Groupe de travail Réseau
Request for Comments : 1123
STD 3
Traduction Claude Brière de L'Isle

Internet Engineering Task Force
R. Braden, éditeur
octobre 1989

Exigences pour les hôtes Internet – Application et prise en charge

Statut du présent mémoire
La présente RFC est une spécification officielle pour la communauté de l'Internet. Elle incorpore par références, amende, corrige et complète les documents de normalisation du protocole principal qui se rapportent aux hôtes. La distribution du présent document n'est soumise à aucune restriction.

Résumé
La présente RFC fait partie d'une paire de documents qui définissent et discutent les exigences pour le logiciel des hôtes Internet. Elle traite des protocoles d'application et de prise en charge ; sa compagne, la RFC 1122 traite des couches de protocoles de communication : couche de liaison, couche IP, et couche transport.

Table des matières

1. Introduction

Le présent document fait partie d'une paire de documents qui définissent et exposent les exigences pour les mises en œuvre de système d'hôte de la suite des protocoles de l'Internet. La présente RFC couvre la couche d'application et les protocoles de soutien. Sa RFC associée, "Exigences pour les hôtes Internet – Couches de communications" [INTRO:1] couvre les protocoles de couche inférieures : couche de transport, couche IP et couche de liaison.

Ces documents sont destinés à fournir des lignes directrices aux fabricants, développeurs et utilisateurs des logiciels de communication Internet. Ils représentent le consensus d'un large corpus d'expérience technique et de réflexion, auquel ont contribué les membres des communautés Internet de la recherche et de l'industrie.

La présente RFC énumère les protocoles standard que doit utiliser un hôte connecté à l'Internet, et elle incorpore par référence les RFC et autres documents qui décrivent les spécifications actuelles de ces protocoles. Elle corrige des erreurs dans les documents référencés et ajoute des exposés et lignes directrices supplémentaires pour un développeur.

Pour chaque protocole, le présent document contient aussi un ensemble explicite d'exigences, recommandations, et options. Le lecteur doit comprendre que la liste des exigences de ce document est incomplète par elle-même ; l'ensemble complet des exigences pour un hôte Internet est principalement défini dans les documents de spécification de protocole standard, avec les corrections, amendements, et suppléments contenus dans la présente RFC.

Une mise en œuvre de bonne foi des protocoles qui ont été produits après une lecture attentive de la RFC et avec une certaine interaction avec la communauté technique de l'Internet, et qui suit de bonnes pratiques d'ingénierie de logiciel de communication, ne devrait différer que de façon mineure des exigences du présent document. Et donc, dans de nombreux cas, les "exigences" de la présente RFC sont déjà établies ou impliquées dans les documents de protocole standard, de sorte que leur inclusion ici est, en un sens, redondante. Cependant, elles ont été incluses parce que certaines mises en œuvre ont fait dans le passé de mauvais choix, causant des problèmes d'interopérabilité, de performance, et/ou de robustesse.

Le présent document inclut des discussions et explications sur beaucoup des exigences et recommandations. Une simple liste des exigences serait dangereuse, parce que:

- Certaines dispositions exigées sont plus importantes que d'autres, et certaines dispositions sont facultatives.

- Il peut y avoir des raisons valides pour que le produit d'un fabricant particulier conçu pour un contexte restrictif choisisse d'utiliser des spécifications différentes.

Cependant, les spécifications du présent document doivent être suivies pour satisfaire au but général d'interopération d'hôtes choisis arbitrairement à travers la diversité et la complexité du système Internet. Bien que la plupart des mises en œuvre actuelles ne réussissent pas à satisfaire de diverses façons à ces exigences, certaines mineures, certaines majeures, la présente spécification est l'idéal vers lequel nous devons tendre.

Ces exigences se fondent sur le niveau actuel de l'architecture Internet. Le présent document sera mis à jour en tant que de besoin pour fournir des éclaircissements supplémentaires ou pour inclure des information additionnelles dans les domaines dans lesquels les spécifications évoluent encore.

Cette section introductive commence par un avis général aux fabricants de logiciel d'hôte, puis donne des indications sur la lecture du reste du document. La section 2 contient les exigences générales qui peuvent être applicables à tous les protocoles d'application et de soutien. Les sections 3, 4, et 5 contiennent les exigences pour les protocoles des trois applications majeures, respectivement Telnet, transfert de fichiers, et messagerie électronique. La section 6 couvre les applications de soutien : le système des noms de domaine, l'initialisation de système, et la gestion. Finalement, toutes les références figurent à la section 7.

1.1 L'architecture de l'Internet

On trouvera une brève introduction à l'architecture de l'Internet du point de vue hôte au paragraphe 1.1 de [INTRO:1]. Ce paragraphe contient aussi les références recommandées sur les fondements de l'architecture de l'Internet.

1.2 Considérations générales

Les fabricants de logiciels d'hôte de l'Internet ont tiré deux importantes leçons que les nouveaux fabricants devraient considérer avec attention.

1.2.1 Continuer l'évolution de l'Internet

L'énorme croissance de l'Internet a révélé des problèmes de gestion et d'échelle dans un grand système de communications fondé sur le datagramme. Ces problèmes sont en voie de résolution et il en résulte qu'il y aura une évolution continue des spécifications décrites dans le présent document. Ces changements seront soigneusement planifiés et contrôlés, car il y a une participation massive des fabricants et des organisations chargées du fonctionnement des réseaux à cette planification.

Développement, évolution, et révision sont les caractéristiques des protocoles de réseau informatique d'aujourd'hui, et cette situation va persister pour plusieurs années. Un fabricant qui développe des logiciels de communication d'ordinateurs pour la suite des protocoles de l'Internet (ou toute autre suite de protocoles !) et échoue ensuite à entretenir et mettre à jour ces logiciels lorsque les spécifications changent, va laisser des cohortes de consommateurs mécontents. L'Internet est un grand réseau de communication, et les usagers sont en constant contact à travers lui. L'expérience montre que la connaissance des déficiences du logiciel d'un fabricant se propage rapidement à travers la communauté technique de l'Internet.

1.2.2 Principe de robustesse

À toutes les couches des protocoles, il y a une règle générale dont l'application peut conduire à d'énormes bénéfices en robustesse et en interopérabilité.

"Soyez libéraux dans ce que vous acceptez, et conservateur dans ce que vous envoyez."

Un logiciel devrait être écrit de façon à traiter toutes les erreurs imaginables, même invraisemblables ; tôt ou tard, viendra un paquet avec cette combinaison particulière d'erreurs et d'attributs, et sauf si le logiciel y est préparé, le chaos peut s'ensuivre. En général, le mieux est de supposer que le réseau est plein d'entités malveillantes qui vont envoyer des paquets conçus pour avoir le pire effet possible. Cette hypothèse va conduire à des concepts de protection convenables, bien que les problèmes les plus sérieux sur l'Internet aient été causés par des mécanismes non envisagés déclenchés par des événements à faible probabilité ; la simple malice de l'homme n'aurait jamais pu suivre un cours aussi vicieux !

L'adaptabilité au changement doit être conçue à tous les niveaux du logiciel d'hôte Internet. Un simple exemple est de considérer une spécification de protocole qui contient une énumération de valeurs pour un champ d'en-tête particulier -- par exemple, un champ de type, un numéro d'accès, ou un code d'erreur ; cette énumération doit être supposée incomplète. Et donc, si une spécification de protocole définit quatre codes d'erreur possibles, le logiciel ne doit pas lâcher si un cinquième code apparaît. Un code indéfini pourrait être noté dans un journal de bord (voir ci-dessous) mais il ne doit pas causer de défaillance.

La seconde partie du principe est presque aussi importante : les logiciels sur les autres hôtes peuvent contenir des imperfections qui rendent peu raisonnable l'exploitation de dispositifs de protocole légaux mais obscurs. Il est malavisé de s'éloigner du simple et de l'évident, de peur que des effets malencontreux n'en résultent ailleurs. Un corollaire de ce conseil est de "surveiller les hôtes qui se conduisent mal" ; les logiciels d'hôtes devraient être prêts, non seulement à survivre à la présence d'hôtes au mauvais comportement, mais aussi à coopérer pour limiter la quantité de perturbations que de tels hôtes peuvent causer aux facilités de communications partagées.

1.2.3 Journalisation des erreurs

L'Internet inclut une grande variété de systèmes d'hôtes et de passerelles, chacun mettant en œuvre de nombreux protocoles et couches de protocole, dont certains contiennent des erreurs et des caractéristiques erronées dans le logiciel de protocole Internet. Par suite de la complexité, de la diversité, et de la distribution des fonctions, le diagnostic des problèmes de l'Internet est souvent très difficile.

Le diagnostic des problèmes sera facilité si la mise en œuvre d'hôte comporte un dispositif bien conçu pour enregistrer les événements de protocole erronés ou "étranges". Il est important d'inclure autant d'informations de diagnostic que possible lors de l'enregistrement d'une erreur. En particulier, il est souvent utile d'enregistrer le ou les en-têtes d'un paquet qui a causé une erreur. Cependant, il faut veiller à s'assurer que l'enregistrement des erreurs ne consomme pas des quantités prohibitives de ressources ou n'interfère pas par ailleurs avec le fonctionnement de l'hôte.

Il y a une tendance à submerger les fichiers d'enregistrement d'erreur avec des événements de protocole anormaux mais anodins ; ceci peut être évité en utilisant un journal "circulaire", ou en ne permettant l'enregistrement que sur diagnostic d'une défaillance reconnue. Cela peut être utile de filtrer et compter les messages dupliqués successifs. Une stratégie qui semble bien fonctionner est de : (1) toujours compter les anomalies et de rendre un tel compte accessibles par le protocole de gestion (voir [INTRO:1]) ; et (2) de permettre sélectivement l'enregistrement d'une grande variété d'événements. Par exemple, il peut être utile d'avoir la capacité de "tout enregistrer" ou de "tout enregistrer pour l'hôte X".

Noter que des gestions différentes peuvent avoir des politiques qui diffèrent quant à la quantité d'enregistrements d'erreurs qu'elles veulent normalement permettre sur un hôte. Certains diront, "si cela ne me fait pas de tort, je ne veux pas en entendre parler", alors que d'autres voudront avoir une attitude plus attentive et agressive quant à la détection et la suppression des anomalies de protocole.

1.2.4 Configuration

L'idéal serait qu'une mise en œuvre d'hôte de la suite des protocoles Internet puisse être entièrement auto-configurable. Cela permettrait que la totalité de la suite soit mise en œuvre sur un CD-ROM ou gravée dans le silicium, cela simplifierait les stations de travail sans disque, et ce serait un immense avantage pour les administrateurs de LAN débordés aussi bien que pour les fabricants. Cet idéal n'a pas été atteint, et en fait nous en sommes loin.

En de nombreux points du présent document, on trouvera l'exigence qu'un paramètre soit une option configurable. Il y a plusieurs raisons à une telle exigence. Dans quelques cas, il y a actuellement une incertitude ou un désaccord sur ce qui serait la meilleure valeur, et il pourra être nécessaire de mettre à jour la valeur recommandée à l'avenir. Dans d'autres cas, la valeur dépend réellement de facteurs externes -- par exemple, de la taille de l'hôte et de la distribution de ses charges de communication, ou de la vitesse et de la topologie des réseaux voisins – et les algorithmes auto-réglables sont indisponibles et peut-être insuffisants. Dans certains cas, la possibilité de configurer est nécessaire à cause d'exigences administratives.

Finalement, certaines options de configuration sont nécessaires pour communiquer avec des mises en œuvre obsolètes ou incorrectes des protocoles, distribuées sans sources, qui persistent malheureusement dans de nombreuses parties de l'Internet. Pour faire coexister les systèmes corrects avec ces systèmes fautifs, les administrateurs ont souvent à "dis-configurer" les systèmes corrects. Ce problème se corrigera de lui-même graduellement avec l'éviction des systèmes fautifs, mais il ne peut être ignoré par les fabricants.

Lorsque nous disons qu'un paramètre doit être configurable, nous n'avons pas l'intention d'exiger que sa valeur soit lue explicitement à partir d'un fichier de configuration à chaque amorçage. Nous recommandons que les développeurs établissent une valeur par défaut pour chaque paramètre, de sorte qu'un fichier de configuration ne soit nécessaire que pour outrepasser ces valeurs par défaut lorsqu'elles sont inappropriées dans une installation particulière. Et donc, l'exigence de configurabilité est une assurance qu'il sera POSSIBLE d'outrepasser la valeur par défaut quand nécessaire, même dans un produit seulement binaire ou fondé sur une mémoire en lecture seule.

Le présent document exige une valeur particulière par défaut dans certains cas. Le choix des valeurs par défaut est une question sensible lorsque l'élément de configuration contrôle l'adaptation aux systèmes fautifs existants. Si l'Internet doit réussir à converger pour réaliser l'interopérabilité, les valeurs par défaut incorporées dans les mises en œuvre doivent suivre le protocole officiel, et non les "dis-configurations" pour s'accommoder des mises en œuvre fautives. Bien que des considérations commerciales aient conduit certains fabricants à choisir des valeurs par défaut dis-configurées, nous invitons les fabricants à choisir des valeurs par défaut conformes à la présente norme.

Finalement, on note que le fabricant a besoin de fournir une documentation adéquate sur tous les paramètres de configuration, leurs limites et leurs effets.

1.3 Lecture du document

1.3.1 Organisation

En général, chaque section majeure du présent document est organisée avec les paragraphes suivants :

(1) Introduction

(2) Survol du protocole -- considère les documents de spécification de protocole section par section, en corrigeant les erreurs, établissant les exigences qui pourraient être ambiguës ou mal définies, et en fournissant des éclaircissements ou des explications.

(3) Questions spécifiques – expose les questions de conception et de mise en œuvre du protocole qui ne sont pas incluses dans le survol du protocole.

(4) Interfaces – expose l'interface de service avec la prochaine couche supérieure.

(5) Résumé -- contient un résumé des exigences de la section.

Sur de nombreux sujets individuels du présent document, il y a des paragraphes entre parenthèses marqués "Discussion" ou "Mise en œuvre". Ces paragraphes sont destinés à apporter des éclaircissements et des explications au texte de l'exigence qui les précède. Ils comportent aussi des suggestions sur des directions ou développements futurs possibles. Les éléments de mise en œuvre contiennent des suggestions d'approches qu'un développeur pourrait vouloir prendre en considération.

Les paragraphes de résumé sont destinés à servir de guides et d'index pour le texte, mais sont nécessairement elliptiques et incomplets. Les résumés ne devraient jamais être utilisés ou référencés séparément de la RFC complète.

1.3.2 Exigences

Dans le présent document, les mots qui sont utilisés pour définir la signification de chaque exigence particulière sont en majuscules.

Ces mots sont :

* "DOIT"
  Ce mot ou l'adjectif "EXIGÉ" signifie que l'élément est une exigence absolue de la spécification.

* "DEVRAIT"
  Ce mot ou l'adjectif "RECOMMANDÉ" signifie qu'il peut exister des raisons valides dans des circonstances particulières pour ignorer cet élément, mais les pleines implications devraient en être comprises et le cas soigneusement soupesé avant de choisir une voie différente.

* "PEUT"
  Ce mot ou l'adjectif "FACULTATIF" signifie que cet élément est vraiment facultatif. Un fabricant peut choisir d'inclure l'élément parce que par exemple, un marché particulier l'exige ou parce qu'il améliore le produit ; un autre fabricant peut omettre le même élément.

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

1.3.3 Terminologie

Le présent document utilise les termes techniques suivants :

Segment : Un segment est l'unité de transmission de bout en bout dans le protocole TCP. Un segment consiste en un en-tête TCP suivi des données d'application. Un segment est transmis par encapsulation au sein d'un datagramme IP.

Message : Ce terme est utilisé par certains protocoles de couche d'application (en particulier SMTP) pour une unité de données d'application.

Datagramme : Un datagramme [UDP] est l'unité de transmission de bout en bout dans le protocole UDP.

Multi rattachement : Un hôte est dit être à rattachement multiple (ou multi rattachements) si il a plusieurs adresses IP sur des réseaux connectés.

1.4 Remerciements

Le présent document incorpore des contributions et commentaires provenant d'un large groupe d'experts du protocole Internet, comprenant des représentants des universités et laboratoires de recherche, de fabricants, et d'agences

gouvernementales. Il a été principalement confectionné par le groupe de travail Exigences pour les hôtes de l'Équipe d'ingénierie de l'Internet (IETF).

L'éditeur remercie tout spécialement le dévouement infatigable des personnes suivantes, qui ont participé à de nombreuses et longues réunions et généré 3 millions d'octets de messagerie électronique pendant les 18 derniers mois d'élaboration de ce document : Philip Almquist, Dave Borman (Cray Research), Noel Chiappa, Dave Crocker (DEC), Steve Deering (Stanford), Mike Karels (Berkeley), Phil Karn (Bellcore), John Lekashman (NASA), Charles Lynn (BBN), Keith McCloghrie (TWG), Paul Mockapetris (ISI), Thomas Narten (Purdue), Craig Partridge (BBN), Drew Perkins (CMU), et James Van Bokkelen (FTP Software).

De plus, les personnes suivantes ont apporté des contributions majeures à l'effort commun : Bill Barns (Mitre), Steve Bellovin (AT&T), Mike Brescia (BBN), Ed Cain (DCA), Annette DeSchon (ISI), Martin Gross (DCA), Phill Gross (NRI), Charles Hedrick (Rutgers), Van Jacobson (LBL), John Klensin (MIT), Mark Lottor (SRI), Milo Medin (NASA), Bill Melohn (Sun Microsystems), Greg Minshall (Kinetics), Jeff Mogul (DEC), John Mullen (CMC), Jon Postel (ISI), John Romkey (Epilogue Technology), et Mike StJohns (DCA).

Les personnes suivantes ont aussi apporté des contributions significatives dans des domaines particuliers : Eric Allman (Berkeley), Rob Austein (MIT), Art Berggreen (ACC), Keith Bostic (Berkeley), Vint Cerf (NRI), Wayne Hathaway (NASA), Matt Korn (IBM), Erik Naggum (Naggum Software, Norway), Robert Ullmann (Prime Computer), David Waitzman (BBN), Frank Wancho (USA), Arun Welch (Ohio State), Bill Westfield (Cisco), et Rayan Zachariassen (Toronto).

Nos remerciements à tous, y compris à tous les contributeurs qui ont été malencontreusement omis de cette liste.

2 Questions générales

Cette section contient les exigences générales qui peuvent être applicables à tous les protocoles de couche application.

2.1 Noms et numéros d'hôtes

La syntaxe d'un nom d'hôte Internet légal est spécifiée dans la RFC-952 [DNS:4]. Un aspect de la syntaxe de nom d'hôte est modifié ici : la restriction sur le premier caractère est relâchée pour permettre une lettre ou un chiffre. Un logiciel d'hôte DOIT accepter cette syntaxe plus libérale.

Les logiciels d'hôte DOIVENT traiter les noms d'hôte jusqu'à 63 caractères et DEVRAIENT traiter les noms d'hôte jusqu'à 255 caractères.

Chaque fois qu'un utilisateur frappe en entrée l'identité d'un hôte Internet, il DEVRAIT être possible d'entrer soit (1) un nom de domaine d'hôte soit (2) une adresse IP en forme décimale séparée par des points ("#.#.#.#"). L'hôte DEVRAIT vérifier la syntaxe de la chaîne en forme de numéro en décimal séparé par des points avant de la chercher dans le système des noms de domaine.

Discussion :
Cette dernière exigence n'est pas destinée à spécifier la forme syntaxique complète pour l'entrée d'un numéro d'hôte en décimal séparé par des points ; ceci est considéré comme une question relevant de l'interface utilisateur. Par exemple, un numéro en décimal séparé par des points doit être entre crochets "[ ]" pour la messagerie SMTP (voir le paragraphe 5.2.17). Cette notation pourrait être rendue universelle au sein d'un système d'hôte, simplifiant la vérification syntaxique pour un numéro en décimal séparé par des points.

Si un numéro en décimal séparé par des points peut être entré sans de tels délimiteurs identificateurs, on doit alors faire une vérification syntaxique complète, parce qu'un segment d'un nom de domaine d'hôte peut maintenant commencer par un chiffre et pourrait légalement être entièrement numérique (voir au paragraphe 6.1.2.4). Cependant, un nom d'hôte valide ne peut jamais avoir la forme décimale séparée par des points #.#.#.#, car au moins l'étiquette du composant de plus haut niveau sera alphabétique.

2.2 Utilisation du Service de noms de domaine

Les noms de domaine d'hôtes DOIVENT être traduits en adresses IP comme décrit au paragraphe 6.1.

Les applications qui utilisent les services de nom de domaine DOIVENT être capables de s'accommoder de conditions d'erreur légère. Les applications DOIVENT attendre un intervalle raisonnable entre des essais successifs dus à une erreur légère, et DOIVENT permettre la possibilité que des problèmes réseau puissent refuser le service pendant des heures ou même des jours.

Une application NE DEVRAIT PAS s'appuyer sur la capacité à localiser un enregistrement de ressource (RR, *Resource Record*) WKS contenant une liste précise de tous les services à l'adresse d'un hôte particulier, car le type de RR WKS n'est pas souvent utilisé par les sites Internet. Pour confirmer qu'un service est présent, essayer simplement de l'utiliser.

2.3 Applications sur les hôtes multi rattachement

Lorsque l'hôte distant est à rattachement multiple, la traduction de nom en adresse va retourner une liste d'adresses IP de remplacement. Comme spécifié au paragraphe 6.1.3.4, cette liste devrait être en ordre de préférence décroissante. Une mise en œuvre de protocoles d'application DEVRAIT être prête à essayer plusieurs adresses de la liste avant de réussir. Des exigences plus spécifiques de SMTP sont données au paragraphe 5.3.4.

