l'autre est le serveur).

En pratique, ces deux utilisations sont si différentes que certaines mises en œuvre de serveurs sont optimisées pour la lecture ou pour le transit et, par suite, n'offrent pas d'autre facilité ou seulement des caractéristiques limitées. D'autres mises en œuvre sont plus générales et offrent les deux. La présente spécification permet cela en mettant en bouquet en conséquence les commandes pertinentes : la commande I HAVE est conçue pour le transit, tandis que les commandes indiquées par la capacité READER sont conçues pour les clients lecteurs.

Sauf par l'effet de la commande MODE READER (paragraphe 5.3) sur un serveur à commutation de mode, une fois qu'un serveur a annoncé l'une ou l'autre capacité I HAVE ou READER, ou les deux, il DOIT continuer à les annoncer pour la session entière.

Un serveur PEUT fournir différents modes de comportement (transit, lecture, ou une combinaison des deux) à différentes connexions client et PEUT utiliser des informations externes, comme l'adresse IP du client, pour déterminer quel mode fournir à la connexion concernée.

L'accès officiel TCP pour le service NNTP est 119. Cependant, si un hôte souhaite offrir des serveurs séparés pour les clients de transit et de lecture, l'accès 433 DEVRAIT être utilisé pour le serveur de transit et le 119 pour le serveur de lecture.

3.4.2 Commutation de mode

Une mise en œuvre PEUT, mais NE DEVRAIT PAS, fournir les deux facilités de transit et de lecture sur le même serveur mais exiger que le client choisisse laquelle il souhaite utiliser. Un tel arrangement est appelé un serveur à "commutation de mode".

Un serveur à commutation de mode a deux modes :

- o mode transit, qui s'applique après la connexion initiale ;
 - * Il DOIT annoncer la capacité MODE-READER.
 - * Il NE DOIT PAS annoncer la capacité READER.
 Cependant, le serveur PEUT cesser d'annoncer la capacité MODE-READER après que le client a utilisé toute commande sauf CAPABILITIES.
- o mode lecture, après une commande MODE READER réussie (voir au paragraphe 5.3).
 - * Il NE DOIT PAS annoncer la capacité MODE-READER.
 - * Il DOIT annoncer la capacité READER.
 - * Il NE PEUT PAS annoncer la capacité I HAVE, même si il l'avait annoncée en mode transit.
 Un client NE DEVRAIT produire une commande MODE READER à un serveur que si il annonce la capacité MODE-READER. Si le serveur ne prend pas en charge CAPABILITIES (et donc ne se conforme pas à la présente spécification) le client PEUT utiliser l'heuristique suivante :
- o Si le client souhaite utiliser une commande "lecture", il DEVRAIT utiliser la commande MODE READER immédiatement après la connexion initiale.
- o Autrement, il NE DEVRAIT PAS utiliser la commande MODE READER.

Dans chaque cas, il devrait être prêt à ce que ne soit pas disponible une commande qui l'aurait été si il avait fait un autre choix.

3.5 Traitement en parallèle

NNTP est conçu pour fonctionner sur une connexion bidirectionnelle fiable, comme TCP. Donc, si une commande ne dépend pas de la réponse à la précédente, cela ne devrait pas avoir d'importance si elle est envoyée avant que cette réponse soit reçue. On appelle cela le "traitement en parallèle" (*pipelining*). Cependant, certaines mises en œuvre de serveur jettent tout le texte reçu du client à la suite de certaines commandes avant d'envoyer leur réponse. Si cela arrive, le traitement en parallèle sera affecté parce qu'une ou plusieurs commandes auront été ignorées ou mal interprétées, et le client va confronter les mauvaises réponses à chaque commande. Comme il y a des avantages significatifs au traitement en parallèle, mais aussi des circonstances où il est raisonnable ou normal que les serveurs se comportent de cette manière, le présent document fait peser certaines exigences sur les clients et les serveurs.

Sauf mention contraire, un client PEUT utiliser le traitement en parallèle. C'est-à-dire qu'il peut envoyer une commande avant d'avoir reçu la réponse à la commande précédente. Le serveur DOIT permettre le traitement en parallèle et NE DOIT

PAS éliminer du texte reçu après une commande. Sans considération de l'utilisation du traitement en parallèle, le serveur DOIT traiter les commandes dans leur ordre d'envoi.

Si la description spécifique d'une commande dit "NE DOIT PAS être traité en parallèle", cette commande DOIT terminer tout traitement en parallèle de commandes. C'est-à-dire que le client NE DOIT PAS envoyer de commande suivante avant d'avoir reçu le CRLF à la fin de la réponse à la commande. Le serveur PEUT ignorer toutes les données reçues après la commande et avant que le CRLF de la fin de la réponse soit envoyé au client.

La connexion initiale ne doit pas faire partie d'un traitement en parallèle ; c'est-à-dire que le client NE DOIT PAS envoyer de commande tant qu'il n'a pas reçu le CRLF à la fin de l'accueil.

Si le client utilise des appels de blocage du système pour envoyer des commandes, il DOIT s'assurer que la quantité de texte envoyé dans le traitement en parallèle ne cause pas une situation impossible entre transmission et réception. La quantité de texte impliquée va dépendre de la taille de la fenêtre dans la couche de transmission ; normalement, elle est de 4 k octets pour TCP. (Comme le serveur envoie seulement des données en réponse aux commandes du client, le problème inverse ne se pose pas.)

3.5.1 Exemples

Exemple d'utilisation correcte du traitement en parallèle :

```
[C] GROUP misc.test
[C] STAT
[C] NEXT
[S] 211 1234 3000234 3002322 misc.test
[S] 223 3000234 <45223423@exemple.com> restitué
[S] 223 3000237 <668929@exemple.org> restitué
```

Exemple d'utilisation incorrecte du traitement en parallèle (la commande MODE READER ne peut pas être traitée en parallèle) :

```
[C] MODE READER
[C] DATE
[C] NEXT
[S] 200 Serveur prêt, envoi permis
[S] 223 3000237 <668929@exemple.org> restitué
```

La commande DATE a été éliminée par le serveur, de sorte qu'il n'y a pas de réponse 111 pour y correspondre.

3.6 Articles

NNTP est destiné à transférer des articles entre clients et serveurs. Pour les besoins de la présente spécification, les articles doivent se conformer aux règles de cette section, et clients et serveurs DOIVENT correctement traiter tout article reçu des autres qui le font. Noter que cette exigence ne s'applique qu'au contenu des communications sur NNTP ; elle n'empêche pas le client ou serveur de rejeter ensuite un article pour des raisons de politique locale. Voir aussi à l'Appendice A d'autres restrictions sur le format des articles dans certaines utilisations de NNTP.

Un article consiste en deux parties : les en-têtes et le corps. Elles sont séparées par une seule ligne vide, ou en d'autres termes par deux paires consécutives de CRLF (si il y a plus d'une ligne vide, la seconde et les suivantes font partie du corps). Afin de satisfaire aux exigences générales de NNTP, un article NE DOIT PAS inclure l'octet NUL, NE DOIT PAS contenir les octets LF et CR autrement qu'au titre d'une paire CRLF, et DOIT se terminer par une paire CRLF. La présente spécification ne fait pas d'autre restriction sur le corps ; en particulier, il PEUT être vide.

Les en-têtes d'un article consistent en une ou plusieurs lignes d'en-tête. Chaque ligne d'en-tête consiste en un nom d'en-tête, deux points, une espace, le contenu de l'en-tête, et un CRLF, dans cet ordre. Le nom consiste en un ou plusieurs caractères US-ASCII imprimables autres que deux points et, pour les besoins de la présente spécification, n'est pas sensible à la casse. Il PEUT y avoir plus d'une ligne d'en-tête avec le même nom. Le contenu NE DOIT PAS comporter de CRLF ; il PEUT être vide. Un en-tête peut "aller à la ligne" ; c'est-à-dire, une paire CRLF peut être placée avant tout caractère TAB ou espace dans la ligne. Il DOIT quand même y avoir d'autres octets entre deux paires CRLF dans une ligne d'en-tête. (Noter qu'aller à la ligne signifie que la ligne d'en-tête occupe plus d'une ligne à l'affichage ou à la transmission ; néanmoins, on l'appelle quand même "une" ligne d'en-tête.) La présence ou l'absence de saut à la ligne n'affecte pas la signification de la ligne d'en-tête ; c'est-à-dire, les paires CRLF introduites par le saut à la ligne ne sont pas considérées comme faisant partie

du contenu d'en-tête. Les lignes d'en-tête NE DEVRAIENT PAS aller à la ligne avant l'espace après les deux points qui suivent le nom d'en-tête et DEVRAIENT inclure au moins un octet autre que %x09 ou %x20 entre les paires CRLF. Cependant, si un article qui ne réussit pas à satisfaire cette exigence a été reçu d'ailleurs, les clients et serveurs PEUVENT le transférer l'un à l'autre sans le refaire sauter à la ligne.

Le contenu d'un en-tête DEVRAIT être en UTF-8. Cependant, si une mise en œuvre reçoit un article d'ailleurs qui utilise des octets dans la gamme 128 à 255 d'une autre manière, il PEUT le passer à un client ou serveur sans modification. Donc, les mises en œuvre DOIVENT être prêtes à recevoir de tels en-têtes, et des données qui en sont dérivées (par exemple, dans les réponses à la commande OVER, paragraphe 8.3) et NE DOIVENT PAS supposer que ce sont toujours de l'UTF-8. Tout traitement externe de ces en-têtes, incluant l'identification du codage utilisé, sort du domaine d'application de ce document.

Chaque article DOIT avoir un identifiant de message univoque ; deux articles offerts par un serveur NNTP NE DOIVENT PAS avoir le même identifiant de message. Pour les besoins de la présente spécification, les identifiants de message sont des chaînes opaques qui DOIVENT satisfaire les exigences suivantes :

- o un identifiant de message DOIT commencer par "<", se terminer par ">", et NE DOIT PAS contenir ce dernier caractère sauf en fin ;
- o un identifiant de message DOIT faire entre 3 et 250 octets ;
- o un identifiant de message NE DOIT PAS contenir d'octets autres que de caractères US-ASCII imprimables.

Deux identifiants de message sont les mêmes si et seulement si ils consistent en la même séquence d'octets.

La présente spécification ne décrit pas comment l'identifiant de message d'un article est déterminé. Si le serveur n'a aucun moyen de déterminer un identifiant de message à partir de l'article lui-même, il DOIT en synthétiser un (la présente spécification n'exige pas que l'article en soit changé). Voir aussi l'Appendice A.2.

4. Format WILDMAT

Le format WILDMAT décrit ici se fonde sur la première version développée par Rich Salz [SALZ1992], qui à son tour était dérivée du format utilisé dans la commande UNIX "find" pour articuler les noms de fichiers. Il a été développé pour fournir un mécanisme uniforme pour confronter les schémas de la même manière que la coquille UNIX confronte les noms de fichiers.

4.1 Syntaxe de wildmat

Un wildmat est décrit par la syntaxe ABNF [RFC4234] suivante, qui est un extrait de celle du paragraphe 9.8.

```
wildmat = wildmat-pattern *(", ["!"] wildmat-pattern)
wildmat-pattern = 1*wildmat-item
wildmat-item = wildmat-exact / wildmat-wildwildmat-exact = %x22-29 / %x2B / %x2D-3E / %x40-5A / %x5E-7E /
UTF8-non-ascii ; cela exclut ! * , ? [ \ ]
wildmat-wild = "*" / "?"
```

Note : les caractères ",", "\", "[", et "]" ne sont pas permis dans les wildmats, tandis que * et ? sont toujours des caractères génériques (*wildcard*). Cela ne devrait pas poser de problème, car ces caractères ne peuvent pas apparaître dans les noms de groupes de nouvelles, qui sont la seule utilisation actuelle des wildmats. La barre oblique inverse (*backslash*) est couramment utilisée pour supprimer la signification spéciale des caractères, tandis que les crochets (*brackets*) sont utilisés pour introduire des ensembles. Cependant, ces usages ne sont pas universels, et l'interprétation de ces caractères dans le contexte de chaînes UTF-8 est potentiellement complexe et diffère de la pratique existante, de sorte qu'ils ont été omis de la présente spécification. Une future extension de la présente spécification pourrait fournir une sémantique pour ces caractères.

4.2 Sémantique de wildmat

Un wildmat est confronté à une chaîne et correspond ou ne correspond pas. Pour ce faire, chaque constituant <wildmat-pattern> est confronté à la chaîne, et le schéma le plus à droite qui correspond est identifié. Si ce <wildmat-pattern> n'est pas précédé de "!", tout le wildmat correspond. Si il est précédé de "!", ou si aucun <wildmat-pattern> ne correspond, le wildmat entier ne correspond pas.

Par exemple, considérons le wildmat "a*,!*b,*c*" :

- o La chaîne "aaa" correspond parce que la correspondance la plus à droite est avec "a*".

- o La chaîne "abb" ne correspond pas parce que la correspondance la plus à droite est avec "*b".
- o La chaîne "ccb" correspond parce que la correspondance la plus à droite est avec "*c*".
- o La chaîne "xxx" ne correspond pas parce que aucun <wildmat-pattern> ne correspond.

Un <wildmat-pattern> correspond à une chaîne si la chaîne peut être cassée en composants, dont chacun correspond au <wildmat-item> correspondant dans le schéma. Les correspondances doivent être dans le même ordre, et la chaîne toute entière doit être utilisée dans la confrontation. Le schéma est "ancré" ; c'est-à-dire que le premier et le dernier caractère de la chaîne doivent correspondre au premier et au dernier élément, respectivement, (sauf si cet élément est un astérisque correspondant à zéro caractère).

Un <wildmat-exact> correspond au même caractère (qui peut faire plus d'un octet en UTF-8).

"?" correspond exactement à un caractère (qui peut faire plus d'un octet).

"*" correspond à zéro, un ou plusieurs caractères. Il peut correspondre à une chaîne vide, mais il ne peut pas correspondre à une sous séquence d'une séquence UTF-8, c'est-à-dire n'être pas aligné sur les limites de caractères.

4.3 Extensions

Un serveur NNTP ou extension PEUT étendre la syntaxe ou la sémantique des wildmats fournie selon lesquelles tous les wildmats qui satisfont aux exigences du paragraphe 4.1 ont la signification décrite pour eux au paragraphe 4.2. De futures éditions du présent document pourraient aussi étendre les wildmats.

4.4 Exemples

Dans ces exemples, § et @ sont utilisés pour représenter, respectivement, les deux octets %xC2 et %xA3 ; §@ est donc le codage UTF-8 pour le symbole de la livre sterling, donné par # dans les descriptions.

Wildmat : description des chaînes qui correspondent

abc : la chaîne "abc"

abc,def : les deux chaînes "abc" et "def"

§@ ; la chaîne de un caractère "#"

a* : toute chaîne qui commence par "a"

a*b : toute chaîne qui commence par "a" et se termine par "b"

a*,*b : toute chaîne qui commence par "a" ou se termine par "b"

a*,!*b : toute chaîne qui commence par "a" et ne se termine pas par "b"

a*,!*b,c* : toute chaîne qui commence par "a" et ne se termine pas par "b", et toute chaîne qui commence par "c" quel que soit ce par quoi elle se termine.

a*,c*,!*b : toute chaîne qui commence par "a" ou "c" et ne se termine pas par "b".

?a* : toute chaîne avec "a" comme second caractère.

??a* : toute chaîne avec "a" comme troisième caractère.

*a? : toute chaîne avec "a" comme avant dernier caractère.

*a?? : toute chaîne avec "a" comme antépénultième caractère

5. Commandes d'administration de session

5.1. Connexion initiale

5.1.1 Usage

Cette commande NE DOIT PAS être traitée en parallèle.

Réponses [1]

200 : Service disponible, envoi permis

201 : Service disponible, envoi interdit

400 : Service temporairement indisponible [2]

502 : Service indisponible en permanence [2]

[1] Ce sont les seuls codes de réponse valides pour l'accueil initial ; le serveur NE DOIT PAS retourner d'autre code de réponse générique.

[2] À la suite d'une réponse 400 ou 502, le serveur DOIT immédiatement clore la connexion.

5.1.2 Description

Il n'y a pas de commande présentée par le client à la connexion initiale au serveur. Le serveur DOIT présenter un code de réponse approprié comme accueil au client. Cette réponse informe le client de la disponibilité du service et si il est permis au client d'envoyer.

Si le serveur accepte d'autres commandes du client incluant POST, le serveur DOIT présenter un code d'accueil 200. Si le serveur va accepter d'autres commandes du client, mais si le client n'est pas autorisé à envoyer des articles en utilisant la commande POST, le serveur DOIT présenter un code d'accueil de 201.

Autrement, le serveur DOIT présenter un code d'accueil 400 ou 502 et ensuite clore immédiatement la connexion. 400 DEVRAIT être utilisé si le problème est seulement temporaire (par exemple, à cause de la charge) et le client peut s'attendre à être capable de se connecter avec succès à l'avenir sans faire de changement. 502 DOIT être utilisé si il n'est pas permis au client, dans aucune circonstance, d'interagir avec le serveur, et PEUT être utilisé si le serveur a des informations insuffisantes pour déterminer si la problème est temporaire ou permanent.

Note : la distinction entre les codes de réponse 200 et 201 s'est révélée en pratique être insuffisante ; par exemple, certains serveurs ne permettent pas d'envoi tant que le client ne s'est pas authentifié, alors que d'autres clients supposent qu'une réponse 201 signifie que l'envoi ne sera jamais possible même après l'authentification. Donc, les clients DEVRAIENT utiliser la commande CAPABILITIES (paragraphe 5.2) plutôt que de s'appuyer sur cette réponse.

5.1.3 Exemples

Exemple d'une connexion normale d'un client autorisé qui termine ensuite la session (voir au paragraphe 5.4) :

```
[Établissement initial de connexion terminé.]
[S] 200 NNTP Service prêt, envoi permis
[C] QUIT
[S] 205 Sortie normale du service NNTP
[Le serveur clôt la connexion.]
```

Exemple d'une connexion normale d'un client autorisé, c'est-à-dire qui n'a pas la permission d'envoyer, qui clôt aussi immédiatement la session :

```
[Établissement initial de connexion terminé.]
[S] 201 NNTP Service prêt, envoi interdit
[C] QUIT
[S] 205 Sortie normale du service NNTP
[Le serveur clôt la connexion.]
```

Exemple d'une connexion normale d'un client non autorisé :

```
[Établissement initial de connexion terminé.]
[S] 502 Service NNTP indisponible en permanence
[Le serveur clôt la connexion.]
```

Exemple d'une connexion d'un client si le serveur est incapable de fournir le service :

```
[Établissement initial de connexion terminé.]
[S] 400 NNTP Service temporairement indisponible
[Le serveur clôt la connexion.]
```

5.2 CAPABILITIES

5.2.1 Usage

Cette commande est obligatoire.

Syntaxe : CAPABILITIES [mot-clé]

Réponse : 101 La liste des capacités suit (multi lignes)

Paramètres : mot-clé, caractéristiques supplémentaires, voir la description.

5.2.2 Description

La commande CAPABILITIES permet à un client de déterminer les capacités du serveur à tout moment.

Cette commande PEUT être produite à tout moment ; le serveur NE DOIT PAS exiger qu'elle soit produite pour faire usage de toute capacité. La réponse générée par cette commande PEUT changer durant une session à cause d'autres informations d'état (qui à leur tour peuvent changer par l'effet d'autres commandes ou d'événements externes). Un client NNTP n'est capable d'obtenir les informations courantes et correctes concernant les capacités disponibles à tout moment d'une session qu'en produisant une commande CAPABILITIES à ce moment de cette session et en traitant la réponse.

La liste de capacités est retournée comme un bloc de données multi lignes suivant le code de réponse 101. Chaque capacité est décrite par une ligne de capacité séparée. Le serveur NE DOIT PAS mentionner deux fois la même capacité dans la réponse, même avec des arguments différents. Excepté que la capacité VERSION DOIT être la première ligne, l'ordre dans lequel apparaissent les lignes de capacité n'a pas de signification ; le serveur n'a même pas besoin de retourner dans le même ordre.

Alors que certaines capacités seront probablement toujours disponibles ou jamais disponibles, d'autres (notamment les extensions) vont apparaître et disparaître selon les changements d'état du serveur au sein de la session ou selon des événements externes entre les sessions. Un client NNTP PEUT mettre en antémémoire les résultats de ces commandes, mais NE DOIT PAS compter que des résultats placés en antémémoire seront corrects, qu'ils proviennent de plus tôt dans cette session ou d'une session précédente, il DOIT accepter de bonne grâce que l'état mis en antémémoire soit périmé, et DEVRAIT (si il met les résultats en antémémoire) donner un moyen de forcer le rafraîchissement des informations mises en antémémoire. De plus, un client NE DOIT PAS utiliser les résultats mis en antémémoire en relation avec les extensions de sécurité, de confidentialité, et d'authentification. Voir au paragraphe 12.6 un exposé sur ce sujet.

L'argument mot-clé n'est pas utilisé par la présente spécification. Il est fourni pour que les extensions ou révisions de la présente spécification puissent inclure des caractéristiques supplémentaires pour cette commande sans exiger que soit utilisée deux fois la commande CAPABILITIES (une fois pour déterminer si les caractéristiques supplémentaires sont disponibles, et une deuxième fois pour les utiliser). Si le serveur ne reconnaît pas l'argument (et qu'il est un mot-clé) il DOIT répondre par le code de réponse 101 comme si l'argument avait été omis. Si le serveur ne reconnaît pas un argument, il PEUT utiliser le code de réponse 101 ou PEUT utiliser un autre code de réponse (qui sera défini dans la spécification de cette caractéristique). Si l'argument n'est pas un mot-clé, le code de réponse générique 501 DOIT être retourné. Le serveur NE DOIT PAS générer d'autre code de réponse à la commande CAPABILITIES.