Lorsque l'hôte local est à rattachement multiple, une demande/réponse d'application fondée sur UDP DEVRAIT envoyer la réponse avec la même adresse IP de source que l'adresse de destination spécifique du datagramme de demande UDP. L'"adresse de destination spécifique" est définie au paragraphe "Adressage IP" de la RFC sœur [INTRO:1].

De même, une application de serveur qui ouvre plusieurs connexions TCP avec le même client DEVRAIT utiliser la même adresse IP locale pour toutes.

2.4 Type-de-Service (TOS)

Les applications DOIVENT choisir des valeurs de TOS appropriées lorsqu'elles invoquent des services de couche transport, et ces valeurs DOIVENT être configurables. Noter qu'une valeur de TOS contient 5 bits, dont seuls les trois bits de plus fort poids sont actuellement définis ; les deux autres bits DOIVENT être à zéro.

Discussion :
Lorsque des algorithmes de routeurs sont développés pour mettre en œuvre le Type-de-Service, les valeurs recommandées pour divers protocoles d'application peuvent changer. De plus, il est vraisemblable que des combinaisons particulières d'utilisateurs et de chemins Internet voudront des valeurs de TOS non standard. Pour ces raisons, les valeurs de TOS doivent être configurables.

Voir dans la dernière version de la RFC "Numéros alloués" [INTRO:5] les valeurs de TOS recommandées pour les protocoles d'application majeurs.

2.5 Résumé des exigences d'application générale

Caractéristique	parag.	DOIT	DEVRAIT	NE DEVRAIT PAS
Interfaces d'utilisateur :				
Permettre au nom d'hôte de commencer par des chiffres	2.1	x		
Nom d'hôte jusqu'à 635 caractères	2.1	x		
Nom d'hôte jusqu'à 255 caractères	2.1		x	
Numéro d'hôte en décimal séparé par des points accepté	2.1		x	
Commence par la vérif. syntaxique du déc. séparé par des points	2.1		x	
Transposition des noms de domaines selon le § 6.1	2.2	x		
S'accommode d'erreurs DNS légères	2.2	x		
Intervalle raisonnable entre les essais	2.2	x		
Permet de longues pannes	2.2	x		
S'attend à ce que les enregistrements WKS soient disponibles	2.2			x
Essaye plusieurs adresses d'hôte distant à rattachement multiple	2.3		x	
Adr src de réponse UDP est adresse spécif. de dest de la demande	2.3		x	
Utilise la même adr IP pour les connexions TCP en rapport	2.3		x	
Spécifie les valeurs de TOS appropriées	2.4	x		
Valeurs de TOS configurables	2.4	x		
Bits de TOS inutilisés à zéro	2.4	x		

3 Connexion distante – Protocole Telnet

3.1 Introduction

Telnet est le protocole standard d'application Internet pour la connexion à distance. Il fournit les règles de codage pour relier un clavier/écran d'usager à un système client ("utilisateur") avec un interprète de commandes sur un système de serveur distant. Un sous-ensemble du protocole Telnet est aussi incorporé au sein d'autres protocoles d'application, par exemple, FTP et SMTP.

Telnet utilise une seule connexion TCP, et son flux de données normal (en mode "terminal réseau virtuel" NVT, *Network Virtual Terminal*) est d'ASCII en 7 bits avec des séquences d'échappement pour incorporer des fonctions de commande. Telnet permet aussi la négociation de nombreux modes et fonctions facultatifs.

La principale spécification Telnet se trouve dans la RFC-854 [TELNET:1], alors que les options sont définies dans de nombreuses autres RFC ; voir les références à la Section 7.

3.2 Survol du protocole

3.2.1 Négociation d'option : RFC-854, pp. 2-3

Toute mise en œuvre Telnet DOIT inclure le mécanisme de négociation et sous-négociation d'option [TELNET:2].

Un hôte DOIT suivre attentivement les règles de la RFC-854 pour éviter les boucles de négociation d'option. Un hôte DOIT refuser (c'est-à-dire, répondre WONT/DONT à un DO/WILL) une option non prise en charge. La négociation d'option DEVRAIT continuer à fonctionner (même si toutes les demandes sont refusées) tout au long de la durée de vie d'une connexion Telnet.

Si toutes les négociations d'option échouent, une mise en œuvre Telnet DOIT par défaut prendre en charge un NVT.

Discussion :
Bien que des "terminaux" plus sophistiqués et la prise en charge de négociations d'options deviennent la norme, toutes les mises en œuvre doivent être prêtes à prendre en charge un NVT pour toute communication usager-serveur.

3.2.2 Fonction Telnet Go-Ahead : RFC-854, p. 5, et RFC-858

Sur un hôte qui n'envoie jamais la commande Telnet Go Ahead (GA) le serveur Telnet DOIT essayer de négocier l'option Suppress Go Ahead (c'est-à-dire, envoyer "WILL Suppress Go Ahead"). Un usager ou serveur Telnet DOIT toujours accepter la négociation de l'option Suppress Go Ahead.

Lorsqu'elle pilote un terminal bidirectionnel pour lequel GA n'a pas de signification, une mise en œuvre d'usager Telnet PEUT ignorer les commandes GA.

Discussion :
Les terminaux en semi-duplex qui font une ligne à la fois ("clavier bloqué") pour lesquels le mécanisme Go-Ahead a été conçu, ont largement disparu de la scène. Il s'est révélé qu'il était difficile de mettre en œuvre l'envoi du signal Go-Ahead dans de nombreux systèmes d'exploitation, même dans des systèmes qui prennent en charge d'origine les terminaux semi-duplex. La difficulté est normalement que le code serveur Telnet n'a pas accès aux informations sur le fait que le processus d'utilisateur est bloqué en attendant une entrée de la part de la connexion Telnet, c'est-à-dire, il ne peut fiablement déterminer quand envoyer une commande GA. Donc, la plupart des hôtes de serveur Telnet n'envoient pas de commandes GA.

L'effet des règles de la présente section est de permettre à l'une ou l'autre extrémité d'une connexion Telnet d'interdire l'utilisation des commandes GA.

Il y a une classe de terminaux semi-duplex qui est toujours commercialement importante : les "terminaux d'entrée de données,"qui interagissent en plein écran. Cependant, la prise en charge des terminaux d'entrée de données qui utilisent le protocole Telnet n'exige pas le signal Go Ahead ; voir au paragraphe 3.3.2.

3.2.3 Fonctions de commande : RFC-854, pp. 7-8

La liste des commandes Telnet a été étendue pour y inclure EOR (End-of-Record, *fin d'enregistrement*) avec le code 239 [TELNET:9].

Aussi bien les utilisateurs que les serveurs Telnet PEUVENT prendre en charge les fonctions de commande EOR, EC, EL, et Break, et DOIVENT prendre en charge AO, AYT, DM, IP, NOP, SB, et SE.

Un hôte DOIT être capable de recevoir et ignorer toute fonction de commande Telnet qu'il ne prend pas en charge.

Discussion :
Noter qu'un serveur Telnet est obligé de prendre en charge la fonction IP (Interrupt Process, *interrompre le processus*) Telnet, même si le serveur hôte a une fonction de flux équivalente (par exemple, Control-C dans de nombreux systèmes). La fonction IP Telnet peut être plus forte qu'une commande d'interruption dans le flux, à cause de l'effet hors bande des données urgentes TCP.

La fonction de commande EOR peut être utilisée pour délimiter le flux. Une application importante est la prise en charge du terminal d'entrée de données (voir au paragraphe 3.3.2). On pouvait craindre que comme EOR n'avait pas été définie dans la RFC-854, un hôte non préparé à ignorer correctement les commandes Telnet inconnues pourrait avoir une défaillance à réception d'une EOR. Pour protéger de tels hôtes, l'option End-of-Record [TELNET:9] a été introduite ; cependant, un programme Telnet correctement mis en œuvre n'aura pas besoin d'une telle protection.

3.2.4 Signal Telnet "Synch" : RFC-854, pp. 8-10

Lorsqu'il reçoit des données TCP "urgentes", un usager ou serveur Telnet DOIT éliminer toutes les données excepté les commandes Telnet jusqu'à atteindre l'état DM *(fin d'urgence)*.

Lors de l'envoi du IP *(Interrompre le processus)* Telnet, un usager Telnet DEVRAIT le faire suivre de la séquence Telnet "Synch", c'est-à-dire, envoyer comme données TCP urgentes la séquence "IAC IP IAC DM". Le pointeur TCP d'urgence pointe sur l'octet DM.

Lorsqu'il reçoit une commande IP Telnet, un serveur Telnet PEUT renvoyer une séquence Telnet "Synch" à l'usager, pour purger le flux de sortie. Le choix devrait être cohérent avec la façon dont le système d'exploitation du serveur se comporte lorsqu'un usager local interrompt un processus.

Lorsqu'il reçoit une commande AO Telnet, un serveur Telnet DOIT renvoyer une séquence Telnet "Synch" à l'usager, pour purger le flux de sortie.

Un usager Telnet DEVRAIT avoir la capacité de purger les sorties lorsqu'il envoie une commande IP Telnet ; voir aussi au paragraphe 3.4.5.

Discussion :
Trois façons de purger le flux des données de sorties du serveur sont possibles pour un usager Telnet :

(1) Envoi de AO après IP.
Cela causera l'envoi par le serveur hôte d'un signal "purger-les-sorties-en-anté-mémoire" à son système d'exploitation. Cependant, la AO peut n'avoir pas d'effet local, c'est-à-dire, stopper les sorties de terminal à l'extrémité usager Telnet, jusqu'à ce que le serveur Telnet ait reçu et traité la AO et renvoyé un "Synch".

(2) Envoi de DO TIMING-MARK [TELNET:7] après IP, et éliminer localement toutes les sorties jusqu'à réception d'un WILL/WONT TIMING-MARK de la part du serveur Telnet.
Comme le DO TIMING-MARK sera traité au serveur après la commande IP, la réponse qui lui est faite devrait être au bon endroit dans le flux des données sortantes. Cependant, le TIMING-MARK ne va pas envoyer de signal "purger les sorties en anté-mémoire" au système d'exploitation du serveur. Que cela soit ou non nécessaire dépend du système serveur.

(3) Faire les deux.
La meilleure méthode n'est pas entièrement évidente, car on doit s'accommoder d'un certain nombre de serveurs hôtes existants qui ne suivent pas les normes Telnet de diverses façons. L'approche la plus sûre est probablement de fournir à l'usager l'option de choisir entre (1), (2), ou (3).

3.2.5 Imprimante et clavier NVT : RFC-854, p. 11

En mode NVT, Telnet NE DEVRAIT PAS envoyer de caractères avec le bit de plus fort poids à 1, et NE DOIT PAS l'envoyer comme bit de parité. Les mises en œuvre qui passent le bit de plus fort poids aux applications DEVRAIENT négocier le mode binaire (voir le paragraphe 3.2.6).

Discussion :
Les développeurs devraient être conscients qu'une stricte lecture de la RFC-854 permet à un client ou un serveur qui attend du NVT ASCII d'ignorer les caractères avec le bit de plus fort poids mis à 1. En général, le mode binaire est attendu pour la transmission d'un jeu de caractères étendu (au-delà de 7 bits) avec Telnet.

Cependant, il existe des applications qui ont réellement besoin d'un mode NVT à huit bits, qui n'est actuellement pas défini, et ces applications existantes mettent effectivement à 1 le bit de plus fort poids durant une partie ou la totalité de la durée de vie d'une connexion Telnet. Noter que le mode binaire n'est pas le même que le mode NVT à 8 bits, car le mode binaire désactive le traitement des fins de ligne. Pour cette raison, les exigences pesant sur le bit de plus fort poids sont formulées comme DEVRAIT, et non DOIT.

La RFC-854 définit un ensemble minimal de propriétés d'un "terminal virtuel de réseau" ou NVT ; ceci n'est pas destiné à empêcher d'inclure des caractéristiques supplémentaires dans un terminal réel. Une connexion Telnet est entièrement transparente à tous les caractères ASCII à 7 bits, y compris les caractères de contrôle ASCII arbitraires.

Par exemple, un terminal pourrait prendre en charge des commandes de plein écran codées comme séquences d'échappement ASCII ; une mise en œuvre Telnet passera ces séquences comme données non interprétées. Et donc, un NVT ne devrait pas être conçu comme un type de terminal très restrictif.

3.2.6 Structure de commande Telnet : RFC-854, p. 13

Comme les options peuvent apparaître en tout point du flux des données, un caractère d'échappement Telnet (appelé IAC, avec la valeur de 255) à envoyer comme donnée DOIT être doublé.

3.2.7 Option Binaire Telnet : RFC-856

Lorsque l'option Binaire a été négociée avec succès, des caractères de 8 bits arbitraires sont permis. Cependant, le flux de données DOIT toujours être examiné à la recherche de caractères IAC, toute commande Telnet incorporée DOIT être respectée, et les octets de données égaux à IAC DOIVENT être doublés. Un autre traitement de caractères (par exemple, remplacer les CR par des CR NUL ou par CR LF) NE DOIT PAS être fait. En particulier, il n'y a pas de convention de fin de ligne (voir le paragraphe 3.3.1) en mode binaire.

Discussion :
L'option Binaire est normalement négociée dans les deux directions, pour changer la connexion Telnet de mode NVT en mode "binaire".

La séquence IAC EOR peut être utilisée pour délimiter les blocs de données au sein d'un flux Telnet en mode binaire.

3.2.8 Option Type de terminal Telnet : RFC-1091

L'option Type de terminal DOIT utiliser les noms de type de terminal définis officiellement par la RFC Numéros alloués [INTRO:5], lorsqu'ils sont disponibles pour le terminal particulier. Cependant, le receveur d'une option Type de terminal DOIT accepter tout nom.

Discussion :
La RFC-1091 [TELNET:10] met à jour une version antérieure de l'option Type de terminal définie dans la RFC-930. La version antérieure permettait à un serveur hôte capable de prendre en charge plusieurs types de terminal d'apprendre le type du terminal d'un client particulier, en supposant que chaque terminal physique a un type intrinsèque. Cependant, aujourd'hui, un "terminal" est souvent en réalité un programme émulateur de terminal qui fonctionne dans un micro ordinateur, peut-être capable d'émuler toute une gamme de types de terminal. Donc, la RFC-1091 étend la spécification pour permettre une négociation de type de terminal plus générale entre usager Telnet et serveur Telnet.

3.3 Questions spécifiques

3.3.1 Convention Telnet de fin de ligne

Le protocole Telnet définit la séquence CR LF comme signifiant "fin-de-ligne". Pour l'entrée au terminal, cela correspond à une commande d'achèvement ou à la frappe d'une touche "fin-de-ligne" sur un terminal d'utilisateur ; sur un terminal ASCII, c'est une touche CR, mais elle peut aussi être marquée "Retour" ou "Entrée".

Lorsqu'un serveur Telnet reçoit la séquence Telnet de fin-de-ligne CR LF comme entrée de la part d'un terminal distant, l'effet DOIT être le même que si l'usager avait frappé la touche "fin-de-ligne" sur un terminal local. Sur les serveurs hôtes qui utilisent ASCII, en particulier, la réception de la séquence Telnet CR LF doit avoir le même effet qu'un usager local frappant la touche CR sur un terminal local. Et donc, CR LF et CR NUL DOIVENT avoir le même effet sur un hôte serveur ASCII lorsque c'est reçu comme entrée sur une connexion Telnet.

Un usager Telnet DOIT être capable d'envoyer toutes les formes : CR LF, CR NUL, et LF. Un usager Telnet sur un hôte ASCII DEVRAIT avoir un mode sous le contrôle de l'usager pour envoyer CR LF ou CR NUL lorsque l'usager frappe la touche "fin-de-ligne", et CR LF DEVRAIT être la valeur par défaut.

La séquence Telnet de fin-de-ligne CR LF DOIT être utilisée pour envoyer des données Telnet qui ne sont pas de terminal à ordinateur (par exemple, pour le serveur Telnet qui envoie des résultats, ou le protocole Telnet qui incorpore un autre protocole d'application).

Discussion :
Pour permettre l'interopérabilité entre des clients et serveurs Telnet arbitraires, le protocole Telnet a défini une représentation standard de terminaison de ligne. Comme le jeu de caractères ASCII n'inclut pas de caractère de fin-de-ligne explicite, les systèmes ont choisi des représentations variées, par exemple, CR, LF, et la séquence CR LF. Le protocole Telnet choisit la séquence CR LF comme standard pour la transmission réseau.

Malheureusement, la spécification du protocole Telnet dans la RFC-854 [TELNET:1] s'est trouvée être quelque peu ambiguë sur le ou les caractères qui devraient être envoyés du client au serveur pour la touche "fin-de-ligne". Le résultat a été des difficultés d'inter fonctionnement massives et continuelles, rendues pires par diverses mises en œuvre fautives à la fois d'usager et de serveur Telnet.

Bien que le protocole Telnet se fonde sur un modèle parfaitement symétrique, dans une session de connexion distante le rôle de l'usager d'un terminal diffère du rôle du serveur hôte. Par exemple, la RFC-854 définit la signification de CR, LF, et CR LF comme sortie du serveur, mais ne spécifie pas ce que l'usager Telnet devrait envoyer lorsque l'utilisateur frappe la touche "fin-de-ligne" du clavier du terminal ; et c'est cela qui est toute la question.

Lorsque l'usager frappe la touche "fin-de-ligne", certaines mises en œuvre d'usager Telnet envoient CR LF, alors que d'autres envoient CR NUL (fondé sur une interprétation différente de la même phrase de la RFC-854). Cela serait équivalent pour un serveur hôte ASCII correctement mis en œuvre, comme exposé ci-dessus. Pour d'autres serveurs, il est nécessaire d'avoir un mode dans l'usager Telnet.

L'existence des usagers Telnet qui n'envoient que CR NUL lorsque la touche CR est frappée crée un dilemme pour les hôtes non ASCII : ils peuvent soit traiter le CR NUL comme équivalent à CR LF en entrée, empêchant par là même la possibilité d'entrer un CR "nu", soit perdre tout inter fonctionnement.

Supposons qu'un usager sur l'hôte A utilise Telnet pour se connecter à un serveur hôte B, et exécute ensuite le programme d'usager Telnet de B pour se connecter au serveur hôte C. Il est souhaitable que la combinaison serveur/usager Telnet sur B soit aussi transparente que possible, c'est-à-dire, d'apparaître comme si A était connecté directement à C. En particulier, une mise en œuvre correcte rendra B transparent pour les séquences Telnet de fin-de-ligne, excepté que CR LF peut être traduit en CR NUL ou vice versa.

Mise en œuvre :
Pour comprendre la question de la fin-de-ligne Telnet, on doit au moins avoir un modèle général de la relation de Telnet avec le système d'exploitation local. Le processus de serveur Telnet est normalement couplé au logiciel de pilote de terminal du système d'exploitation comme un pseudo-terminal. Une séquence Telnet de fin-de-ligne reçue par le serveur Telnet doit avoir le même effet que de frapper la touche fin-de-ligne sur un terminal connecté localement réel.

Les systèmes d'exploitation qui prennent en charge les applications interactives à un caractère à la fois (par exemple, les éditeurs) ont normalement deux modes internes pour leur entrée/sortie de terminal : un mode formaté, dans lequel les

conventions locales de fin-de-ligne et autres règles de formatage ont été appliquées au flux de données, et un mode "brut", dans lequel l'application a un accès direct à chaque caractère au moment où il est entré. Un serveur Telnet doit être mis en œuvre de telle façon que ces modes aient le même effet pour le terminal local ou distant. Par exemple, supposons qu'un CR LF ou CR NUL soit reçu par le serveur Telnet sur un hôte ASCII. En mode brut, un caractère CR est passé à l'application ; en mode formaté, on utilise la convention de fin-de-ligne du système local.

3.3.2 Terminaux d'entrée de données

Discussion :
En plus des terminaux ASCII orientés ligne et orientés caractère pour lesquels Telnet a été conçu, il y a plusieurs familles de terminaux d'affichage vidéo qui sont parfois appelés "terminaux d'entrée de données" ou DET *(data entry terminal)*. La famille IBM 3270 en est un exemple bien connu.

Deux protocoles Internet ont été conçus pour prendre en charge les DET génériques : SUPDUP [TELNET:16, TELNET:17], et l'option DET [TELNET:18, TELNET:19]. L'option DET pilote un terminal d'entrée de données sur une connexion Telnet en utilisant la (sous-) négociation. SUPDUP est un protocole de terminal complètement distinct, qui peut être accédé à partir de Telnet par négociation.

Bien que SUPDUP et l'option DET aient tous deux été utilisés avec succès dans des environnements particuliers, aucun d'eux n'a obtenu un consensus général ou une large mise en œuvre.

Une approche différente de l'interaction de DET a été développée pour la prise en charge de la famille IBM 3270 à travers Telnet, bien que la même approche soit applicable à tout DET. L'idée est d'entrer en mode "DET natif", dans lequel le flux d'entrée/sortie natif du DET est envoyé comme des données binaires. La commande EOR de Telnet est utilisée pour délimiter des enregistrements logiques (par exemple, "des écrans") au sein de ce flux binaire.

Mise en œuvre :
Les règles pour entrer et quitter le mode DET natif sont les suivantes :
- o Le serveur utilise l'option Type-de-terminal [TELNET:10] pour apprendre que le client est un DET.
- o Il est convenu, mais pas exigé, que les deux extrémités négocient l'option EOR [TELNET:9].
- o Les deux extrémités négocient l'option binaire [TELNET:3] pour entrer en mode DET natif.
- o Quand l'une des extrémités négocie le mode binaire, l'autre extrémité le fait aussi, et le mode revient alors au NVT normal.

3.3.3 Exigences pour les options

Toute mise en œuvre Telnet DOIT prendre en charge l'option Binaire [TELNET:3] et l'option Suppress Go Ahead [TELNET:5], et DEVRAIT accepter les options Echo [TELNET:4], Status [TELNET:6], End-of-Record [TELNET:9], et Extended Options List [TELNET:8].

Un usager ou serveur Telnet DEVRAIT accepter l'option Window Size [TELNET:12] si le système d'exploitation local fournit la capacité correspondante.

Discussion :
Noter que l'option End-of-Record signifie seulement que Telnet peut recevoir une EOR Telnet sans défaillance ; donc, toute mise en œuvre Telnet devrait accepter la négociation de l'option End-of-Record. Voir aussi l'exposé du paragraphe 3.2.3.

3.3.4 Initialisation d'option

Lorsque le protocole Telnet est utilisé dans une situation client/serveur, le serveur DEVRAIT initier la négociation du mode d'interaction de terminal qu'il attend.

Discussion :
Le protocole Telnet a été défini comme parfaitement symétrique, mais son application est généralement asymétrique. La connexion à distance est connue pour échouer parce que AUCUNE des extrémités n'initialise la négociation des modes de terminal exigés en dehors de ceux par défaut . C'est généralement le serveur qui détermine le mode préféré, aussi le serveur a-t-il besoin d'initier la négociation ; comme la négociation est symétrique, l'usager peut aussi l'initier.

Un client (Usager Telnet) DEVRAIT fournir un moyen pour que les utilisateurs activent et désactivent l'initiation de la négociation d'options.

Discussion :
Un usager a parfois besoin de se connecter à un service d'application (par exemple, FTP ou SMTP) qui utilise Telnet pour son flux de commande mais ne prend pas en charge les options Telnet. L'usager Telnet peut être utilisé à cette fin si l'initiation de la négociation d'option est désactivée.

3.3.5 Option Telnet Linemode

Discussion :
Une importante nouvelle option Telnet, LINEMODE [TELNET:12], a été proposée. L'option LINEMODE fournit un moyen normalisé pour que l'usager Telnet et le serveur Telnet se mettent d'accord pour que le client plutôt que le serveur effectue le traitement de caractères du terminal. Lorsque le client a préparé une ligne de texte complète, il va l'envoyer au serveur en un (généralement) seul paquet TCP. Cette option va largement diminuer le coût du paquet des sessions Telnet et va aussi donner une bien meilleure réponse de l'usager sur les réseaux encombrés ou à long délai.

L'option LINEMODE permet la commutation dynamique entre le traitement local et distant de caractères. Par exemple, la connexion Telnet va négocier automatiquement le mode à un seul caractère lors du fonctionnement d'un éditeur de plein écran, puis va retourner au mode par ligne lorsque l'éditeur a terminé.

On s'attend à ce que lors de la publication de cette RFC, les hôtes mettent en œuvre le côté client de cette option, et éventuellement qu'il puissent mettre en œuvre le côté serveur de cette option. Pour mettre en œuvre correctement le côté serveur, celui-ci doit être capable de dire au système local de ne faire aucun traitement de caractère en entrée, mais de se souvenir de son état actuel de terminal et de notifier au processus de serveur Telnet chaque fois que l'état change. Cela permettra que, par exemple, l'écho de mot de passe et les éditeurs de plein écran soient traités correctement.

3.4 Interface Telnet/usager

3.4.1 Transparence du jeu de caractères

Une mise en œuvre d'usager Telnet DEVRAIT être capable d'envoyer ou recevoir tout caractère ASCII de 7 bits. Lorsque c'est possible, toute interprétation de caractères spéciaux par le système d'exploitation de l'hôte utilisateur DEVRAIT être outrepassée de sorte que ces caractères puissent être commodément envoyés et reçus sur la connexion.

Certaines valeurs de caractère DOIVENT être réservées comme "échappées en mode commande" ; par convention, le doublement de ce caractère lui permet d'être entré comme donnée. Le caractère spécifique utilisé DEVRAIT être choisi par l'usager.

Sur les connexions en mode binaire, un programme d'usager Telnet PEUT fournir un mécanisme d'échappement pour entrer des valeurs arbitraires de 8 bits, si le système d'exploitation de l'hôte ne leur permet pas d'être frappées directement au clavier.

Mise en œuvre :
Les questions de transparence sont moins pressantes sur les serveurs, mais les développeurs devraient faire attention lors du traitement de questions comme le masquage des bits de parité (envoyé par un vieux client non conforme) avant qu'ils n'atteignent des programmes qui attendent seulement de l'ASCII NVT, et le traitement correct des programmes qui exigent des flux de données à 8 bits.

3.4.2 Commandes Telnet

Un programme d'usager Telnet DOIT fournir à l'usager la capacité d'entrer toute fonction IP, AO ou AYT de contrôle Telnet, et DEVRAIT fournir la capacité d'entrer EC, EL, et Break.

3.4.3 Erreurs de connexion TCP

Un programme d'usager Telnet DEVRAIT rapporter à l'usager toute erreur TCP qui est rapportée par la couche transport (voir la section "Interface de couche TCP/application" dans [INTRO:1]).

3.4.4 Accès de contact Telnet autre que par défaut

Un programme d'usager Telnet DEVRAIT permettre à l'usager de spécifier facultativement un numéro d'accès de contact non standard chez l'hôte serveur Telnet.

3.4.5 Purge des sorties

Un programme d'usager Telnet DEVRAIT fournir à l'usager la capacité de spécifier si une sortie devrait être purgée ou non lorsqu'une IP est envoyée ; voir au paragraphe 3.2.4.

Pour tout schéma de purge de sorties qui causerait la purge locale des sorties par l'usager Telnet jusqu'à ce qu'un signal Telnet soit reçu du serveur, il DEVRAIT y avoir un moyen pour que l'usager restaure manuellement la sortie normale, au cas où le serveur échouerait à envoyer le signal attendu.

3.5 Résumé des exigences TELNET

Caractéristique	**parag.**	**DOIT**	**DVRT**	**PEUT**	**NE DVRT PAS**	**NE DT PAS**
Négociation d'option	3.2.1	x				
Évite les boucles de négociation	3.2.1	x				
Refuse les options non prises en charge	3.2.1	x				
Négociation possible à tout moment en connexion	3.2.1		x			
NVT par défaut	3.2.1	x				
Envoi du nom officiel dans l'option Term-Type	3.2.8	x				
Accepte tout nom dans l'option Term-Type	3.2.8	x				
Accepte les options Binary, Suppress-GA	3.3.3	x				
Options Echo, Status, EOL, Ext-Opt-List	3.3.3		x			
Accepte l'option Window-Size si approprié	3.3.3		x			
Le serveur initie les négociations de mode	3.3.4		x			
L'usager peut activer/désactiver négociations d'init.	3.3.4		x			
Go-Ahead						
Un serveur non-GA négocie l'option SUPPRESS-GA	3.2.2	x				
L'usager ou serveur accepte l'option SUPPRESS-GA	3.2.2	x				
Usager Telnet ignore les GA	3.2.2			x		
Fonctions de contrôle						
Accepte SE NOP DM IP AO AYT SB	3.2.3	x				
Accepte EOR EC EL Break	3.2.3			x		
Ignore les fonctions de contrôle non prises en charge	3.2.3	x				
Usager, serveur élimine données urgentes jusqu'à DM	3.2.4	x				
Usager Telnet envoie "Synch" après IP, AO, AYT	3.2.4		x			
Serveur Telnet répond Synch à IP	3.2.4			x		
Serveur Telnet répond Synch à AO	3.2.4	x				
Usager Telnet peut purger les sorties après l'envoi IP	3.2.4		x			
Codage						
Envoi du bit de plus fort poids en mode NVT	3.2.5				x	
Envoi du bit de plus fort poids comme bit de parité	3.2.5					x
Négocie BINARY si passe bit de pfp à applic	3.2.5		x			
Double toujours l'octet de données IAC	3.2.6	x				
Double l'octet de données IAC en mode binaire	3.2.7	x				
Respecte commandes Telnet en mode binaire	3.2.7	x				
Fin-de-ligne, CR NUL en mode binaire	3.2.7					x
Fin-de-ligne (EOL)						
EOL au serveur comme fin-de-ligne local	3.3.1	x				
Serveur ASCII accepte CR LF ou CR NUL pour EOL	3.3.1	x				
Usager Telnet capable envoi CR LF, CR NUL, ou LF	3.3.1	x				
Usager ASCII capable de choisir CR LF/CR NUL	3.3.1		x			
Mode par défaut d'usager Telnet est CR LF	3.3.1		x			
Non interactif utilise CR LF pour EOL	3.3.1	x				
Interface d'usager Telnet						
Entrée & sortie tout en caractères de 7 bits	3.4.1		x			
Outrepasse interprétation du système d'exploitation	3.4.1		x			
Caractère d'échappement	3.4.1	x				
Caractère d'échappement réglable par l'usager	3.4.1		x			

Échappement pour entrer des valeurs de 8 bits	3.4.1			x
Peut entrer IP, AO, AYT	3.4.2	x		
Peut entrer EC, EL, Break	3.4.2		x	
Rapporte les erreurs de connexion TCP à l'usager	3.4.3		x	
Accès de contact autre que défaut facultatif	3.4.4		x	
Peut spécifier purge des sorties quand IP envoyé	3.4.5		x	
Peut restaurer manuellement le mode de sortie	3.4.5		x	

4 Transfert de fichier

4.1 Protocole de transfert de fichier -- FTP

4.1.1 Introduction

Le protocole de transfert de fichiers (FTP, *File Transfer Protocol*) est la principale norme de l'Internet pour les transferts de fichiers. La spécification actuelle est contenue dans la RFC-959 [FTP:1].

FTP utilise des connexions TCP séparées simultanées pour le contrôle des données et le transfert. Le protocole FTP comporte de nombreuses caractéristiques dont certaines ne sont pas couramment mises en œuvre. Cependant, pour chaque caractéristique de FTP, il existe au moins une mise en œuvre. La mise en œuvre minimum définie dans la RFC-959 était trop petite, aussi une mise en œuvre minimum un peu plus conséquente est définie ici.

Les usagers de l'Internet ont été encombrés sans nécessité pendant des années par des mises en œuvre TCP déficientes. Les développeurs de protocoles ont souffert de l'opinion erronée que la mise en œuvre de TCP devait être une petite tâche triviale. C'est faux, parce que FTP a une interface d'utilisateur, parce qu'il doit traiter (correctement) toute une variété d'erreurs de communication et de systèmes d'exploitation potentielles, et parce qu'il doit traiter une grande diversité de systèmes de fichiers réels dans le monde.

4.1.2 Survol du protocole

4.1.2.1 Type LOCAL : RFC-959 paragraphe 3.1.1.4

Un programme FTP DOIT prendre en charge le TYPE I (type "IMAGE" ou binaire) aussi bien que le TYPE L 8 (type "LOCAL" avec une taille d'octet logique de 8). Une machine dont la mémoire est organisée en mots de m-bits, où m n'est pas multiple de 8, PEUT aussi prendre en charge le TYPE L m.

Discussion :
La commande "TYPE L 8" est souvent exigée pour transférer des données binaires entre une machine dont la mémoire est organisée en (par exemple) mots de 36 bits et une machine avec une organisation en octets de 8 bits. Pour une machine à octets de 8 bits, le TYPE L 8 est équivalent à IMAGE.

Le "TYPE L m" est parfois spécifié pour les programmes FTP sur des machines à mots de m-bits pour assurer le transfert correct d'un fichier binaire en mode natif d'une machine à l'autre. Cependant, cette commande devrait avoir le même effet sur ces machines que le "TYPE I".

4.1.2.2 Contrôle de format Telnet : RFC-959 paragraphe 3.1.1.5.2

Un hôte qui ne fait pas de distinction entre TYPE N et TYPE T DEVRAIT mettre en œuvre le TYPE T comme identique au TYPE N.

Discussion : Cette disposition devrait faciliter l'interopération avec les hôtes qui font cette distinction.

De nombreux hôtes représentent en interne les fichiers texte comme des chaînes de caractères ASCII, en utilisant les caractères effecteurs de format ASCII incorporés (LF, BS, FF, ...) pour contrôler le format lors de l'impression d'un fichier. Pour de tels hôtes, il n'y a pas de distinction entre les fichiers "à imprimer" et les autres fichiers. Cependant, les systèmes qui utilisent des fichiers structurés en enregistrement ont normalement besoin d'un format particulier pour les fichiers imprimables (par exemple, la commande de chariot ASA). Pour ces derniers hôtes, FTP permet un choix de TYPE N ou TYPE T.

4.1.2.3 Structure de page : RFC-959 paragraphe 3.1.2.3 et Appendice I

La mise en œuvre de la structure de page est NON RECOMMANDÉE en général. Cependant, si un système d'hôte a besoin de mettre en œuvre FTP pour des fichiers "en accès aléatoire" ou "troués", il DOIT utiliser le format de structure de page défini plutôt que de définir un nouveau format FTP privé.

4.1.2.4 Transformations de structure de données : RFC-959 paragraphe 3.1.2

Une transformation FTP entre structure d'enregistrement et structure de fichier DEVRAIT être réversible, dans la mesure du possible tout en rendant le résultat utile sur l'hôte cible.

Discussion :
La RFC-959 exigeait une réversibilité stricte entre la structure d'enregistrement et la structure de fichier, mais en pratique, l'efficacité et la commodité l'empêchent souvent. Donc, l'exigence a été atténuée. Il y a deux objectifs différents au transfert d'un fichier : le traiter sur un hôte cible, ou seulement le mémoriser. Pour la mémorisation, la stricte réversibilité est importante. Pour le traitement, le fichier créé sur l'hôte cible a besoin d'être dans le format attendu par les programmes d'application de cet hôte.

Comme exemple du conflit, imaginons un système d'exploitation orienté enregistrement qui exige que des fichiers de données aient exactement 80 octets dans chaque enregistrement. Pour mémoriser un fichier sur un tel hôte, un serveur FTP doit être capable de bourrer chaque ligne ou enregistrement à 80 octets ; une restitution ultérieure d'un tel fichier ne peut pas être strictement réversible.

4.1.2.5 Gestion de la connexion de données : RFC-959 paragraphe 3.3

Un usager FTP qui utilise le mode STREAM DEVRAIT envoyer une commande PORT pour allouer un accès de données autre que par défaut avant la production de chaque commande de transfert.

Discussion : Ceci est exigé parce que le long délai après une connexion TCP est clos jusqu'à ce que sa paire de prises puisse être réutilisée, pour permettre plusieurs transferts durant une seule session FTP. L'envoi d'une commande d'accès peut être évité si un mode de transfert autre que le flux est utilisé, en laissant la connexion de transfert de données ouverte entre les transferts.

4.1.2.6 Commande PASV : RFC-959 paragraphe 4.1.2

Un serveur FTP DOIT mettre en œuvre la commande PASV.

Si plusieurs transferts de tierce partie sont à exécuter durant la même session, une nouvelle commande PASV DOIT être produite avant chaque commande de transfert, pour obtenir une paire d'accès unique.

Mise en œuvre :
Le format de la réponse 227 à une commande PASV n'est pas bien normalisé. En particulier, un client FTP ne peut pas supposer que les parenthèses indiquées à la page 40 de la RFC-959 seront présentes (et en fait, la Figure 3 de la page 43 les omet). Donc, un programme d'usager FTP qui interprète la réponse PASV doit examiner le premier chiffre des numéros d'hôte et d'accès de la réponse.

Noter que le numéro d'hôte h1,h2,h3,h4 est l'adresse IP du serveur hôte qui envoie la réponse, et que p1,p2 est un accès de transfert de données que PASV a alloué qui n'est pas par défaut.

4.1.2.7 Commandes LIST et NLST : RFC-959 paragraphe 4.1.3

Les données retournées par une commande NLST DOIVENT ne contenir qu'une simple liste des noms de chemin légaux, de telle sorte que le serveur puisse les utiliser directement comme les arguments des commandes de transfert de données ultérieures pour les fichiers individuels.

Les données retournées par une commande LIST ou NLST DEVRAIENT utiliser un TYPE AN implicite, sauf si le type actuel est EBCDIC, auquel cas un TYPE EN implicite DEVRAIT être utilisé.

Discussion : De nombreux clients FTP prennent en charge des macro-commandes qui vont obtenir ou envoyer des fichiers satisfaisant à une spécification de caractère générique, en utilisant NLST pour obtenir une liste des noms de chemin. L'expansion de "l'envoi multiple" est locale pour le client, mais "l'obtention multiple" exige la coopération du serveur.

Le type implicite pour LIST et NLST est destiné à fournir la compatibilité avec les usagers FTP existants, et en particulier avec les commandes d'obtention multiple.

4.1.2.8 Commande SITE : RFC-959 paragraphe 4.1.3

Un serveur FTP DEVRAIT utiliser la commande SITE pour des caractéristiques non standard, plutôt que d'inventer de nouvelles commandes privées ou extensions non normalisées aux commandes existantes.

4.1.2.9 Commande STOU : RFC-959 paragraphe 4.1.3

La commande STOU mémorise dans un fichier dénommé de façon univoque. Lorsqu'il reçoit une commande STOU, un serveur FTP DOIT retourner le nom de fichier réel dans le message "125 Début de transfert" ou "150 Ouverture de connexion de données" qui précède le transfert (le code de réponse 250 mentionné dans la RFC-959 est incorrect). Le format exact de ces messages est défini ici comme suit :

125 FILE: pppp
150 FILE: pppp

où pppp représente le nom univoque de chemin du fichier qui sera écrit.

4.1.2.10 Code Telnet de Fin-de-ligne : RFC-959, Page 34

Les développeurs NE DOIVENT PAS supposer de correspondance entre les frontières de READ sur la connexion de contrôle et les séquences EOL de Telnet (CR LF).

Discussion : Donc, un serveur FTP (ou usager FTP) doit continuer de lire les caractères provenant de la connexion de contrôle jusqu'à ce qu'il rencontre une séquence EOL Telnet complète, avant de traiter la commande (ou respectivement, la réponse). À l'inverse, un seul READ provenant de la connexion de contrôle peut inclure plus d'une commande FTP.

4.1.2.11 Réponses FTP : RFC-959 paragraphe 4.2, page 35

Un serveur FTP DOIT n'envoyer que des réponses correctement formatées sur la connexion de contrôle. Noter que la RFC-959 (à la différence de versions antérieures de la spécification FTP) ne contient aucune disposition pour un message de réponse "spontané".

Un serveur FTP DEVRAIT utiliser les codes de réponse définis dans la RFC-959 chaque fois qu'ils s'appliquent. Cependant, un serveur FTP PEUT utiliser un code de réponse différent lorsque c'est nécessaire, tant que les règles générales du paragraphe 4.2 sont respectées. Lorsque le développeur a le choix entre un code de réponse 4xx et 5xx, un serveur FTP DEVRAIT envoyer un 4xx (échec temporaire) lorsqu'il y a une possibilité raisonnable qu'un FTP défaillant réussisse quelques heures plus tard.

Un usager FTP DEVRAIT généralement n'utiliser que le chiffre d'ordre le plus élevé pour prendre une décision de procédure, pour empêcher des difficultés lorsqu'un serveur FTP utilise des codes de réponse non standard.

Un usager FTP DOIT être capable de traiter des réponses multilignes. Si la mise en œuvre impose une limite au nombre de lignes et si cette limite est dépassée, l'usager FTP DOIT récupérer, par exemple, en ignorant les lignes en excès jusqu'à ce que la fin de la réponse multi-lignes soit atteinte.

Un usager FTP NE DEVRAIT PAS interpréter un code de réponse 421 ("Service non disponible, clôture de la connexion de contrôle") de façon particulière, mais DEVRAIT détecter la clôture de la connexion de contrôle par le serveur.

Discussion : Une mise en œuvre de serveur qui échoue à suivre strictement les règles de réponse cause souvent le raccrochement des programmes d'usager FTP. Noter que la RFC-959 résout des ambiguïtés dans les règles de réponse figurant dans les spécifications FTP antérieures et qu'elle doit être suivie.

Il est important de choisir des codes de réponse FTP qui font une distinction appropriée entre défaillance temporaire et permanente, pour permettre l'utilisation réussie des automates client de transfert de fichier. Ces programmes dépendent des codes de réponse pour décider de réessayer ou non un transfert raté ; utiliser un code d'échec permanent (5xx) pour une erreur temporaire causera un abandon non nécessaire par ces programmes.

Lorsque la signification d'une réponse correspond exactement au texte donné dans la RFC-959, l'uniformité sera améliorée en utilisant le texte mot à mot de la RFC-959. Cependant, une mise en œuvre de serveur FTP est invitée à choisir le texte de réponse qui convoie les informations spécifiques dépendantes du système, lorsque c'est approprié.

4.1.2.12 Connexions : RFC-959 paragraphe 5.2

Les mots "et l'accès utilisé" dans le second alinéa de ce paragraphe de la RFC-959 sont erronés (historique), et devraient être ignorés.

Sur un serveur hôte à rattachements multiples, l'accès de transfert de données par défaut (L-1) DOIT être associé à la même adresse IP locale que la connexion de contrôle correspondante à l'accès L.