5.2.3 Exemples

Exemple d'une réponse minimale (un serveur en lecture seule) :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] LIST ACTIVE NEWSGROUPS
[S] .
```

Exemple d'une réponse d'un serveur qui a une gamme de facilités et qui se décrit aussi :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] IHAVE
[S] POST
[S] NEWNEWS
[S] LIST ACTIVE NEWSGROUPS ACTIVE.TIMES OVERVIEW.FMT
[S] IMPLEMENTATION INN 4.2 2004-12-25
[S] OVER MSGID
[S] STREAMING
[S] XSECRET
[S] .
```

Exemple d'un serveur qui prend en charge plus d'une version de NNTP :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2 3
[S] READER
[S] LIST ACTIVE NEWSGROUPS
[S] .
```

Exemple d'un client qui tente d'utiliser une caractéristique de la commande CAPABILITIES que le serveur ne prend pas en charge :

```
[C] CAPABILITIES AUTOUPDATE
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] IHAVE
[S] LIST ACTIVE NEWSGROUPS OVERVIEW.FMT HEADERS
[S] OVER MSGID
[S] HDR
[S] NEWNEWS
[S] .
```

5.3 MODE READER

5.3.1 Usage

Indication de capacité : MODE-READER

Cette commande NE DOIT PAS être traitée en parallèle.

Syntaxe : MODE READER

Réponses

200 : envoi permis

201 : envoi interdit

502 : service lecture indisponible en permanence [1]

[1] À la suite d'une réponse 502, le serveur DOIT immédiatement clore la connexion.

5.3.2 Description

La commande MODE READER donne pour instruction à un serveur à commutation de mode de changer de mode, comme décrit au paragraphe 3.4.2.

Si le serveur est à commutation de mode, il passe de son mode transit à son mode lecture, indiquant cela en changeant en conséquence la liste de capacités. Il DOIT alors retourner une réponse 200 ou 201 avec la même signification que pour l'accueil initial (comme décrit au paragraphe 5.1.1). Noter que la réponse n'a pas besoin d'être la même que celle présentée durant l'accueil initial. Le client NE DOIT PAS produire de MODE READER plus d'une fois dans une session ou après la production de toute commande de sécurité ou de confidentialité. Lorsque la commande MODE READER est produite, le serveur PEUT rétablir l'état qu'il avait immédiatement après la connexion initiale avant le changement de mode.

Si le serveur n'est pas à commutation de mode, ce qui suit s'applique alors :

- o Si il annonce la capacité READER, il DOIT retourner une réponse 200 ou 201 avec la même signification que pour l'accueil initial ; dans ce cas, la commande NE DOIT affecter en aucune façon l'état du serveur.
- o Si il n'annonce pas la capacité READER, il DOIT retourner une réponse 502 et ensuite clore immédiatement la connexion.

5.3.3 Exemples

Exemple d'utilisation de la commande MODE READER sur un serveur en transit seul (qui ne fournit donc pas les facilités de lecture) :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] IHAVE
[S] .
[C] MODE READER
[S] 502 Seulement service Transit
[Le serveur clôt la connexion.]
```

Exemple d'utilisation de la commande MODE READER sur un serveur qui fournit les facilités de lecture :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] LIST ACTIVE NEWSGROUPS
[S] .
[C] MODE READER
[S] 200 Mode lecture, envoi permis
[C] IHAVE <je.suis.un.article.que.tu.as@Exemple.com>
[S] 500 Commande inconnue
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
```

Noter que dans ces deux situations, le client NE DEVRAIT PAS utiliser MODE READER.

Exemple d'utilisation de la commande MODE READER sur un serveur à commutation de mode :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] IHAVE
[S] MODE-READER
[S] .
[C] MODE READER
[S] 200 Mode lecture, envoi permis
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] NEWNEWS
[S] LIST ACTIVE NEWSGROUPS
[S] STARTTLS
[S] .
```

Dans ce cas, le serveur offre (mais n'exige pas) la confidentialité TLS dans son mode lecture mais pas dans son mode transit.

Exemple d'utilisation de la commande MODE READER où le client n'a pas la permission d'envoi :

```
[C] MODE READER
[S] 201 Service NNTP prêt, envoi interdit
```

5.4 QUIT

5.4.1 Usage

Cette commande est obligatoire.

Syntaxe : QUIT

Réponse : 205 : Fermeture de la connexion

5.4.2 Description

Le client utilise la commande QUIT pour terminer la session. Le serveur DOIT accuser réception de la commande QUIT et ensuite clore la connexion avec le client. C'est la méthode préférée pour qu'un client indique qu'il a fini toutes ses transactions avec le serveur NNTP.

Si un client se déconnecte simplement (ou si la connexion arrive en fin de temporisation ou si une autre faute se produit) le serveur DOIT cesser en douceur ses tentatives de service au client, en se déconnectant de son extrémité si nécessaire.

Le serveur NE DOIT PAS générer de code de réponse à la commande QUIT autre que 205 ou, si des arguments sont fournis, 501.

5.4.3 Exemple

```
[C] QUIT
[S] 205 clôture de la connexion
[Le serveur clôt la connexion.]
```

6. Envoi et restitution d'article

Les clients lecteurs de nouvelles ont à leur disposition divers mécanismes pour récupérer des articles via NNTP. Les articles de nouvelles sont mémorisés et indexés en utilisant trois types de clés. Le premier type de clé est l'identifiant de message d'un article et est unique au monde. Le second type de clé est composé d'un nom de groupe de nouvelles (*newsgroup*) et d'un numéro d'article au sein de ce groupe de nouvelles. Sur un certain serveur, il DOIT seulement y avoir un article avec un certain numéro au sein de tout groupe de nouvelles, et un article NE DOIT PAS avoir deux numéros différents dans le même groupe de nouvelles. Un article peut faire l'objet d'un envoi croisé à plusieurs groupes de nouvelles, de sorte qu'il peut y avoir plusieurs clés qui pointent sur le même article sur le même serveur ; ils PEUVENT avoir des numéros différents dans chaque groupe de nouvelles. Cependant, il n'est pas exigé que ce type de clé soit unique au monde, de sorte que la même clé PEUT se référer à différents articles sur différents serveurs. (Noter que les termes "groupe" et "groupe de nouvelles" sont équivalents.)

Le type de clé final est l'horodatage d'arrivée, qui donne l'heure à laquelle l'article est arrivé sur le serveur. Le serveur DOIT s'assurer que les numéros d'article sont produits dans l'ordre des horodatages d'arrivée ; c'est-à-dire que les articles qui arrivent plus tard DOIVENT avoir des numéros plus élevés que ceux qui sont arrivés plus tôt. Le serveur DEVRAIT allouer le prochain numéro de séquence non utilisé à chaque nouvel article.

Les numéros d'article DOIT sont entre 1 et 2 147 483 647, inclus. Le client et le serveur PEUVENT utiliser des zéros en tête en spécifiant les numéros d'article mais NE DOIVENT PAS utiliser plus de 16 chiffres. Dans certaines situations, la valeur zéro remplace un numéro d'article pour montrer une situation particulière.

Noter qu'il est probable que la limite de numéro d'article de 2 147 483 647 sera rehaussée dans une future révision ou extension de la présente spécification. Bien que les serveurs NE DOIVENT PAS envoyer de numéros d'article supérieurs à cette limite actuelle, il est conseillé aux développeurs de clients et serveurs d'utiliser des structures internes et des types de données capables de traiter de plus grandes valeurs en anticipation d'un tel changement.

6.1 Choix de groupe et d'article

Les commandes suivantes sont utilisées pour établir le "groupe de nouvelles actuellement choisi" et le "numéro d'article actuel", qui sont utilisés par diverses commandes. Au début d'une session NNTP, ces deux valeurs sont réglées à la valeur spéciale "invalide".

6.1.1 GROUP

6.1.1.1 Usage

Indication de capacité : READER

Syntaxe : groupe GROUP

Réponses

211 : numéro de groupe bas haut ; groupe retenu avec succès

411 : Pas de tel groupe de nouvelles

Paramètres

groupe : nom du groupe de nouvelles

nombre : nombre estimé d'articles dans le groupe

bas : marque inférieure rapportée

haut : marque supérieure rapportée

6.1.1.2 Description

La commande GROUP choisit un groupe de nouvelles comme groupe de nouvelles actuellement choisi et retourne des informations sommaires sur lui.

L'argument exigé est le nom du groupe de nouvelles à choisir (par exemple, "news.software.nntp"). Une liste des groupes de nouvelles valides peut être obtenu en utilisant la commande LIST ACTIVE (voir au paragraphe 7.6.3). La réponse de choix réussi va retourner les numéros d'article du premier et du dernier article du groupe au moment du choix (ces numéros sont mentionnés comme "marque inférieure rapportée" et "marque supérieure rapportée") et une estimation du nombre d'articles actuellement disponibles dans le groupe.

Si le groupe n'est pas vide, l'estimation DOIT être d'au moins le nombre réel d'articles disponibles et NE DOIT PAS être supérieur de un à la différence entre les marques inférieures et supérieures rapportées. (Certaines mises en œuvre vont en fait compter le nombre d'articles actuellement mémorisés. D'autres vont juste soustraire la marque inférieure de la marque supérieure et ajouter un pour avoir une estimation.)

Si le groupe est vide, une des trois situations suivantes va se produire. Les clients DOIVENT accepter les trois cas ; les serveurs NE DOIVENT PAS représenter un groupe vide d'une autre façon.

- o La marque supérieure sera inférieure de un à la marque inférieure, et le compte estimé d'articles sera zéro. Les serveurs DEVRAIENT utiliser cette méthode pour montrer un groupe vide. C'est la seule fois où la marque supérieure peut être inférieure à la marque inférieure.
- o Les trois nombres sont tous à zéro.
- o La marque supérieure est supérieure ou égale à la marque inférieure. Le compte d'articles estimé peut être zéro ou non zéro ; si il est non zéro, la même exigence s'applique que pour un groupe non vide.

L'ensemble d'articles dans un groupe peut changer après l'exécution de la commande GROUP :

- o des articles peuvent être retirés du groupe,
- o des articles peuvent être réinstallés dans le groupe avec le même numéro d'article, mais ces articles DOIVENT avoir des numéros qui ne sont pas inférieurs à la marque inférieure rapportée (noter que c'est une réinstallation de l'article précédent, et non un nouvel article réutilisant le numéro),
- o de nouveaux articles peuvent être ajoutés avec des numéros d'article supérieurs à la marque supérieure rapportée. (Si un article qui était celui qui avait le plus haut numéro a été retiré, et si la marque supérieure a été ajustée en conséquence, le prochain nouvel article n'aura pas le numéro supérieur de un à la marque supérieure rapportée.)

Sauf lorsque le groupe est vide et que les trois numéros sont à zéro, chaque fois qu'est produite une commande GROUP suivante pour le même groupe de nouvelles, soit par le même client, soit par un client différent, la marque inférieure rapportée dans la réponse NE DOIT PAS être inférieure au numéro de toute réponse antérieure pour ce groupe de nouvelles dans cette session, et elle NE DEVRAIT PAS être inférieure au numéro de toute réponse antérieure jamais envoyée à tout client pour ce groupe de nouvelles. Tout échec à satisfaire cette dernière condition DEVRAIT seulement être transitoire. Le client peut faire usage de la marque inférieure pour retirer toutes les informations mémorisées sur les articles qui ont des numéros inférieurs, car ils ne vont jamais réapparaître. Cela inclut la situation où la marque supérieure est inférieure de un à la marque inférieure. Aucune hypothèse similaire ne peut être faite sur la marque supérieure, car elle peut décroître si un article est retiré et ensuite augmenter à nouveau si il est réinstallé ou si de nouveaux articles arrivent.

Lorsque un groupe valide est choisi au moyen de cette commande, le groupe de nouvelles actuellement choisi DOIT être établi dans ce groupe, et le numéro d'article actuel DOIT être réglé au premier article dans le groupe (ceci s'applique même si le groupe est déjà le groupe de nouvelles actuellement choisi). Si un groupe de nouvelles vide est choisi, le numéro d'article actuel est rendu invalide. Si un groupe invalide est spécifié, le groupe de nouvelles actuellement choisi et le numéro d'article courant NE DOIVENT PAS être changés.

La commande GROUP ou LISTGROUP (voir au paragraphe 6.1.2) DOIT être utilisée par un client, et une réponse de

réussite reçue, avant que soit utilisée toute autre commande qui dépend de la valeur du groupe de nouvelles actuellement choisi ou du numéro d'article actuel.

Si le groupe spécifié n'est pas disponible sur le serveur, une réponse 411 DOIT être retourné.

6.1.1.3 Exemples

Exemple pour un groupe connu du serveur :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
```

Exemple pour un groupe inconnu du serveur :

```
[C] GROUP exemple.is.sob.bradner.ou.barber
[S] 411 exemple.is.sob.bradner.ou.barber est inconnu
```

Exemple de groupe vide utilisant la réponse préférée :

```
[C] GROUP exemple.currently.vide.newsgroup
[S] 211 0 4000 3999 exemple.currently.vide.newsgroup
```

Exemple d'un groupe vide utilisant une réponse de remplacement :

```
[C] GROUP exemple.currently.vide.newsgroup
[S] 211 0 0 0 exemple.currently.vide.newsgroup
```

Exemple d'un groupe vide utilisant une réponse de remplacement différente :

```
[C] GROUP exemple.currently.vide.newsgroup
[S] 211 0 4000 4321 exemple.currently.vide.newsgroup
```

Exemple faisant un nouveau choix du groupe de nouvelles actuellement choisi :

```
[C] GROUP misc.test
[S] 211 123 234 567 misc.test
[C] STAT 444
[S] 223 444 <123456@exemple.net> restitué
[C] GROUP misc.test
[S] 211 123 234 567 misc.test
[C] STAT
[S] 223 234 <different@exemple.net> restitué
```

6.1.2 LISTGROUP

6.1.2.1 Usage

Indication de capacité : READER

Syntaxe : LISTGROUP [groupe [gamme]]

Réponses

211 : numéro de groupe haut bas. Les numéros d'article suivent (multi lignes)

411 : pas de tel groupe de nouvelles

412 : pas de groupe de nouvelles choisi [1]

[1] La réponse 412 ne peut se produire que si aucun groupe n'a été spécifié.

Paramètres :

groupe : Nom du groupe de nouvelles

gamme : Gamme d'articles à rapporter

nombre : Estimation du nombre d'articles dans le groupe

bas : Marque inférieure rapportée

haut : Marque supérieure rapportée

6.1.2.2 Description

La commande LISTGROUP choisit un groupe de nouvelles de la même manière que la commande GROUP (voir au paragraphe 6.1.1) mais fournit aussi une liste des numéros d'article dans le groupe de nouvelles. Si aucun groupe n'est spécifié, on utilise le groupe de nouvelles actuellement choisi.

Lorsque elle réussit, une liste de numéros d'article est retournée comme un bloc de données multi lignes suivant le code de réponse 211 (les arguments sur la ligne initiale de réponse sont les mêmes que pour la commande GROUP). La liste contient un numéro par ligne et est dans l'ordre numérique. Elle mentionne précisément les articles qui existent dans le groupe au moment de la sélection (donc, un groupe vide produit une liste vide). Si l'argument gamme facultative est spécifié, seuls les articles qui sont dans la gamme sont inclus dans la liste (donc, la liste PEUT être vide même si le groupe ne l'est pas).

L'argument gamme peut être un des suivants :

- o un numéro d'article,
- o un numéro d'article suivi par un tiret pour indiquer tous les suivants,
- o un numéro d'article suivi par un tiret suivi par un autre numéro d'article.

Dans ce dernier cas, si le second numéro est inférieur au premier numéro, la gamme ne contient alors pas d'article. Omettre la gamme est équivalent à spécifier la gamme 1-.

Si le groupe spécifié n'est pas disponible sur le serveur, une réponse 411 DOIT être retournée. Si aucun groupe n'est spécifié et si le groupe de nouvelles actuellement choisi est invalide, une réponse 412 DOIT être retournée.

Excepté que l'argument groupe est facultatif, qu'un argument gamme peut être spécifié, et qu'un bloc de données multi lignes suit le code de réponse 211, la commande LISTGROUP est identique à la commande GROUP. En particulier, lorsque elle réussit, la commande règle le numéro d'article courant au premier article du groupe, si il y en a, même si ce n'est pas dans la gamme spécifiée par le second argument.

Noter que l'argument gamme est une nouvelle caractéristique de la présente spécification et les serveurs qui ne prennent pas en charge CAPABILITIES (et donc ne se conforment pas à la présente spécification) ont peu de chances de la prendre en charge.

6.1.2.3 Exemples

Exemple de LISTGROUP utilisé pour choisir un groupe :

```
[C] LISTGROUP misc.test
[S] 211 2000 3000234 3002322 La liste misc.test suit
[S] 3000234
[S] 3000237
[S] 3000238
[S] 3000239
[S] 3002322
[S] .
```

Exemple de LISTGROUP sur un groupe vide :

```
[C] LISTGROUP exemple.vide.newsgroup
[S] 211 0 0 0 La liste exemple.vide.nexsgroup suit
[S] .
```

Exemple de LISTGROUP sur un groupe de nouvelles actuellement choisi valide :

```
[C] GROUP misc.test
[S] 211 2000 3000234 3002322 misc.test
[C] LISTGROUP
[S] 211 2000 3000234 3002322 la liste misc.test suit
[S] 3000234
[S] 3000237
```

[S] 3000238
 [S] 3000239
 [S] 3002322
 [S] .

Exemple de LISTGROUP avec une gamme :

[C] LISTGROUP misc.test 3000238-3000248
 [S] 211 2000 3000234 3002322 La liste misc.test suit
 [S] 3000238
 [S] 3000239
 [S] .

Exemple de LISTGROUP avec une gamme vide :

[C] LISTGROUP misc.test 12345678-
 [S] 211 2000 3000234 3002322 la liste misc.test suit
 [S] .

Exemple de LISTGROUP avec une gamme invalide :

[C] LISTGROUP misc.test 9999-111
 [S] 211 2000 3000234 3002322 La liste misc.test suit
 [S] .

6.1.3 LAST

6.1.3.1. Usage

Indication de capacité : READER

Syntaxe : LAST

Réponses :

223 n identifiant de message : article trouvé
 412 : aucun groupe de nouvelles choisi
 420 : numéro d'article courant invalide
 422 : pas d'article précédent dans ce groupe

Paramètres

n : nombre d'article

identifiant de message : identifiant de message de l'article

6.1.3.2 Description

Si le groupe de nouvelles actuellement choisi est valide, le numéro d'article courant DOIT être réglé à l'article précédent dans ce groupe de nouvelles (c'est-à-dire, le plus fort numéro d'article existant moins le numéro d'article courant). En cas de succès, une réponse indiquant le nouveau numéro d'article courant et l'identifiant de message de cet article DOIT être retournée. Aucun texte d'article n'est envoyé en réponse à cette commande.

Il PEUT n'y avoir pas d'article précédent dans le groupe, bien que le numéro d'article courant ne soit pas la marque inférieure rapportée. Il NE DOIT PAS y avoir d'article précédent lorsque le numéro d'article courant est la marque inférieure rapportée.

Parce que les articles peuvent être supprimés et ajoutés, le résultat de plusieurs commandes LAST et NEXT PEUT n'être pas cohérent sur la durée de vie d'une session NNTP particulière.

Si le numéro d'article courant est déjà le premier article du groupe de nouvelles, une réponse 422 DOIT être retournée. Si le numéro d'article courant est invalide, une réponse 420 DOIT être retournée. Si le groupe de nouvelles actuellement choisi est invalide, une réponse 412 DOIT être retournée. Dans les trois cas, le groupe de nouvelles actuellement choisi et le numéro d'article courant NE DOIVENT PAS être altérés.

6.1.3.3 Exemples

Exemple d'une restitution d'article réussie en utilisant LAST :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] NEXT
[S] 223 3000237 <668929@exemple.org> restitué
[C] LAST
[S] 223 3000234 <45223423@exemple.com> restitué
```

Exemple de tentative de restitution d'un article sans avoir d'abord choisi un groupe (via la commande GROUP) :

```
[On suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] LAST
[S] 412 pas de groupe de nouvelles choisi
```

Exemple de tentative de restitution d'un article en utilisant la commande LAST lorsque le numéro d'article courant est celui du premier article du groupe :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] LAST
[S] 422 pas d'article précédent à restituer
```

Exemple de tentative de restitution d'un article en utilisant la commande LAST lorsque le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] LAST
[S] 420 pas d'article courant choisi
```

6.1.4 NEXT

6.1.4.1 Usage

Indication de capacité : READER

Syntaxe : NEXT

Réponses :

```
223 n identifiant de message : article trouvé
412 : pas de groupe de nouvelles choisi
420 : numéro d'article courant invalide
421 : pas d'article suivant dans ce groupe
```

Paramètres

n : nombre d'articles

identifiant de message : identifiant de message de l'article

6.1.4.2 Description

Si le groupe de nouvelles actuellement choisi est valide, le numéro d'article courant DOIT être réglé au prochain article du groupe de nouvelles (c'est-à-dire, le plus bas numéro d'article existant supérieur au numéro d'article courant). En cas de succès, une réponse indiquant le nouveau numéro d'article courant et l'identifiant de message de cet article DOIT être retournée. Aucun texte d'article n'est envoyé en réponse à cette commande.

Si le numéro d'article courant est déjà le dernier article du groupe de nouvelles, une réponse 421 DOIT être retournée. Sous tous les autres aspects (à part, bien sûr, l'absence de réponse 422) cette commande est identique à la commande LAST (paragraphe 6.1.3).