Un usager FTP NE DOIT PAS envoyer de commandes Telnet autres que SYNCH et IP sur une connexion de contrôle FTP. En particulier, il NE DOIT PAS tenter de négocier les options Telnet sur la connexion de contrôle. Cependant, un serveur FTP DOIT être capable d'accepter et refuser les négociations Telnet (c'est-à-dire, d'envoyer DONT/WONT).

Discussion : Bien que la RFC dise : "Les processus de serveur et d'usager devraient suivre les conventions du protocole Telnet...[sur la connexion de contrôle]", l'intention n'est pas d'employer la négociation d'option Telnet.

4.1.2.13 Mise en œuvre minimum : RFC-959 paragraphe 5.1

Les commandes et options suivantes DOIVENT être prise en charge par tout serveur et usager FTP, excepté dans les cas où le système de fichier ou système d'exploitation sous-jacent ne permet pas ou ne prend pas en charge une commande particulière.

Type : ASCII Non-print, IMAGE, LOCAL 8
Mode : Stream
Structure : File, Record*
Commandes : USER, PASS, ACCT, PORT, PASV, TYPE, MODE, STRU, RETR, STOR, APPE, RNFR, RNTO, DELE, CWD, CDUP, RMD, MKD, PWD, LIST, NLST, SYST, STAT, HELP, NOOP, QUIT.

La structure *Record n'est EXIGÉE que pour les hôtes dont le système de fichiers accepte la structure d'enregistrement.

Discussion : Les fabricants sont invités à mettre en œuvre un plus grand sous-ensemble du protocole. Par exemple, il y a un important dispositif de robustesse dans le protocole (par exemple, Restart, ABOR, mode bloc) qui serait une aide pour certains usagers de l'Internet mais qui n'est pas largement mis en œuvre.

Un hôte qui n'a pas de structures d'enregistrement dans son système de fichiers peut quand même accepter des fichiers avec STRU R, enregistrant littéralement le flux d'octet.

4.1.3 Questions spécifiques

4.1.3.1 Verbes de commande non standard

FTP permet des commandes "expérimentales", dont les noms commencent par "X". Si ces commandes sont ultérieurement adoptées comme normes, il pourrait toujours y avoir des mises en œuvre qui utilisent la forme "X". À présent, ceci est vrai pour les commandes de répertoire :

RFC-959	"Expérimental"
MKD	XMKD
RMD	XRMD
PWD	XPWD
CDUP	[2]XCUP
CWD	XCWD

Toute mise en œuvre FTP DEVRAIT reconnaître les deux formes de ces commandes, simplement en les mettant égales aux entrées supplémentaires dans le tableau de recherche des commandes.

Mise en œuvre :
Un usager FTP peut accéder à un serveur qui ne prend en charge que les formes "X" en mettant en œuvre un commutateur de mode, ou en utilisant automatiquement la procédure suivante : si la forme de la RFC-959 d'une des commandes ci-dessus est rejetée avec un code de réponse 500 ou 502, essayer alors la forme expérimentale ; toute autre réponse devrait être passée à l'usager.

4.1.3.2 Temporisation d'inactivité

Un processus de serveur FTP DEVRAIT avoir une temporisation d'inactivité, qui va terminer le processus et clore la connexion de contrôle si le serveur est inactif (c'est-à-dire, aucune commande ou transfert de données en cours) pendant une longue période. La durée de la temporisation d'inactivité DEVRAIT être configurable, et la valeur par défaut devrait être d'au moins 5 minutes.

Un processus de client FTP ("Usager-PI" dans la RFC-959) n'aura besoin de temporisation sur les réponses que si c'est invoqué à partir d'un programme.

Discussion : Sans temporisation, un processus de serveur FTP peut être laissé en instance indéfiniment si le client correspondant est arrêté sans avoir clos la connexion de contrôle.

4.1.3.3 Concurrence entre données et contrôle

Discussion : L'intention des concepteurs de FTP était qu'un usager devrait être capable d'envoyer une commande STAT à tout moment pendant le cours d'un transfert de données et que le serveur FTP répondrait immédiatement par l'état -- par exemple, le nombre d'octets transférés jusqu'à présent. De même, une commande ABOR devrait être possible à tout moment pendant un transfert de données.

Malheureusement, certains systèmes d'exploitation de petites machines rendent cette programmation concurrente difficile, et certains autres développeurs cherchent des solutions minimales, aussi certaines mises en œuvre FTP ne permettent pas l'utilisation en concurrence entre les données et les connexions de contrôle. Même de tels serveurs minimaux doivent être prêts à accepter et obéir à la commande STAT ou ABOR arrivant durant un transfert de données.

4.1.3.4 Mécanisme de redémarrage FTP

La description de la réponse 110 aux pages 40-41 de la RFC-959 est incorrecte ; la description correcte est la suivante. Un message de réponse Redémarrage, envoyé sur la connexion de contrôle de la part du FTP qui reçoit à l'usager FTP, a le format :

110 MARK ssss = rrrr

Ici ssss est une chaîne de texte qui apparaît dans un marqueur de redémarrage dans le flux de données et code une position dans le système de fichiers de l'envoyeur ; rrrr code la position correspondante dans le système de fichiers du receveur.

Le codage, qui est spécifique d'un système de fichier et d'une mise en œuvre réseau particuliers, est toujours généré et interprété par le même système, envoyeur ou receveur.

Lorsqu'un FTP qui met en œuvre un redémarrage reçoit un marqueur de Redémarrage dans le flux de données, il DEVRAIT forcer à partir de ce point l'écriture des données dans une mémorisation stable avant de coder la position rrrr correspondante. Un FTP qui envoie des marqueurs de redémarrage NE DOIT PAS supposer que des réponses 110 seront retournées en synchronisme avec les données, c'est-à-dire qu'il ne doit pas attendre une réponse 110 avant d'avoir envoyé plus de données.

Deux nouveaux codes de réponse sont définis pour les erreurs rencontrées dans le redémarrage d'un transfert :

554 Action demandée non effectuée : paramètre REST invalide.
Une réponse 554 peut résulter d'une commande de service FTP qui suit une commande REST. La réponse indique que le fichier existant chez le serveur FTP ne peut pas être repositionnée comme spécifié dans REST.

555 Action demandée non effectuée : discordance de type ou stru.
Une réponse 555 peut résulter d'une commande APPE ou de toute commande de service FTP suivie par une commande REST. La réponse indique qu'il y a un désaccord entre les paramètres de transfert courants (type et stru) et les attributs du fichier existant.

Discussion : Noter que le mécanisme FTP Restart exige que le mode Bloc ou Compressé soit utilisé pour le transfert de données, pour permettre que les marqueurs de redémarrage soient inclus dans le flux des données. La fréquence des marqueurs de redémarrage peut être faible.

Un marqueur de redémarrage marque un endroit dans le flux des données, mais le receveur peut effectuer certaines transformations sur les données lorsqu'il les sauvegarde dans une mémorisation stable. En général, le codage du receveur doit inclure toute information d'état nécessaire pour redémarrer cette transformation en tout point du flux des données FTP. Par exemple, dans les transferts de TYPE A, certains hôtes receveurs transforment les séquences CR LF en un seul caractère LF sur disque. Si un marqueur de redémarrage se trouve tomber entre le CR et le LF, le receveur doit coder en rrrr que le transfert doit être redémarré dans un état "CR a été vu et supprimé".

Noter que le marqueur de redémarrage doit obligatoirement être codé comme une chaîne de caractères ASCII imprimables, quel que soit le type des données.

La RFC-959 dit que les informations de redémarrage sont à retourner "à l'usager". Ceci ne devrait pas être pris au pied de la lettre. En général, l'usager FTP devrait sauvegarder les informations de redémarrage (ssss,rrrr) dans une mémorisation stable, par exemple, ajoutées à un fichier de contrôle de redémarrage. Un fichier de contrôle de redémarrage vide devrait être créé lorsque débute le transfert et être supprimé automatiquement lorsque le transfert s'achève avec succès. Il est suggéré que ce fichier ait un nom dérivé d'une façon facilement identifiable du nom du fichier en cours de transfert et du nom d'hôte distant ; ceci est analogue au moyen utilisé par de nombreux éditeurs de texte pour nommer leurs fichiers de "sauvegarde".

Il y a trois cas pour Redémarrage FTP :

(1) Transfert d'usager à serveur
L'usager FTP met les marqueurs de Redémarrage <ssss> aux endroits appropriés dans le flux des données. Lorsque le serveur FTP reçoit un marqueur, il inscrit toutes les données antérieures sur le disque, code sa position de système fichier et son état de transformation comme rrrr, et retourne une réponse "110 MARK ssss = rrrr" sur la connexion de contrôle. L'usager FTP ajoute la paire (ssss,rrrr) à son fichier de commande de redémarrage.
Pour redémarrer le transfert, l'usager FTP va chercher la dernière paire (ssss,rrrr) dans le fichier de commande de redémarrage, repositionne son système de fichier local et son état de transformation en utilisant ssss, et envoie la commande "REST rrrr" au serveur FTP.

(2) Transfert de serveur à usager
Le serveur FTP met les marqueurs de redémarrage <ssss> aux endroits appropriés dans le flux des données. Lorsque l'usager FTP reçoit un marqueur, il inscrit toutes les données antérieures sur le disque, code sa position de système de fichier et son état de transformation comme rrrr, et ajoute la paire (rrrr,ssss) à son fichier de commande de redémarrage.
Pour redémarrer le transfert, l'usager FTP va chercher la dernière paire (rrrr,ssss) dans le fichier de commande de redémarrage, repositionne son système de fichier local et son état de transformation en utilisant rrrr, et envoie la commande "REST ssss" au serveur FTP.

(3) Transfert de serveur à serveur ("Tierce partie")
Le serveur FTP envoyeur met les marqueurs de redémarrage <ssss> aux endroits appropriés dans le flux des données. Quand il reçoit un marqueur, le serveur FTP receveur inscrit toutes les données antérieures sur le disque, code sa position de système fichier et son état de transformation comme rrrr, et envoie une réponse "110 MARK ssss = rrrr" sur la connexion de contrôle à l'usager. L'usager FTP ajoute la paire (ssss,rrrr) à son fichier de commande de redémarrage.
Pour redémarrer le transfert, l'usager FTP va chercher la dernière paire (ssss,rrrr) dans le fichier de commande de redémarrage, envoie "REST ssss" au serveur FTP envoyeur, et envoie "REST rrrr" au serveur FTP receveur.

4.1.4 Interface FTP/usager

Ce paragraphe discute de l'interface d'utilisateur pour programme d'usager FTP.

4.1.4.1 Spécification du nom de chemin

Comme FTP est destiné à être utilisé dans un environnement hétérogène, une mise en œuvre d'usager FTP DOIT prendre en charge les noms de chemins distants comme des chaînes de caractères arbitraires, de sorte que leur forme et contenu ne soient pas limités par les conventions du système d'exploitation local.

Discussion : En particulier, les noms de chemins distants peuvent être de longueur arbitraire, et tous les caractères ASCII d'impression ainsi que l'espace (0x20) doivent être admis. La RFC-959 permet à un nom de chemin de contenir tout caractère ASCII de 7 bits excepté CR ou LF.

4.1.4.2 Commande "QUOTE"

Un programme d'usager FTP DOIT mettre en œuvre une commande "QUOTE" qui va passer une chaîne de caractères arbitraire au serveur et afficher à l'usager tous les messages de réponse qui en résultent.

Pour rendre la commande "QUOTE" utile, un usager FTP DEVRAIT envoyer des commandes de contrôle de transfert au serveur lorsque l'usager les entre, plutôt que de sauvegarder toutes les commandes et de ne les envoyer au serveur que lors du début du transfert.

Discussion : La commande "QUOTE" est essentielle pour permettre à l'usager d'accéder aux serveurs qui requièrent des commandes spécifiques du système (par exemple, SITE ou ALLO) ou qui invoquent des dispositifs nouveaux ou facultatifs qui non mis en œuvre par l'usager FTP. Par exemple, "QUOTE" peut être utilisée pour spécifier "TYPE A T" pour envoyer un fichier d'impression à des hôtes qui exigent la distinction, même si l'usager FTP ne reconnaît pas ce TYPE.

4.1.4.3 Affichage des réponses à l'usager

Un usager FTP DEVRAIT afficher à l'usager le texte complet de tous les messages de réponse qu'il reçoit. Il DEVRAIT avoir un mode "verbeux" dans lequel toutes les commandes qu'il envoie et le texte complet et les codes de réponse qu'il reçoit sont affichés, pour le diagnostic des problèmes.

4.1.4.4 Maintien de la synchronisation

Dans un usager FTP, l'automate à états DEVRAIT pardonner l'omission des messages de réponse et l'arrivée de messages de réponse inattendus, de façon à conserver la synchronisation des commandes avec le serveur.

4.1.5 Résumé des exigences pour FTP

Caractéristique	parag.	DOIT	DVRT	PEUT	NE DVRT PAS	NE DT PAS
Met en œuvre TYPE T si même que TYPE N	4.1.2.2		x			
Transform. File/Record réversible si poss.	4.1.2.4		x			
Usager FTP envoie cmd PORT pour mode flux	4.1.2.5		x			
Le serveur FTP met en œuvre PASV	4.1.2.6	x				
PASV est transfert par transfert	4.1.2.6	x				
Réponse NLST utilisable dans les cmds RETR	4.1.2.7	x				
Type implicite pour LIST et NLST	4.1.2.7		x			
Cmd SITE pour caractéristiques non standard	4.1.2.8		x			
Cmd STOU retourne nom de ch. comme spécifié	4.1.2.9	x				
Utilise limites READ TCP sur connexion de cont.	4.1.2.10					x
Serveur FTP n'envoie que format de rép. correct	4.1.2.11	x		|		
Serveur FTP utilise si poss. code de rép défini	4.1.2.11		x			
Nouveau code de réponse suivant parag 4.2	4.1.2.11			x		
Usager FTP n'utilise que n° d'ordre élevé de rép.	4.1.2.11		x			
Usager FTP traite lignes réponse multi-lignes	4.1.2.11	x				
Usager FTP traite spécial réponses 421	4.1.2.11				x	
Accès données défaut a même adr IP que conn ctl	4.1.2.12	x				
L'us. FTP envoie cmds Telnet exc. SYNCH, IP	4.1.2.12					x
L'usager FTP négocie les options Telnet	4.1.2.12					x
Le serveur FTP traite les options Telnet	4.1.2.12	x				
Traite des cmds de répertoire "expérimentales"	4.1.3.1		x			
Tempo. d'inactivité dans serveur FTP	4.1.3.2		x			

Tempo. d'inactivité configurable	4.1.3.2		x			
Receveur pointe données au marqueur de redém.	4.1.3.4		x			
L'envoyeur suppose réponses 110 synchrones	4.1.3.4					x
Accepte TYPE :						
ASCII - Non-imprimable (AN)	4.1.2.13	x				
ASCII - Telnet (AT) – si même que AN	4.1.2.2		x			
ASCII – Commande chariot (AC)	3.1.1.5.2	(959)		x		
EBCDIC - (toute forme)	3.1.1.2	(959)		x		
IMAGE	4.1.2.1	x				
LOCAL 8	4.1.2.1	x				
LOCAL m *(note 2)*	4.1.2.1			x		
Accepte MODE :						
Stream	4.1.2.13	x				
Block	3.4.2	(959)		x		
Accepte STRUCTURE :						
File	4.1.2.13	x				
Record *(note 3)*	4.1.2.13	x				
Page	4.1.2.3				x	
Accepte les commandes :						
USER	4.1.2.13	x				
PASS	4.1.2.13	x				
ACCT	4.1.2.13	x				
CWD	4.1.2.13	x				
CDUP	4.1.2.13	x				
SMNT	5.3.1	(959)		x		
REIN	5.3.1	(959)		x		
QUIT	4.1.2.13	x				
PORT	4.1.2.13	x				
PASV	4.1.2.6	x				
TYPE *(note 1)*	4.1.2.13	x				
STRU *(note 1)*	4.1.2.13	x				
MODE *(note 1)*	4.1.2.13	x				
RETR	4.1.2.13	x				
STOR	4.1.2.13	x				
STOU	5.3.1	(959)		x		
APPE	4.1.2.13	x				
ALLO	5.3.1	(959)		x		
REST	5.3.1	(959)		x		
RNFR	4.1.2.13	x				
RNTO	4.1.2.13	x				
ABOR	5.3.1	(959)		x		
DELE	4.1.2.13	x				
RMD	4.1.2.13	x				
MKD	4.1.2.13	x				
PWD	4.1.2.13	x				
LIST	4.1.2.13	x				
NLST	4.1.2.13	x				
SITE	4.1.2.8			x		
STAT	4.1.2.13	x				
SYST	4.1.2.13	x				
HELP	4.1.2.13	x				
NOOP	4.1.2.13	x				
Interface d'usager :						
Noms de chemin arbitraires	4.1.4.1	x				
Accepte commande "QUOTE"	4.1.4.2	x				
Transfert immédiat des commandes de contr.	4.1.4.2		x			
Affiche messages d'erreur à l'usager	4.1.4.3		x			
Mode verbeux	4.1.4.3		x			
Maintien de synchronisation avec serveur	4.1.4.4		x			

Notes :
(1) Pour les valeurs données précédemment.
(2) Ici m est un nombre de bits dans un mot de mémoire.
(3) Exigé pour un hôte avec un système de fichier structuré en enregistrement, facultatif autrement.

4.2 Protocole trivial de transfert de fichier -- TFTP

4.2.1 Introduction

Le protocole trivial de transfert de fichier (TFTP, *Trivial File Transfer Protocol*) est défini dans la RFC-783 [TFTP:1].

TFTP fournit sa propre livraison fiable avec UDP comme protocole de transport, en utilisant un simple système d'accusé de réception d'arrêt et attente. Comme TFTP n'a une fenêtre efficace que d'un seul segment de 512 octets, il ne peut fournir de bonnes performances que sur des chemins qui ont un faible produit délai*bande passante. L'interface de fichier TFTP est très simple, et ne fournit ni contrôle ni sécurité d'accès.

L'application la plus importante de TFTP est l'amorçage d'un hôte sur un réseau local, car il est assez simple et petit pour être facilement mis en œuvre dans EPROM [BOOT:1], [BOOT:2]. Les développeurs sont invités à accepter TFTP pour l'amorçage.

4.2.2 Survol du protocole

La spécification TFTP [TFTP:1] est écrite en style ouvert, et ne spécifie pas complètement de nombreux aspects du protocole.

4.2.2.1 Modes de transfert : RFC-783, page 3

Le mode de transfert "mail" NE DEVRAIT PAS être accepté.

4.2.2.2 En-tête UDP : RFC-783, page 17

Le champ Longueur d'un en-tête UDP est défini de façon incorrecte ; il inclut la longueur de l'en-tête UDP (8).

4.2.3 Questions spécifiques

4.2.3.1 Syndrome de l'apprenti sorcier

Il y a une erreur sérieuse, connue sous le nom de "Syndrome de l'apprenti sorcier", dans la spécification du protocole. Bien qu'elle ne cause pas un fonctionnement incorrect du transfert (le fichier sera toujours transféré correctement si le transfert s'achève) cette erreur peut causer une retransmission excessive, qui peut provoquer le dépassement de temporisation par le transfert.

Les mises en œuvre DOIVENT contenir la réparation de ce problème : l'envoyeur (c'est-à-dire, le côté à l'origine des paquets de données) ne doit jamais renvoyer le paquet de données en cours à réception d'un ACK dupliqué.

Discussion : L'erreur est causée par la règle du protocole selon laquelle l'un ou l'autre côté, à réception d'un vieux datagramme dupliqué, peut renvoyer le datagramme en cours. Si un paquet est retardé dans le réseau mais est bien livré ultérieurement après que chaque côté est arrivé en fin de temporisation et a retransmis un paquet, une copie dupliquée de la réponse peut être générée. Si l'autre côté répond à ce duplicata par un duplicata de sa propre réponse, chaque datagramme sera alors envoyé en duplicata pour le reste du transfert (à moins qu'un datagramme ne soit perdu, ce qui rompt la répétition). Ce qui est pire encore, car le retard est souvent causé par l'encombrement, c'est que cette transmission de duplicata va normalement causer encore plus d'encombrement, conduisant à plus de paquets retardés, etc.

L'exemple suivant peut aider à éclaircir ce problème.

	TFTP A	**TFTP B**
(1)	Reçoit ACK X-1 Envoie DATA X	
(2)		Reçoit DATA X Envoie ACK X

(ACK X est retardé dans le réseau, et A arrive en fin de temporisation) :

(3) Retransmet DATA X
(4) Reçoit à nouveau DATA X
Envoie à nouveau ACK X
(5) Reçoit (en retard) ACK X
Envoi de DATA X+1
(6) Reçoit DATA X+1
Envoie ACK X+1
(7) Reçoit à nouveau ACK X
Envoi à nouveau de DATA X+1
(8) Reçoit à nouveau DATA X+1
Envoi à nouveau de ACK X+1
(9 Reçoit ACK X+1
Envoi de DATA X+2
(10) Reçoit DATA X+2
Envoi de ACK X+3
(11) Reçoit à nouveau ACK X+1
Envoi à nouveau de DATA X+2
(12) Reçoit à nouveau DATA X+2

Remarquer qu'une fois qu'arrive l'accusé de réception retardé, le protocole se met à dupliquer tous les paquets qui suivent (séquences 5-8 et 9-12). Le problème n'est pas causé par l'arrivée en fin de temporisation de l'un ou l'autre côté, mais par le fait que les deux côtés retransmettent le paquet en cours quand ils reçoivent un dupliqué.

La solution est de casser la boucle de retransmission, comme indiqué ci-dessus. C'est un comportement analogue à celui de TCP. Il est alors possible de retirer la temporisation de retransmission chez le receveur, car l'accusé de réception renvoyé ne provoquera jamais d'action ; c'est une simplification utile dans laquelle TFTP est utilisé dans un programme d'amorçage. Il est parfait pour permettre de conserver la temporisation, et il peut être utile si l'accusé de réception retransmis en remplace un qui a vraiment été perdu dans le réseau. L'envoyeur a, bien sûr, toujours besoin d'un temporisateur de retransmission.

4.2.3.2 Algorithmes de fin de temporisation

Une mise en œuvre de TFTP DOIT utiliser une temporisation adaptable.

Mise en œuvre : Les algorithmes de retransmission de TCP fournissent une base de départ utile. Une augmentation exponentielle d'attente de temporisation de retransmission est au moins nécessaire.

4.2.3.3 Extensions

Diverses extensions non standard ont été apportées à TFTP, y compris des modes de transfert additionnels et un mode de fonctionnement sécurisé (avec des mots de passe). Aucune d'elles n'a été normalisée.

4.2.3.4 Contrôle d'accès

Une mise en œuvre de serveur TFTP DEVRAIT inclure un contrôle d'accès configurable sur les noms de chemin qui sont permis dans les opérations TFTP.

4.2.3.5 Demande de diffusion

Une demande TFTP dirigée sur une adresse de diffusion DEVRAIT être ignorée en silence.

Discussion : Du fait de la faiblesse de la capacité de contrôle d'accès de TFTP, les demandes de TFTP dirigées sur la diffusion à des réseaux aléatoires pourraient créer de sérieux trous pour la sécurité.

4.2.4 Résumé des exigences pour TFTP

Caractéristique	Parag.	Doit	Devrait	Peut	Ne devrait pas	Ne doit pas
Régler le syndrome de l'apprenti sorcier	4.2.3.1	x				
Modes de transfert :						

netascii	RFC-783	x				
octet	RFC-783	x				
mail	4.2.2.1					x
extensions	4.2.3.3			x		
Utilise une temporisation adaptable	4.2.3.2	x				
Contrôle d'accès configurable	4.2.3.4		x			
Ignore en silence les demandes de diffusion	4.2.3.5		x			

5 Messagerie électronique -- SMTP et la RFC-822

5.1 Introduction

Dans la suite des protocoles TCP/IP, la messagerie électronique dans le format spécifié par la RFC-822 [SMTP:2] est transmise en utilisant le protocole simple de transfert de messagerie (SMTP, *Simple Mail Transfer Protocol*) défini dans la RFC-821 [SMTP:1].

Bien que SMTP soit resté inchangé au fil des ans, la communauté de l'Internet a apporté plusieurs changements à la façon de l'utiliser. En particulier, la conversion au système des noms de domaine (DNS) a causé le changement des formats d'adresse et de l'acheminement de la messagerie. Dans cette section, on suppose que le lecteur est familier des concepts et terminologie du DNS, dont les exigences sont données au paragraphe 6.1.

La RFC-822 spécifie le format standard de l'Internet pour les messages de la messagerie électronique. La RFC-822 se substitue à une norme plus ancienne, la RFC-733, qui peut toujours être utilisée dans quelques endroits, bien qu'elle soit obsolète. Les deux formats sont parfois mentionnés par un simple nombre ("822" et "733").

La RFC-822 est utilisée dans certains environnements de messagerie non Internet avec des protocoles de transfert de messagerie différents de SMTP, et SMTP a aussi été adapté pour être utilisé dans des environnements non Internet. Noter que le présent document ne présente les règles que pour une utilisation de SMTP et de la RFC-822 dans l'environnement Internet ; les autres environnements de messagerie qui utilisent ces protocoles sont supposés avoir leurs propres règles.

5.2 Survol du protocole

Ce paragraphe couvre à la fois la RFC-821 et la RFC-822.

La spécification SMTP dans la RFC-821 est claire et contient de nombreux exemples, de sorte que les développeurs ne devraient pas la trouver difficile à comprendre. Ce paragraphe se contente de mettre à jour ou d'annoter des portions de la RFC-821 pour se conformer à l'usage courant.

La RFC-822 est un document long et dense, qui définit une syntaxe riche. Malheureusement, des mises en œuvre incomplètes ou déficientes de la RFC-822 sont courantes. En fait, presque tous les nombreux formats de la RFC-822 sont utilisés, de sorte qu'une mise en œuvre a généralement besoin de reconnaître et d'interpréter correctement toute la syntaxe de la RFC-822.

5.2.1 Modèle SMTP : RFC-821 paragraphe 2

Discussion : La messagerie est envoyée par une série de transactions de demande/réponse entre un client, l'"envoyeur-SMTP," et un serveur, le "receveur SMTP". Ces transactions passent (1) le message propre, qui se compose de l'en-tête et du corps, et (2) la source et adresse SMTP de destination, appelée "enveloppe".

Les programmes SMTP sont analogues aux agents de transfert de message (MTA, *Message Transfer Agent*) de X.400. Il y a un autre niveau de logiciel de protocole, plus proche de l'utilisateur final, qui est chargé de composer et analyser les en-têtes de message de la RFC-822 ; ce composant est appelé "Agent d'utilisateur" dans X.400, et on utilise ce terme dans le présent document. Il y a une claire distinction logique entre l'agent d'utilisateur et la mise en œuvre SMTP, car ils fonctionnent sur des niveaux de protocole différents. Noter cependant que cette distinction peut n'être pas exactement reflétée dans la structure d'une mise en œuvre normale de messagerie Internet. Il y a souvent un programme appelé le "messageur" qui met en œuvre SMTP et aussi certaines fonctions d'agent d'utilisateur ; le reste des fonctions d'agent d'utilisateur est inclus dans une interface d'utilisateur utilisée pour entrer et lire la messagerie.

L'enveloppe SMTP est construite dans le site d'origine, normalement par l'agent d'utilisateur lorsque le message est d'abord mis en file d'attente pour le programme d'envoyeur SMTP. Les adresses de l'enveloppe peuvent être déduites

d'informations contenues dans l'en-tête du message, fournies par l'interface d'utilisateur (par exemple, mettre en œuvre une demande bcc:) ou déduites d'informations de configuration locales (par exemple, expansion d'une liste de diffusion). L'enveloppe SMTP ne peut en général pas être à nouveau déduite de l'en-tête à un stade ultérieur de la délivrance du message, de sorte que l'enveloppe est transmise séparément du message lui-même en utilisant les commandes MAIL et RCPT de SMTP.

Le texte de la RFC-821 suggère que cette messagerie est à délivrer à un utilisateur individuel chez un hôte. Avec l'avènement du système des domaines et de l'acheminement de la messagerie à l'aide des enregistrements de ressource d'échange de messagerie (MX, *mail-exchange*) les développeurs devraient penser que la messagerie se livre à un utilisateur dans un domaine, qui peut être ou non un hôte particulier. Cela NE CHANGE PAS le fait que SMTP est un protocole d'échange de messagerie d'hôte à hôte.

5.2.2 Canonisation : RFC-821 paragraphe 3.1

Les noms de domaine qu'un envoyeur SMTP envoie dans les commandes MAIL et RCPT DOIVENT avoir été "canonisés," c'est-à-dire qu'ils doivent être des noms principaux pleinement qualifiés ou des domaines littéraux, et pas des surnoms ou des abréviations du domaine. Un nom canonisé identifie directement un hôte ou est un nom MX ; il ne peut pas être un CNAME.

5.2.3 Commandes VRFY et EXPN : RFC-821 paragraphe 3.3

Un receveur SMTP DOIT mettre en œuvre VRFY et DEVRAIT mettre en œuvre EXPN (cette exigence subroge celles de la RFC-821). Cependant, il PEUT y avoir des informations de configuration pour désactiver VRFY et EXPN dans une installation particulière ; elles peuvent même permettre de désactiver EXPN pour les listes choisies.

Un nouveau code de réponse est défini pour la commande VRFY :

252 ne peut pas VRFY l'usager (par exemple, les infos ne sont pas locales) mais va prendre le message pour cet usager et tenter la livraison.

Discussion : Les utilisateurs et administrateurs de SMTP utilisent régulièrement ces commandes pour faire le diagnostic des problèmes de livraison de messagerie. Avec l'utilisation croissante de l'expansion multi niveaux des listes de diffusion (parfois plus de deux niveaux) EXPN a une importance croissante pour le diagnostic des boucles de messagerie involontaires. D'un autre côté, certains pensent que EXPN représente un risque significatif pour la confidentialité et même peut-être la sécurité.

5.2.4 Commandes SEND, SOML, et SAML : RFC-821 paragraphe 3.4

SMTP PEUT mettre en œuvre les commandes pour envoyer un message au terminal d'un utilisateur : SEND, SOML, et SAML.

Discussion : Il a été suggéré que l'utilisation d'un relais de messagerie à travers un enregistrement MX est incohérente avec l'intention de SEND de délivrer immédiatement et directement un message au terminal d'un utilisateur. Cependant, un receveur SMTP qui n'est pas capable d'écrire directement au terminal de l'utilisateur peut retourner une réponse "251 Usager non local" au RCPT qui suit un SEND, pour informer l'origine d'un retard possible de la livraison.

5.2.5 Commande HELO : RFC-821 paragraphe 3.5

L'envoyeur SMTP DOIT s'assurer que le paramètre <domaine> dans une commande HELO est un nom principal de domaine d'hôte valide pour l'hôte client. Il en résulte que le receveur SMTP n'aura pas à effectuer une résolution MX sur ce nom afin de valider le paramètre HELO.

Le receveur de HELO PEUT vérifier que le paramètre HELO correspond réellement à l'adresse IP de l'envoyeur. Cependant, le receveur NE DOIT PAS refuser d'accepter un message, même si la commande HELO de l'envoyeur échoue à la vérification.

Discussion : Vérifier le paramètre HELO exige une recherche de nom de domaine et peut donc prendre un temps considérable. Un autre outil pour détecter les sources de messagerie erronées est suggéré ci-dessous (voir à "Commande DATA").

Noter aussi que l'argument HELO est toujours exigé pour une syntaxe <domain> valide, car il va apparaître dans une ligne Received: ; autrement, une erreur 501 doit être envoyée.

Mise en œuvre :
Lors de l'échec de la validation d'un paramètre HELO, la procédure suggérée est d'insérer une note sur l'authenticité douteuse de l'envoyeur dans l'en-tête de message (par exemple, dans la ligne "Received:").

5.2.6 Relais de messagerie : RFC-821 paragraphe 3.6

On distingue trois types de transmission de messagerie (en différé) :

(1) Un simple transmetteur ou "échangeur de messages" transmet un message en utilisant des connaissances particulières sur le receveur ; voir le paragraphe 3.2 de la RFC-821.

(2) Un "relais" de messagerie SMTP transmet un message au sein d'un environnement de messagerie SMTP par suite d'une route de source explicite (comme défini au paragraphe 3.6 de la RFC-821). La fonction de relais SMTP utilise la forme "@...:" de route de source de la RFC-822 (voir le paragraphe 5.2.19 ci-dessous).

(3) Une "passerelle"de messagerie passe un message entre différents environnements. Les règles pour les passerelles de messagerie sont exposées ci-dessous au paragraphe 5.3.7.

Un hôte Internet qui transmet un message mais n'est pas une passerelle vers un environnement de messagerie différent (c'est-à-dire, qui rentre dans (1) ou (2)) NE DEVRAIT PAS altérer un des champs d'en-tête existants, bien que l'hôte aille ajouter une ligne Received: appropriée comme exigé au paragraphe 5.2.8.

Un envoyeur SMTP NE DEVRAIT PAS envoyer de commande RCPT TO: contenant une route de source explicite en utilisant la forme d'adresse "@...:". Et donc, la fonction de relais définie au paragraphe 3.6 de la RFC-821 ne devrait pas être utilisée.

Discussion : L'intention est de décourager tous les acheminements de source et d'abolir l'acheminement de source explicite pour la livraison de messagerie au sein de l'environnement Internet. L'acheminement de source est inutile ; la simple adresse cible "usager@domaine" devrait toujours suffire. Ceci est le résultat d'une décision architecturale explicite d'utiliser les dénominations universelles plutôt que l'acheminement de source pour la messagerie. Et donc, SMTP fournit une connexité de bout en bout, et le DNS fournit des noms uniques au monde, indépendants de la localisation. Les enregistrements MX traitent les principaux cas où l'acheminement de source pourrait être autrement nécessaire.

Un receveur SMTP DOIT accepter la syntaxe de route de source explicite dans l'enveloppe, mais il PEUT mettre en œuvre la fonction relais comme défini au paragraphe 3.6 de la RFC-821. Si il ne met pas en œuvre la fonction relais, il DEVRAIT tenter de livrer le message directement à l'hôte qui est à la droite du signe "@" le plus à droite.

Discussion : Par exemple, supposons qu'un hôte qui ne met pas en œuvre la fonction relais reçoive un message avec la commande SMTP : "RCPT TO:<@ALPHA,@BETA:joe@GAMMA>", où ALPHA, BETA, et GAMMA représentent des noms de domaines. Plutôt que d'immédiatement refuser le message avec une réponse d'erreur 550 comme suggéré à la page 20 de la RFC-821, l'hôte devrait essayer de transmettre le message directement à GAMMA, en utilisant : "RCPT TO:<joe@GAMMA>". Comme cet hôte n'accepte pas le relayage, il n'est pas obligé de mettre à jour le chemin inverse.

Certains ont suggéré que l'acheminement de source pourrait être nécessaire lors de l'acheminement manuel de messagerie en cas d'échec ; cependant, la réalité et l'importance de ce besoin sont controversées. L'utilisation à cette fin du relais explicite de messagerie SMTP est déconseillée, et en fait, elle peut ne pas réussir, car de nombreux systèmes d'hôte ne l'acceptent pas. Certains ont utilisé le "hachage en %" (voir le paragraphe 5.2.16) pour cela.

5.2.7 Commande RCPT : RFC-821 paragraphe 4.1.1

Un hôte qui prend en charge un receveur SMTP DOIT prendre en charge la boîte aux lettres réservée "Postmaster".

Le receveur SMTP PEUT vérifier les paramètres RCPT lors de leur arrivée ; cependant, les réponses RCPT NE DOIVENT PAS être retardées au-delà d'un délai raisonnable (voir le paragraphe 5.3.2).

Donc, une réponse "250 OK" à un RCPT n'implique pas nécessairement que les adresses de livraison sont valides. Les erreurs découvertes après l'acceptation du message seront rapportées en envoyant un message de notification à l'adresse appropriée (voir le paragraphe 5.3.3).

Discussion : L'ensemble des conditions dans lesquelles un paramètre RCPT peut être validé immédiatement est un choix des conceptions d'ingénierie. Faire rapport des erreurs de boîte aux lettres de destination à l'envoyeur SMTP avant le transfert des messages est généralement souhaitable pour gagner du temps et épargner la bande passante du réseau, mais cet avantage est perdu si la vérification RCPT prend trop de temps.

Par exemple, le receveur peut vérifier immédiatement une référence locale, comme une boîte aux lettres enregistrée localement. D'un autre côté, la limitation à un "délai raisonnable" implique généralement de différer la vérification d'une liste de diffusion jusqu'après le transfert et l'acceptation du message, car la vérification d'une grosse liste de diffusion peut prendre très longtemps. Une mise en œuvre peut choisir de différer ou non la validation des adresses qui ne sont pas locales et donc exigent une recherche de DNS. Si une recherche de DNS est effectuée mais qu'une erreur logique du système des domaines survient (par exemple, l'expiration d'une temporisation) la validité doit être supposée.

5.2.8 Commande DATA : RFC-821 paragraphe 4.1.1

Tout receveur SMTP (pas seulement celui qui "accepte un message pour le relayer ou pour livraison finale" [SMTP:1]) DOIT insérer une ligne "Received:" au début d'un message. Dans cette ligne, appelée "ligne d'horodatage" dans la RFC-821 :

* Le champ FROM DEVRAIT contenir à la fois (1) le nom de l'hôte de source tel que présenté dans la commande HELO et (2) un domaine littéral contenant l'adresse IP de la source, déterminée à partir de la connexion TCP.

* Le champ ID PEUT contenir un "@" comme suggéré dans la RFC-822, mais ce n'est pas exigé.

* Le champ FOR PEUT contenir une liste des entrées de <path> lorsque plusieurs commandes RCPT ont été données.

Un programme de messagerie Internet NE DOIT PAS changer une ligne Received: qui a été précédemment ajoutée à l'en-tête du message.

Discussion : L'inclusion à la fois de l'hôte de source et de l'adresse IP de source dans la ligne Received: peut fournir suffisamment d'informations pour traquer les sources de messages illicites et éliminer le besoin de vérifier explicitement le paramètre HELO.

Les lignes Received: sont principalement destinées au traçage des routes de messagerie par des hommes, essentiellement pour le diagnostic des défauts. Voir aussi l'exposé du paragraphe 5.3.7.

Lorsque le receveur SMTP fait la "livraison finale" d'un message, il DOIT alors passer l'adresse MAIL FROM: de l'enveloppe SMTP avec le message, au cas où un message de notification d'erreur devrait être envoyé ultérieurement (voir au paragraphe 5.3.3). Il y a la même exigence lorsqu'il y a une passerelle de l'Internet vers un environnement de messagerie différent ; voir au paragraphe 5.3.7.

Discussion : Noter que la réponse finale à la commande DATA ne dépend que de la réussite du transfert et de la mémorisation du message. Tout problème posé par la ou les adresses de destination doit être (1) soit rapporté dans une réponse d'erreur SMTP à la ou les commandes RCPT, (2) soit rapporté dans un message d'erreur ultérieur envoyé à l'origine du message.

Mise en œuvre : Les informations de MAIL FROM: peuvent être passées comme paramètre ou dans une ligne Return-Path: insérée au début du message.

5.2.9 Syntaxe de commande : RFC-821 paragraphe 4.1.2

La syntaxe indiquée dans la RFC-821 pour la commande MAIL FROM: omet le cas du chemin vide : "MAIL FROM: <>" (voir la RFC-821, page 15). Un chemin inverse vide DOIT être accepté.

5.2.10 Réponses SMTP : RFC-821 paragraphe 4.2

Un receveur SMTP DEVRAIT n'envoyer que les codes de réponse dont la liste figure au paragraphe 4.2.2 de la RFC-821 ou dans le présent document. Un receveur SMTP DEVRAIT utiliser le texte donné dans les exemples de la RFC-821 chaque fois qu'approprié.

Un envoyeur SMTP DOIT ne déterminer ses actions que par le code de réponse, et non par le texte (excepté pour les réponses 251 et 551) ; tout texte, y compris pas de texte du tout, doit être acceptable. L'espace (blanche) suivant le code de réponse est considéré comme partie du texte. Chaque fois que possible, un envoyeur SMTP DEVRAIT tester seulement le premier chiffre du code de réponse, comme spécifié à l'appendice E de la RFC-821.

Discussion : Des problèmes d'interopérabilité sont apparus avec les systèmes SMTP qui utilisent des codes de réponses qui ne figurent pas explicitement sur la liste du paragraphe 4.3 de la RFC-821 mais sont légaux conformément à la théorie des codes de réponse expliquée dans l'appendice E.

5.2.11 Transparence : RFC-821 paragraphe 4.5.2

Les développeurs DOIVENT être sûrs que leur système de messagerie ajoute et supprime toujours les points pour assurer la transparence des messages.

5.2.12 Utilisation de WKS dans le traitement MX : RFC-974, page 5

La RFC-974 [SMTP:3] recommandait que le système des domaines soit interrogé sur les enregistrements de WKS ("*Well-Known Service*", services bien connus) pour vérifier que chaque cible de message proposée accepte bien SMTP. Les expériences ultérieures ont montré que WKS ne jouit pas d'un grand soutien, de sorte que l'étape WKS dans le traitement MX NE DEVRAIT PAS être utilisée.

Ci-après figurent des notes sur la RFC-822, organisées par section de ce document.

5.2.13 Spécification du message : RFC-822 section 4

La syntaxe indiquée pour la ligne Return-path omet la possibilité d'un chemin de retour nul, qui est utilisé pour empêcher les boucles de notifications d'erreurs (voir au paragraphe 5.3.3). La syntaxe complète est :

return = "Return-path" ":" route-addr / "Return-path" ":" "<" ">"

L'ensemble des champs d'en-tête facultatifs est étendu ici pour comporter le champ Content-Type défini dans la RFC-1049 [SMTP:7]. Ce champ "permet aux systèmes de lecture de messagerie d'identifier automatiquement le type d'un corps de message structuré et de le traiter pour l'afficher en conséquence" [SMTP:7]. Un agent d'utilisateur PEUT accepter ce champ.

5.2.14 Spécification de la date et de l'heure : RFC-822 section 5

La syntaxe pour la date est changée ici en :

date = 1*2CHIFFRE mois 2*4CHIFFRE

Tous les logiciels de messagerie DEVRAIT utiliser des années de quatre chiffres dans les dates, pour faciliter la transition au siècle suivant.

Il y a une tendance forte en faveur de l'utilisation d'indicateurs numériques de fuseau horaire, et les mises en œuvre DEVRAIENT utiliser des zones horaires numériques au lieu de noms de zone horaire. Cependant, toutes les mises en œuvre DOIVENT accepter l'une ou l'autre notation. Si les noms de zone horaire sont utilisés, ils DOIVENT être exactement comme définis dans la RFC-822.