6.1.4.3 Exemples

Exemple d'une restitution réussie de l'article en utilisant NEXT :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] NEXT
[S] 223 3000237 <668929@exemple.org> restitué
```

Exemple d'une tentative de restitution d'un article sans avoir d'abord choisi un groupe (via la commande GROUP) :

```
[On suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] NEXT
[S] 412 pas de groupe de nouvelles choisi
```

Exemple d'une tentative de restitution d'un article en utilisant la commande NEXT lorsque le numéro d'article courant est celui du dernier article dans le groupe :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] STAT 3002322
[S] 223 3002322 <411@exemple.net> restitué
[C] NEXT
[S] 421 pas d'article suivant à restituer
```

Exemple d'une tentative de restitution d'un article en utilisant la commande NEXT lorsque le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] NEXT
[S] 420 pas d'article courant choisi
```

6.2 Restitution d'articles et de paragraphes d'articles

Les commandes ARTICLE, BODY, HEAD, et STAT sont très similaires. Elles diffèrent seulement par les parties de l'article qui sont présentées au client et dans le code de réponse de succès. La commande ARTICLE est décrite en entier, tandis que les trois autres commandes sont décrites par leurs différences. Comme spécifié au paragraphe 3.6, un article consiste en deux parties : les en-têtes d'article et le corps de l'article.

Lors de la réponse à une de ces commandes, le serveur DOIT présenter l'article entier ou la partie appropriée et NE DOIT PAS tenter de l'altérer ou le traduire d'aucune façon.

6.2.1 ARTICLE

6.2.1.1 Usage

Indication de capacité : READER

Syntaxe
 identifiant de message ARTICLE
 numéro d'ARTICLE
 ARTICLE

Réponses :

Première forme (identifiant de message spécifié)
 220 0\n identifiant de message : l'article suit (multi lignes)
 430 : pas d'article avec cet identifiant de message

Seconde forme (numéro d'article spécifié)
 220 n identifiant de message : l'article suit (multi lignes)
 412 : pas de groupe de nouvelles choisi

423 : pas d'article avec ce numéro

Troisième forme (numéro d'article courant utilisé)
220 n identifiant de message : l'article suit (multi lignes)
412 : pas de groupe de nouvelles choisi
420 : numéro d'article courant invalide

Paramètres :
numéro : numéro d'article demandé
n : numéro d'article retourné
identifiant de message : identifiant de message de l'article

6.2.1.2 Description

La commande ARTICLE choisit un article conformément aux arguments et présente au client l'article entier (c'est-à-dire, les en-têtes, une ligne vide, et le corps, dans cet ordre). La commande a trois formes.

Dans la première forme, un identifiant de message est spécifié, et le serveur présente l'article avec cet identifiant de message. Dans ce cas, le serveur NE DOIT PAS altérer le groupe de nouvelles actuellement choisi ou le numéro d'article courant. C'est à la fois pour faciliter la présentation des articles qui peuvent être référencés au sein d'un autre article lu, et à cause des difficultés sémantiques de déterminer la séquence appropriée et les appartenances d'un article qui peut avoir fait l'objet d'envois croisés à plus d'un groupe de nouvelles.

Dans la réponse, le numéro d'article DOIT être remplacé par zéro, sauf si il y a un groupe de nouvelles actuellement choisi et si l'article est présent dans ce groupe, et dans ce cas, le serveur PEUT utiliser le numéro de l'article dans ce groupe. (Le serveur n'est pas obligé de déterminer si l'article est dans le groupe de nouvelles actuellement choisi ou, si il l'est, quel numéro d'article il a ; le client DOIT toujours être prêt à ce que zéro soit spécifié.) Le serveur NE DOIT PAS fournir de numéro d'article sauf si l'utilisation de ce numéro dans une seconde commande ARTICLE suivant immédiatement celui-ci retournerait le même article. Même si le serveur choisit de retourner les numéros d'article dans ces circonstances, il n'est pas obligé de le faire systématiquement ; il PEUT retourner zéro à toute commande de cette sorte (voir aussi les exemples de STAT, paragraphe 6.2.4.3).

Dans la seconde forme, un numéro d'article est spécifié. Si il y a un article avec ce numéro dans le groupe de nouvelles actuellement choisi, le serveur DOIT régler le numéro d'article courant à ce numéro.

Dans la troisième forme, l'article indiqué par le numéro d'article courant dans le groupe de nouvelles actuellement choisi est utilisé.

Noter qu'un numéro d'article précédemment valide PEUT devenir invalide si l'article a été supprimé. Un numéro d'article précédemment invalide PEUT devenir valide si l'article a été réinstallé, mais ce numéro d'article NE DOIT PAS être inférieur à la marque inférieure rapportée pour ce groupe.

Le serveur NE DOIT PAS changer le groupe de nouvelles actuellement choisi par suite de cette commande. Le serveur NE DOIT PAS changer le numéro d'article courant excepté quand un argument de numéro d'article a été fourni et que l'article existe ; en particulier, il NE DOIT PAS le changer à la suite d'une réponse d'échec.

Comme l'identifiant de message est unique pour chaque article, il peut être utilisé par un client pour sauter des affichages dupliqués d'articles qui ont été envoyés plus d'une fois, ou à plus d'un groupe de nouvelles.

L'article est retourné comme un bloc de données multi lignes suivant le code de réponse 220.

Si l'argument est un identifiant de message et qu'un tel article n'existe pas, une réponse 430 DOIT être retournée. Si l'argument est un numéro ou est omis et si le groupe de nouvelles actuellement choisi est invalide, une réponse 412 DOIT être retournée. Si l'argument est un numéro et si cet article n'existe pas dans le groupe de nouvelles actuellement choisi, une réponse 423 DOIT être retournée. Si l'argument est omis et si le groupe de nouvelles actuellement choisi est valide mais si le numéro d'article courant est invalide, une réponse 420 DOIT être retournée.

6.2.1.3 Exemples

Exemple de restitution réussie d'un article (n'utilisant explicitement pas un numéro d'article) :

[C] GROUP misc.test

```
[S] 211 1234 3000234 3002322 misc.test
[C] ARTICLE
[S] 220 3000234 <45223423@exemple.com>
[S] Path: pathost!demo!whitehouse!not-for-mail
[S] From: "Demo User" <personne@exemple.net>
[S] Newsgroups: misc.test
[S] Subject: C'est juste un article d'essai
[S] Date: 6 Oct 1998 04:38:40 -0500
[S] Organisation: Un exemple Net, Uncertain, Texas
[S] Message-ID: <45223423@exemple.com>
[S]
[S] C'est juste un article d'essai.
[S] .
```

Exemple de restitution réussie d'un article par l'identifiant de message :

```
[C] ARTICLE <45223423@exemple.com>
[S] 220 0 <45223423@exemple.com>
[S] Path: pathost!demo!whitehouse!not-for-mail
[S] From: "Demo User" <personne@exemple.net>
[S] Newsgroups: misc.test
[S] Subject: C'est juste un article d'essai
[S] Date: 6 Oct 1998 04:38:40 -0500
[S] Organisation: Un exemple Net, Uncertain, Texas
[S] Message-ID: <45223423@exemple.com>
[S]
[S] C'est juste un article d'essai.
[S] .
```

Exemple d'échec de restitution d'un article par l'identifiant de message :

```
[C] ARTICLE <je.ne.suis.pas.là@exemple.com>
[S] 430 Cet article n'a pas été trouve
```

Exemple d'échec de la restitution d'un article par le numéro :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 news.groups
[C] ARTICLE 300256
[S] 423 Pas d'article avec ce numéro
```

Exemple d'échec de la restitution d'un article par le numéro parce qu'aucun groupe de nouvelles n'a été choisi d'abord :

```
[Suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] ARTICLE 300256
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple d'une tentative de restitution d'un article lorsque le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] ARTICLE
[S] 420 Pas d'article courant choisi
```

6.2.2 HEAD

6.2.2.1 Usage

Cette commande est obligatoire.

Syntaxe :

identifiant de message HEAD

numéro HEAD
HEAD

Réponses
Première forme (identifiant de message spécifié)
221 0\n identifiant de message ; les en-têtes suivent (multi lignes)
430 : pas d'article avec cet identifiant de message

Seconde forme (numéro d'article spécifié)
221 n identifiant de message : les en-têtes suivent (multi lignes)
412 : pas de groupe de nouvelles choisi
423 : pas d'article ayant de numéro

Troisième forme (numéro d'article courant utilisé)
221 n identifiant de message: les en-têtes suivent (multi lignes)
412 : pas de groupe de nouvelles choisi
420 : numéro d'article courant invalide

Paramètres :
numéro : numéro d'article demandé
n : numéro d'article retourné
identifiant de message : identifiant de message de l'article

6.2.2.2 Description

La commande HEAD se comporte de façon identique à la commande ARTICLE excepté que, si l'article existe, le code de réponse est 221 au lieu de 220 et seuls les en-têtes sont présentés (la ligne vide qui sépare les en-têtes et le corps NE DOIT PAS être incluse).

6.2.2.3 Exemples

Exemple d'une restitution réussie des en-têtes d'un article (n'utilisant explicitement pas un numéro d'article) :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HEAD
[S] 221 3000234 <45223423@exemple.com>
[S] Path: pathost!demo!whitehouse!not-for-mail
[S] From: "Demo User" <personne@exemple.net>
[S] Newsgroups: misc.test
[S] Subject: c'est juste un article d'essai
[S] Date: 6 Oct 1998 04:38:40 -0500
[S] Organisation: Un exemple Net, Uncertain, Texas
[S] Message-ID: <45223423@exemple.com>
[S] .
```

Exemple d'une restitution réussie des en-têtes d'un article par identifiant de message :

```
[C] HEAD <45223423@exemple.com>
[S] 221 0 <45223423@exemple.com>
[S] Path: pathost!demo!whitehouse!not-for-mail
[S] From: "Demo User" <personne@exemple.net>
[S] Newsgroups: misc.test
[S] Subject: c'est juste un article d'essai0
[S] Date: 6 Oct 1998 04:38:40 -0500
[S] Organisation: Un exemple Net, Uncertain, Texas
[S] Message-ID: <45223423@exemple.com>
[S] .
```

Exemple d'un échec de restitution des en-têtes d'un article par les identifiants de message:

```
[C] HEAD <je.ne.suis.pas.là@exemple.com>
```

[S] 430 Cet article n'a pas été trouvé

Exemple d'un échec de restitution des en-têtes d'un article par numéro :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HEAD 300256
[S] 423 Pas d'article ayant ce numéro
```

Exemple d'échec de restitution des en-têtes d'un article par numéro parce que aucun groupe de nouvelles n'avait d'abord été choisi :

```
[On suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] HEAD 300256
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple d'une tentative de restitution des en-têtes d'un article alors que le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] HEAD
[S] 420 pas d'article courant choisi
```

6.2.3 BODY

6.2.3.1 Usage

Indication de capacité : READER

Syntaxe :
 identifiant de message BODY
 numéro de BODY
 BODY

Réponses

Première forme (identifiant de message spécifié)
 222 0\n identifiant de message : le corps suit (multi lignes)
 430 : pas d'article avec cet identifiant de message

Seconde forme (numéro d'article spécifié)
 222 n identifiant de message : le corps suit (multi lignes)
 412 : pas de groupe de nouvelles choisi
 423 : pas d'article avec ce numéro

Troisième forme (numéro d'article courant utilisé)
 222 n identifiant de message : le corps suit (multi lignes)
 412 : pas de groupe de nouvelles choisi
 420 : numéro d'article courant invalide

Paramètres :

numéro : numéro d'article demandé
 n : numéro d'article retourné
 identifiant de message : identifiant de message de l'article

6.2.3.2 Description

La commande BODY se comporte comme la commande ARTICLE à l'exception que, si l'article existe, le code de réponse est 222 au lieu de 220 et seul le corps est présenté (la ligne vide qui sépare les en-têtes et le corps NE DOIT PAS être incluse).

6.2.3.3 Exemples

Exemple d'une restitution réussie du corps d'un article (n'utilisant explicitement pas de numéro d'article) :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] BODY
[S] 222 3000234 <45223423@exemple.com>
[S] C'est juste un article d'essai.
[S] .
```

Exemple d'une restitution réussie du corps d'un article par identifiant de message :

```
[C] BODY <45223423@exemple.com>
[S] 222 0 <45223423@exemple.com>
[S] C'est juste un article d'essai.
[S] .
```

Exemple d'un échec de restitution du corps d'un article par identifiant de message :

```
[C] BODY <je.ne.suis.pas.là@exemple.com>
[S] 430 Cet article n'a pas été trouvé
```

Exemple d'un échec de restitution du corps d'un article par numéro :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] BODY 300256
[S] 423 Pas d'article avec ce numéro
```

Exemple d'un échec de restitution du corps d'un article par numéro parce que un groupe de nouvelles n'a pas été choisi d'abord :

```
[Suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] BODY 300256
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple d'une tentative de restitution du corps d'un article alors que le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.groupedenouvelles
[S] 211 0 0 0 exemple.vide.groupedenouvelles
[C] BODY
[S] 420 Pas d'article courant choisi
```

6.2.4 STAT

6.2.4.1 Usage

Cette commande est obligatoire.

Syntaxe :

```
identifiant de message STAT
numéro STAT
STAT
```

Réponses

Première forme (identifiant de message spécifié)
 223 0\n identifiant de message : l'article existe
 430 : pas d'article avec cet identifiant de message

Seconde forme (numéro d'article spécifié)
 223 n identifiant de message : l'article existe
 412 : pas de groupe de nouvelles choisi

423 : pas d'article avec ce numéro

Troisième forme (numéro d'article courant utilisé)

223 n identifiant de message : l'article existe

412 : pas de groupe de nouvelles choisi

420 : numéro d'article courant invalide

Paramètres :

numéro : numéro d'article demandé

n : numéro d'article retourné

identifiant de message : identifiant de message de l'article

6.2.4.2 Description

La commande STAT se comporte de la même façon que la commande ARTICLE excepté que, si l'article existe, il N'EST PAS présenté au client et le code de réponse est 223 au lieu de 220. Noter que la réponse N'EST PAS multi lignes.

Cette commande permet au client de déterminer si un article existe et, dans les secondes et troisièmes formes, quel est son identifiant de message, sans avoir à traiter une quantité arbitraire de texte.

6.2.4.3 Exemples

Exemple de STAT sur un article existant (n'utilisant explicitement pas de numéro d'article) :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] STAT
[S] 223 3000234 <45223423@exemple.com>
```

Exemple de STAT sur un article existant par identifiant de message :

```
[C] STAT <45223423@exemple.com>
[S] 223 0 <45223423@exemple.com>
```

Exemple de STAT sur un article qui n'est pas sur le serveur par identifiant de message :

```
[C] STAT <je.ne.suis.pas.là@exemple.com>
[S] 430 Pas d'article avec cet identifiant de message
```

Exemple de STAT sur un article qui n'est pas dans le serveur par numéro :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] STAT 300256
[S] 423 Pas d'article avec ce numéro
```

Exemple de STAT sur un article par numéro lorsque un groupe de nouvelles n'a pas été choisi d'abord :

```
[Suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] STAT 300256
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple de STAT sur un article lorsque le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] STAT
[S] 420 Pas d'article courant choisi
```

Exemple de STAT par identifiant de message sur un serveur qui fait parfois rapport du numéro d'article réel :

```
[C] GROUP misc.test
```



```
[S] 211 1234 3000234 3002322 misc.test
[C] STAT
[S] 223 3000234 <45223423@exemple.com>
[C] STAT <45223423@exemple.com>
[S] 223 0 <45223423@exemple.com>
[C] STAT <45223423@exemple.com>
[S] 223 3000234 <45223423@exemple.com>
[C] GROUP exemple.vide.newsgroup
[S] 211 0 0 0 exemple.vide.newsgroup
[C] STAT <45223423@exemple.com>
[S] 223 0 <45223423@exemple.com>
[C] GROUP alt.crossposts
[S] 211 9999 111111 222222 alt.crossposts
[C] STAT <45223423@exemple.com>
[S] 223 123456 <45223423@exemple.com>
[C] STAT
[S] 223 111111 <23894720@exemple.com>
```

La première commande STAT établit l'identité d'un article dans le groupe. La seconde et la troisième montrent que le serveur peut, sans y être obligé, donner le numéro d'article lorsque l'identifiant de message est spécifié. La quatrième commande STAT montre que zéro doit être spécifié si l'article n'est pas dans le groupe de nouvelles actuellement choisi. La cinquième montre que le numéro, si il est fourni, doit être en relation avec le groupe de nouvelles actuellement choisi. Le dernier montre que le numéro d'article courant n'est toujours pas changé par l'utilisation de STAT avec un identifiant de message même si il retourne un numéro d'article.

6.3 Envoi d'article

L'envoi d'article est fait d'une de deux façons : envoi individuel d'article de clients lecteurs de nouvelles en utilisant POST, et transfert d'article provenant d'autres serveurs de nouvelles en utilisant IHAVE.

6.3.1 POST

6.3.1.1 Usage

Indication de capacité : POST

Cette commande NE DOIT PAS être traitée en parallèle.

Syntaxe : POST

Réponses :

Réponses initiales :

340 : Envoi de l'article à poster

440 : Envoi non permis

Réponses suivantes :

240 : Article bien reçu

441 : Échec de l'envoi

6.3.1.2 Description

Si l'envoi est permis, une réponse 340 DOIT être retournée pour indiquer que l'article à poster devrait être envoyé. Si l'envoi est interdit pour une raison qui dépend de l'installation, une réponse 440 DOIT être retournée.

Si l'envoi est permis, l'article DOIT être dans le format spécifié au paragraphe 3.6 et DOIT être envoyé par le client au serveur comme un bloc de données multi lignes (voir au paragraphe 3.1.1). Donc, un seul point (".") sur une ligne indique la fin du texte, et les lignes qui commencent par un point dans le texte original ont ce point doublé durant la transmission.

À la suite de la présentation de la séquence de terminaison par le client, le serveur DOIT retourner une réponse indiquant le succès ou l'échec du transfert de l'article. Noter que les codes de réponse 340 et 440 sont utilisés dans les réponses directes à la commande POST tandis que les réponses 240 et 441 sont retournées après l'envoi de l'article.

Une réponse de 240 DEVRAIT indiquer que, sauf erreurs imprévues du serveur, l'article envoyé sera rendu disponible sur le serveur et/ou transféré à d'autres serveurs, comme approprié, éventuellement en suivant un autre traitement. En d'autres termes, les articles non voulus par le serveur DEVRAIENT être rejetés avec une réponse 441, plutôt que d'être acceptés et ensuite éliminés en silence. Cependant, le client NE DEVRAIT PAS supposer que l'article a été bien transféré sauf si il reçoit une réponse affirmative du serveur et NE DEVRAIT PAS supposer qu'il est rendu disponible aux autres clients sans le vérifier explicitement (par exemple, en utilisant la commande STAT).

Si la session est interrompue avant la réception de la réponse, il est possible qu'une réponse affirmative ait été envoyée mais ait été perdue. Donc, dans toute session suivante, le client DEVRAIT soit vérifier si l'article a été envoyé avec succès avant de le renvoyer, soit s'assurer que le serveur va allouer le même identifiant de message à la nouvelle tentative (voir l'Appendice A.2). Cette dernière approche est préférée car l'article peut n'avoir pas encore été rendu disponible à la lecture (par exemple, il peut devoir passer par un processus de modération).

6.3.1.3 Exemples

Exemple d'envoi réussi :

```
[C] POST
[S] 340 Entrer l'article ; terminer avec <CR-LF>.<CR-LF>
[C] From: "Demo User" <personne@exemple.net>
[C] Newsgroups: misc.test
[C] Subject: C'est juste un article d'essai
[C] Organisation: Un exemple Net
[C]
[C] C'est juste un article d'essai.
[C] .
[S] 240 Article bien reçu
```

Exemple d'échec de l'envoi :

```
[C] POST
[S] 340 Entrer l'article ; terminer avec <CR-LF>.<CR-LF>
[C] From: "Demo User" <personne@exemple.net>
[C] Newsgroups: misc.test
[C] Subject: C'est juste un article d'essai
[C] Organisation: Un exemple Net
[C]
[C] C'est juste un article d'essai.
[C] .
[S] 441 Échec de l'envoi
```

Exemple de tentative d'envoi alors que l'envoi n'est pas permis :

```
[Établissement de la connexion initiale achevé.]
[S] 201 Service NNTP prêt, envoi interdit
[C] POST
[S] 440 Envoi non permis
```

6.3.2 IHAVE

6.3.2.1 Usage

Indication de capacité : IHAVE

Cette commande NE DOIT PAS être traitée en parallèle.

Syntaxe : IHAVE identifiant de message

Réponses :

Réponses initiales

335 : Envoi de l'article à transférer

435 : Article non voulu

436 : Transfert impossible; réessayer plus tard
 Réponses suivantes
 235 : Article bien transféré
 436 : Échec du transfert, réessayer plus tard
 437 : Transfert rejeté; ne pas réessayer

Paramètres : identifiant de message : identifiant de message de l'article

6.3.2.2 Description

La commande IHAVE informe le serveur que le client a un article avec l'identifiant de message spécifié. Si le serveur désire une copie de cet article, une réponse 335 DOIT être retournée, donnant pour instruction au client d'envoyer l'article entier. Si le serveur ne veut pas l'article (si, par exemple, le serveur en a déjà une copie) une réponse 435 DOIT être retournée, indiquant que l'article n'est pas voulu. Finalement, si l'article n'est pas voulu immédiatement mais que le client devrait réessayer plus tard si possible (si, par exemple, un autre client est en train d'envoyer le même article au serveur) une réponse 436 DOIT être retournée.

Si la transmission de l'article est demandée, le client DOIT envoyer au serveur l'article entier, incluant les en-têtes et le corps, comme bloc de données multi lignes (voir au paragraphe 3.1.1). Donc, un seul point (".") sur une ligne indique la fin du texte, et les lignes commençant par un point dans le texte d'origine ont ce point doublé durant la transmission. Le serveur DOIT retourner une réponse 235, indiquant que l'article a réussi à être transféré ; une réponse 436, indiquant que le transfert a échoué mais devrait être réessayé plus tard ; ou une réponse 437, indiquant que l'article a été rejeté.

Cette fonction diffère de la commande POST en ce qu'elle est destinée à être utilisée au transfert d'articles déjà postés entre des hôtes. Elle NE DEVRAIT PAS être utilisée lorsque le client est un programme personnel de lecture de nouvelles, car l'utilisation de cette commande indique que l'article a déjà été posté sur un autre site et est simplement transmis d'un autre hôte. Cependant, en dépit de cela, le serveur PEUT choisir de ne pas poster ou transmettre l'article si, après un examen plus approfondi de l'article, il estime qu'il n'est pas approprié de le faire. Les raisons pour un tel rejet ultérieur peuvent inclure des problèmes tels que des groupes de nouvelles ou des distributions inappropriées, des limitations d'espace de disque, la longueur de l'article, des en-têtes tronqués, et ainsi de suite. Ce sont les restrictions typiques appliquées par le logiciel de nouvelles de l'hôte serveur et pas nécessairement par le serveur NNTP lui-même.

Le client NE DEVRAIT PAS supposer que l'article a bien été transféré tant qu'il n'a pas reçu une réponse affirmative du serveur. L'absence de réponse (comme une connexion réseau abandonnée ou une fin de temporisation du réseau) DEVRAIT être traitée comme une réponse 436.

Parce que certains nouveaux logiciels de serveur peuvent ne pas être immédiatement capables de déterminer si un article convient pour l'envoi ou la transmission, un serveur NNTP PEUT accuser réception de la réussite du transfert de l'article (avec une réponse 235) mais l'éliminer en silence ultérieurement.

6.3.2.3 Exemples

Exemple d'envoi réussi d'un article à un autre site :

```
[C] IHAVE <je.suis.un.article.que.tu.voudras@exemple.com>
[S] 335 Envoyer ; terminer par <CR-LF>.<CR-LF>
[C] Path: pathost!demo!somewhere!not-for-mail
[C] From: "Demo User" <personne@exemple.com>
[C] Newsgroups: misc.test
[C] Subject: C'est juste un article d'essai
[C] Date: 6 Oct 1998 04:38:40 -0500
[C] Organisation: Un exemple Com, San Jose, CA
[C] Message-ID: <je.suis.un.article.que.tu.voudras@exemple.com>
[C]
[C] C'est juste un article d'essai.
[C] .
[S] 235 Article bien transféré
```

Exemple d'envoi d'un article à un autre site qui le rejette. Noter que l'identifiant de message dans la commande IHAVE n'est pas le même que celui des en-têtes de l'article ; bien que ce soit une mauvaise pratique et NE DEVRAIT PAS être fait, ce n'est pas interdit.

```
[C] IHAVE <je.suis.un.article.que.tu.voudras@exemple.com>
[S] 335 Envoyer ; terminer par <CR-LF>.<CR-LF>
[C] Path: pathost!demo!somewhere!not-for-mail
[C] From: "Demo User" <personne@exemple.com>
[C] Newsgroups: misc.test
[C] Subject: C'est juste un article d'essai
[C] Date: 6 Oct 1998 04:38:40 -0500
[C] Organisation: Un exemple Com, San Jose, CA
[C] Message-ID: <je.suis.un.article.que.tu.as@exemple.com>
[C]
[C] C'est juste un article d'essai.
[C] .
[S] 437 Article rejeté ; ne pas le renvoyer
```

Exemple d'envoi d'un article à un autre site et le transfert échoue :

```
[C] IHAVE <je.suis.un.article.que.tu.voudras@Exemple.com>
[S] 335 Envoyer ; terminer par <CR-LF>.<CR-LF>
[C] Path: pathost!demo!somewhere!not-for-mail
[C] From: "Demo User" <personne@exemple.com>
[C] Newsgroups: misc.test
[C] Subject: C'est juste un article d'essai
[C] Date: 6 Oct 1998 04:38:40 -0500
[C] Organisation: Un exemple Com, San Jose, CA
[C] Message-ID: <je.suis.un.article.que.tu.voudras@Exemple.com>
[C]
[C] C'est juste un article d'essai.
[C] .
[S] 436 Échec du transfert
```

Exemple d'envoi d'un article à un site qui l'a déjà :

```
[C] IHAVE <je.suis.un.article.que.tu.as@Exemple.com>
[S] 435 Dupliqué
```

Exemple d'envoi d'un article à un site qui demande que l'article soit réessayé plus tard :

```
[C] IHAVE <je.suis.un.article.que.tu.diffères@exemple.com>
[S] 436 Réessayer plus tard
```

7. Commandes d'information

Cette section fait la liste des autres commandes qui peuvent être utilisées à tout moment entre le début d'une session et sa fin. L'utilisation de ces commandes n'altère aucune information d'état, mais la réponse générée de leur utilisation peut fournir des informations utiles aux clients.