Les zones horaires militaires sont spécifiées de façon incorrecte dans la RFC-822 : elles comptent à l'envers du Temps Universel (les signes sont inversés). Il en résulte que les zones horaires militaires dans les en-têtes de la RFC-822 ne portent aucune information.

Finalement, noter qu'il y a une faute de frappe dans la définition de "zone" dans le résumé de la syntaxe de l'appendice D ; la définition correcte est celle de la section 3 de la RFC-822.

5.2.15 Changement de syntaxe : RFC-822 paragraphe 6.1

La définition syntaxique de "mailbox" dans la RFC-822 est changée ici en :

mailbox = addr-spec ; adresse simple

/ [phrase] route-addr ; nom & spécification d'adresse

C'est à dire que la phrase qui précède une adresse d'acheminement est maintenant FACULTATIVE. Ce changement rend légal l'en-tête suivant, par exemple :

From: <craig@nnsc.nsf.net>

5.2.16 Partie locale : RFC-822 paragraphe 6.2

La spécification d'adresse de boîte aux lettres de base est de la forme : "local-part@domaine". Ici "local-part", parfois appelée le "côté gauche" de l'adresse, est fonction du domaine.

Un hôte qui transmet le message mais n'est pas l'hôte de destination impliqué par le "domaine" du côté droit NE DOIT PAS interpréter ou modifier la "partie locale" de l'adresse.

Lorsque de la messagerie est à faire passer d'un environnement de messagerie Internet à un environnement de messagerie étranger (voir au paragraphe 5.3.7) les informations d'acheminement pour cet environnement étranger PEUVENT être incorporées au sein de la "partie locale" de l'adresse. La passerelle va alors interpréter cette partie locale de la façon appropriée pour l'environnement de messagerie étranger.

Discussion : Bien que les routes de source soient déconseillées au sein de l'Internet (voir au paragraphe 5.2.6) il y a des environnements de messagerie non Internet dont les mécanismes de livraison dépendent des routes de source. Les routes de source pour les environnements extra Internet peuvent généralement être enfouis dans la "partie locale" de l'adresse (voir au paragraphe 5.2.16) alors que le message traverse l'Internet. Lorsque le message atteint la passerelle appropriée de messagerie Internet, la passerelle interprète la partie locale et construit l'adresse ou chemin nécessaire pour l'environnement de messagerie visé.

Par exemple, un hôte Internet peut vouloir envoyer un message à : "a!b!c!user@gateway-domain". La partie locale complexe "a!b!c!user" ne sera pas interprétée au sein du domaine Internet, mais pourrait être analysée et comprise par la passerelle de messagerie spécifiée.

Une route de source incorporée est parfois codée dans la "partie locale" en utilisant "%" comme opérateur de séparation à droite. Par exemple, dans :

user%domain%relay3%relay2@relay1

la convention "%" implique que le message soit acheminé à travers les "relay1", "relay2", puis "relay3", et finalement à "user" chez "domain". C'est ce qui est habituellement appelé "hachage en %". Il est suggéré que "%" ait une priorité inférieure à celle de tout autre opérateur d'acheminement (par exemple, "!") caché dans la partie locale ; par exemple, "a!b %c" serait à interpréter comme "(a!b)%c".

Seul l'hôte cible (dans ce cas, "relay1") est autorisé à analyser la partie locale "user%domain%relay3%relay2".

5.2.17 Domaines littéraux : RFC-822 paragraphe 6.2.3

Un messageur DOIT être capable d'accepter et analyser un littéral de domaine Internet dont le contenu ("dtext"; voir la RFC-822) est une adresse d'hôte en décimal séparé par des points. Cela satisfait à l'exigence du paragraphe 2.1 pour le cas de la messagerie.

Une mise en œuvre de SMTP DOIT accepter et reconnaître un littéral de domaine pour toutes ses propres adresses IP.

5.2.18 Erreurs courantes de format d'adresse : RFC-822 paragraphe 6.1

Les erreurs de format ou d'analyse des adresses de la RFC 822 sont malheureusement courantes. Ce paragraphe ne mentionne que les plus communes. Un agent d'utilisateur DOIT accepter tous les formats d'adresse valides de la RFC-822, et NE DOIT PAS générer de syntaxe d'adresse illégale.

- Une erreur courante est d'oublier le deux-points après un identifiant de groupe.
- Certains systèmes ne donnent pas la pleine qualification des noms de domaine dans les messages qu'ils génèrent. Le côté droit d'un signe "@" dans un champ adresse d'en-tête DOIT être un nom de domaine pleinement qualifié.

Par exemple, certains systèmes manquent à qualifier pleinement l'adresse From: ; cela empêche une commande "reply" de l'interface d'utilisateur de construire automatiquement une adresse de retour.

Discussion : Bien que la RFC-822 permette l'utilisation locale de noms de domaine abrégés au sein d'un domaine, l'application de la RFC-822 dans la messagerie Internet ne le permet pas. L'intention est qu'un hôte Internet n'envoie pas d'en-tête de message SMTP contenant un nom de domaine abrégé dans un champ d'adresse. Cela permet que les champs d'adresse de l'en-tête soient passés sans altération à travers l'Internet, comme exigé au paragraphe 5.2.6.

o Certains systèmes font une mauvaise analyse des routes de source à plusieurs bonds explicites telles que :

@relay1,@relay2,@relay3:user@domain.

o Certains systèmes surqualifient des noms de domaine en ajoutant un point en queue à certains noms de domaine ou à tous dans les noms de domaine ou dans les identifiants de message. C'est une violation de la syntaxe de la RFC-822.

5.2.19 Routes de source explicite : RFC-822 paragraphe 6.2.7

Un logiciel d'hôte Internet NE DEVRAIT PAS créer un en-tête de RFC-822 contenant une adresse avec une route de source explicite, mais DOIT accepter de tels en-têtes pour la compatibilité avec les systèmes antérieurs.

Discussion : La RFC-822 dit en forme de litote "L'utilisation de l'acheminement explicite de source est déconseillé". De nombreux hôtes mettaient incorrectement en œuvre les routes de source de la RFC-822, de sorte que sa syntaxe ne peut pas être utilisée sans ambiguïté en pratique. De nombreux utilisateurs estiment que la syntaxe est laide. Les routes de source explicites ne sont pas nécessaires dans l'enveloppe de messagerie pour la livraison ; voir au paragraphe 5.2.6. Pour toutes ces raisons, les routes de source explicites qui utilisent les notations de la RFC-822 ne sont pas à utiliser dans les en-têtes de messagerie Internet.

Comme exposé au paragraphe 5.2.16, il est nécessaire de permettre qu'une route de source explicite soit enfouie dans la partie locale d'une adresse, par exemple, en utilisant le "hachage en %", afin de permettre que la messagerie soit passée dans un autre environnement dans lequel l'acheminement explicite de source est nécessaire. Un utilisateur vigilant observera qu'il n'y a aucun moyen pour un agent d'utilisateur de détecter et empêcher l'utilisation d'un tel acheminement implicite de source lorsque la destination est au sein de l'Internet. On peut seulement déconseiller l'acheminement de source de quelque sorte qu'il soit au sein de l'Internet, comme inutile et indésirable.

5.3 Questions spécifiques

5.3.1 Stratégies SMTP de mise en file d'attente

La structure courante d'une mise en œuvre d'hôte SMTP comporte des boîtes aux lettres d'usager, une ou plusieurs zones pour la mise en file d'attente des messages en transit, et un ou plusieurs traitements d'automate pour l'envoi et la réception des messages. La structure exacte va varier selon les besoins des utilisateurs sur l'hôte et le nombre et la taille des listes de diffusion prises en charge par l'hôte. On décrit plusieurs optimisations qui se sont révélées utiles, en particulier pour les messageurs qui supportent de hauts niveaux de trafic.

Toute stratégie de mise en file d'attente DOIT inclure :
o Des temporisations sur toutes les activités. Voir au paragraphe 5.3.2.
o De ne jamais envoyer de message d'erreur en réponse à des messages d'erreur.

5.3.1.1 Stratégie d'envoi

Le modèle général d'un envoyeur SMTP est un ou plusieurs processus qui essayent périodiquement de transmettre les messages sortants. Dans un système normal, le programme qui compose un message a des méthodes pour demander une attention immédiate à une nouvelle séquence de messages sortants, alors que les messages qui ne peuvent pas être transmis immédiatement DOIVENT être mis en file d'attente et réessayés périodiquement par l'envoyeur. Une entrée de file d'attente de messagerie comportera non seulement le message lui-même mais aussi les informations d'enveloppe.

L'envoyeur DOIT retarder les essais sur une destination particulière après l'échec d'un essai. En général, l'intervalle d'essai DEVRAIT être au moins de 30 minutes ; cependant, des stratégies plus sophistiquées et variables seront bénéfiques lorsque l'envoyeur SMTP peut déterminer la raison de la non livraison.

Les essais continuent jusqu'à ce que le message soit transmis ou que l'envoyeur abandonne ; le délai d'abandon doit généralement être d'au moins 4 à 5 jours. Les paramètres de l'algorithme de réessai DOIVENT être configurables.

Un envoyeur DEVRAIT garder une liste des hôtes qu'il ne peut pas atteindre et les temporisations correspondantes, plutôt que de simplement réessayer les éléments de messagerie en file d'attente.

Discussion :
L'expérience suggère que les échecs sont normalement temporaires (le système cible est en panne) en faveur d'une politique de deux tentatives de connexion dans la première heure de présence du message dans la file d'attente, et puis d'y revenir toutes les deux ou trois heures.

L'envoyeur SMTP peut raccourcir le délai de mise en file d'attente en coopérant avec le receveur SMTP. En particulier, si un message est reçu d'une adresse particulière, il est évident que tout message pour cet hôte dans la file d'attente peut maintenant être envoyé.

La stratégie peut encore être modifiée pour tenir compte de la présence de plusieurs adresses pour l'hôte (voir le paragraphe 5.3.4) pour optimiser les délais de livraison par rapport à l'utilisation des ressources.

Un envoyeur SMTP peut avoir une grande file d'attente de messages pour chaque hôte de destination indisponible, et si il réessaye tous ces messages à chaque cycle d'essai, il y aurait une surcharge excessive de l'Internet et l'automate serait bloqué pour une longue période. Noter qu'une mise en œuvre de SMTP peut généralement déterminer qu'une tentative de livraison a échoué après une temporisation de seulement une minute ou plus ; une temporisation d'une minute par connexion répétée pour des douzaines ou même des centaines de messages en file d'attente résultera en un très gros délai.

Lorsque le même message est à livrer à plusieurs usagers sur le même hôte, une seule copie du message DEVRAIT être transmise. C'est à dire que l'envoyeur SMTP devrait utiliser la séquence de commandes : RCPT, RCPT,... RCPT, DATA au lieu de la séquence : RCPT, DATA, RCPT, DATA,... RCPT, DATA. La mise en œuvre de ce dispositif efficace est vivement recommandée.

De même, l'envoyeur SMTP PEUT accepter que plusieurs transactions de messagerie sortante concurrentes réalisent une livraison en temps et en heure. Cependant, certaines limites DEVRAIENT être imposées pour empêcher l'hôte de consacrer toutes ses ressources à la messagerie.

L'utilisation de différentes adresses d'un hôte à rattachement multiple est exposée ci-dessous.

5.3.1.2 Stratégie de réception

Le receveur SMTP DEVRAIT tenter de garder une écoute active en permanence sur l'accès SMTP. Cela exige la prise en charge de plusieurs connexions TCP entrantes pour SMTP. Des limites PEUVENT être imposées.

Mise en œuvre : Lorsque le receveur SMTP reçoit un message d'une adresse d'hôte particulière, il pourrait notifier à l'envoyeur SMTP de réessayer tous les messages en attente pour cette adresse d'hôte.

5.3.2 Temporisations dans SMTP

Deux approches des temporisations sont possibles chez l'envoyeur SMTP : (a) limiter séparément le temps pour chaque commande SMTP , ou (b) limiter le temps du dialogue SMTP tout entier pour un seul message. Un envoyeur SMTP DEVRAIT utiliser l'option (a) de temporisation par commande. Les temporisations DEVRAIENT être facilement reconfigurables, de préférence sans recompiler le code SMTP.

Discussion : Les temporisations sont une caractéristique essentielle d'une mise en œuvre de SMTP. Si les temporisations sont trop longues (ou pire, s'il n'y a pas de temporisation) les échecs de communication Internet ou les erreurs de logiciel dans les programmes de receveur SMTP peuvent entraver indéfiniment les processus SMTP. Si les temporisations sont trop courtes, des ressources seront gâchées avec les tentatives qui arrivent en fin de temporisation à mi-parcours de la livraison du message.

Si l'option (b) est utilisée, la temporisation doit être très longue, par exemple, une heure, pour donner le temps de développer de très grosses listes de diffusion. La temporisation peut aussi devoir être augmentée de façon linéaire avec la taille du message, pour tenir compte du temps de transmission d'un très long message. Une durée de temporisation fixe

longue amène deux problèmes : un échec peut encore lier l'envoyeur pour une très longue durée, et de très gros messages peuvent encore arriver à dépasser le temps imparti (ce qui est un échec malencontreux !).

Si on utilise l'option (a) recommandée, un temporisateur est établi pour chaque commande SMTP et pour chaque mémoire tampon du transfert des données. Cette dernière exigence signifie que la temporisation globale est par nature proportionnelle à la taille du message.

Sur la base des nombreuses expériences des hôtes actifs de relais de messagerie, les valeurs minimum de temporisation par commande DEVRAIENT être les suivantes :

- Message 220 initial : 5 minutes
  Un processus d'envoyeur SMTP a besoin de distinguer entre un échec de connexion TCP et un retard de réception du message initial 220 d'accueil. De nombreux receveurs SMTP vont accepter une connexion TCP mais retarder la livraison du message 220 jusqu'à ce que la charge de leur système leur permette de traiter plus de messages.

- Commande MAIL : 5 minutes

- Commande RCPT : 5 minutes
  Une temporisation plus longue serait nécessaire si le traitement des listes de diffusion et des alias n'est pas différé jusqu'après l'acceptation du message.

- Initialisation des données : 2 minutes
  C'est en attendant la réponse "354 Début d'entrée" à une commande DATA.

- Bloc de données : 3 minutes
  C'est en attendant l'achèvement de chaque appel TCP SEND qui transmet un tronçon de données.

- Fin de DATA : 10 minutes.
  C'est en attendant la réponse "250 OK". Lorsque le receveur obtient le point final qui termine les données du message, il effectue normalement le traitement pour livrer le message à la boîte aux lettres de l'usager. Une temporisation parasite à ce moment serait une grosse perte de temps car le message a bien été envoyé.

Un receveur SMTP DEVRAIT avoir une temporisation d'au moins 5 minutes lorsqu'il attend la prochaine commande de la part de l'envoyeur.

5.3.3 Réception fiable de messagerie

Lorsque le receveur SMTP accepte un élément de messagerie (en envoyant un message "250 OK" en réponse à DATA) il accepte la responsabilité de la livraison ou du relais du message. Il doit prendre cette responsabilité au sérieux, c'est-à-dire, il NE DOIT PAS perdre le message pour des raisons futiles, par exemple, à cause d'une panne ultérieure de l'hôte ou à cause d'une insuffisance de ressources prévisible.

S'il y a une défaillance de livraison après l'acceptation d'un message, le receveur SMTP DOIT formuler un message de notification et l'envoyer. Cette notification DOIT être envoyée en utilisant un chemin inverse nul ("<>") dans l'enveloppe ; voir le paragraphe 3.6 de la RFC-821. Le receveur de cette notification DEVRAIT être l'adresse tirée du chemin de retour de l'enveloppe (ou de la ligne Return-Path:). Cependant, si cette adresse est nulle ("<>") le receveur SMTP NE DOIT PAS envoyer de notification. Si l'adresse est une route de source explicite, elle DEVRAIT être effacée à son bond final.

Discussion :
Par exemple, supposons qu'une notification d'erreur doive être envoyée pour un message qui est arrivé avec : "MAIL FROM:<@a,@b:user@d>". Le message de notification devrait être envoyé à : "RCPT TO:<user@d>".

Certains échecs de livraison après l'acceptation du message par SMTP seront inévitables. Par exemple, il peut être impossible au receveur SMTP de valider toutes les adresses de livraison dans la ou les commandes RCPT du fait d'une erreur de logiciel du système des domaines ou parce que la cible est une liste de diffusion (voir l'exposé précédent sur RCPT).

Pour éviter de recevoir des messages dupliqués par suite de fins de temporisations, un receveur SMTP DOIT chercher à minimiser le temps requis pour répondre au "." final qui termine un transfert de message. Voir dans la RFC-1047 [SMTP:4] un exposé sur ce problème.

5.3.4 Transmission fiable de messagerie

Pour transmettre un message, un envoyeur SMTP détermine l'adresse IP de l'hôte cible à partir de l'adresse de destination dans l'enveloppe. Précisément, il transpose la chaîne qui est à droite du signe "@" en une adresse IP. Cette transposition ou le transfert lui-même peuvent échouer à cause d'une erreur logique, auquel cas l'envoyeur SMTP remettra le message sortant dans la file d'attente pour un essai ultérieur, comme exigé au paragraphe 5.3.1.1.

Quand elle réussit, la transposition peut résulter en une liste d'adresses de livraison de remplacement plutôt qu'une seule adresse, à cause de (a) plusieurs enregistrements MX, (b) du rattachement multiple, ou des deux. Pour fournir une transmission de messagerie fiable, l'envoyeur SMTP DOIT être capable d'essayer (et réessayer) chacune des adresses de la liste dans l'ordre, jusqu'à ce que réussisse une tentative de livraison. Cependant, il PEUT aussi y avoir une limite configurable du nombre d'adresses de remplacement qui peuvent être essayées. Dans tous les cas, un hôte DEVRAIT essayer au moins deux adresses.

Les informations suivantes sont à utiliser pour ordonner les adresses d'hôte :

(1) Plusieurs enregistrements MX – ils contiennent une indication de préférence qui devrait être utilisée pour le tri. Si il y a plusieurs destinations qui ont la même préférence et qu'il n'y a pas de raisons claires pour en favoriser une (par exemple, par préférence d'adresse) l'envoyeur SMTP DEVRAIT alors en choisir une au hasard pour répartir la charge sur plusieurs points d'échange de messagerie pour une organisation spécifique ; noter que ceci est une amélioration de la procédure de [DNS:3].

(2) Hôte à rattachement multiple – L'hôte de destination (qui peut être pris dans l'enregistrement MX préféré) peut être à rattachements multiples, auquel cas le résolveur de nom de domaine va retourner une liste d'adresses IP de remplacement. Il est de la responsabilité de l'interface de résolveur de nom de domaine (voir au paragraphe 6.1.3.4 ci-dessous) d'avoir ordonné cette liste par préférence décroissante, et SMTP DOIT les essayer dans l'ordre présenté.

Discussion : Bien que la capacité d'essayer plusieurs adresses de remplacement soit exigée, il peut y avoir des circonstances dans lesquelles des installations spécifiques veulent limiter ou désactiver l'utilisation des adresses de remplacement. La question de savoir si un envoyeur devrait faire des essais en utilisant les différentes adresses d'un hôte à rattachements multiples a fait l'objet de controverses. Le principal argument pour l'utilisation de plusieurs adresses est que cela maximise la probabilité d'une livraison à temps, et bien sûr, parfois la probabilité qu'il soit livré tout court ; le contre argument est qu'il peut en résulter une consommation inutile de ressources.

Noter que l'utilisation des ressources est aussi fortement déterminée par la stratégie d'envoi exposée au paragraphe 5.3.1.

5.3.5 Prise en charge des noms de domaine

Une mise en œuvre SMTP DOIT utiliser le mécanisme défini au paragraphe 6.1 pour les transpositions entre noms de domaines et adresses IP. Cela signifie que tout SMTP Internet DOIT comporter la prise en charge du DNS Internet.

En particulier, un envoyeur SMTP DOIT prendre en charge le schéma d'enregistrement MX [SMTP:3]. Voir aussi au paragraphe 7.4 de [DNS:2] des informations sur la prise en charge des noms de domaine pour SMTP.

5.3.6 Listes de diffusion et alias

Un hôte à capacité SMTP DEVRAIT prendre en charge à la fois les formes d'alias et de liste d'expansion d'adresse pour la livraison multiple. Lorsqu'un message est livré ou transmis à chaque adresse d'une forme d'expansion de liste, l'adresse de retour dans l'enveloppe ("MAIL FROM:") DOIT être changée en l'adresse d'une personne qui administre la liste, mais l'en-tête du message DOIT être laissé inchangé ; en particulier, le champ "From" du message n'est pas affecté.

Discussion : Une facilité importante de la messagerie est le mécanisme pour la livraison multi-destinations d'un seul message, par la transformation ou "expansion" d'une adresse de pseudo boîte aux lettres dans une liste d'adresses de boîtes aux lettres de destination. Lorsqu'un message est envoyé à une telle pseudo boîte aux lettres (parfois appelée un "exploseur") des copies sont transmises ou redistribuées à chaque boîte aux lettres de la liste d'expansion. On classe une telle pseudo boîte aux lettres comme "alias" ou "liste", selon les règles d'expansion :

(a) Alias : Pour provoquer l'expansion d'un alias, le messageur de réception remplace simplement l'adresse de la pseudo boîte aux lettres qui figure dans l'enveloppe par chacune des adresses développée tour à tour ; le reste de l'enveloppe et le corps du message restent inchangés. Le message est alors livré ou transmis à chaque adresse développée.