7.1 DATE

7.1.1 Usage

Indication de capacité : READER

Syntaxe : DATE

Réponse : 111 aaaammjjhhmss : date et heure du serveur

Paramètres : aaaammjjhhmss : date et heure courante en UTC sur le serveur

7.1.2 Description

Cette commande existe pour aider les clients à trouver le temps coordonné universel (UTC) [TF.686-1] du point de vue du serveur. Cette commande NE DEVRAIT PAS être utilisée comme substitut de NTP [RFC1305] mais pour fournir des informations qui peuvent être utiles lorsque on utilise la commande NEWNEWS (voir au paragraphe 7.4).

La commande DATE DOIT retourner un horodatage de la même horloge qu'utilisée pour déterminer les heures d'arrivée de l'article et de création du groupe (voir au paragraphe 6). Cette horloge DEVRAIT être à accroissement monotone, et les ajustements DEVRAIENT être faits en l'accélérant ou en la ralentissant par rapport à l'heure "réelle" plutôt que par des sauts brusques. Un système qui fournit le service NNTP DEVRAIT garder l'horloge système aussi précise que possible, soit avec NTP, soit par une autre méthode.

Le serveur DOIT retourner une réponse 111 spécifiant la date et l'heure sur le serveur sous la forme aaaammjjhhmmss. Cette date et cette heure sont en temps universel coordonné.

7.1.3 Exemples

[C] DATE

[S] 111 19990623135624

7.2 HELP

7.2.1 Usage

Cette commande est obligatoire.

Syntaxe : HELP

Réponse : 100 Un texte d'aide suit (multi lignes)

7.2.2 Description

Cette commande fournit un court résumé des commandes qui sont comprises par cette mise en œuvre de serveur. Le texte d'aide sera présenté comme un bloc de données multi lignes suivant le code de réponse 100.

Il n'est pas garanti que ce texte soit dans un format particulier (mais il doit être en UTF-8) et NE DOIT PAS être utilisé par les clients comme remplacement de la commande CAPABILITIES décrite au paragraphe 5.2.

7.2.3 Exemples

[C] HELP

[S] 100 Le texte d'aide suit

[S] Voici du texte d'aide. Il n'y a pas d'exigence spécifique

[S] de format pour ce texte, bien qu'il soit

[S] courant qu'il fasse la liste des commandes valides

[S] et donne une brève définition de ce qu'elles font.

[S] .

7.3 NEWGROUPS

7.3.1 Usage

Indication de capacité : READER

Syntaxe : Date et heure NEWGROUPS [GMT]

Réponse : 231 Une liste des nouveaux groupes de nouvelles suit (multi lignes)

Paramètres :

date : Date en format aammjj ou aaaammjj

heure : Heure en format hhmmss

7.3.2 Description

Cette commande retourne une liste de groupes de nouvelles créés sur le serveur depuis la date et l'heure spécifiées. Le résultat est dans le même format que la commande LIST ACTIVE (voir au paragraphe 7.6.3). Cependant, elle PEUT inclure des groupes non disponibles sur le serveur (et donc non retournés par LIST ACTIVE) et PEUT omettre des groupes pour lesquels la date de création n'est pas disponible.

La date est spécifiée par 6 ou 8 chiffres dans le format [xx]aammjj, où xx sont les deux premiers chiffres de l'année (19-99), aa sont les deux derniers chiffres de l'année (00-99), mm est le mois (01-12), et jj est le jour du mois (01-31). Les clients DEVRAIENT spécifier les quatre chiffres de l'année. Si les deux premiers chiffres de l'année ne sont pas spécifiés (ceci n'est accepté que pour la rétro compatibilité) l'année est prise comme étant dans le siècle en cours si aa est inférieur ou égal à l'année en cours, et dans le siècle précédent autrement.

L'heure est spécifiée par 6 chiffres dans le format hhmms, où hh est l'heure dans l'horloge de 24 heures (00-23), mm sont les minutes (00-59), et ss sont les secondes (00-60, pour permettre les sauts de secondes). Le jeton "GMT" spécifie que la date et l'heure sont données en temps universel coordonné [TF.686-1] ; si il est omis, la date et l'heure sont alors spécifiés dans la zone horaire locale du serveur. Noter qu'il n'y a pas de moyen d'utiliser le protocole spécifié dans le présent document pour établir la zone horaire locale du serveur.

Noter que une liste vide est une réponse valide possible et indique qu'il n'y a pas de nouveau groupe de nouvelles depuis cette date et heure.

Les clients DEVRAIT faire toutes les interrogations en utilisant le temps universel coordonné (c'est-à-dire, en incluant l'argument "GMT") lorsque possible.

7.3.3 Exemples

Exemple où il y a de nouveaux groupes :

```
[C] NEWGROUPS 19990624 000000 GMT
[S] 231 La liste des nouveaux groupes de nouvelles suit
[S] alt.rfc-writers.recovery 4 1 y
[S] tx.natives.recovery 89 56 y
[S] .
```

Exemple où il n'y a pas de nouveau groupe :

```
[C] NEWGROUPS 19990624 000000 GMT
[S] 231 La liste des nouveaux groupes de nouvelles suit
[S] .
```

7.4 NEWNEWS

7.4.1 Usage

Indication de capacité : NEWNEWS

Syntaxe : NEWNEWS wildmat date heure [GMT]

Réponse : 230 La liste des nouveaux articles suit (multi lignes)

Paramètres :

wildmat : Groupes de nouvelles intéressants
date : Date en format aammjj ou aaaammjj
heure : Heure en format hhmms

7.4.2 Description

Cette commande retourne une liste d'identifiants de message des articles postés ou reçus sur le serveur, dans les groupes de nouvelles dont les noms correspondent au wildmat, depuis la date et l'heure spécifiées. Un identifiant de message est

envoyé sur chaque ligne ; l'ordre de la réponse n'a pas de signification spécifique et peut varier d'une réponse à l'autre dans la même session. Un identifiant de message PEUT apparaître plus d'une fois ; si il le fait, il a la même signification que si il n'apparaissait qu'une fois.

La date et l'heure sont dans le même format que dans la commande NEWGROUPS (voir au paragraphe 7.3).

Noter qu'une liste vide est une réponse valide possible et indique qu'il n'y a actuellement aucune nouvelle nouvelle dans les groupes concernés.

Les clients DEVRAIENT faire toutes les interrogations en temps universel coordonné (c'est-à-dire, en utilisant l'argument "GMT") lorsque possible.

7.4.3 Exemples

Exemple où il y a de nouveaux articles :

```
[C] NEWNEWS news.*,sci.* 19990624 000000 GMT
[S] 230 La liste des nouveaux articles par identifiant de message suit
[S] <je.suis.un.nouvel.article@exemple.com>
[S] <je.suis.un.autre.nouvel.article@exemple.com>
[S] .
```

Exemple où il n'y a pas de nouvel article :

```
[C] NEWNEWS alt.* 19990624 000000 GMT
[S] 230 La liste des nouveaux articles par identifiant de message suit
[S] .
```

7.5 Heure

Comme décrit à la Section 6, chaque article a un horodatage d'arrivée. Chaque groupe de nouvelles a aussi un horodatage de création. Ces horodatages sont utilisés par les commandes NEWNEWS et NEWGROUP pour construire leurs réponses.

Les clients peuvent s'assurer qu'il n'ont pas de trous dans les listes d'articles ou de groupes en utilisant la commande DATE de la manière suivante :

Première session :

Produire la commande DATE et enregistrer le résultat.

Produire une commande NEWNEWS en utilisant un horodatage choisi précédemment.

Sessions suivantes :

Produire la commande DATE et conserver le résultat dans une mémorisation temporaire.

Produire une commande NEWNEWS en utilisant un horodatage sauvegardé d'une session précédente.

Écraser l'horodatage sauvegardé avec celui de la mémorisation temporaire actuelle.

Afin de permettre des erreurs mineures, les clients PEUVENT vouloir ajuster l'horodatage de deux ou trois minutes avant de l'utiliser dans NEWNEWS.

7.5.1 Exemples

Première session :

```
[C] DATE
[S] 111 20010203112233
[C] NEWNEWS local.chat 20001231 235959 GMT
[S] 230 La liste suit
[S] <article.1@local.service>
[S] <article.2@local.service>
[S] <article.3@local.service>
[S] .
```

Seconde session (le client a soustrait 3 minutes de l'horodatage retourné précédemment) :

```
[C] DATE
[S] 111 20010204003344
[C] NEWS local.chat 20010203 111933 GMT
[S] 230 La liste suit
[S] <article.3@local.service>
[S] <article.4@local.service>
[S] <article.5@local.service>
[S] .
```

Noter comment <article.3@local.service> arrivé dans le trou de 3 minutes est mentionné dans les deux réponses.

7.6 Commandes LIST

La famille de commandes LIST retourne des informations c'est-à-dire des réponses multi lignes et dont on peut, en général s'attendre qu'elles ne changent pas durant la session. Souvent, les informations se rapportent aux groupes de nouvelles, et dans ce cas la réponse a une ligne par groupe de nouvelles et un wildmat PEUT être fourni pour restreindre les groupes pour lesquels ces informations sont retournées.

L'ensemble des mots-clés disponibles (incluant ceux fournis par des extensions) est donné dans la liste de capacités avec l'étiquette de capacité LIST.

7.6.1 LIST

7.6.1.1 Usage

Indication de capacité : LIST

Syntaxe : LIST [mot-clé [wildmat|argument]]

Réponse : 215 Des informations suivent (multi lignes)

Paramètres :

mot-clé : Informations demandées [1]

argument : Spécifique du mot-clé

wildmat : Groupes d'intérêt

[1] Si aucun mot-clé n'est fourni, c'est ACTIVE par défaut.

7.6.1.2 Description

La commande LIST permet au serveur de fournir des blocs d'informations au client. Ces informations peuvent être globales ou se rapporter aux groupes de nouvelles ; dans ce cas, les informations peuvent être retournées soit pour tous les groupes, soit seulement pour ceux qui correspondent à un wildmat. Chaque bloc d'informations est représenté par un mot-clé différent. La commande retourne les informations spécifiques identifiées par le mot-clé.

Si les informations sont disponibles, elles sont retournées comme un bloc de données multi lignes suivant le code de réponse 215. Le format des informations dépend du mot-clé. Les informations PEUVENT être affectées par l'argument supplémentaire, mais le format NE DOIT PAS l'être.

Si les informations se fondent sur des groupes de nouvelles et si l'argument wildmat facultatif est spécifié, la réponse est limitée aux seuls groupes (si il en est) dont les noms correspondent et pour lesquels les informations sont disponibles.

Noter qu'une liste vide est une réponse valide possible ; pour un mot-clé fondé sur un groupe de nouvelles, cela indique qu'il n'y a pas de groupe qui satisfait aux critères.

Si le mot-clé n'est pas reconnu, ou si un argument est spécifié et si le mot-clé n'en attend pas, un code de réponse 501 DOIT être retourné. Si le mot-clé est reconnu mais si le serveur ne conserve pas les informations, un code de réponse 503 DOIT être retourné.

La commande LIST NE DOIT changer en aucune façon l'état visible du serveur ; c'est-à-dire, le comportement des

commandes suivantes NE DOIT PAS être affecté par la production ou non de la commande LIST. Par exemple, elle NE DOIT PAS rendre disponibles des groupes qui autrement ne l'auraient pas été.

7.6.1.3 Exemples

Exemple de LIST avec le mot-clé ACTIVE :

```
[C] LIST ACTIVE
[S] 215 La liste des groupes de nouvelles suit
[S] misc.test 3002322 3000234 y
[S] comp.risks 442001 441099 m
[S] alt.rfc-writers.recovery 4 1 y
[S] tx.natives.recovery 89 56 y
[S] tx.natives.recovery.d 11 9 n
[S] .
```

Exemple de LIST sans mot-clé :

```
[C] LIST
[S] 215 La liste des groupes de nouvelles suit
[S] misc.test 3002322 3000234 y
[S] comp.risks 442001 441099 m
[S] alt.rfc-writers.recovery 4 1 y
[S] tx.natives.recovery 89 56 y
[S] tx.natives.recovery.d 11 9 n
[S] .
```

Le résultat est identique à celui de l'exemple précédent .

Exemple de LIST sur un mot-clé fondé sur un groupe de nouvelles avec et sans wildmat :

```
[C] LIST ACTIVE.TIMES
[S] 215 Les informations suivent
[S] misc.test 930445408 <creatme@isc.org>
[S] alt.rfc-writers.recovery 930562309 <m@exemple.com>
[S] tx.natives.recovery 930678923 <sob@academ.com>
[S] .
[C] LIST ACTIVE.TIMES tx.*
[S] 215 Les informations suivent
[S] tx.natives.recovery 930678923 <sob@academ.com>
[S] .
```

Exemple de LIST retournant une erreur où le mot-clé est reconnu mais le logiciel ne conserve pas cette information :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] LIST ACTIVE NEWSGROUPS ACTIVE.TIMES XTRA.DATA
[S] .
[C] LIST XTRA.DATA
[S] 503 Élément de données non mémorisé
```

Exemple de LIST où le mot-clé n'est pas reconnu :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] LIST ACTIVE NEWSGROUPS ACTIVE.TIMES XTRA.DATA
[S] .
[C] LIST DISTRIB.PATS
[S] 501 Erreur de syntaxe
```

7.6.2 Mots clés de LIST standard

La présente spécification définit les mots-clés LIST suivants :

Mot clé	Définition	Statut
ACTIVE	paragraphe 7.6.3	Obligatoire si la capacité READER est annoncée
ACTIVE.TIMES	paragraphe 7.6.4	Facultatif
DISTRIB.PATS	paragraphe 7.6.5	Facultatif
HEADERS	paragraphe 8.6	Obligatoire si la capacité HDR est annoncée
NEWSGROUPS	paragraphe 7.6.6	Obligatoire si la capacité READER est annoncée
OVERVIEW.FMT	paragraphe 8.4	Obligatoire si la capacité OVER est annoncée

Lorsque ces mots-clés LIST sont pris en charge par un serveur, ils DOIVENT avoir la signification du paragraphe pertinent.

7.6.3 LIST ACTIVE

Ce mot-clé DOIT être pris en charge par les serveurs qui annoncent la capacité READER.

LIST ACTIVE retourne une liste de groupes de nouvelles valides et des informations associées. Si aucun wildmat n'est spécifié, le serveur DOIT inclure tout groupe qu'il est permis au client de choisir avec la commande GROUP (paragraphe 6.1.1). Chaque ligne de cette liste consiste en quatre champs séparés les uns des autres par une ou plusieurs espaces :

- o le nom du groupe de nouvelles,
- o la marque supérieure rapportée pour le groupe,
- o la marque inférieure rapportée pour le groupe,
- o le statut actuel du groupe sur ce serveur.

Les marques supérieures et inférieures rapportées sont comme décrit dans la commande GROUP (voir au paragraphe 6.1.1) mais noter qu'elles sont dans l'ordre opposé de la réponse 211 à cette commande.

Le champ Statut est normalement un des suivants :

"y" : L'envoi est permis.

"n" : L'envoi n'est pas permis.

"m" : Les envois seront transmis au modérateur du groupe de nouvelles.

Le serveur DEVRAIT utiliser ces valeurs lorsque ces significations sont exigées et NE DOIT PAS les utiliser avec une autre signification. D'autres valeurs de statut peuvent exister ; la définition de ces autres valeurs et les circonstances dans lesquelles elles sont retournées peuvent être spécifiées dans une extension ou peuvent être privées sur le serveur. Un client DEVRAIT traiter un statut non reconnu comme ne donnant pas d'information.

Le statut d'un groupe de nouvelles indique seulement comment sont normalement traités les envois à ce groupe de nouvelles et n'est pas nécessairement personnalisé pour le client spécifique. Par exemple, si il est interdit au client actuel d'envoyer, cela va alors s'appliquer également aux groupes qui ont le statut "y". À l'inverse, un client avec des privilèges particuliers (non définis par la présente spécification) pourrait être capable d'envoyer à un groupe qui a le statut "n".

Par exemple:

```
[C] LIST ACTIVE
[S] 215 La liste des groupes de nouvelles suit
[S] misc.test 3002322 3000234 y
[S] comp.risks 442001 441099 m
[S] alt.rfc-writers.recovery 4 1 y
[S] tx.natives.recovery 89 56 y
[S] tx.natives.recovery.d 11 9 n
[S] .
```

ou, sur une mise en œuvre qui inclut des zéros en tête :

```
[C] LIST ACTIVE
[S] 215 La liste des groupes de nouvelles suit
[S] misc.test 0003002322 0003000234 y
[S] comp.risks 0000442001 0000441099 m
[S] alt.rfc-writers.recovery 0000000004 0000000001 y
[S] tx.natives.recovery 0000000089 0000000056 y
[S] tx.natives.recovery.d 0000000011 0000000009 n
```

[S] .

Les informations sont fondées sur le groupe de nouvelles, et un wildmat PEUT être spécifié, auquel cas la réponse se limite aux seuls groupes (si il en est) dont le nom correspond au wildmat. Par exemple :

```
[C] LIST ACTIVE *.recovery
[S] 215 La liste des groupes de nouvelles suit
[S] alt.rfc-writers.recovery 4 1 y
[S] tx.natives.recovery 89 56 y
[S] .
```

7.6.4 LIST ACTIVE.TIMES

Ce mot-clé est facultatif.

La liste active.times est entretenue par des serveurs NNTP pour contenir des informations sur qui a créé un groupe de nouvelles particulier et quand. Chaque ligne de cette liste consiste en trois champs séparés l'un de l'autre par une ou plusieurs espaces. Le premier champ est le nom du groupe de nouvelles. Le second est l'heure à laquelle ce groupe a été créé sur ce serveur de nouvelles, mesuré en secondes depuis le début du 1^{er} janvier 1970. Le troisième est du texte destiné à décrire l'entité qui a créé le groupe de nouvelles ; c'est souvent une boîte aux lettres, comme défini dans la [RFC2822]. Par exemple :

```
[C] LIST ACTIVE.TIMES
[S] 215 Les informations suivent
[S] misc.test 930445408 <creatme@isc.org>
[S] alt.rfc-writers.recovery 930562309 <m@example.com>
[S] tx.natives.recovery 930678923 <sob@academ.com>
[S] .
```

La liste PEUT omettre des groupes de nouvelles pour lesquels les informations sont indisponible et PEUT inclure des groupes non disponibles sur le serveur ; en particulier, elle PEUT omettre tous les groupes créés avant la date et l'heure de la plus ancienne entrée. Le client NE DOIT PAS supposer que la liste est complète ou qu'elle correspond à la liste retournée par la commande LIST ACTIVE (paragraphe 7.6.3). La commande NEWGROUPS (paragraphe 7.3) peut fournir un meilleur moyen d'accéder à ces informations, et le résultat des deux commandes DEVRAIT être cohérent sauf que, si cette dernière est invoquée avec une date et une heure antérieure à la plus ancienne entrée dans la liste active.times, son résultat peut inclure des groupes supplémentaires.

Les informations se fondent sur le groupe de nouvelles, et un wildmat PEUT être spécifié, auquel cas la réponse se limite aux seuls groupes (si il en est) dont le nom correspond au wildmat.

7.6.5 LIST DISTRIB.PATS

Ce mot-clé est facultatif.

La liste distrib.pats est entretenue par certains serveurs NNTP pour aider les clients à choisir une valeur pour le contenu de l'en-tête Distribution d'un article de nouvelles envoyé. Chaque ligne de cette liste consiste en trois champs séparés les uns des autres par deux points (":"). Le premier champ est un poids, le second champ est un wildmat (qui peut être un simple nom de groupe de nouvelles) et le troisième champ est une valeur pour le contenu d'en-tête Distribution. Par exemple :

```
[C] LIST DISTRIB.PATS
[S] 215 Les informations suivent :
[S] 10:local.*:local
[S] 5*:monde
[S] 20:local.ici.*:ceSite
[S] .
```

Le client PEUT utiliser ces informations pour construire un en-tête Distribution approprié donné dans le nom d'un groupe de nouvelles. Pour ce faire, il devrait déterminer les lignes dont le second champ correspond au nom du groupe de nouvelles, choisir parmi elles la ligne avec le plus fort poids (0 étant le plus faible) et utiliser la valeur du troisième champ pour construire l'en-tête Distribution.

Les informations ne sont pas fondées sur le groupe de nouvelles, et un argument NE DOIT PAS être spécifié.

7.6.6 LIST NEWSGROUPS

Ce mot-clé DOIT être pris en charge par les serveurs qui annoncent la capacité READER.

La liste des groupes de nouvelles est tenue par les serveurs NNTP pour contenir le nom de chaque groupe de nouvelles qui est disponible sur le serveur et une brève description de l'objet du groupe. Chaque ligne de cette liste consiste en deux champs séparés par un ou plusieurs caractères espace ou (la pratique courante est un seul TAB). Le premier champ est le nom du groupe de nouvelles, et le second est une brève description du groupe. Par exemple:

```
[C] LIST NEWSGROUPS
[S] 215 Les informations suivent :
[S] misc.test Essai général Usenet
[S] alt.rfc-writers.recovery Récupération des auteurs de RFC
[S] tx.natives.recovery Texas Natives Recovery
[S] .
```

La liste PEUT omettre des groupes de nouvelles pour lesquels les informations sont indisponibles et PEUT inclure des groupes non disponibles sur le serveur. Le client NE DOIT PAS supposer que la liste est complète ou qu'elle correspond à la liste retournée par LIST ACTIVE.

La description DEVRAIT être en UTF-8. Cependant, les serveurs obtiennent souvent les informations de sources externes. Ces sources peuvent avoir utilisé des codages différents (qui utilisent les octets dans la gamme 128 à 255 d'une autre manière) et dans ce cas, le serveur PEUT les passer inchangés. Donc, les clients DOIVENT être prêts à recevoir de telles descriptions.

Les informations sont fondées sur le groupe de nouvelles, et un wildmat PEUT être spécifié, auquel cas la réponse se limite aux seuls groupes (si il en est) dont le nom correspond au wildmat.

8. Commandes d'accès aux champs d'articles

Cette section fait la liste des commandes qui peuvent être utilisées pour accéder à des champs d'article spécifiques; c'est-à-dire, des en-têtes d'articles et des métadonnées sur les articles. Ces commandes vont normalement chercher des données dans une "base de données de vue d'ensemble", qui est une base de données d'en-têtes extraits des articles entrants plus les métadonnées déterminées lorsque l'article arrive. Seuls certains champs sont inclus dans la base de données.

Cette section se fonde sur la base de données Overview/NOV [ROBE1995] développée par Geoff Collyer.

8.1 Métadonnées d'article

Les "métadonnées" d'articles sont des données sur des articles qui n'apparaissent pas au sein de l'article lui-même. Chaque élément de métadonnées a un nom qui DOIT commencer par deux points (et qui NE DOIT PAS contenir de deux points ailleurs en son sein). Comme avec les noms d'en-tête, les noms de métadonnées ne sont pas sensibles à la casse.

Lorsque on génère un élément de métadonnées, le serveur DOIT le calculer pour lui-même et NE DOIT PAS s'appuyer sur une valeur qui s'y rapporte fournie dans l'article. (En particulier, un en-tête Lignes ou Octets dans l'article NE DOIT PAS être supposé spécifier le nombre correct de lignes ou d'octets dans l'article.) Si le serveur a accès à plusieurs copies non identiques d'un article, la valeur retournée DOIT être correcte pour toute copie de cet article restituée durant la même session.

La présente spécification définit deux éléments de métadonnées : ":bytes" et ":lines". D'autres éléments de métadonnées peuvent être définis par des extensions. Les noms des éléments de métadonnées définis par des extensions enregistrées NE DOIVENT PAS commencer par ":x-". Pour éviter le risque de collisions avec de futures extensions enregistrées, les noms d'éléments de métadonnées définis par des extensions privées DEVRAIENT commencer par ":x-".

8.1.1 Élément de métadonnées :bytes

L'élément de métadonnées :bytes pour un article est un entier décimal. Il DEVRAIT être égal au nombre d'octets dans l'article entier : en-têtes, corps, et ligne vide de séparation (comptant une paire CRLF comme deux octets, et excluant à la fois le "." CRLF qui termine la réponse et tout "." ajouté pour les besoins de "bourrage de point").

Note pour les mises en œuvre de client : certains serveurs existants retournent une valeur différente de ce qui est mentionné ci-dessus. Les raisons les plus courantes pour cela sont les suivantes :

- o ils comptent une paire CRLF comme un octet,
- o ils incluent le caractère "." utilisé pour le bourrage de point dans le compte,
- o ils incluent le "." CRLF de terminaison dans le compte,
- o ils utilisent une copie d'un article pour compter les octets mais en retournent une autre qui diffère d'une certaine manière (permise).

Les mises en œuvre devraient être prêtes à de telles variations et NE DOIVENT PAS compter que la valeur sera précise.

8.1.2 Élément de métadonnées :lines

L'élément de métadonnées :lines pour un article est un entier décimal. Il DOIT être égal au nombre de lignes dans le corps de l'article (excluant la ligne vide qui sépare en-têtes et corps). De façon équivalente, il fait deux de moins que le nombre de paires CRLF que la commande BODY aurait retournées pour cet article (les deux de plus sont celles qui suivent le code de réponse et l'octet de terminaison).

8.2 Cohérence de la base de données

Les informations mémorisées dans la base de données de vue d'ensemble peuvent changer au fil du temps. Si la base de données enregistre le contenu ou l'absence d'un certain champ (c'est-à-dire, un en-tête ou un élément de métadonnées) pour tous les articles, elle est dite être "cohérente" pour ce champ. Si elle enregistre le contenu d'un en-tête pour certains articles mais pas pour d'autres qui incluent néanmoins cet en-tête, ou si elle enregistre un élément de métadonnées pour certains articles mais pas pour d'autres auxquels cet élément s'applique, elle est dite être "incohérente" pour ce champ.

La commande LIST OVERVIEW.FMT DEVRAIT faire la liste de tous les champs pour lesquels la base de données est cohérente à ce moment. Elle PEUT omettre de tels champs (par exemple, si on ne sait pas si la base de données est cohérente ou pas). Elle NE DOIT PAS inclure de champs pour lesquels la base de données est incohérente ou qui ne sont pas mémorisés dans la base de données. Donc, si un en-tête apparaît dans le résultat de LIST OVERVIEW.FMT mais pas dans le résultat de OVER pour un certain article, cet en-tête n'apparaît pas dans l'article (de même pour les éléments de métadonnées).

Ces règles supposent que les champs mémorisés dans la base de données restent constants pour de longues périodes, et donc la base de données sera cohérente. Lorsque l'ensemble de champs à mémoriser est changé, elle sera incohérente jusqu'à ce que la base de données soit reconstruite ou que les seuls articles restants soient ceux reçus depuis le changement. Donc, le résultat de LIST OVERVIEW.FMT doit être altéré deux fois. D'abord, avant que tous les champs arrêtent d'être mémorisés ils DOIVENT être retirés du résultat ; puis, lorsque la base de données est de nouveau connue pour être cohérente, les nouveaux champs DEVRAIENT être ajoutés au résultat.

Si la commande HDR utilise la base de données de vue d'ensemble plutôt que de prendre directement les informations des articles, les mêmes problèmes de cohérence et d'incohérence s'appliquent, et la commande LIST HEADERS DEVRAIT prendre la même approche que la commande LIST OVERVIEW.FMT pour les résoudre.

8.3 OVER

8.3.1 Usage

Indication de capacité : OVER

Syntaxe :

OVER identifiant de message

OVER gamme

OVER

Réponses :

Première forme (identifiant de message spécifié)

224 : les informations de vue d'ensemble suivent (multi lignes)

430 : pas d'article avec cet identifiant de message

Seconde forme (gamme spécifiée)

224 : les informations de vue d'ensemble suivent (multi lignes)

412 : pas de groupe de nouvelles choisi

423 : pas d'article dans cette gamme

Troisième forme (numéro d'article courant utilisé)

224 : les informations de vue d'ensemble suivent (multi lignes)

412 : pas de groupe de nouvelles choisi

420 : le numéro d'article courant est invalide

Paramètres :

gamme : numéros des articles

identifiant de message : identifiant de message de l'article

8.3.2 Description

La commande OVER retourne le contenu de tous les champs dans la base de données pour un article spécifié par l'identifiant de message, ou d'un article ou gamme d'articles spécifié dans le groupe de nouvelles actuellement choisi.

L'argument identifiant de message indique un article spécifique. L'argument gamme peut avoir une des valeurs suivantes :

- o un numéro d'article,
- o un numéro d'article suivi d'un tiret pour indiquer tout ce qui suit,
- o un numéro d'article suivi par un tiret suivi par un autre numéro d'article.

Si aucun n'est spécifié, le numéro d'article courant est utilisé.

La prise en charge de la première forme (identifiant de message) est facultative. Si elle est prise en charge, la ligne de capacité OVER DOIT inclure l'argument "MSGID". Autrement, la ligne de capacité NE DOIT PAS inclure cet argument, et la commande OVER DOIT retourner le code de réponse générique 503 lorsque cette forme est utilisée.

Si les informations sont disponibles, elles sont retournées comme un bloc de données multi lignes suivant le code de réponse 224 et contenant une ligne par article, triés dans l'ordre numérique des numéros d'article. (Noter que sauf si l'argument est une gamme incluant un tiret, il y aura exactement une ligne dans le bloc de données.) Chaque ligne consiste en un certain nombre de champs séparés par une tabulation. Un champ peut être vide (auquel cas il y aura deux tabulations adjacentes) et une séquence de tabulations en queue peut être omise.

Les huit premiers champs DOIVENT être les suivants, dans l'ordre :

"0" ou numéro d'article (voir ci-dessous)

contenu de l'en-tête Subject

contenu de l'en-tête From

contenu de l'en-tête Date

contenu de l'en-tête Message-ID

contenu de l'en-tête References

élément de métadonnées :bytes

élément de métadonnées :lines

Si l'article est spécifié par l'identifiant de message (la première forme de la commande) le numéro d'article DOIT être remplacé par zéro, sauf si il y a un groupe de nouvelles actuellement choisi et si l'article est présent dans ce groupe, le serveur PEUT utiliser le numéro de l'article dans ce groupe. (Voir la commande ARTICLE (paragraphe 6.2.1) et les exemples de STAT (paragraphe 6.2.4.3) pour les détails.) Dans les deux autres formes de la commande, le numéro d'article DOIT être retourné.

Tous les champs suivants sont les contenus des autres en-têtes et les métadonnées détenues dans la base de données.

Pour les cinq en-têtes obligatoires, le contenu de chaque champ DOIT se fonder sur le contenu de l'en-tête (c'est-à-dire, avec le nom d'en-tête et les deux points et l'espace suivants retirés). Si l'article ne contient pas cet en-tête, ou si le contenu est vide, le champ DOIT être vide. Pour les deux éléments de métadonnées obligatoires, le contenu du champ DOIT être juste la valeur, sans autre texte.

Pour tous les champs suivants qui contiennent des en-têtes, le contenu DOIT être la ligne d'en-tête entière sauf le CRLF de queue. Pour tous les champs suivants qui contiennent des métadonnées, le champ consiste en le nom des métadonnées, une seule espace, et ensuite la valeur.

Pour tous les champs, la valeur est traitée en retirant d'abord toutes les paires CRLF (c'est-à-dire, en supprimant tous les sauts à la ligne et en supprimant le CRLF de terminaison) et ensuite en remplaçant chaque tabulation par une seule espace.

Si il n'y a pas un tel en-tête dans l'article, aucun de ces éléments de métadonnées, ou pas d'en-tête ou d'élément mémorisé dans la base de données pour cet article, le champ correspondant DOIT être vide.

Noter que, après la suppression des sauts à la ligne, les caractères NUL, LF, et CR ne peuvent plus survenir dans l'en-tête d'un article offert par un serveur conforme. Néanmoins, les serveurs DEVRAIENT vérifier ces caractères et remplacer chacun d'eux par une seule espace (afin que, par exemple, CR LF LF TAB deviennent deux espaces, car le CR et le premier LF seront supprimés par le processus de suppression de saut à la ligne). Cela va renforcer la robustesse face à des données non conformes ; il est aussi possible que de futures versions de la présente spécification permettent que ces caractères apparaissent dans les articles.

Le serveur NE DEVRAIT PAS produire de résultat pour des articles qui n'existent plus.

Si l'argument est un identifiant de message et qu'un tel article n'existe pas, une réponse 430 DOIT être retournée. Si l'argument est une gamme ou est omis et si le groupe de nouvelles actuellement choisi est invalide, une réponse 412 DOIT être retournée. Si l'argument est une gamme et si aucun article n'existe dans cette gamme de nombres dans le groupe de nouvelles actuellement choisi, incluant le cas où le second nombre est inférieur au premier, une réponse 423 DOIT être retournée. Si l'argument est omis et si le groupe de nouvelles actuellement choisi est valide mais si le numéro d'article courant est invalide, une réponse 420 DOIT être retournée.

8.3.3 Exemples

Dans les quatre premiers exemples, TAB a été remplacé par une barre verticale et certaines lignes ont été coupées pour améliorer la lisibilité.

Exemple d'une restitution réussie d'informations de vue d'ensemble pour un article (n'utilisant explicitement pas de numéro d'article) :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] OVER
[S] 224 Les informations de vue d'ensemble suivent
[S] 3000234|C'est juste un article d'essai|"Demo User" <personne@exemple.com>|6 Oct 1998 04:38:40 -0500|
    <45223423@exemple.com>|<45454@exemple.net>|1234|17|Xref: news.exemple.com misc.test:3000363
[S] .
```

Exemple d'une restitution réussie d'informations de vue d'ensemble pour un article par identifiant de message :

```
[C] CAPABILITIES
[S] 101 Liste des capacités :
[S] VERSION 2
[S] READER
[S] OVER MSGID
[S] LIST ACTIVE NEWSGROUPS OVERVIEW.FMT
[S] .
[C] OVER <45223423@exemple.com>
[S] 224 Les informations de vue d'ensemble suivent
[S] 0|C'est juste un article d'essai|"Demo User" <personne@exemple.com>|6 Oct 1998 04:38:40 -0500|
    <45223423@exemple.com>|<45454@exemple.net>|1234|17|Xref: news.exemple.com misc.test:3000363
[S] .
```

Noter que le numéro d'article a été remplacé par "0".

Exemple des mêmes commandes sur un système qui ne met pas en œuvre la restitution par identifiant de message :

```
[C] CAPABILITIES
[S] 101 Liste des capacités :
[S] VERSION 2
[S] READER
[S] OVER
[S] LIST ACTIVE NEWSGROUPS OVERVIEW.FMT
[S] .
[C] OVER <45223423@exemple.com>
```

[S] 503 Vue d'ensemble par identifiant de message non prise en charge

Exemple d'une restitution réussie d'informations de vue d'ensemble pour une gamme d'articles :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] OVER 3000234-3000240
[S] 224 Les informations de vue d'ensemble suivent
[S] 3000234|C'est juste un article d'essai|"Demo User"<personne@exemple.com>|6 Oct 1998 04:38:40 -0500|
    <45223423@exemple.com>|<45454@exemple.net>|1234|17|Xref: news.exemple.com misc.test:3000363
[S] 3000235|Un autre article d'essai|personne@nullepart.to(Demo User)|6 Oct 1998 04:38:45 -0500|<45223425@to.to>||
    4818|37||Distribution: fi
[S] 3000238|Re: C'est juste un article d'essai|quelqu'un@ailleurs.to|7 Oct 1998 11:38:40 +1200|<kfwer3v@ailleurs.to>|
    <45223423@to.to>|9234|51
[S] .
```

Noter les en-têtes "References" et Xref manquants dans la seconde ligne, les champs en queue manquants dans la première et la dernière ligne, et qu'il n'y a de résultat que pour les articles qui existent toujours.

Exemple d'un échec de restitution d'informations de vue d'ensemble sur un article par numéro :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] OVER 300256
[S] 423 Pas cet article dans ce groupe
```

Exemple d'une gamme invalide :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] OVER 3000444-3000222
[S] 423 Gamme vide
```

Exemple d'un échec de restitution d'informations de vue d'ensemble par numéro parce que aucun groupe de nouvelles n'a été choisi d'abord :

```
[On suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] OVER
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple d'une tentative de restitution d'informations lorsque le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.groupedenouvelles
[S] 211 0 0 0 exemple.vide.groupedenouvelles
[C] OVER
[S] 420 Pas d'article courant choisi
```

8.4 LIST OVERVIEW.FMT

8.4.1 Usage

Indication de capacité : OVER

Syntaxe : LIST OVERVIEW.FMT

Réponse : 215 Les informations suivent (multi lignes)

8.4.2 Description

Voir au paragraphe 7.6.1 les exigences générales de la commande LIST.

La commande LIST OVERVIEW.FMT retourne une description des champs de la base de données pour lesquels elle est cohérente (comme décrit ci-dessus). Les informations sont retournées comme bloc de données multi lignes à la suite du code de réponse 215. Les informations contiennent une ligne par champ dans l'ordre dans lequel elles sont retournées par la commande OVER ; les sept premières lignes DOIVENT (sauf pour la casse des lettres) être exactement comme suit :

Subject:
From:
Date:
Message-ID:
References:
:bytes
:lines

Pour la compatibilité avec les mises en œuvre existantes, les deux dernières lignes PEUVENT être :

Bytes:
Lines:
même si elle se rapportent aux métadonnées, et non aux en-têtes.

Toutes les lignes suivantes DOIVENT consister en un nom d'en-tête suivi par ":full", ou par le nom d'un élément de métadonnées.

Il n'y a pas d'espaces en tête ou en queue dans le résultat.

Noter que les sept lignes fixées décrivent les champs du second au huitième du résultat de OVER. Le suffixe "full" (qui peut être en majuscules, minuscules, ou un mélange des deux) est un rappel que les champs correspondants comportent le nom d'en-tête.

Cette commande PEUT générer des résultats différents si elle est utilisée plus d'une fois dans une session.

Si la commande OVER n'est pas mise en œuvre, la signification du résultat de cette commande n'est pas spécifiée, mais elle doit quand même satisfaire aux exigences syntaxiques ci-dessus.

8.4.3 Exemples

Exemple de résultat LIST OVERVIEW.FMT correspondant au résultat de l'exemple OVER ci-dessus, dans le format préféré :

```
[C] LIST OVERVIEW.FMT
[S] 215 Ordre des champs dans la base de données de vue d'ensemble.
[S] Subject:
[S] From:
[S] Date:
[S] Message-ID:
[S] References:
[S] :bytes
[S] :lines
[S] Xref:full
[S] Distribution:full
[S] .
```

Exemple de résultat de LIST OVERVIEW.FMT correspondant au résultat de l'exemple OVER ci-dessus, dans le format de remplacement :

```
[C] LIST OVERVIEW.FMT
[S] 215 Ordre des champs dans la base de données de vue d'ensemble.
[S] Subject:
[S] From:
[S] Date:
[S] Message-ID:
[S] References:
[S] Bytes:
[S] Lines:
[S] Xref:FULL
```

[S] Distribution:FULL

[S] .

8.5 HDR

8.5.1 Usage

Indication de capacité : HDR

Syntaxe :

identifiant de message de champ HDR

gamme de champ HDR

champ HDR

Réponses :

Première forme (identifiant de message spécifié)

225 : Les en-têtes suivent (multi lignes)

430 : Pas d'article avec cet identifiant de message

Seconde forme (gamme spécifiée)

225 : Les en-têtes suivent (multi lignes)

412 : Pas de groupe de nouvelles choisi

423 : Pas d'article dans cette gamme

Troisième forme (numéro d'article courant utilisé)

225 : Les en-têtes suivent (multi lignes)

412 : Pas de groupe de nouvelles choisi

420 : Numéro d'article courant invalide

Paramètres :

champ : Nom du champ

gamme : Numéros des articles

identifiant de message : Identifiant de message de l'article

8.5.2 Description

La commande HDR donne accès aux champs spécifiques d'un article spécifié par identifiant de message, ou à partir d'un article ou gamme d'articles spécifié dans le groupe de nouvelles actuellement choisi. Elle PEUT prendre les informations directement des articles ou de la base de données de vue d'ensemble. Dans le cas des en-têtes, une mise en œuvre PEUT restreindre l'utilisation de cette commande à une liste spécifique d'en-têtes ou PEUT lui permettre d'être utilisée avec tout en-tête ; elle peut se comporter différemment lorsque elle est utilisée avec un argument d'identifiant de message et quand elle est utilisée avec une gamme d'arguments ou pas d'argument du tout.

L'argument de champ requis est le nom d'un en-tête dont les deux points sont omis (par exemple, "subject") ou le nom d'un élément de métadonnées incluant les deux points en tête (par exemple, ":bytes") et est insensible à la casse.

L'argument identifiant de message indique un article spécifique. L'argument gamme peut être un des suivants :

- o un numéro d'article,
- o un numéro d'article suivi par un tiret pour indiquer tout ce qui suit,
- o un numéro d'article suivi par un tiret suivi par un autre numéro d'article.

Si aucun n'est spécifié, le numéro d'article courant est utilisé.

Si l'information est disponible, elle est retournée comme bloc de données multi lignes qui suit le code de réponse 225 et contient une ligne pour chaque article qui existe dans la gamme. (Noter que sauf si l'argument est une gamme incluant un tiret, il y aura exactement une ligne dans le bloc de données.) La ligne consiste en le numéro d'article, une espace, et ensuite le contenu du champ. Dans le cas d'un en-tête, le nom d'en-tête, les deux points, et la première espace après les deux points sont tous omis.

Si l'article est spécifié par identifiant de message (la première forme de la commande) le numéro d'article DOIT être remplacé par zéro, sauf si c'est un groupe de nouvelles actuellement choisi et si l'article est présent dans ce groupe, le serveur PEUT utiliser le numéro de l'article dans ce groupe. (Pour plus de détails, voir la commande ARTICLE (paragraphe 6.2.1) et les exemples de STAT (paragraphe 6.2.4.3).) Dans les deux autres formes de la commande, le numéro

d'article DOIT être retourné.

Les contenus d'en-tête sont modifiés comme suit : toutes les paires CRLF sont retirées, et ensuite chaque TAB est remplacée par une seule espace. (Noter que ceci est la même transformation qu'effectuée par la commande OVER (paragraphe 8.3.2) et le même commentaire concernant NUL, CR, et LF s'applique.)

Noter la distinction entre en-têtes et métadonnées qui paraissent avoir la même signification. Les en-têtes sont toujours pris inchangés dans l'article ; les métadonnées sont toujours calculées. Par exemple, une demande pour "Lines" retourne le contenu de l'en-tête "Lines" des articles spécifiés, si il en est, peu importe qu'ils déclarent de façon précise le nombre de lignes, tandis qu'une demande ".lines" retourne les métadonnées de compte de lignes, qui est toujours le nombre réel de lignes sans considération de ce que peut déclarer tout en-tête.

Si l'en-tête demandé n'est pas présent dans l'article, ou si il est présent mais vide, une ligne pour cet article est incluse dans le résultat, mais le contenu de la portion d'en-tête de la ligne est vide (l'espace après le numéro d'article PEUT être conservée ou omise). Si l'en-tête survient plus d'une fois dans un certain article, seul le contenu de la première occurrence est retourné par HDR. Si un numéro d'article dans la gamme fournie n'existe pas dans le groupe, aucune ligne n'est incluse dans le résultat pour ce numéro d'article.

Si le second argument est un identifiant de message et qu'il n'existe pas de tel article, une réponse 430 DOIT être retournée. Si le second argument est une gamme ou est omis et si le groupe de nouvelles actuellement choisi est invalide, une réponse 412 DOIT être retournée. Si le second argument est une gamme et si aucun article n'existe dans cette gamme de numéros dans le groupe de nouvelles actuellement choisi, incluant le cas où le second numéro est inférieur au premier, une réponse 423 DOIT être retournée. Si l'argument est omis et si le groupe de nouvelles actuellement choisi est valide mais le numéro d'article courant est invalide, une réponse 420 DOIT être retournée.

Un serveur PEUT seulement permettre les commandes HDR pour un ensemble de champs limité ; il peut se comporter différemment à cet égard pour la première forme (identifiant de message) de ce qu'il ferait pour les autres formes. Si c'est ainsi, il DOIT répondre avec la réponse générique 503 aux tentatives de demande d'autres champs, plutôt que retourner des résultats erronés, comme une réponse de succès vide.

Si HDR utilise la base de données de vue d'ensemble et si elle est incohérente pour le champ demandé, le serveur PEUT retourner les résultats qu'il peut, ou il PEUT répondre par la réponse générique 503. Dans ce cas, le champ NE DOIT PAS apparaître dans le résultat de LIST HEADERS.

8.5.3 Exemples

Exemple d'une restitution réussie de lignes de Subject d'une gamme d'articles (3000235 n'a pas d'en-tête Subject, et 3000236 manque) :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HDR Subject 3000234-3000238
[S] 225 Les en-têtes suivent
[S] 3000234 C'est juste un article d'essai
[S] 3000235
[S] 3000237 Re: C'est juste un article d'essai
[S] 3000238 Ditto
[S] .
```

Exemple d'une restitution réussie de comptes de lignes à partir d'une gamme d'articles :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HDR :lines 3000234-3000238
[S] 225 Les en-têtes suivent
[S] 3000234 42
[S] 3000235 5
[S] 3000237 11
[S] 3000238 2378
[S] .
```

Exemple d'une restitution réussie de la ligne Subject d'un article par identifiant de message :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HDR subject <je.suis.un.article.d'essai@exemple.com>
[S] 225 Les informations d'en-tête suivent
[S] 0 C'est juste un article d'essai
[S] .
```

Exemple d'une restitution réussie de la ligne Subject de l'article courant :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HDR subject
[S] 225 Les informations d'en-tête suivent
[S] 3000234 C'est juste un article d'essai
[S] .
```

Exemple d'un échec de restitution d'un en-tête d'un article par identifiant de message :

```
[C] HDR subject <je.ne.suis.pas.là@exemple.com>
[S] 430 Cet article n'a pas été trouvé
```

Exemple d'un échec de restitution d'en-têtes d'articles par numéro parce qu'aucun groupe de nouvelles n'a été choisi :

```
[Suppose que le groupe de nouvelles actuellement choisi est invalide.]
[C] HDR subject 300256-
[S] 412 Pas de groupe de nouvelles choisi
```

Exemple d'un échec de restitution d'en-tête parce que le groupe de nouvelles actuellement choisi est vide :

```
[C] GROUP exemple.vide.groupedenouvelles
[S] 211 0 0 0 exemple.vide.groupedenouvelles
[C] HDR subject 1-
[S] 423 Pas d'article dans cette gamme
```

Exemple d'un échec de restitution d'en-têtes parce que le serveur ne permet pas les commandes HDR pour cet en-tête :

```
[C] GROUP misc.test
[S] 211 1234 3000234 3002322 misc.test
[C] HDR Content-Type 3000234-3000238
[S] 503 HDR pas permis sur ce type de contenu
```

8.6 LIST HEADERS

8.6.1 Usage

Indication de capacité : HDR

Syntaxe : LIST HEADERS [MSGID|RANGE]

Réponse : 215 La liste des champs suit (multi lignes)

Paramètres :

MSGID : demande la liste pour l'accès par identifiant de message

RANGE : demande la liste pour l'accès par gamme

8.6.2 Description

Voir au paragraphe 7.6.1 les exigences générales de la commande LIST.