(b) Liste : une liste de diffusion peut être dite fonctionner par "redistribution" plutôt que par "transmission". Pour provoquer l'expansion d'une liste, le messageur de réception remplace l'adresse de la pseudo boîte au lettres qui figure dans l'enveloppe par chacune des adresses développée tour à tour. L'adresse de retour dans l'enveloppe est changée de sorte que tous les messages d'erreur générés par les livraisons finales soient retournés à l'administrateur de la liste, et non à l'origine du message, qui n'a généralement aucun contrôle sur le contenu de la liste et va normalement trouver dérangeants les messages d'erreur.

5.3.7 Passerelle de messagerie

Assurer les passerelles de messagerie entre différents environnements de messagerie, c'est-à-dire, différents formats et protocoles de messagerie, est complexe et ne donne pas facilement lieu à normalisation. Voir par exemple [SMTP:5a], [SMTP:5b]. Cependant, certaines exigences générales peuvent être formulées pour la passerelle entre l'Internet et un autre environnement de messagerie.

(A) Les champs d'en-tête PEUVENT être réécrits lorsque nécessaire quand les messages sont passés à travers les frontières d'un environnement de messagerie.

Discussion : Cela peut impliquer l'interprétation de la partie locale de l'adresse de destination, comme suggéré au paragraphe 5.2.16.

Les autres systèmes de messagerie passés sur l'Internet utilisent généralement un sous-ensemble des en-têtes de la RFC-822, mais certains d'entre eux n'ont pas d'équivalent à l'enveloppe SMTP. Donc, lorsqu'un message quitte l'environnement Internet, il peut être nécessaire de ranger les informations de l'enveloppe SMTP dans l'en-tête de message. Une solution possible serait de créer de nouveaux champs d'en-tête pour porter les informations d'enveloppe (par exemple, "X-SMTP-MAIL:" et "X-SMTP-RCPT:") ; cependant, cela exigerait des changements des programmes de messagerie dans l'environnement étranger.

(B) Lors de la transmission d'un message dans ou vers l'environnement Internet, une passerelle DOIT ajouter une ligne Received:, mais elle NE DOIT PAS altérer une ligne Received: qui serait déjà dans l'en-tête.

Discussion : Cette exigence est un sous-ensemble de l'exigence générale de la ligne "Received:" du paragraphe 5.2.8 ; elle est rappelée ici pour la souligner.

Les champs Received: des messages provenant d'autres environnements peuvent ne pas se conformer exactement à la RFC822. Cependant, l'utilisation la plus importante des lignes Received: est le débogage des erreurs de messagerie, et ce débogage peut être sévèrement entravé par la bienveillance de passerelles qui essayer de "réparer" une ligne Received:.

La passerelle est vivement invitée à indiquer l'environnement et le protocole dans les clauses "via" du ou des champs Received : qu'elle fournit.

(C) Du côté de l'Internet, la passerelle DEVRAIT accepter tous les formats d'adresse valides dans les commandes SMTP et dans les en-têtes de la RFC-822, et tous les messages valides de la RFC-822. Bien qu'une passerelle doive accepter une route de source explicite de la RFC-822 (format "@...:") aussi bien dans l'en-tête de la RFC-822 que dans l'enveloppe, elle PEUT ou non agir sur la route de source ; voir les paragraphes 5.2.6 et 5.2.19.

Discussion : Il est souvent tentant de restreindre la gamme des adresses acceptées à la passerelle de messagerie pour simplifier la traduction en adresses de l'environnent distant. Cette pratique se fonde sur l'hypothèse que les utilisateurs de la messagerie exercent un contrôle sur les adresses que leurs messageurs envoient à la passerelle de messagerie. En pratique, cependant, les utilisateurs ont peu de contrôle sur les adresses qui sont finalement envoyées ; leurs messageurs ont toute liberté pour changer les adresses dans tout format légal de la RFC-822.

(D) La passerelle DOIT s'assurer que tous les champs d'en-tête d'un message qu'elle transmet dans l'Internet satisfont aux exigences de la messagerie Internet. En particulier, toutes les adresses dans les champs "From:", "To:", "Cc:", etc., doivent être transformées (si nécessaire) pour satisfaire à la syntaxe de la RFC-822, et elles doivent être efficaces et utiles pour l'envoi des réponses.

(E) L'algorithme de traduction utilisé pour convertir des messages du protocole Internet en protocoles d'autres environnements DEVRAIT essayer de s'assurer que les messages d'erreur provenant de l'environnement de

messagerie étranger sont livrés sur le chemin de retour provenant de l'enveloppe SMTP, et non à l'envoyeur figurant sur la liste du champ "From:" du message de la RFC-822.

Discussion : Les listes de diffusion Internet placent généralement l'adresse de celui qui assure la maintenance de la liste de diffusion dans l'enveloppe mais laissent intact l'en-tête original du message (avec le champ "From:" qui contient l'envoyeur d'origine). Cela donne le comportement attendu du receveur moyen : une réponse à l'en-tête est envoyée à l'expéditeur d'origine, et non à celui qui assure la maintenance de la liste de diffusion ; cependant, les erreurs sont envoyées à celui-ci (qui peut régler le problème) et non à l'envoyeur (qui ne le peut probablement pas).

(F) De même, lors de la transmission d'un message d'un autre environnement vers l'Internet, la passerelle DEVRAIT établir le chemin de retour de l'enveloppe en conformité avec une adresse de retour de message d'erreur, s'il en est, fournie par l'environnement étranger.

5.3.8 Taille maximum de message

Un logiciel de messageur DOIT être capable d'envoyer et recevoir des messages d'une longueur d'au moins 64 ko (y compris l'en-tête) et une longueur maximum bien supérieure est très souhaitable.

Discussion : Bien que SMTP ne définisse pas de taille maximum de message, de nombreux systèmes imposent des limites de mise en œuvre.

La limite minimum courante de fait de l'Internet est de 64 ko. Cependant, la messagerie électronique est utilisée pour divers objets qui créent de bien plus grands messages. Par exemple, la messagerie est souvent utilisée à la place de FTP pour transmettre des fichiers ASCII, et en particulier transmettre des documents entiers. Il en résulte que des messages peuvent atteindre 1 M octets ou même plus. On note que le présent document avec son compagnon pour la couche inférieure contient 0,5 M octets.

5.4 Résumé des exigences pour SMTP

Caractéristique	**Parag.**	**Doit**	**Devrait**	**Peut**	**Ne devrait pas**	**Ne doit pas**
Receveur SMTP :						
Met en œuvre VRFY	5.2.3	x				
Met en œuvre EXPN	5.2.3		x			
EXPN, VRFY configurable	5.2.3			x		
Met en œuvre SEND, SOML, SAML	5.2.4			x		
Vérifie le paramètre HELO	5.2.5			x		
Refuse un message avec un mauvais HELO	5.2.5					x|
Accepte la syntaxe rte de src explicite en env.	5.2.6	x				
Accepte "postmaster"	5.2.7	x				
Traite RCPT lorsque reçu (sauf listes)	5.2.7			x		
Long délai de réponses RCPT	5.2.7					x|
Ajout de la ligne Received:	5.2.8	x				
Ligne Received: comporte le domaine littéral	5.2.8		x			
Change la ligne Received: précédente	5.2.8					x|
Passe les infos de Return-Path (livrais/passer. finale)	5.2.8	x				
Accepte chemin inverse vide	5.2.9	x				
Envoie seulement les codes de réponse officiels	5.2.10		x			
Envoie le texte de la RFC-821 quand approprié	5.2.10		x			
Supprime "." pour la transparence	5.2.11	x				
Accepte et reconnaît le domaine littéral autonome	5.2.17	x				
Message d'erreur sur un message d'erreur	5.3.1					x
Garde l'écoute sur l'accès SMTP	5.3.1.2		x			
Met une limite à la réception en concurrence	5.3.1.2			x		
Attend au moins 5 m pour prochaine cmd de l'envoyeur	5.3.2		x			
Échec de livraison après "250 OK"	5.3.3					x
Envoi de msg de notification d'erreur après acceptation	5.3.3	x				
Envoi avec chemin de retour nul	5.3.3	x				
Envoi avec chemin de retour de l'enveloppe	5.3.3		x			
Envoi à l'adresse nulle	5.3.3					x

Élimine la route de source explicite	5.3.3		x				
Minimise le délai d'acceptation (RFC-1047)	5.3.3	x					
Envoyeur SMTP :							
Noms de domaine canonisés dans MAIL, RCPT	5.2.2	x					
Met en œuvre SEND, SOML, SAML	5.2.4			x			
Envoie nom d'hôte principal valide dans HELO	5.2.5	x					
Envoie route de source explicite dans RCPT TO:	5.2.6				x		
Utilise seulement le code de rép pour déterminer action	5.2.10	x					
Utilise seult le premier chiffre du code de rép qd poss.	5.2.10		x				
Ajout de "." pour la transparence	5.2.11	x					
Réessaye les messages après échec logiciel	5.3.1.1	x					
Délai avant nouvel essai	5.3.1.1	x					
Paramètres de nouvel essai configurables	5.3.1.1	x					
Essai une fois pour chaque hôte de dest en file d'att.	5.3.1.1		x				
Plusieurs RCPT pour les mêmes DATA	5.3.1.1		x				
Accepte plusieurs transactions concurrentes	5.3.1.1			x			
Met une limite à la concurrence	5.3.1.1		x				
Temporisations sur toutes les activités	5.3.1	x					
Temporisations par commande	5.3.2		x				
Temporisation facilement reconfigurable	5.3.2		x				
Délais recommandés	5.3.2		x				
Essaye les adresses de remplacement dans l'ordre	5.3.4	x					
Limite configurable sur les essais de remplacement	5.3.4			x			
Essaye au moins deux adresses de remplacement	5.3.4		x				
Partage de charge entre MX égaux	5.3.4		x				
Utilise le système des noms de domaine	5.3.5	x					
Prend en charge les enregistrements MX	5.3.5	x					
Utilise enregistr WKS dans traitement MX	5.2.12				x		
Transmission de messagerie :							
Altère les champs d'en-tête existants	5.2.6				x		
Met en œuvre la fonction de relais : RFC821/§ 3.6	5.2.6			x			
Sinon, livre au domaine de droite	5.2.6		x				
Interprète la 'partie locale' de l'adresse	5.2.16						x
Listes de diffusion et alias							
Prend en charge les deux	5.3.6		x				
Rapporte l'erreur de liste de diff à l'admin. local	5.3.6	x					
Passerelles de messagerie :							
Incorpore route de messag. étrangère dans partie locale	5.2.16			x			
Réécrit les champs d'en-tête quand nécessaire	5.3.7			x			
Ajoute la ligne Received:	5.3.7	x					
Change la ligne Received: existante	5.3.7						x|
Accepte RFC-822 sur le côté Internet	5.3.7		x				
Agit sur route de source explicite de RFC-822	5.3.7			x			
Envoie seulement RFC-822 valide sur côté Internet	5.3.7	x					
Livre les msgs d'erreur à l'adresse de l'enveloppe	5.3.7		x				
Règle chemin de retour d'env d'après adr de retour d'err	5.3.7		x				
Agent d'utilisateur -- RFC-822							
Permet à l'usager d'entrer l'adresse <route>	5.2.6				x		
Accepte le champ Content Type de la RFC-1049	5.2.13			x			
Utilise l'année à quatre chiffres	5.2.14		x				
Génère des zones horaires numériques	5.2.14		x				
Accepte toutes les zones horaires	5.2.14	x					
Utilise les zones horaires non numériques de RFC-822	5.2.14	x					
Omet phrase devant route-addr	5.2.15			x			
Accepte et analyse les domaines littéraux en déc. à point	5.2.17	x					
Accepte tous les formats d'adresse de la RFC-822	5.2.18	x					
Génère un format d'adresse RFC-822 invalide	5.2.18						x
Noms de domaine pleinement qualifiés dans l'en-tête	5.2.18	x					
Crée une route de source explicite dans l'en-tête	5.2.19				x		
Accepte une route de source explicite dans l'en-tête	5.2.19	x					

Envoi/réception de messages d'au moins 64 koctet	5.3.8	x

6 Services de soutien

6.1 Traduction de nom de domaine

6.1.1 Introduction

Chaque hôte DOIT mettre en œuvre un résolveur pour le système des noms de domaine (DNS) et il DOIT mettre en œuvre un mécanisme utilisant ce résolveur DNS pour convertir les noms d'hôte en adresses IP et vice-versa [DNS:1], [DNS:2].

En plus du DNS, un hôte PEUT aussi mettre en œuvre un mécanisme de traduction de nom d'hôte qui cherche sur un tableau d'hôtes Internet locaux. Voir au paragraphe 6.1.3.8 des informations supplémentaires sur cette option.

Discussion : La traduction de nom d'hôte Internet était à l'origine effectuée en cherchant sur des copies locales du tableau de tous les hôtes. Ce tableau est devenu trop gros à mettre à jour et à distribuer à temps et trop gros pour tenir dans de nombreux hôtes, c'est pourquoi le DNS a été inventé.

Le DNS crée une base de données distribuée utilisée principalement pour la traduction entre nom d'hôte et adresse d'hôte. La mise en œuvre du logiciel de DNS est exigée. Le DNS comporte deux parties logiquement distinctes : les serveurs de noms et les résolveurs (bien que les mises en œuvre combinent souvent ces deux parties logiques dans l'intérêt de l'efficacité) [DNS:2].

Les serveurs de noms de domaines mémorisent les données d'autorisation sur certaines sections de la base de données et répondent aux interrogations sur les données. Les résolveurs de domaine interrogent les serveurs de noms de domaine sur les données au nom des processus d'utilisateur. Chaque hôte a donc besoin d'un résolveur DNS ; certaines machines d'hôte vont aussi avoir besoin de faire fonctionner des serveurs de noms de domaine. Comme aucun serveur de noms ne dispose des informations complètes, il est en général nécessaire d'obtenir des informations de la part de plus d'un serveur de noms pour résoudre une interrogation.

6.1.2 Survol du protocole

Un développeur doit étudier soigneusement les références [DNS:1] et [DNS:2]. Elles fournissent une description approfondie de la théorie, du protocole, et de la mise en œuvre du système des noms de domaine, et capitalisent plusieurs années d'expérience.

6.1.2.1 Enregistrements de ressource avec TTL de zéro : RFC-1035 paragraphe 3.2.1

Tous les serveurs de noms et résolveurs DNS DOIVENT traiter correctement les enregistrements de ressource (RR) avec un TTL de zéro : retourner le RR au client mais ne pas le mettre en anté-mémoire.

Discussion : Les valeurs de TTL de zéro sont interprétées comme signifiant que le RR ne peut être utilisé que pour la transaction en cours, et ne devrait pas être mis en anté-mémoire ; elles sont utiles pour des données extrêmement volatiles.

6.1.2.2 Valeurs de QCLASS : RFC-1035 paragraphe 3.2.5

Une interrogation avec "QCLASS=*" NE DEVRAIT PAS être utilisée à moins que celui qui fait l'interrogation ne cherche des données à partir de plus d'une classe. En particulier, si l'interrogateur est seulement intéressé par des types de données Internet, QCLASS=IN DOIT être utilisé.

6.1.2.3 Champs non utilisés : RFC-1035 paragraphe 4.1.1

Les champs non utilisés dans un message d'interrogation ou de réponse DOIVENT être à zéro.

6.1.2.4 Compression : RFC-1035 paragraphe 4.1.4

Les serveurs de nom DOIVENT utiliser la compression dans les réponses.

Discussion : La compression est essentielle pour éviter les débordements de datagrammes UDP ; voir en 6.1.3.2.

6.1.2.5 Mauvais usage des informations de configuration : RFC-1035 paragraphe 6.1.2

Les serveurs de noms récurrents et les résolveurs de plein exercice ont généralement des informations de configuration qui contiennent des indications sur la localisation des serveurs de noms racine ou local. Une mise en œuvre NE DOIT PAS comporter une de ces indications dans une réponse.

Discussion : De nombreux développeurs ont trouvé pratique de mémoriser ces indications comme si elles étaient des données en anté-mémoire, mais certains négligent de s'assurer que ces "données d'anté-mémoire" ne sont pas incluses dans les réponses. Cela a causé de sérieux problèmes dans l'Internet lorsque l'indication est obsolète ou incorrecte.

6.1.3 Questions spécifiques

6.1.3.1 Mise en œuvre de résolveur

Un résolveur de nom DEVRAIT être capable de multiplexer des demandes concurrentes si l'hôte accepte des processus en concurrence.

Dans la mise en œuvre d'un résolveur de DNS, on PEUT choisir entre deux modèles différents : un résolveur de plein exercice, ou un résolveur de bout.

(A) Résolveur de plein exercice
Un résolveur de plein exercice est une mise en œuvre complète du service de résolution, et est capable de traiter les défaillances de communication, les défaillances des serveurs individuels de noms, la localisation du serveur de noms approprié pour un nom donné, etc. Il doit satisfaire aux exigences suivantes :
- o Le résolveur DOIT mettre en œuvre une fonction de mise en anté-mémoire locale pour éviter des accès distants répétés pour des demandes identiques, et DOIT avoir une temporisation des informations de l'anté-mémoire.
- o Le résolveur DEVRAIT être configurable avec les informations de démarrage pointées sur plusieurs serveurs de noms racines et plusieurs serveurs de noms pour le domaine local. Cela assure que le résolveur sera capable d'accéder à tout l'espace de noms dans les cas normaux, et sera capable d'accéder aux informations de domaine local si le réseau devait être déconnecté du reste de l'Internet.

(B) Résolveur de bout
Un "résolveur de bout" s'appuie sur les services d'un serveur de noms récurrent sur le réseau connecté ou un réseau "voisin". Ce schéma permet à l'hôte de passer la charge de la fonction de résolveur au serveur de noms sur un autre hôte. Ce modèle est souvent essentiel pour les hôtes à moindres capacités, tels que les micro-ordinateurs, et est aussi recommandé lorsque l'hôte est une station parmi d'autres sur un réseau, parce qu'il permet à toutes les stations de travail de partager l'antémémoire du serveur de noms récurrent et donc de réduire le nombre de demandes de domaine exportées par le réseau local.

Au minimum, le résolveur de bout DOIT être capable de diriger ses demandes sur les serveurs de noms récurrents redondants. Noter que les serveurs de noms récurrents ont la permission de restreindre les sources de demandes qu'ils vont honorer, de sorte que l'administrateur de l'hôte doit vérifier que le service sera fourni. Les résolveurs de bout PEUVENT mettre en œuvre la mise en antémémoire si ils le veulent, mais si il en est ainsi, ils DOIVENT mettre une temporisation sur les informations de l'antémémoire.

6.1.3.2 Protocoles de transport

Les résolveurs DNS et les serveurs récurrents DOIVENT prendre en charge UDP, et DEVRAIT prendre en charge TCP, pour l'envoi des interrogations (transfert hors zone). Précisément, un résolveur DNS ou un serveur qui envoie une interrogation de transfert hors zone DOIT envoyer d'abord une interrogation UDP. Si la section Answer de la réponse est tronquée et si le demandeur prend en charge TCP, il DEVRAIT essayer à nouveau l'interrogation en utilisant TCP.

Les serveurs DNS DOIVENT être capables de servir les interrogations UDP et DEVRAIENT être capables de servir les interrogations TCP. Un serveur de noms PEUT limiter les ressources qu'il dédie aux interrogations TCP, mais il NE DEVRAIT PAS refuser de servir une interrogation TCP seulement parce qu'il aurait réussi avec UDP.

Les réponses tronquées NE DOIVENT PAS être sauvegardées (en anté-mémoire) et utilisées ultérieurement d'une façon qui oublie le fait qu'elles sont tronquées.

Discussion : UDP est préféré à TCP pour les interrogations parce que les interrogations UDP ont beaucoup moins de redondance, à la fois en compte de paquets et en état de connexion. L'utilisation de UDP est essentielle pour les serveurs

lourdement chargés, et en particulier les serveurs racine. UDP offre aussi une robustesse supplémentaire, car un résolveur peut essayer plusieurs interrogations UDP sur différents serveurs pour le coût d'une seule interrogation TCP.

Il est possible qu'une réponse DNS soit tronquée, bien que cela soit très rare dans le DNS Internet actuel. En pratique, la troncature ne peut être prévue, car elle dépend des données. Cette dépendance est aussi liée au nombre de RR dans la réponse, à la taille de chaque RR, et aux économies d'espace réalisées par l'algorithme de compression de noms. Approximativement, la troncature dans les listes NS et MX ne devrait pas se produire pour les réponses contenant 15 RR ou moins.

Savoir s'il est possible d'utiliser une réponse tronquée dépend de l'application. Un messageur ne doit pas utiliser une réponse MX tronquée, car cela peut conduire à des messages en boucle.

Une pratique responsable peut rendre UDP suffisant dans la vaste majorité des cas. Les serveurs de noms doivent utiliser la compression dans les réponses. Les résolveurs doivent différencier la troncature de la section additionnelle d'une réponse (qui ne perd que des informations supplémentaires) de la troncature de la section Réponse (qui pour les enregistrements MX rend la réponse inutilisable par les messageurs). Les administrateurs de base de données devraient faire une liste comportant seulement un nombre raisonnable de noms principaux dans les listes de serveurs de noms, de MX de remplacement, etc.

Cependant, il est aussi clair que certains nouveaux types d'enregistrements DNS qui seront définis à l'avenir contiendront des informations excédant la limite de 512 octets qui s'applique à UDP, et requièreront donc TCP. Et donc, les résolveurs et serveurs de noms devraient mettre en œuvre les services TCP comme solution de repli à l'UDP d'aujourd'hui, sachant qu'ils auront besoin du service TCP à l'avenir.