La commande LIST HEADERS retourne une liste de champs qui peuvent être restitués en utilisant la commande HDR.

Les informations sont retournées comme un bloc de données multi lignes à la suite du code de réponse 215 et contient une ligne pour chaque nom de champ (excluant les deux points en queue pour les en-têtes et incluant les deux points en tête pour les éléments de métadonnées). Si la mise en œuvre permet qu'un en-tête soit restitué, il NE DOIT PAS inclure de nom d'en-tête dans la liste mais DOIT inclure l'entrée spéciale ":" (un seul deux points tout seul). Il DOIT quand même faire la liste explicite de tous les éléments de métadonnées qui sont disponibles. L'ordre des éléments dans la liste n'a pas de signification ; le serveur n'a même pas besoin de retourner dans le même ordre. La liste PEUT être vide (bien que dans ce cas il y ait peu de sens à fournir une commande HDR).

Une mise en œuvre qui prend aussi en charge la commande OVER DEVRAIT au moins permettre tous les en-têtes et éléments de métadonnées mentionnés dans le résultat de la commande LIST OVERVIEW.FMT.

Si le serveur traite la première forme de la commande HDR (identifiant de message spécifié) différemment des deux autres formes (gamme spécifiée ou numéro d'article courant utilisé) par rapport aux en-têtes ou éléments de métadonnées qui sont disponibles, on applique alors ce qui suit :

- o si l'argument MSGID est spécifié, le résultat DOIT être celui disponible pour la première forme de la commande HDR,
- o si l'argument RANGE est spécifié, le résultat DOIT être celui disponible pour les secondes et troisièmes formes de la commande HDR,
- o si aucun argument n'est spécifié, le résultat DOIT être celui disponible dans toutes les formes de la commande HDR (c'est-à-dire, il DOIT seulement faire la liste des éléments mentionnés dans les deux cas précédents).

Si le serveur ne traite pas les diverses formes différemment, il DOIT alors ignorer tout argument et toujours produire le même résultat (bien que pas nécessairement toujours dans le même ordre).

Si la commande HDR n'est pas mise en œuvre, la signification du résultat de cette commande n'est pas spécifié, mais il doit néanmoins satisfaire les exigences syntaxiques ci-dessus.

8.6.3 Exemples

Exemple d'une mise en œuvre fournissant l'accès à seulement quelques en-têtes :

```
[C] LIST HEADERS
[S] 215 en-têtes pris en charge :
[S] Subject
[S] Message-ID
[S] Xref
[S] .
```

Exemple d'une mise en œuvre fournissant l'accès aux mêmes champs que le premier exemple du paragraphe 8.4.3 :

```
[C] CAPABILITIES
[S] 101 Liste de capacités :
[S] VERSION 2
[S] READER
[S] OVER
[S] HDR
[S] LIST ACTIVE NEWSGROUPS HEADERS OVERVIEW.FMT
[S] .
[C] LIST HEADERS
[S] 215 en-têtes et éléments de métadonnées pris en charge :
[S] Date
[S] Distribution
[S] From
[S] Message-ID
[S] References
[S] Subject
[S] Xref
[S] :bytes
[S] :lines
[S] .
```

Exemple d'une mise en œuvre fournissant l'accès à tous les en-têtes :

[C] LIST HEADERS

[S] 215 éléments de métadonnées pris en charge :

[S] :

[S] :lines

[S] :bytes

[S] :x-article-number

[S] .

Exemple d'une mise en œuvre qui distingue la première forme de la commande HDR des deux autres formes :

[C] LIST HEADERS RANGE

[S] 215 éléments de métadonnées pris en charge :

[S] :

[S] :lines

[S] :bytes

[S] .

[C] LIST HEADERS MSGID

[S] 215 en-têtes et éléments de métadonnées pris en charge :

[S] Date

[S] Distribution

[S] From

[S] Message-ID

[S] References

[S] Subject

[S] :lines

[S] :bytes

[S] :x-article-number

[S] .

[C] LIST HEADERS

[S] 215 en-têtes et éléments de métadonnées pris en charge :

[S] Date

[S] Distribution

[S] From

[S] Message-ID

[S] References

[S] Subject

[S] :lines

[S] :bytes

[S] .

Noter que :x-article-number n'apparaît pas dans le dernier ensemble de résultats.

9. Syntaxe en BNF augmenté pour NNTP

9.1 Introduction

Chacun des paragraphes suivants décrit la syntaxe d'un élément majeur de NNTP. Cette syntaxe étend et précise les descriptions données ailleurs dans la présente spécification et devrait avoir la préséance lors de la résolution de conflits apparents. Noter que les chaînes ABNF [RFC4234] sont insensibles à la casse. Les non terminaux utilisés en plusieurs endroits sont définis dans un paragraphe distinct à la fin.

Entre eux, les non terminaux <commande-ligne>, <commande-datastream>, <commande-continuation>, et <réponse> spécifient le texte qui s'écoule entre clients et serveurs. Une manière cohérente de schéma de désignation est utilisée dans le présent document pour les non terminaux qui se rapportent à chaque commande, et DEVRAIT être utilisée par la spécification des extensions enregistrées.

Pour chaque commande, la séquence est la suivante :

- o Le client envoie une instance de <commande-ligne> ; la syntaxe pour la commande EXEMPLE est <exemple-commande>.
- o Si le client est un de ceux qui envoient immédiatement des données, il envoie une instance de <commande-datastream> ; la syntaxe pour la commande EXEMPLE est <exemple-datastream>.

- o Le serveur envoie une instance de <réponse>.
 - * La ligne de réponse initiale est indépendante de la commande qui la génère ; si la réponse 000 a des arguments, la syntaxe de la ligne initiale est <réponse-000-contenu>.
 - * Si la réponse est multi lignes, la ligne initiale est suivie par un <bloc-de-données-multi-lignes>. La syntaxe pour le contenu de ce bloc après que le "bourrage de point" a été retiré est (pour la réponse 000 à la commande EXEMPLE) <exemple-000-ml-contenu> et est une instance de <contenu-réponse-multi-lignes>.
 - o Lorsque la dernière réponse indique que plus de données sont requises (en général, une réponse 3xx) :
 - * le client envoie une instance de <commande-continuation> ; la syntaxe pour la continuation de EXEMPLE suivant une réponse 333 est <exemple-333-continuation> ;
 - * le serveur envoie une autre instance de <réponse>, comme ci-dessus.
- (Il n'y a pas de commande dans la présente spécification qui écoule immédiatement les données, mais ce non terminal est défini pour les besoins des extensions.)

9.2 Commandes

Cette syntaxe définit la <ligne-de-commande> non terminale, qui représente ce qui est envoyé du client au serveur (voir au paragraphe 3.1 les limites sur les longueurs).

ligne-de-commande = commande EOL

commande = X-commande

X-commande = mot-clé *(WS jeton)

commande = / article-commande / corps-commande / capacités-commande / date-commande / groupe-commande / hdr-commande / head-commande / help-commande / ihave-commande / dernière-commande / liste-commande / liste-groupe-commande / mode-reader-commande / newgroups-commande / newnews-commande / prochaine-commande / over-commande / post-commande / quit-commande / stat-commande

article-commande = "ARTICLE" [WS article-ref] ; (WS = espace blanche)

corps-commande = "BODY" [WS article-ref]

capacités-commande = "CAPABILITIES" [WS mot-clé]

date-commande = "DATE"

groupe-commande = "GROUP" WS nom-de-groupe-de-nouvelles

hdr-commande = "HDR" WS nom-métadonnées-en-tête [WS gamme-ref]

head-commande = "HEAD" [WS article-ref] help-commande = "HELP"

ihave-commande = "IHAVE" WS identifiant-de-message

dernière-commande = "LAST"

liste-commande = "LIST" [WS liste-arguments]

liste-groupe-commande = "LISTGROUP" [WS nom-de-groupe-de-nouvelles [WS gamme]]

mode-reader-commande = "MODE" WS "READER"

newgroups-commande = "NEWGROUPS" WS date-heure

newnews-commande = "NEWNEWS" WS wildmat WS date-heure

prochaine-commande = "NEXT"

over-commande = "OVER" [WS gamme-ref]

post-commande = "POST"

quit-commande = "QUIT"

stat-commande = "STAT" [WS article-ref]

article-ref = numéro-article / identifiant-de-message

date = date2y / date4y

date4y = 4DIGIT 2DIGIT 2DIGIT

date2y = 2DIGIT 2DIGIT 2DIGIT

date-heure = date WS heure [WS "GMT"]

header-meta-name = nom-en-tête / nom-métadonnées

liste-arguments = mot-clé [WS jeton]

nom-métadonnées = ":" 1*A-NOTCOLON

gamme = numéro-article ["-" [numéro-article]]

gamme-ref = gamme / identifiant-de-message

heure = 2DIGIT 2DIGIT 2DIGIT

9.3 Continuation de commande

Cette syntaxe définit la suite du matériel envoyé par le client dans le cas de commandes multi-étapes et celles qui écoulent les données.

commande-datastream = INDÉFINIE ; non utilisée, fournie pour accrocher des extensions
 commande-continuation = ihave-335-continuation / post-340-continuation
 ihave-335-continuation = article-codé
 post-340-continuation = article-codé
 article-codé = bloc-de-données-multi-lignes ; après avoir supprimé le "bourrage de point", DOIT correspondre à <article>

9.4 Réponses

9.4.1 Réponses génériques

Cette syntaxe définit le non terminal <réponse>, qui représente la forme générique des réponses ; c'est-à-dire, ce qui est envoyé du serveur au client en réponse à une <commande> ou une <commande-continuation>.

réponse = réponse simple / réponse multi lignes
 réponse simple = ligne de réponse initiale
 réponse multi lignes = ligne de réponse initiale bloc de données multi lignes

ligne de réponse initiale = contenu de réponse initiale [SP commentaire en queue] CRLF
 contenu de réponse initiale = contenu de réponse initiale-X
 contenu de réponse initiale-X = 3DIGIT *(SP argument de réponse)
 argument de réponse = 1*A-CHAR
 commentaire en queue = *U-CHAR

9.4.2 Contenu de ligne de réponse initiale

Cette syntaxe définit les lignes de réponse initiale spécifiques pour les diverses commandes de la présente spécification (voir au paragraphe 3.1 les limites de longueurs). Seuls sont mentionnés les codes de réponse avec des arguments.

contenu de réponse initiale =/ contenu de réponse-111 / contenu de réponse-211 / contenu de réponse-220 / contenu de réponse-221 / contenu de réponse-222 / contenu de réponse-223 / contenu de réponse-401

contenu de réponse-111 = "111" SP date4y heure
 contenu de réponse-211 = "211" 3(SP numéro d'article) SP nom de groupe de nouvelles
 contenu de réponse-220 = "220" SP numéro d'article SP identifiant de message
 contenu de réponse-221 = "221" SP numéro d'article SP identifiant de message
 contenu de réponse-222 = "222" SP numéro d'article SP identifiant de message
 contenu de réponse-223 = "223" SP numéro d'article SP identifiant de message
 contenu de réponse-401 = "401" SP étiquette de capacité

9.4.3 Contenu de réponse multi lignes

Cette syntaxe définit le contenu de diverses réponses multi lignes ; plus précisément, elle définit la partie de la réponse dans le bloc de données multi lignes après avoir ôté tout "bourrage de point". La portion numérique de chaque nom non terminal indique le code de réponse de la ligne de réponse initiale qui est suivie par ces données.

contenu-de-réponse-multi-lignes = contenu-article-220-ml / corps-222-ml-contenu / capacités-101-ml-contenu / hdr-225-ml-contenu / head-221-ml-contenu / help-100-ml-contenu / liste-215-ml-contenu / liste-groupe-211-ml-contenu / newgroups-231-ml-contenu / newnews-230-ml-contenu / over-224-ml-contenu

article-220-ml-contenu = article
 corps-222-ml-contenu = corps
 capacités-101-ml-contenu = version-ligne CRLF *(capacité-ligne CRLF)
 hdr-225-ml-contenu = *(numéro-article SP contenu-en-tête CRLF)
 head-221-ml-contenu = 1*en-tête
 help-100-ml-contenu = *(U-CHAR CRLF)
 liste-215-ml-contenu = liste-contenu
 liste-groupe-211-ml-contenu = *(numéro-article CRLF)

newgroups-231-ml-contenu = liste-groupes-actifs
 newnews-230-ml-contenu = *(identifiant-de-message CRLF)
 over-224-ml-contenu = *(numéro-article over-contenu CRLF)

liste-groupes-actifs = *(nom-de-groupe-de-nouvelles SPA numéro-article SPA numéro-article SPA statut-groupe-de-nouvelles CRLF)
 contenu-en-tête = *S-NONTAB
 hdr-n-contenu = [(nom-en-tête ":" / nom-métadonnées) SP contenu-en-tête]
 contenu-liste = corps
 statut-groupe-de-nouvelles = %x79 / %x6E / %x6D / statut-privé
 over-contenu = 7(TAB contenu-en-tête) *(TAB hdr-n-contenu)
 statut-privé = jeton ; excepté les valeurs dans statut-groupe-de-nouvelles

9.5 Lignes de capacités

Cette syntaxe définit la forme générique d'une ligne de capacité dans la liste des capacités (voir au paragraphe 3.3.1).

ligne de capacité = entrée-capacité
 entrée-capacité = X-entrée-capacité
 X-entrée-capacité = étiquette-de-capacité *(WS argument-de-capacité)
 étiquette-de-capacité = mot-clé
 argument-de-capacité = jeton

Cette syntaxe définit les entrées spécifiques de capacité pour les capacités dans la présente spécification.

entrée-capacité = / hdr-capacité / ihave-capacité / implementation-capacité / list-capacité / mode-reader-capacité / newnews-capacité / over-capacité / post-capacité / reader-capacité

hdr-capacité = "HDR"
 ihave-capacité = "IHAVE"
 implementation-capacité = "IMPLEMENTATION" *(WS jeton)
 list-capacité = "LIST" 1*(WS mot-clé)
 mode-reader-capacité = "MODE-READER"
 newnews-capacité = "NEWNEWS"
 over-capacité = "OVER" [WS "MSGID"]
 post-capacité = "POST"
 reader-capacité = "READER"

version-ligne = "VERSION" 1*(WS numéro-version)
 numéro-version = nzDIGIT *5DIGIT

9.6 Variantes de LIST

Ce paragraphe définit plus spécifiquement les mots-clés pour la commande LIST et la syntaxe du contenu de la réponse correspondante.

; active
 liste-arguments = / "ACTIVE" [WS wildmat]
 liste-contenu = / liste-active-contenu
 liste-active-contenu = liste-groupes-actifs

; active.times
 liste-arguments = / "ACTIVE.TIMES" [WS wildmat]
 liste-contenu = / liste-active-times-contenu
 liste-active-times-contenu = *(nom-de-groupe-de-nouvelles SPA 1*DIGIT SPA créateur-du-groupe-de-nouvelles CRLF)
 créateur-du-groupe-de-nouvelles = U-TEXT

; distrib.pats
 liste-arguments = / "DISTRIB.PATS"
 liste-contenu = / liste-distrib-pats-contenu
 liste-distrib-pats-contenu = *(1*DIGIT ":" wildmat ":" distribution CRLF)

distribution = jeton

; headers

liste-arguments =/ "HEADERS" [WS ("MSGID" / "RANGE")]

liste-contenu =/ liste-en-têtes-contenu

liste-en-têtes-contenu = *(header-meta-nom CRLF) / *((metadata-nom / ":") CRLF)

; groupes-de -nouvelles

liste-arguments =/ "NEWSGROUPS" [WS wildmat]

liste-contenu =/ liste-groupes-de-nouvelles-contenu

liste-groupes-de-nouvelles-contenu = *(nom-groupe-de-nouvelles WS description-groupe-de-nouvelles CRLF)

description-groupe-de-nouvelles = S-TEXT

; overview.fmt

liste-arguments =/ "OVERVIEW.FMT"

liste-contenu =/ liste-overview-fmt-contenu

liste-overview-fmt-contenu = "Subject:" CRLF

"From:" CRLF

"Date:" CRLF

"Message-ID:" CRLF

"References:" CRLF

(":bytes" CRLF ":lines" / "Bytes:" CRLF "Lines:") CRLF

*((nom-en-tête ":full" / nom-métadonnées) CRLF)

9.7 Articles

Cette syntaxe définit le non terminal <article>, qui représente le format d'un article décrit au paragraphe 3.6.

article = 1*en-tête CRLF corps

en-tête = nom d'en-tête ":" [CRLF] SP contenu d'en-tête CRLF

contenu d'en-tête = *(S-CHAR / [CRLF] WS)

corps = *(B-CHAR CRLF)

9.8 Non terminaux généraux

Ces non terminaux sont utilisés en divers endroits dans la syntaxe et sont rassemblés ici pour des raisons pratiques. Quelques uns de ces non terminaux ne sont pas utilisés dans la présente spécification mais sont fournis pour la cohérence et pour les auteurs d'extensions.

bloc-données-multi-lignes = terminaison-lignes-contenu

lignes-contenu = *([texte-contenu] CRLF)

texte-contenu = (".." / B-NONDOT) *B-CHAR

terminaison = "." CRLF

numéro-article = 1*16DIGIT

nom-en-tête = 1*A-NOTCOLON

mot-clé = ALPHA 2*(ALPHA / DIGIT / "." / "-")

identifiant de message = "<" 1*248A-NOTGT ">"

nom-de-groupe-de-nouvelles = 1*wildmat-exact

jeton = 1*P-CHAR

wildmat = wildmat-pattern *(", ["!"] wildmat-pattern)

wildmat-pattern = 1*wildmat-item

wildmat-item = wildmat-exact / wildmat-wild

wildmat-exact = %x22-29 / %x2B / %x2D-3E / %x40-5A / %x5E-7E /

UTF8-non-ascii ; exclut ! * , ? [\]

wildmat-wild = "*" / "?"

base64 = *(4base64-char) [base64-terminal]

base64-char = UPPER / LOWER / DIGIT / "+" / "/"

base64-terminal = 2base64-char "=" / 3base64-char "="

; Jeux de caractères spéciaux assortis :
 ; A- signifie fondé sur US-ASCII, exclut les caractères de contrôle et les SP
 ; P- signifie fondé sur UTF-8, exclut les caractères de contrôle et les SP
 ; U- signifie fondé sur UTF-8, exclut les caractères NUL CR et LF
 ; B- signifie fondé sur les octets, exclut les caractères NUL CR et LF
 A-CHAR = %x21-7E
 A-NOTCOLON = %x21-39 / %x3B-7E ; exclut ":"
 A-NOTGT = %x21-3D / %x3F-7E ; exclut ">"
 P-CHAR = A-CHAR / UTF8-non-ascii
 U-CHAR = CTRL / TAB / SP / A-CHAR / UTF8-non-ascii
 U-NONTAB = CTRL / SP / A-CHAR / UTF8-non-ascii
 U-TEXT = P-CHAR *U-CHAR
 B-CHAR = CTRL / TAB / SP / %x21-FF
 B-NONDOT = CTRL / TAB / SP / %x21-2D / %x2F-FF ; exclut "."

ALPHA = UPPER / LOWER ; utilisé seulement quand c'est insensible à la casse
 CR = %x0D
 CRLF = CR LF
 CTRL = %x01-08 / %x0B-0C / %x0E-1F
 DIGIT = %x30-39
 nzDIGIT = %x31-39
 EOL = *(SP / TAB) CRLF
 LF = %x0A
 LOWER = %x61-7A
 SP = %x20
 SPA = 1*SP
 TAB = %x09
 UPPER = %x41-5A
 UTF8-non-ascii = UTF8-2 / UTF8-3 / UTF8-4
 UTF8-2 = %xC2-DF UTF8-tail
 UTF8-3 = %xE0 %xA0-BF UTF8-tail / %xE1-EC 2UTF8-tail / %xED %x80-9F UTF8-tail / %xEE-EF 2UTF8-tail
 UTF8-4 = %xF0 %x90-BF 2UTF8-tail / %xF1-F3 3UTF8-tail / %xF4 %x80-8F 2UTF8-tail
 UTF8-tail = %x80-BF
 WS = 1*(SP / TAB)

Les non terminaux suivants exigent une considération particulière. Ils représentent des situations où le matériel DEVRAIT être restreint à l'UTF-8, mais les mises en œuvre DOIVENT être capables de traiter d'autres codages de caractères. Donc, il y a pour eux deux ensembles de définitions.

Les mises en œuvre DOIVENT accepter tout contenu qui satisfait cette syntaxe :

S-CHAR = %x21-FF
 S-NONTAB = CTRL / SP / S-CHAR
 S-TEXT = (CTRL / S-CHAR) *B-CHAR

et PEUVENT passer un tel contenu non altéré.

Lorsque elles génèrent un nouveau contenu ou recodent un contenu existant, les mises en œuvre DEVRAIENT se conformer à cette syntaxe :

S-CHAR = P-CHAR
 S-NONTAB = U-NONTAB
 S-TEXT = U-TEXT

9.9 Extensions et validation

La spécification d'une extension enregistrée DOIT inclure la syntaxe formelle qui définit les formes supplémentaires pour les non terminaux suivants:

commande

pour chaque nouvelle commande autre qu'une variante de la commande LIST - la syntaxe de chaque commande DOIT être compatible avec la définition de <X-commande> ;

commande-datastream

pour chaque nouvelle commande qui écoule immédiatement des données ;

commande-continuation

pour chaque nouvelle commande qui envoie d'autre matériel après la ligne de commande initiale - la syntaxe de chaque continuation DOIT être exactement ce qui est envoyé au serveur, incluant tous les mécanismes d'échappement tels que le "bourrage de points" ;

contenu-réponse-initiale

pour chaque nouveau code de réponse qui a des arguments - la syntaxe de chaque réponse DOIT être compatible avec la définition de <X-contenu-réponse-initiale> ;

contenu-réponse-multi-lignes

pour chaque nouveau code de réponse qui a une réponse multi lignes - la syntaxe DOIT montrer la réponse après que les lignes qui contiennent le code de réponse et l'octet de terminaison ont été retirées et tout "bourrage de point" défaut ;

entrée-capacité

pour chaque nouvelle étiquette de capacité - la syntaxe de chaque entrée DOIT être compatible avec la définition de <X-entrée-capacité> ;

liste-arguments

pour chaque nouvelle variante de la commande LIST - la syntaxe de chaque entrée DOIT être compatible avec la définition de <X-commande> ;

liste-contenu

pour chaque nouvelle variante de la commande LIST - la syntaxe DOIT montrer la réponse après que les lignes contenant le code de réponse 215 et l'octet de terminaison ont été retirés et que tout "bourrage de points" a été défaut.

La notation =/ de l'ABNF [RFC4234] et les conventions de désignation décrites au paragraphe 9.1 DEVRAIENT être utilisées pour cela.

Lorsque la syntaxe de la présente spécification, ou la syntaxe qui s'appuie sur elle, est validée, on devrait noter que :

- o les non terminaux <commande-ligne>, <commande-datastream>, <commande-continuation>, <réponse>, et <contenu-réponse-multi-lignes> décrivent les concepts de base du protocole et aucune autre règle ne s'y réfère ;
- o le non terminal <base64> est fourni pour les besoins des auteurs d'extensions et aucune règle de la présente spécification n'y fait référence ;
- o pour les raisons ci-dessus, les non terminaux <S-CHAR>, <S-NONTAB>, et <S-TEXT> ont chacun deux définitions ;
- o le non terminal <INDÉFINI> est délibérément non défini.

10. Considérations d'internationalisation

10.1 Introduction et situation historique

La [RFC977] a été écrite à une époque où l'internationalisation n'était pas perçue comme une question significative. À ce titre, elle a été écrite avec l'hypothèse que toutes les communications seraient en ASCII et utiliseraient seulement une couche de transport à 7 bits, bien qu'en pratique toutes les mises en œuvre connues soient à 8 bits.

Depuis lors, Usenet et NNTP se sont répandus dans le monde entier. En l'absence de normes pour traiter les questions de langage et de jeux de caractères, les pays, hiérarchies de groupes de nouvelles, et les individus, ont trouvé diverses solutions qui fonctionnent pour eux mais ne sont pas nécessairement appropriées ailleurs. Par exemple, certains ont adopté un jeu de caractères à 8 bits par défaut approprié à leurs besoins (comme l'ISO/CEI 8859-1 en Europe de l'ouest ou KOI-8 en Russie) ; d'autres ont utilisé l'ASCII (soit l'US-ASCII, soit des variantes nationales) dans les en-têtes mais des jeux de caractères locaux à 16 bits dans les corps d'articles, et d'autres encore ont choisi une combinaison de MIME [RFC2045] et d'UTF-8. Avec l'utilisation accrue de MIME dans la messagerie électronique, il est devenu beaucoup plus courant de trouver des articles NNTP contenant des en-têtes MIME qui identifient le jeu de caractères du corps, mais c'est loin d'être universel.

La confusion qui en résulte ne sert pas l'interopérabilité.

Un point qui a été généralement accepté est que les articles peuvent contenir des octets avec le bit de poids fort établi, et NNTP est seulement supposé fonctionner sur des chemins de transport en 8 bits pur.

10.2 Cette spécification

Une partie du rôle de la présente spécification est d'éliminer cette confusion et de promouvoir l'interopérabilité autant que possible. En même temps, il est nécessaire d'accepter l'existence de la situation actuelle et de ne pas casser gratuitement les mises en œuvre et arrangements existants, même si elles sont tout sauf optimales. Donc, la pratique actuelle décrite ci-dessus a été prise en considération pour produire la présente spécification.

Cette spécification étend NNTP de l'US-ASCII [ANSI1986] à l'UTF-8 [RFC3629]. Sauf pour les deux domaines discutés ci-dessous, UTF-8 (qui est un sur ensemble de l'US-ASCII) est obligatoire, et les mises en œuvre NE DOIVENT PAS utiliser d'autre codage.

D'abord, l'utilisation de MIME pour les en-têtes d'articles et les corps est fortement recommandée. Cependant, sachant qu'il existe des pratiques largement divergentes, une tentative d'exiger une norme particulière de codage et d'étiquetage serait prématurée pour l'instant. En conséquence, la présente spécification permet l'utilisation de données de 8 bits arbitraires dans les articles sous réserve des exigences et recommandations suivantes :

- o Les noms des en-têtes (par exemple, "From" ou "Subject") DOIVENT être en US-ASCII.
- o Les valeurs d'en-têtes DEVRAIENT utiliser l'US-ASCII ou un codage fondé sur lui, comme celui de la [RFC2047], jusqu'au moment où une autre approche sera normalisée. À présent, les codages sur 8 bits (incluant l'UTF-8) NE DEVRAIENT PAS être utilisés parce qu'il est probable qu'ils vont causer des problèmes d'interopérabilité.
- o Le jeu de caractères des corps d'articles DEVRAIT être indiqué dans les en-têtes d'articles, et ce DEVRAIT être fait en accord avec MIME.
- o lorsque un article est obtenu d'une source externe, une mise en œuvre PEUT le passer et en déduire des données (comme la réponse à la commande HDR) même si l'article ou les données ne satisfont pas aux exigences ci-dessus. Les mises en œuvre DOIVENT transférer correctement de tels articles et données sans les changer ; elles NE DOIVENT PAS tenter de convertir ou recoder l'article ou les données dérivées. (Néanmoins, un client ou serveur PEUT choisir de ne pas poster ou transmettre l'article si, après examen de l'article, il estime inapproprié de le faire.)

Cette exigence affecte les commandes ARTICLE (paragraphe 6.2.1), BODY (paragraphe 6.2.3), HDR (paragraphe 8.5), HEAD (paragraphe 6.2.2), IHAVE (paragraphe 6.3.2), OVER (paragraphe 8.3), et POST (paragraphe 6.3.1).

Ensuite, les exigences suivantes s'appliquent à la liste de groupes de nouvelles retournée par la commande LIST NEWSGROUPS (paragraphe 7.6.6) :

- o Bien que la présente spécification permette l'UTF-8 pour les noms de groupes de nouvelles, ils DEVRAIENT se restreindre à l'US-ASCII jusqu'à ce qu'un successeur de la [RFC1036] normalise une autre approche. Les codages sur 8 bits NE DEVRAIENT PAS être utilisés parce qu'ils vont probablement causer des problèmes d'interopérabilité.
- o La description de groupe de nouvelles DEVRAIT être en US-ASCII ou UTF-8 jusqu'à ce qu'un successeur de la RFC 1036 normalise d'autres arrangements de codage. Les codages sur 8 bits autres que UTF-8 NE DEVRAIENT PAS être utilisés parce qu'ils vont probablement causer des problèmes d'interopérabilité.
- o Les mises en œuvre qui obtiennent des données d'une source externe DOIVENT les traiter correctement même si elles ne satisfont pas aux exigences ci-dessus. Les mises en œuvre (en particulier, les clients) DOIVENT traiter correctement de telles données.

10.3 Problèmes en instance

Bien que la principale utilisation de NNTP soit de transmettre des articles qui se conforment à la RFC 1036 (articles Netnews) il est aussi utilisé pour d'autres formats (voir l'Appendice A). Il est donc très approprié que les problèmes d'internationalisation relatifs aux formats d'article soient traités dans les spécifications pertinentes. Pour les articles Netnews, c'est tout successeur de la RFC 1036. Pour les messages électroniques, c'est la [RFC2822].

Bien sûr, tout article transmis via NNTP doit se conformer aussi à la présente spécification.

Restreindre les noms de groupes de nouvelles à l'UTF-8 n'est pas une solution complète. En particulier, lorsque de nouveaux noms de groupes de nouvelles sont créés ou lorsque il est demandé à un utilisateur d'entrer un nom de groupe de nouvelles, certains schémas de canonisation devront avoir lieu. La présente spécification ne tente pas de définir cette canonisation ; d'autres travaux sont nécessaires dans ce domaine, en conjonction avec les spécifications de format d'article. Jusqu'à la publication de telles spécifications, les mises en œuvre DEVRAIENT confronter les noms de groupes de nouvelles octet par octet. On prévoit que tout schéma approuvé sera appliqué "sur les bords", et donc que les comparaisons octet par octet vont continuer de s'appliquer à la plupart, sinon à toutes, les utilisations de noms de groupes de nouvelles

dans NNTP.

En attendant, toutes les mises en œuvre qui expérimentent des noms de groupe de nouvelles en UTF-8 sont averties qu'une future spécification pourrait exiger que ces noms soient canonisés lorsque utilisés avec NNTP d'une certaine façon non compatible avec leurs expériences.

Comme l'utilisation principale de NNTP est avec Netnews, et comme les descriptions de groupe de nouvelles sont normalement distribuées par des articles spécialement formatés, il est recommandé que les questions d'internationalisation en rapport avec eux soient traités par tout successeur de la RFC 1036.

11. Considérations relatives à l'IANA

La présente spécification demande à l'IANA de tenir un registre des étiquettes de capacité. Le contenu initial de ce registre est spécifié au paragraphe 3.3.4. Comme décrit au paragraphe 3.3.3, les étiquettes qui commencent par X sont réservées à une utilisation privée, tandis que tous les autres noms sont supposés être associés à une spécification dans une RFC sur la voie de la normalisation ou définissant un protocole expérimental approuvé par l'IESG.

Les différentes entrées dans le registre DOIVENT utiliser des étiquettes de capacité différentes.

Les différentes entrées dans le registre NE DOIVENT PAS utiliser le même nom de commande. À cette fin, les variantes distinguées par un second mot-clé ou un mot-clé suivant (par exemple, "LIST HEADERS" et "LIST OVERVIEW.FMT") comptent comme des commandes différentes. Si il y a besoin que deux extensions utilisent la même commande, une seule spécification harmonisée DOIT être enregistrée.

12. Considérations sur la sécurité

Cette section est destinée à informer les développeurs d'applications, les fournisseurs d'informations, et les utilisateurs des limitations de la sécurité de NNTP comme décrit par ce document. L'exposé ne comporte pas de solutions définitives aux problèmes révélés, bien qu'il fasse des suggestions pour réduire les risques pour la sécurité.

12.1 Informations personnelles et propriétaires

NNTP, parce qu'il a été créé pour distribuer des articles de nouvelles du réseau, va transmettre toutes les informations quelles qu'elles soient qui sont mémorisées dans ces articles. La spécification de ces informations sort du domaine d'application du présent document, mais il est probable que certaines informations personnelles et/ou propriétaires sont disponibles dans certains de ces articles. Il est très important que les concepteurs et développeurs fournissent des informations d'avertissement aux utilisateurs afin que des informations personnelles et/ou propriétaires dans le matériel qui sont ajoutées automatiquement aux articles (par exemple, dans les en-têtes) ne soient pas divulguées par inadvertance. De plus, des mécanismes efficaces et facilement compréhensibles pour gérer la distribution des nouveaux articles DEVRAIENT être fournis aux administrateurs de service NNTP afin qu'ils soient capables de rapporter en toute confiance la diffusion probable de tout ensemble particulier de nouveaux articles.

12.2 Abus des informations du serveur d'enregistrement

Un serveur est en position de sauvegarder les données de session sur les demandes d'un utilisateur et peut identifier leurs schémas de lecture ou les sujets qui l'intéressent. Ces informations sont clairement de nature confidentielle, et leur traitement peut être restreint par la loi dans certains pays. Les gens qui utilisent ce protocole pour fournir des données sont responsables de s'assurer qu'un tel matériel n'est pas distribué sans la permission de tout individu identifiable par les résultats publiés.

12.3 Authentification faible et contrôle d'accès

Il n'y a pas d'authentification fondée sur l'utilisateur ou sur un jeton dans la spécification NNTP de base. L'accès est normalement contrôlé par les fichiers de configuration du serveur. Ces fichiers spécifient l'accès en utilisant des noms de domaine ou des adresses IP. Cependant, la présente spécification ne permet pas la création d'extensions à NNTP à de telles fins ; une de ces extensions est la [RFC4643]. Bien que l'inclusion de tels mécanismes soit facultative, le faire est vivement encouragé.

D'autres mécanismes sont aussi disponibles. Par exemple, un serveur mandataire pourrait être mis en place pour exiger l'authentification avant la connexion via le mandataire au serveur NNTP.

12.4 Usurpation de DNS

De nombreuses mises en œuvre existantes de NNTP autorisent les connexions entrantes en vérifiant l'adresse IP de cette connexion par rapport aux adresses IP obtenues via des recherches dans le DNS des noms de domaines donnés dans les fichiers de configuration locale. Les serveurs qui utilisent ce type d'authentification et les clients qui trouvent un serveur en faisant une recherche du nom du serveur dans le DNS dépendent très fortement du service des noms de domaines, et sont donc généralement enclins à des attaques contre la sécurité fondées sur des mauvaises associations délibérées d'adresses IP et de noms DNS. Les clients et les serveurs doivent prendre des précautions pour s'assurer de la validité d'une association d'un numéro IP et d'un nom DNS.

En particulier, les clients et serveurs NNTP DEVRAIENT s'appuyer sur leur résolveur de noms pour avoir la confirmation d'une association d'un numéro IP et d'un nom DNS, plutôt que de mettre en antémémoire le résultat des recherches précédentes de nom d'hôte. De nombreuses plateformes peuvent déjà mettre en antémémoire les recherches de nom d'hôte en local lorsque c'est approprié, et elles DEVRAIENT être configurées à le faire. Il est le propre de ces recherches d'être mises en antémémoire, cependant, c'est seulement lorsque les informations de durée de vie restante (TTL, *Time To Live*) rapportées par le serveur de noms les rendent vraisemblables que les informations mises en antémémoire restent utiles.

Si les clients ou serveurs NNTP mettent en antémémoire les résultats des recherches de noms d'hôtes afin de réaliser des améliorations de performances, ils DOIVENT observer les informations de TTL rapportées par le DNS. Si les clients ou serveurs NNTP n'observent pas cette règle, ils pourraient être mystifiés lorsque l'adresse IP d'un serveur joint précédemment a changé. Comme on s'attend à ce que les dénumérotages deviennent de plus en plus fréquents dans les réseaux, la possibilité de cette forme d'attaque va augmenter. Observer cette exigence réduit donc cette vulnérabilité potentielle de la sécurité.

Cette exigence améliore aussi le comportement d'équilibrage de charge des clients pour les serveurs dupliqués qui utilisent le même nom DNS et réduit la probabilité qu'un utilisateur échoue à accéder à des sites qui utilisent cette stratégie.

12.5 Problèmes de UTF-8

UTF-8 [RFC3629] permet seulement certaines séquences d'octets et en désigne d'autres comme mal formées ou "illégales". La norme Unicode identifie un certain nombre de problèmes de sécurité en rapport avec les séquences illégales et interdit leur génération par les mises en œuvre conformes.

Les mises en œuvre de la présente spécification NE DOIVENT PAS générer de séquences mal formées ou illégales et DEVRAIENT les détecter et prendre les mesures appropriées. Cela pourrait inclure ce qui suit :

- o générer un code de réponse 501,
- o remplacer de telles séquences par la séquence %xEF.BF.BD, qui code le "caractère de remplacement" U+FFFD,
- o fermer la connexion,
- o remplacer de telles séquences par une séquence valide "devinée" (sur la base des propriétés du codage UTF-8).

Dans le dernier cas, la mise en œuvre DOIT s'assurer que tout remplacement ne peut pas être utilisé pour outrepasser la validité ou les vérifications de sécurité. Par exemple, la séquence illégale %xC0.A0 est un codage trop long pour une espace (%x20). Si il est remplacé par le codage correct dans une ligne de commande, cela doit se faire avant que la ligne de commande soit analysée en arguments individuels. Si le remplacement intervient après l'analyse, il serait possible de générer un argument avec une espace incorporée, ce qui est interdit. L'utilisation de "caractère de remplacement" n'a pas ce problème, car elle est permise partout où des caractères non US-ASCII le sont. Les mises en œuvre DEVRAIENT utiliser une des deux premières solutions lorsque la structure générale du flux NNTP reste intacte et DEVRAIENT clore la connexion si il n'est plus possible de l'analyser de façon intelligente.

12.6 Mise en antémémoire des listes de capacités

La commande CAPABILITIES fournit une liste de capacités, qui sont des informations sur les capacités actuelles du serveur. Chaque fois qu'il y a un changement pertinent de l'état du serveur, le résultat de cette commande est obligé de changer en conséquence.

Dans la plupart des situations, la liste des capacités dans un certain état du serveur ne va pas changer d'une session à l'autre ; par exemple, une certaine extension sera installée de façon permanente sur un serveur. Certains clients peuvent donc souhaiter se souvenir des extensions qu'accepte un serveur pour éviter de se retarder d'une commande et réponse

supplémentaire, en particulier si ils ouvrent plusieurs connexions dans la même session.

Cependant, les informations sur les extensions relatives à la sécurité et la confidentialité NE DOIVENT PAS être mises en antémémoire, car cela pourrait permettre diverses attaques.

Par exemple, considérons un serveur qui permet l'utilisation de mots de passe en clair sur des liaisons qui sont chiffrées mais pas autrement :

```
[Établissement de la connexion initiale achevé.]
[S] 200 NNTP Service prêt, envoi permis
[C] CAPABILITIES
[S] 101 Liste des capacités :
[S] VERSION 2
[S] READER
[S] NEWNEWS
[S] POST
[S] XENCRYPT
[S] LIST ACTIVE NEWSGROUPS
[S] .
[C] XENCRYPT
[Client et serveur négocient le chiffrement sur la liaison]
[S] 283 Liaison chiffrée établie
```

```
[C] CAPABILITIES
[S] 101 Liste des capacités :
[S] VERSION 2
[S] READER
[S] NEWNEWS
[S] POST
[S] XSECRET
[S] LIST ACTIVE NEWSGROUPS
[S] .
[C] XSECRET fred flintstone
[S] 290 Mot de passe pour fred accepté
```

Si le client met en antémémoire la dernière liste des capacités, il va alors à la prochaine session tenter d'utiliser XSECRET sur une liaison non chiffrée :

```
[Établissement de la connexion initiale achevé.]
[S] 200 NNTP Service prêt, envoi permis
[C] XSECRET fred flintstone
[S] 483 N'est permis que sur les liaisons sécurisées
```

Ceci expose le mot de passe à tout espion. Bien que la principale cause de cela soit de passer un secret sans d'abord vérifier la sécurité de la liaison, mettre en antémémoire une liste de capacités peut augmenter le risque.

Toute extension de sécurité devrait inclure l'exigence de vérifier l'état de sécurité de la liaison d'une manière appropriée à cette extension.

La mise en antémémoire ne devrait normalement être envisagée que pour les clients anonymes qui n'utilisent aucune extension de sécurité ou de confidentialité et pour lesquels le temps nécessaire pour une commande et réponse supplémentaire est un vrai problème.

13. Remerciements

Le présent document est le résultat des efforts des membres présents et passés du groupe de travail NNTP présidé par Russ Allbery et Ned Freed. Il n'aurait pas pu être produit sans eux.

L'auteur remercie les auteurs originaux de NNTP tel que documenté dans la [RFC0977] : Brian Kantor et Phil Lapsley.

L'auteur adresse ses remerciements chaleureux pour les travaux au comité NNTP présidé par Eliot Lear. L'organisation de

ce document a été influencée par la dernière version disponible de ce groupe de travail. Des remerciements particuliers à Eliot qui a généreusement fourni les sources originales lisibles par la machine pour ce document. Les travaux du groupe de travail DRUMS, en particulier la [RFC1869], qui a soutenu les réflexions originales qui ont conduit à la commande CAPABILITIES et aux mécanismes d'extensions détaillés dans ce document. Les auteurs de la [RFC2616] qui ont fourni des exemples spécifiques et pertinents des problèmes de sécurité qui devraient être pris en considération pour HTTP. Comme beaucoup des mêmes considérations existent pour NNTP, ces exemples ont été inclus ici avec des modifications mineures. Les commentaires et informations supplémentaires fournies par les personnes suivantes dans la préparation d'un ou plusieurs des prédécesseurs de ce document : Russ Allbery rra@stanford.edu, Wayne Davison davison@armory.com, Chris Lewis clewis@bnr.ca, Tom Limoncelli tal@mars.superlink.net, Eric Schnoebelen eric@egsner.cirr.com, Rich Salz rsalz@osf.org.

Ce travail a été motivé par les travaux de divers auteurs de lecteurs de nouvelles et d'auteurs de serveurs de nouvelles, incluant : Rick Adams, auteur original des extensions NNTP au lecteur de nouvelles RN et dernier gestionnaire de Bnews ; Stan Barber, auteur original des extensions NNTP aux lecteurs de nouvelles qui font partie de Bnews ; Geoff Collyer, auteur original de la proposition de base de données OVERVIEW et un des auteurs originaux de CNEWS ; Dan Curry, auteur original du lecteur de nouvelles xvnews ; Wayne Davison, auteur des premières extensions au lecteur de nouvelles RN (couramment appelé TRN) ; Geoff Huston, auteur original de hANU NEWS ; Phil Lapsey, auteur original de la mise en œuvre de référence UNIX pour NNTP ; Iain Lea, gestionnaire original du lecteur de nouvelles TIN ; Chris Lewis, premier auteur de la mise en œuvre de l'extension AUTHINFO GENERIC ; Rich Salz, auteur original de INN ; Henry Spencer, un des auteurs originaux de CNEWS ; Kim Storm, auteur original du lecteur de nouvelles NN.

D'autres personnes ont contribué à ce document : Matthias Andree, Greg Andruk, Daniel Barclay, Maurizio Codogno, Mark Crispin, Andrew Gierth, Juergen Helbing, Scott Hollenbeck, Urs Janssen, Charles Lindsey, Ade Lovett, David Magda, Ken Murchison, Francois Petillon, Peter Robinson, Rob Siemborski, Howard Swinehart, Ruud van Tol, Jeffrey Vinocur, Erik Warming. L'auteur les remercie et présente ses excuses à tous ceux qui ont été omis.

Finalement, l'auteur présent exprime sa reconnaissance pour la grande quantité de travail effectué sur les précédentes versions par leur auteur : Stan Barber sob@academ.com.

14. Références

14.1 Références normatives

- [ANSI1986] American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.
- [RFC2045] N. Freed et N. Borenstein, "[Extensions de messagerie Internet](#) multi-objets (MIME) Partie 1 : Format des corps de message Internet", novembre 1996. (*D. S., MàJ par [2184](#), [2231](#), [5335](#).*)
- [RFC2047] K. Moore, "MIME ([Extensions de messagerie Internet](#) multi-objets) Partie trois : extensions d'en-tête de message pour texte non ASCII", novembre 1996. (*MàJ par [RFC2184](#), [RFC2231](#)*) (D.S.)
- [RFC2119] S. Bradner, "[Mots clés à utiliser](#) dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997.
- [RFC3629] F. Yergeau, "[UTF-8, un format de transformation](#) de la norme ISO 10646", STD 63, novembre 2003.
- [RFC4234] D. Crocker et P. Overell, "[BNF augmenté pour les spécifications de syntaxe](#) : ABNF", octobre 2005. (*Remplace RFC2234, remplacée par RFC5234*)
- [RFC4648] S. Josefsson, "[Codages de données Base16, Base32 et Base64](#)", octobre 2006. (*Remplace [RFC3548](#)*) (P.S.)
- [TF.686-1] Union Internationale des Télécommunications – Secteur des Radiocommunications, "Glossaire", Recommandation UIT-R TF.686-1, octobre 1997.

14.2 Références pour information

- [RFC0977] B. Kantor et P. Lapsley, "Protocole de transfert des nouvelles du réseau", février 1986. (*Obsolète, voir [RFC3977](#)*)

- [RFC1036] M. Horton et R. Adams, "Norme pour l'[échange de messages](#) USENET", décembre 1987. (*Remplacée par [RFC5536](#)*)
- [RFC1305] D. Mills, "[Protocole de l'heure du réseau](#), version 3, spécification, mise en œuvre et analyse", STD 12, mars 1992. (*Remplacée par [RFC5905](#)*)
- [RFC1869] J. Klensin, N. Freed, M. Rose, E. Stefferud, D. Crocker, "Extensions de service à SMTP", novembre 1995. (*Obsolète, voir [RFC5321](#), [STD0010](#)*)
- [RFC2616] R. Fielding et autres, "[Protocole de transfert hypertexte](#) -- HTTP/1.1", juin 1999. (*D.S., MàJ par [2817](#), [6585](#)*)
- [RFC2629] M. Rose, "Écrire des I-D et des RFC en utilisant XML", juin 1999. (*Information*)
- [RFC2822] P. Resnick, "[Format de message Internet](#)", avril 2001. (*Remplace la [RFC0822](#), STD 11, Remplacée par [RFC5322](#)*)
- [RFC2980] S. Barber, "Extensions communes à NNTP", octobre 2000. (*MàJ par [RFC3977](#), [RFC4643](#), [RFC4644](#)*) (*Information*)
- [RFC4642] K. Murchison et autres, "Utilisation de la sécurité de la couche transport (TLS) avec le protocole de transfert des nouvelles du réseau (NNTP)", octobre 2006. (*P.S.*)
- [RFC4643] J. Vinocur, K. Murchison, "Extension d'authentification pour le protocole de transfert des nouvelles du réseau (NNTP)", octobre 2006. (*MàJ [RFC2980](#)*) (*P.S.*)
- [RFC4644] J. Vinocur, K. Murchison, "Extension d'alimentation en direct pour le protocole de transfert des nouvelles du réseau (NNTP)", octobre 2006. (*MàJ [RFC2980](#)*) (*P.S.*)
- [ROBE1995] Robertson, R., "FAQ: Overview database / NOV General Information", January 1995. Il n'y a pas de copie définitive de ce document connue de l'auteur. Il a été précédemment posté comme article Usenet <news:nov-faq-1-930909720@agate.Berkeley.EDU>
- [SALZ1992] Salz, R., "Manual Page for wildmat(3) from the INN 1.4 distribution, Revision 1.10", avril 1992. Il n'y a pas de copie définitive de ce document connue de l'auteur.

Appendice A. Interaction avec d'autres spécifications

NNTP est le plus souvent utilisé pour transférer des articles qui se conforment à la [RFC1036] (de tels articles sont appelés "articles Netnews" ici). Il est aussi parfois utilisé pour transférer des messages électroniques qui se conforment à la [RFC2822] (de tels articles sont appelés "articles de messagerie électronique" ici). Dans cette situation, les articles doivent se conformer à la fois à la présente spécification et à l'autre ; cet appendice décrit quelques problèmes qui s'y rapportent.

A.1 Saut à la ligne d'en-tête

NNTP permet qu'une ligne d'en-tête saute à la ligne (en insérant une paire CRLF) avant tout caractère espace ou TAB.

Il est exigé des articles de messagerie électronique et Netnews qu'ils aient tous deux au moins un octet autre qu'espace ou TAB sur chaque ligne d'en-tête. Donc, le saut à la ligne ne peut se produire qu'à un point de chaque séquence d'espaces consécutives ou de tabulations. Les articles Netnews sont de plus obligés d'avoir le nom d'en-tête, deux points, et l'espace suivant tous sur la première ligne ; le saut à la ligne ne peut intervenir qu'au delà de cette espace. Finalement, certains logiciels non conformes vont retirer les espaces et tabulations en queue d'une ligne. Donc, il peut être déconseillé de faire sauter à la ligne un en-tête après une espace ou une tabulation.

Pour un maximum de sûreté, les lignes d'en-tête DEVRAIENT se conformer à la syntaxe suivante plutôt qu'à celle du paragraphe 9.7.

en-tête = nom-d'en-tête ":" SP [contenu-d'en-tête] CRLF
 contenu-d'en-tête = [WS] jeton *([CRLF] WS jeton)

A.2 Identifiants de message

Chaque article traité par un serveur NNTP DOIT avoir un identifiant de message univoque. Pour les besoins de la présente spécification, un identifiant de message est une chaîne arbitraire opaque qui doit simplement satisfaire certaines exigences syntaxiques, et c'est juste un moyen de se référer à l'article.

Parce qu'il y a un risque significatif que de vieux articles soient réinjectés dans le système Usenet mondial, la [RFC1036] exige que les identifiants de message soient uniques au monde pour toujours.

La présente spécification déclare que les identifiants de message sont les mêmes si et seulement si ils consistent en la même séquence d'octets. D'autres spécifications peuvent définir deux séquences différentes comme étant égales parce qu'elles font une interprétation de caractères particuliers. La [RFC2822] a un concept de caractères "entre guillemets" (*quoted*) et "échappés" (*escaped*). Elle considère donc ces trois identifiants de message :

```
<ab.cd@exemple.com>
<"ab.cd"@exemple.com>
<"ab.\cd"@exemple.com>
```

comme étant identiques. Donc, une mise en œuvre de NNTP qui traite des articles de messagerie électronique doit s'assurer qu'un seul de ces trois apparaît dans le protocole et que les deux autres sont convertis en celui là comme et lorsque nécessaire, comme lorsque un client vérifie les résultats d'une commande NEWNEWS par rapport à une base de données interne d'identifiants de message. Noter que la [RFC1036] ne traite jamais deux chaînes différentes comme étant identiques. Son successeur (au moment de la rédaction) restreint la syntaxe des identifiants de message de telle sorte que, chaque fois que la RFC2822 traiterai deux chaînes comme équivalentes, une seule d'entre elles est valide (dans l'exemple ci-dessus, seule la première chaîne est valide).

La présente spécification ne décrit pas comment est déterminé l'identifiant de message d'un article ; il peut être déduit du contenu de l'article ou dérivé d'une source externe. Si le serveur se conforme aussi à une autre spécification qui contient une définition d'identifiant de message compatible avec celle-ci, le serveur DEVRAIT utiliser ces identifiants de message. Une approche commune, et une qui DEVRAIT être utilisée pour les articles de messagerie électronique et Netnews, est d'extraire l'identifiant de message du contenu d'un en-tête dont le nom est "Message-ID". Cela peut n'être pas aussi simple que de copier les contenus d'en-tête entiers ; il peut être nécessaire d'éliminer des commentaires et de défaire les guillemets, ou de réduire des identifiants de message "équivalents" à une forme canonique.

Si un article est obtenu grâce à la commande IHAVE, un identifiant de message sera fourni avec la commande. Le serveur PEUT soit l'utiliser, soit en déterminer un à partir du contenu de l'article. Cependant, quoi qu'il fasse, il DEVRAIT s'assurer que, si la commande IHAVE est répétée avec les mêmes arguments et articles, elle sera reconnue comme dupliquée.

Si un article ne contient pas d'identifiant de message que le serveur puisse identifier, il DOIT en synthétiser un. Ce pourrait, par exemple, être un simple numéro de séquence ou être fondé sur la date et l'heure d'arrivée de l'article. Lors du traitement d'articles de messagerie électronique ou Netnews, un en-tête Message-ID DEVRAIT être ajouté pour s'assurer de la cohérence et l'unicité mondiales.

Noter que, parce que l'identifiant de message pourrait n'avoir pas été déduit de l'en-tête Message-ID de l'article, l'exemple suivant est légitime (bien qu'inhabituel) :

```
[C] HEAD <45223423@exemple.com>
[S] 221 0 <45223423@exemple.com>
[S] Path: pathost!demo!whitehouse!not-for-mail
[S] Message-ID: <1234@exemple.net>
[S] From: "Demo User" <personne@exemple.net>
[S] Newsgroups: misc.test
[S] Subject: C'est juste un article d'essai
[S] Date: 6 Oct 1998 04:38:40 -0500
[S] Organisation: Un exemple Net, Uncertain, Texas
[S] .
```

A.3 Envoi d'article

Pour autant que NNTP soit concerné, les commandes POST et IHAVE fournissent les mêmes facilités de base d'une façon légèrement différente. Cependant, elles ont des intentions assez différentes.

La commande IHAVE est destinée à la transmission d'articles conformes entre un système de serveurs NNTP, avec tous les articles peut-être aussi conformes à une autre spécification (par exemple, tous les articles sont des articles Netnews). On

s'attend à ce que le client ait déjà fait toutes les validations nécessaires (ou qu'il ait à son tour obtenu l'article d'un tiers qui l'a fait) ; donc, le contenu DEVRAIT rester inchangé.

À l'opposé, la commande POST est destinée à être utilisée lorsque un usager final injecte un article de création nouvelle dans un tel système. L'article transféré peut n'être pas un article de messagerie électronique ou Netnews conforme, et le serveur est supposé le valider, et si nécessaire, le convertir au bon format pour la distribution à venir. Ceci est souvent fait par un autre élément de logiciel sur l'installation du serveur ; si il en est ainsi, le serveur NNTP DEVRAIT passer l'article entrant inaltéré à ce logiciel, en ne faisant aucune tentative de filtrage des caractères, de saut à la ligne ou de limitation de lignes, ou de traiter autrement le texte entrant.

La commande POST peut échouer de diverses façons, et les clients devraient être prêts à envoyer à nouveau un article. Lorsque on le fait, il est cependant souvent important de s'assurer (autant que possible) que le même identifiant de message est alloué aux deux tentatives afin que le serveur, ou les autres serveurs, puissent reconnaître les deux articles comme des duplications. Dans le cas d'articles de messagerie électronique ou Netnews, les articles postés DEVRAIENT donc contenir un en-tête du nom de "Message-ID", et le contenu de cet en-tête DEVRAIT être identique à chaque tentative. Le serveur DEVRAIT s'assurer que deux articles postés avec le même contenu pour cet en-tête sont reconnus comme identiques et que le même identifiant de message est alloué, que ce contenu soit ou non convenable comme identifiant de message.

Appendice B. Sommaire des commande

Cette section contient une liste de toutes les commandes définie dans le présent document, rangées par ordre alphabétique des noms de commande, et avec l'indication de la capacité.

Ordonnées par nom de commande

Commande	Capacité indiquée	définition
ARTICLE	READER	paragraphe 6.2.1
BODY	READER	paragraphe 6.2.3
CAPABILITIES	obligatoire	paragraphe 5.2
DATE	READER	paragraphe 7.1
GROUP	READER	paragraphe 6.1.1
HDR	HDR	paragraphe 8.5
HEAD	obligatoire	paragraphe 6.2.2
HELP	obligatoire	paragraphe 7.2
IHAVE	IHAVE	paragraphe 6.3.2
LAST	READER	paragraphe 6.1.3
LIST	LIST	paragraphe 7.6.1
LIST ACTIVE.TIMES	LIST	paragraphe 7.6.4
LIST ACTIVE	LIST	paragraphe 7.6.3
LIST DISTRIB.PATS	LIST	paragraphe 7.6.5
LIST HEADERS	HDR	paragraphe 8.6
LIST NEWSGROUPS	LIST	paragraphe 7.6.6
LIST OVERVIEW.FMT	OVER	paragraphe 8.4
LISTGROUP	READER	paragraphe 6.1.2
MODE READER	MODE-READER	paragraphe 5.3
NEWGROUPS	READER	paragraphe 7.3
NEWNEWS	NEWNEWS	paragraphe 7.4
NEXT	READER	paragraphe 6.1.4
OVER	OVER	paragraphe 8.3
POST	POST	paragraphe 6.3.1
QUIT	obligatoire	paragraphe 5.4
STAT	obligatoire	paragraphe 6.2.4

Ordonnées par capacité indiquée

Commande	Capacité indiquée	Définition
CAPABILITIES	obligatoire	paragraphe 5.2
HEAD	obligatoire	paragraphe 6.2.2
HELP	obligatoire	paragraphe 7.2
QUIT	obligatoire	paragraphe 5.4
STAT	obligatoire	paragraphe 6.2.4

HDR	HDR	paragraphe 8.5
LIST HEADERS	HDR	paragraphe 8.6
IHAVE	IHAVE	paragraphe 6.3.2
LIST	LIST	paragraphe 7.6.1
LIST ACTIVE	LIST	paragraphe 7.6.3
LIST ACTIVE.TIMES	LIST	paragraphe 7.6.4
LIST DISTRIB.PATS	LIST	paragraphe 7.6.5
LIST NEWSGROUPS	LIST	paragraphe 7.6.6
MODE READER	MODE-READER	paragraphe 5.3
NEWNEWS	NEWNEWS	paragraphe 7.4
OVER	OVER	paragraphe 8.3
LIST OVERVIEW.FMT	OVER	paragraphe 8.4
POST	POST	paragraphe 6.3.1
ARTICLE	READER	paragraphe 6.2.1
BODY	READER	paragraphe 6.2.3
DATE	READER	paragraphe 7.1
GROUP	READER	paragraphe 6.1.1
LAST	READER	paragraphe 6.1.3
LISTGROUP	READER	paragraphe 6.1.2
NEWGROUPS	READER	paragraphe 7.3
NEXT	READER	paragraphe 6.1.4

Appendice C. Sommaire des codes de réponse

Cette section contient la liste de tous les codes de réponse définis dans ce document et indique si ils sont multi lignes, quelles commandes peuvent le générer, quels arguments il a, et quelle est sa signification.

Code de réponse 100 (multi lignes)

génééré par : HELP

signification : le texte d'aide suit.

Code de réponse 101 (multi lignes)

génééré par : CAPABILITIES

signification : la liste des capacités suit.

Code de réponse 111

génééré par : DATE

1 argument: aaaammjjhhmmss

signification : date et heure du serveur.

Code de réponse 200

génééré par : connexion initiale, MODE READER

signification : service disponible, envoi permis.

Code de réponse 201

génééré par : connexion initiale, MODE READER

signification : service disponible, envoi interdit.

Code de réponse 205

génééré par : QUIT

signification : clôture de la connexion (le serveur clôt immédiatement la connexion).

Code de réponse 211

Le code de réponse 211 a deux formes complètement différentes, selon la commande qui le génère :

(pas multi lignes)

génééré par : GROUP

4 arguments : numéro bas haut groupe

signification : groupe choisi.

(multi lignes)

génééré par : LISTGROUP

4 arguments : numéro bas haut groupe

signification : les numéros d'article suivent.

Code de réponse 215 (multi lignes)
génééré par : LIST
signification : des informations suivent.

Code de réponse 220 (multi lignes)
génééré par : ARTICLE
2 arguments: n identifiant de message
signification : l'article suit.

Code de réponse 221 (multi lignes)
génééré par : HEAD
2 arguments : n identifiant de message
signification : les en-têtes d'article suivent.

Code de réponse 222 (multi lignes)
génééré par : BODY
2 arguments: n identifiant de message
signification : le corps de l'article suit.

Code de réponse 223
génééré par : LAST, NEXT, STAT
2 arguments: n identifiant de message
signification : l'article existe et est choisi.

Code de réponse 224 (multi lignes)
génééré par : OVER
signification : les informations de vue d'ensemble suivent.

Code de réponse 225 (multi lignes)
génééré par : HDR
signification : les en-têtes suivent.

Code de réponse 230 (multi lignes)
génééré par : NEWNEWS
signification : la liste des nouveaux articles suit.

Code de réponse 231 (multi lignes)
génééré par : NEWGROUPS
signification : la liste des nouveaux groupes de nouvelles suit.

Code de réponse 235
génééré par : IHAVE (seconde étape)
signification : l'article a bien été transféré.

Code de réponse 240
génééré par : POST (seconde étape)
signification : article bien reçu.

Code de réponse 335
génééré par : IHAVE (première étape)
signification : envoi de l'article à transférer.

Code de réponse 340
génééré par : POST (première étape)
signification : envoi de l'article à poster.

Code de réponse 400
Réponse générique et génééré par la connexion initiale
signification : service non disponible ou plus disponible (le serveur clôt immédiatement la connexion).

Code de réponse 401
Réponse générique

1 argument : étiquette de capacité

signification : le serveur est dans le mauvais mode ; la capacité indiquée devrait être utilisée pour changer le mode.

Code de réponse 403

Réponse générique

signification : faute ou problème interne empêchant d'effectuer l'action.

Code de réponse 411

génééré par : GROUP, LISTGROUP

signification : pas de tel groupe de nouvelles.

Code de réponse 412

génééré par : ARTICLE, BODY, GROUP, HDR, HEAD, LAST, LISTGROUP, NEXT, OVER, STAT

signification : pas de groupe de nouvelles choisi.

Code de réponse 420

génééré par : ARTICLE, BODY, HDR, HEAD, LAST, NEXT, OVER, STAT

signification : le numéro d'article actuel est invalide.

Code de réponse 421

génééré par : NEXT

signification : pas de prochain article dans ce groupe.

Code de réponse 422

génééré par : LAST

signification : pas d'article précédent dans ce groupe.

Code de réponse 423

génééré par : ARTICLE, BODY, HDR, HEAD, OVER, STAT

signification : pas d'article avec de numéro ou dans cette gamme.

Code de réponse 430

génééré par : ARTICLE, BODY, HDR, HEAD, OVER, STAT

signification : pas d'article avec cet identifiant de message.

Code de réponse 435

génééré par : IHAVE (premier stade)

signification : article non voulu.

Code de réponse 436

génééré par : IHAVE (l'un ou l'autre stade)

signification : transfert impossible (premier stade) ou échoué (second stade) ; réessayer plus tard.

Code de réponse 437

génééré par : IHAVE (second stade)

signification : transfert rejeté ; ne pas réessayer.

Code de réponse 440

génééré par : POST (premier stade)

signification : postage non permis.

Code de réponse 441

génééré par : POST (second stade)

signification : échec du postage.

Code de réponse 480

Réponse générique

signification : commande indisponible jusqu'à ce que le client se soit authentifié.

Code de réponse 483

Réponse générique

signification : commande indisponible jusqu'à ce que une confidentialité convenable ait été arrangée.

Code de réponse 500
Réponse générique
signification : commande inconnue.

Code de réponse 501
Réponse générique
signification : erreur de syntaxe dans la commande.

Code de réponse 502
Réponse générique et générée par la connexion initiale
Signification pour la connexion initiale et la commande MODE READER : service indisponible en permanence (le serveur clôt immédiatement la connexion).
Signification pour toutes les autres commandes : commande non permise (et le client n'a aucun moyen de la changer).

Code de réponse 503
Réponse générique
signification : caractéristique non prise en charge.

Code de réponse 504
Réponse générique
signification : erreur de codage base64 [RFC4648] d'un argument.

Appendice D. Changements depuis la RFC 977

En général tout a été fait pour s'assurer que la spécification de protocole dans le présent document est compatible avec la version spécifiée dans la [RFC977] et les diverses facilités adoptées de la [RFC2980]. Cependant, il y a eu un certain nombre de changements, certains compatibles et d'autres pas.

Le présent appendice fait la liste de ces changements. Il n'est pas garanti qu'il soit exhaustif ou correct et on NE DOIT PAS s'y fier.

- o Une spécification de syntaxe formelle (Section 9) a été ajoutée.
- o Le jeu de caractères par défaut est changé de l'US-ASCII [ANSI1986] à l'UTF-8 [RFC3629] (noter que l'US-ASCII est un sous ensemble de l'UTF-8). Cette question est plus discutée à la Section 10.
- o Il est exigé que tous les articles aient un identifiant de message, éliminant le fourre tout "<0>" utilisé par la RFC 977 dans certaines réponses.
- o Les capacités de correspondance de nom de groupe de nouvelles déjà mentionnées dans la RFC 977 ("wildmats", Section 4) sont précisées et étendues. Les nouvelles facilités (par exemple, l'utilisation de virgules et de points d'exclamation) sont permises chaque fois que des wildmats apparaissent dans le protocole.
- o La prise en charge du traitement en parallèle des commandes (paragraphe 3.5) est rendue obligatoire.
- o Les principes des codes de réponses (paragraphe 3.2) ont été ordonnés. En particulier :
 - * La famille de codes de réponse x8x, anciennement utilisée pour les extensions privées, est maintenant réservée pour les extensions d'authentification et de confidentialité ;
 - * la famille de codes de réponse x9x, anciennement destinée aux facilités de débogage, est maintenant réservée aux extensions privées ;
 - * les codes de réponse génériques 502 et 503 (paragraphe 3.2.1) ont été redéfinis ;
 - * les codes de réponse génériques nouveaux 401, 403, 480, 483, et 504 ont été ajoutés.
- o Les règles pour le numérotage d'articles (Section 6) ont été précisées (voir aussi au paragraphe 6.1.1.2).
- o La commande SLAVE (qui était mal définie) est retirée du protocole.
- o Les années à quatre chiffres sont permises dans les commandes NEWNEWS (paragraphe 7.4) et NEWGROUPS (paragraphe 7.3) (les années à deux chiffres sont toujours permises). Les paramètres facultatifs de distribution de ces commandes ont été supprimés.
- o La commande LIST (paragraphe 7.6.1) est largement étendue ; l'original est disponible comme LIST ACTIVE, tandis que la nouvelle variante inclut ACTIVE.TIMES, DISTRIB.PATS, et NEWSGROUPS. Un nouveau fanion d'état "m" est ajouté à la réponse à LIST ACTIVE.
- o Une nouvelle commande CAPABILITIES (paragraphe 5.2) permet aux clients de déterminer quelles facilités sont prises en charge par un serveur.
- o La commande DATE (paragraphe 7.1) est adoptée de la RFC 2980 sans changement.
- o La commande LISTGROUP (paragraphe 6.1.2) est adoptée de la RFC 2980. Un argument facultatif "range" a été ajouté, et la ligne de réponse initiale 211 a maintenant le même format que la réponse 211 de la commande GROUP.
- o La commande MODE READER (paragraphe 5.3) est adoptée de la RFC 2980 et sa signification et ses effets sont précisés.

- o La commande XHDR de la RFC 2980 a été formalisée comme nouvelles commandes HDR (paragraphe 8.5) et LIST HEADERS (paragraphe 8.6).
- o La commande XOVER de la RFC 2980 a été formalisée comme nouvelles commandes OVER (paragraphe 8.3) et LIST OVERVIEW.FMT (paragraphe 8.4). L'ancienne peut être appliquée à un identifiant de message ainsi qu'à une gamme.
- o Le concept de métadonnées d'article (paragraphe 8.1) a été formalisé, permettant aux pseudo en-têtes Bytes et Lines d'être déconseillés.

Les auteurs de clients devraient noter en particulier que la non prise en charge de la commande CAPABILITIES est un bon indicateur que le serveur ne prend pas en charge la présente spécification.

Adresse de l'auteur

Clive D.W. Feather
THUS plc
322 Regents Park Road
London
N3 2QQ
United Kingdom

téléphone : +44 20 8495 6138
Fax : +44 870 051 9937
mél : clive@demon.net
URI : <http://www.davros.org/>

Déclaration complète de droits de reproduction

Copyright (C) The Internet Society (2006).

Le présent document est soumis aux droits, licences et restrictions contenus dans le BCP 78, et à www.rfc-editor.org, et sauf pour ce qui est mentionné ci-après, les auteurs conservent tous leurs droits.

Le présent document et les informations contenues sont fournis sur une base "EN L'ÉTAT" et le contributeur, l'organisation qu'il ou elle représente ou qui le/la finance (s'il en est), la INTERNET SOCIETY et la INTERNET ENGINEERING TASK FORCE déclinent toutes garanties, exprimées ou implicites, y compris mais non limitées à toute garantie que l'utilisation des informations ci-encloses ne violent aucun droit ou aucune garantie implicite de commercialisation ou d'aptitude à un objet particulier.

Propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourraient être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur le répertoire en ligne des IPR de l'IETF à <http://www.ietf.org/ipr> .

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org .

Remerciement

Le financement de la fonction d'édition des RFC est actuellement fourni par l'Internet Society