Par accord mutuel, les serveurs de noms et résolveurs PEUVENT s'arranger pour utiliser TCP pour tout le trafic entre eux. TCP DOIT être utilisé pour les transferts de zone.

Un serveur DNS DOIT avoir une capacité interne suffisante pour pouvoir continuer à traiter des interrogations UDP tout en attendant une réponse ou en effectuant un transfert de zone sur une connexion TCP ouverte [DNS:2].

Un serveur PEUT prendre en charge une interrogation UDP qui est livrée en utilisant une adresse IP de diffusion ou de diffusion groupée. Cependant, le bit de récurrence souhaitée NE DOIT PAS être établi dans une interrogation qui est en diffusion groupée, et DOIT être ignoré par les serveurs de noms qui reçoivent des interrogations via une adresse de diffusion ou de diffusion groupée. Un hôte qui envoie des interrogations de DNS en diffusion ou diffusion groupée DEVRAIT ne les envoyer que comme preuves occasionnelles, en mettant en anté-mémoire la ou les adresses IP qu'il obtient de la part de la ou des réponses, de sorte qu'il puisse normalement envoyer des interrogations en envoi individuel.

Discussion : La diffusion IP ou (tout particulièrement) la diffusion groupée peut fournir un moyen de localiser des serveurs de noms voisins sans savoir leurs adresses IP à l'avance. Cependant, la diffusion générale d'interrogations récurrentes peut résulter en une charge excessive et inutile à la fois pour les réseaux et les serveurs.

6.1.3.3 Utilisation efficace des ressources

Les exigences suivantes pour les serveurs et résolveurs sont très importantes pour la santé de l'Internet global, en particulier lorsque des services de DNS sont invoqués de façon répétée par des serveurs automatiques de haut niveau, tels que les messageurs.

(1) Le résolveur DOIT mettre en œuvre des contrôles de retransmission pour s'assurer qu'il ne gaspille pas la bande passante de communication, et DOIT imposer des limites finies aux ressources consommées pour répondre à une seule demande. Voir aux pages 43-44 de [DNS:2] les recommandations spécifiques.

(2) Après qu'une interrogation a été retransmise plusieurs fois sans réponse, une mise en œuvre DOIT abandonner et retourner une erreur légère à l'application.

(3) Tous les serveurs de noms et résolveurs DNS DEVRAIENT mettre en anté-mémoire les défaillances temporaires, avec une période de temporisation de l'ordre de quelques minutes.

Discussion : Cela empêche les applications qui réessayent immédiatement après une défaillance légère (en violation du paragraphe 2.2 du présent document) de générer un trafic DNS excessif.

(4) Tous les serveurs de noms et résolveurs DNS DEVRAIENT mettre en anté-mémoire les réponses négatives qui indiquent que le nom spécifié, ou les données du type spécifié, n'existent pas, comme décrit dans [DNS:2].

(5) Lorsqu'un serveur ou résolveur DNS réessaye une interrogation UDP, l'intervalle d'essai DEVRAIT être contraint par un algorithme de retard exponentiel, et DEVRAIT aussi avoir des limites supérieures et inférieures.

Mise en œuvre : On devrait utiliser un délai d'aller retour et une variance (si disponible) mesurés pour calculer un intervalle initial de retransmission. Si ces informations ne sont pas disponibles, une valeur par défaut qui ne devrait pas être inférieure à 5 s devrait être utilisée. Les mises en œuvre peuvent limiter l'intervalle de retransmission, mais cette limite doit excéder deux fois la durée de vie du segment Internet maximum plus le délai de service du serveur de noms.

(6) Lorsqu'un résolveur ou serveur reçoit un "Source éteinte" pour une interrogation qu'il a produite, il DEVRAIT prendre des mesures pour réduire le débit d'interrogations sur ce serveur à brève échéance. Un serveur PEUT ignorer un Source éteinte qu'il reçoit suite à l'envoi d'un datagramme de réponse.

Mise en œuvre :
Une action recommandée pour réduire le débit est d'envoyer la prochaine tentative d'interrogation à un serveur de remplacement, si il en est un disponible. Une autre est de reculer l'intervalle d'essai pour ce même serveur.

6.1.3.4 Hôtes à rattachements multiples

Lorsque la fonction de traduction de nom d'hôte en adresse rencontre un hôte avec plusieurs adresses, elle DEVRAIT ranger ou trier les adresses en utilisant sa connaissance du ou des numéros de réseau immédiatement connectés et toutes les autres informations applicables de performance ou d'historique.

Discussion : Les différentes adresses d'un hôte à rattachements multiples impliquent généralement différents chemins Internet, et certains chemins peuvent être préférables à d'autres pour leurs performances, fiabilité, ou restrictions administratives. Il n'y a pas de façon générale pour que le système des domaines détermine le meilleur chemin. Une approche recommandée est de fonder cette décision sur les informations de configuration locales établies par l'administrateur de système.

Mise en œuvre :
Le schéma suivant a été utilisé avec succès :
(a) Incorporer dans les données de configuration de l'hôte une liste de réseaux préférés, qui est simplement une liste des réseaux par ordre de préférence. Cette liste peut être vide si il n'y a aucune préférence.
(b) Lorsqu'un nom d'hôte est transposé dans une liste d'adresses IP, ces adresses devraient être triées par numéro de réseau, dans le même ordre que les réseaux correspondants de la liste des réseaux préférés. Les adresses IP dont les réseaux n'apparaissent pas dans la liste des réseaux préférés devraient être placées à la fin de la liste.

6.1.3.5 Extensibilité

Un logiciel de DNS DOIT prendre en charge tous les formats bien connus indépendants des classes [DNS:2], et DEVRAIT être écrit de façon à minimiser l'impact de l'introduction de nouveaux types bien connus et l'expérimentation locale de types non normalisés.

Discussion :
Les types et classes de données utilisés par le DNS sont extensibles, et donc de nouveaux types seront ajoutés et de vieux types seront supprimés ou redéfinis. L'introduction de nouveaux types de données ne devrait dépendre que des règles de compression des noms de domaines au sein des messages DNS, et de traduction entre formats imprimables (c'est-à-dire, fichier maître) et internes des enregistrements de ressource (RR, *Resource Record*).

La compression s'appuie sur la connaissance du format des données au sein d'un RR particulier. Et donc la compression ne doit être utilisée que pour les contenus de RR bien connus, indépendants de la classe, et ne doit jamais être utilisée pour des RR spécifiques d'une classe ou des types de RR qui ne sont pas bien connus. Le nom du propriétaire d'un RR est toujours susceptible d'être compressé.

Un serveur de noms peut acquérir, via un transfert de zone, des RR que le serveur ne sait pas convertir en format imprimable. Un résolveur peut recevoir des informations similaires suite à des interrogations. Pour un fonctionnement approprié, ces données doivent être préservées, et donc, cela implique que le logiciel DNS ne puisse pas utiliser de formats textuels pour la mémorisation interne.

Le DNS définit de façon très générale une syntaxe de nom de domaine – une chaîne d'étiquettes contenant chacune jusqu'à 63 octets de 8 bits, séparés par des points, et avec un maximum total de 255 octets. Des applications particulières du DNS peuvent imposer des contraintes syntaxiques supplémentaires aux noms de domaines qu'elles utilisent, bien que le développement du DNS ait conduit à ce que certaines applications permettent des noms plus généraux. En particulier, le paragraphe 2.1 du présent document libéralise légèrement la syntaxe du nom d'hôte Internet légal défini dans la RFC-952 [DNS:4].

6.1.3.6 Statut des types de RR

Les serveurs de noms DOIVENT être capables de charger tous les types de RR excepté MD et MF à partir des fichiers de configuration. Les types MD et MF sont obsolètes et NE DOIVENT PAS être mis en œuvre ; en particulier, les serveurs de noms NE DOIVENT PAS charger ces types à partir des fichiers de configuration.

Discussion : Les types de RR MB, MG, MR, NULL, MINFO et RP sont considérés comme expérimentaux, et les applications qui utilisent le DNS ne peuvent pas s'attendre à ce que ces types de RR soient acceptés par la plupart des domaines. De plus, ces types seront redéfinis.

Les types de TT TXT et WKS n'ont pas été largement utilisés par les sites Internet ; il en résulte qu'une application ne peut pas s'appuyer sur l'existence d'un RR TXT ou WKS dans la plupart des domaines.

6.1.3.7 Robustesse

Un logiciel de DNS peut avoir besoin de fonctionner dans des environnements où les serveurs racines sur les autres serveurs sont indisponibles du fait de problèmes de connexité du réseau ou autres. Dans une telle situation, les serveurs de noms et résolveurs du DNS DOIVENT continuer de fournir le service pour la partie joignable de l'espace de noms, tout en indiquant une défaillance temporaire pour le reste.

Discussion : Bien que le DNS soit destiné à être principalement utilisé dans l'Internet connecté, il devrait être possible d'utiliser le système dans des réseaux qui ne sont pas connectés à l'Internet. Et donc les mises en œuvre ne doivent pas dépendre de l'accès aux serveurs racines pour pouvoir servir les noms locaux.

6.1.3.8 Tableau des hôtes locaux

Discussion : Un hôte peut utiliser un tableau des hôtes locaux comme sauvegarde ou supplément au DNS. Cela soulève la question de savoir qui prend le pas, du DNS ou du tableau des hôtes ; l'approche la plus souple ferait de cette question une option de configuration.

Normalement, le contenu d'un tel tableau supplémentaire des hôtes sera déterminé localement par le site. Cependant, un tableau des hôtes Internet disponible au public est entretenu par le Centre des informations réseau du DDN (DDN NIC, *DDN Network Information Center*) dans un format documenté dans [DNS:4]. Ce tableau peut être restitué à partir du DDN NIC avec un protocole décrit dans [DNS:5]. On doit noter que ce tableau ne contient qu'une petite fraction de tous les hôtes de l'Internet. Les hôtes qui utilisent ce protocole pour restituer le tableau des hôtes du DDN NIC devraient utiliser la commande VERSION pour vérifier si le tableau a été changé avant de demander le téléchargement de la totalité du tableau avec la commande ALL. L'identificateur VERSION devrait être traité comme une chaîne arbitraire et vérifié seulement en égalité ; aucune séquence numérique ne peut être supposée.

Le tableau des hôtes DDN NIC comporte des informations administratives qui ne sont pas utiles au fonctionnement des hôtes et ne sont donc pas actuellement incluses dans la base de données du DNS ; par exemple, les entrées de réseau et de routeurs. Cependant, beaucoup de ces informations supplémentaires seront ajoutées à l'avenir au DNS. À l'inverse, le DNS fournit des services essentiels (en particulier, les enregistrements MX) qui ne sont pas disponibles à partir du tableau des hôtes DDN NIC.

6.1.4 Interface d'utilisateur DNS

6.1.4.1 Administration du DNS

Le présent document s'occupe des questions de conception et de mise en œuvre dans les logiciel des hôtes, mais pas des questions administratives ou opérationnelles. Cependant, les questions administratives sont d'une importance particulière dans le DNS, car des erreurs dans des segments particuliers de cette grande base de données distribuée peuvent causer des performances erronées ou de mauvaise qualité sur de nombreux sites. Ces questions sont discutées dans [DNS:6] et [DNS:7].

6.1.4.2 Interface d'utilisateur DNS

Les hôtes DOIVENT fournir une interface au DNS pour tous les programmes d'application qui fonctionnent sur l'hôte. Cette interface va normalement diriger les demandes sur un processus système pour effectuer la fonction de résolveur [DNS:1, 6.1:2].

Au minimum, l'interface de base DOIT prendre en charge une demande pour toutes les informations d'un type spécifique et la classe associée à un nom spécifique, et elle DOIT retourner toutes les informations demandées, un code d'erreur lourde, ou une indication d'erreur légère. Lorsqu'il n'y a pas d'erreur, l'interface de base retourne les informations complètes de réponse sans modification, suppression, ou mise en ordre, de sorte que l'interface de base n'aura pas besoin d'être changée pour s'accommoder de nouveaux types de données.

Discussion :
L'indication d'erreur légère est une partie essentielle de l'interface, car il peut n'être pas toujours possible d'accéder à des informations particulières provenant du DNS ; voir au paragraphe 6.1.3.3.

Un hôte PEUT fournir d'autres interfaces du DNS conçues pour des fonctions particulières, transformant les données brutes de domaine en formats mieux adaptés à ces fonctions. En particulier, un hôte DOIT fournir une interface DNS pour faciliter la traduction entre les adresses et les noms d'hôtes.

6.1.4.3 Facilités d'abréviation de l'interface

Les interfaces d'utilisateur PEUVENT fournir une méthode pour que les usagers entrent des abréviations pour les noms d'usage courant. Bien que la définition de telles méthodes sorte du domaine d'application de la spécification du DNS, certaines règles sont nécessaires pour garantir que ces méthodes permettent l'accès à l'espace de noms du DNS tout entier et empêcher une utilisation excessive des ressources de l'Internet.

Si une méthode d'abréviation est fournie, alors :
(a) Il DOIT y avoir une convention pour noter qu'un nom est déjà complet, de sorte que la ou les méthodes d'abréviation soient supprimées. Un point en queue est la méthode habituelle.
(b) L'expansion d'une abréviation DOIT être faite exactement une fois, et elle DOIT être faite dans le contexte dans lequel le nom a été entré.

Discussion : Par exemple, si une abréviation est utilisée dans un programme de messagerie pour une destination, l'abréviation devrait être développée sous la forme d'un nom de domaine complet et mémorisée dans le message mis en file d'attente avec une indication comme quoi il est déjà complet. Autrement, l'abréviation pourrait être développée par une liste de recherche de système de messagerie, pas celle de l'utilisateur, ou un nom pourrait s'augmenter du fait de tentatives répétées de canonisation interagissant avec des caractères génériques.

Les deux méthodes d'abréviation les plus courantes sont :
(1) Alias de niveau interface
Les alias de niveau interface sont mis en œuvre conceptuellement comme une liste de paires d'alias/domaines. La liste peut être par usager ou par hôte, et des listes séparées peuvent être associées à différentes fonctions, par exemple une liste pour la traduction de nom d'hôte en adresse, et une liste différente pour les domaines de messagerie. Lorsque l'usager entre un nom, l'interface essaye de faire correspondre le nom avec le composant alias d'une entrée de liste, et si on peut trouver une entrée correspondante, le nom est remplacé par le nom de domaine trouvé dans la paire.

Noter que les alias de niveau interface et les CNAME sont des mécanismes complètement distincts ; les alias de niveau interface sont une affaire locale alors que les CNAME sont un mécanisme d'alias au niveau de l'Internet qui est une partie requise de toute mise en œuvre du DNS.

(2) Listes de recherche
Une liste de recherche est un concept mis en œuvre comme une liste ordonnée de noms de domaines. Lorsque l'usager entre un nom, les noms de domaine de la liste de recherche sont utilisés comme suffixes pour le nom fourni par l'usager, un par un, jusqu'à ce qu'un nom de domaine ayant les données associées désirées soit trouvé, ou que la liste de recherche soit épuisée. Les listes de recherche contiennent souvent le nom du domaine parent de l'hôte local ou d'autres domaines ancêtres. Les listes de recherche sont souvent par usager ou par processus.

Il DEVRAIT être possible à un administrateur de désactiver un dispositif de liste de recherche du DNS. Un refus administratif peut être prévu dans certains cas, pour empêcher un abus du DNS.

Il existe un danger que le mécanisme de liste de recherche génère des interrogations excessives aux serveurs racines lorsqu'on vérifie si une entrée d'usager est un nom de domaine complet, lorsqu'il manque le point final qui marque qu'il est complet. Un mécanisme de liste de recherche DOIT avoir une des dispositions suivantes, et DEVRAIT avoir les deux, pour empêcher cela :

(a) Le résolveur local/serveur de noms peut mettre en œuvre la mise en anté-mémoire des réponses négatives (voir au paragraphe 6.1.3.3).

(b) Le développeur de liste de recherche peut exiger deux points intérieurs ou plus dans un nom de domaine généré avant qu'il essaye d'utiliser le nom dans une interrogation de serveurs de domaines non locaux, tels que la racine.

Discussion :
L'intention de cette exigence est d'éviter des délais excessifs pour l'usager lorsque la liste de recherche est testée, et plus important, d'empêcher un trafic excessif avec le serveur racine et autres serveurs de haut niveau. Par exemple, si l'usager a fourni un nom "X" et si la liste de recherche contenait la racine comme composant, une interrogation devrait consulter un serveur racine avant que la solution de remplacement suivante de la liste de recherche puisse être essayée. La charge résultante vue par les serveurs racines et les passerelles proches de la racine pourrait être multipliée par le nombre des hôtes de l'Internet.

La solution de remplacement de mise en anté-mémoire négative limite les effets à la première fois qu'un nom est utilisé. La règle du point intérieur est plus simple à mettre en œuvre mais peut empêcher une utilisation facile de certains noms de niveau supérieur.

6.1.5 Résumé des exigences du système de noms de domaines

Caractéristique	Parag.	Doit	Devrait	Peut	Ne devrait pas	Ne doit pas
Questions générales						
Met en œuvre la conversion DNS nom/adresse	6.1.1	x				
Met en œuvre la conversion DNS adresse/nom	6.1.1	x				
Accepte les conversions utilisant le tableau des hôtes	6.1.1			x		
Traite correctement un RR avec TTL de zéro	6.1.2.1	x				
Utilise QCLASS=* inutilement	6.1.2.2		x			
Utilise QCLASS=IN pour la classe Internet	6.1.2.2	x				
Les champs inutilisés sont à zéro	6.1.2.3	x				
Utilise la compression dans les réponses	6.1.2.4	x				
Inclut les info de config dans les réponses	6.1.2.5					x|
Accepte tous les types bien connus, indép. de la classe	6.1.3.5	x				
Liste de types facilement développée	6.1.3.5		x			
Charge tous les types de RR (excepté MD et MF)	6.1.3.6	x				
Charge le type MD ou MF	6.1.3.6					x|
Fonctionne lorsque serveurs racine, etc. indisponibles	6.1.3.7	x				
Problèmes du résolveur :						
Le résolveur accepte plusieurs demandes concurrentes	6.1.3.1		x			
Résolveur de plein exercice :	6.1.3.1			x		
Mise en anté-mémoire locale	6.1.3.1	x				
Temporisation des infos en anté-mémoire locale	6.1.3.1	x				
Configurable avec infos de démarrage	6.1.3.1		x			
Résolveur de bout :	6.1.3.1			x		
Utilise des serveurs de nom récurrents redondants	6.1.3.1	x				
Mise en anté-mémoire locale	6.1.3.1			x		
Temporisation des infos en anté-mémoire locale	6.1.3.1	x				
Accepte des hôtes distants à rattachements multiples :						
Trie les adresses multiples par liste de préférence	6.1.3.4		x			
Protocoles de transport :						
Accepte les interrogations UDP	6.1.3.2	x				
Accepte les interrogations TCP	6.1.3.2		x			
Envoie l'interrogation d'abord avec UDP (note 1)	6.1.3.2	x				
Essaye TCP si réponse UDP tronquée	6.1.3.2		x			
Serveur de nom limite les ressources d'interro TCP	6.1.3.2			x		
Punit les interrogations TCP inutiles	6.1.3.2				x	

Utilise données tronquées comme si ne l'étaient pas	6.1.3.2					x|
Accord privé pour n'utiliser que TCP	6.1.3.2			x		
Utilise TCP pour transferts de zone	6.1.3.2	x				
Usage de TCP ne bloque pas interrogations UDP	6.1.3.2	x				
Accepte interrogations en diffusion ou diff. groupée	6.1.3.2			x		
Bit RD établi dans l'interrogation	6.1.3.2					x|
Bit RD ignoré par serveur est diffusion/diff. groupée	6.1.3.2	x				
Envoyé seult comme preuve occas pour les adresses	6.1.3.2		x			
Utilisation des ressources :						
Contrôles de transmission, selon [DNS:2]	6.1.3.3	x				
Limites finies par demande	6.1.3.3	x				
Échec après essai => erreur légère	6.1.3.3	x				
Mise en anté-mémoire des échecs temporaires	6.1.3.3		x			
Mise en anté-mémoire des réponses négatives	6.1.3.3		x			
Les réessais utilisent le retard exponentiel	6.1.3.3		x			
Limites supérieure et inférieure	6.1.3.3		x			
Le client traite Source éteinte	6.1.3.3		x			
Le serveur ignore Source éteinte	6.1.3.3			x		
Interface d'utilisateur :						
Tous les programmes ont accès à l'interface DNS	6.1.4.2	x				
Capable demander toutes les infos pour un nom donné	6.1.4.2	x				
Retourne les infos complètes ou erreur	6.1.4.2	x				
Interfaces particulières	6.1.4.2			x		
Traduction nom<->adresse	6.1.4.2	x				
Facilités d'abréviation :	6.1.4.3			x		
Convention pour les noms complets	6.1.4.3	x				
Conversion exactement une fois	6.1.4.3	x				
Conversion dans contexte approprié	6.1.4.3	x				
Liste de recherche :	6.1.4.3			x		
Administrateur peut désactiver	6.1.4.3		x			
Empêche les interrogations excessives de racine	6.1.4.3	x				
Les deux méthodes	6.1.4.3		x			

note 1. Sauf s'il y a un accord privé entre un résolveur particulier et un serveur particulier.

6.2 Initialisation de l'hôte

6.2.1 Introduction

Cette section discute de l'initialisation du logiciel d'hôte à travers un réseau connecté, ou plus généralement à travers un chemin Internet. Ceci est nécessairement pour un hôte sans disque, et peut facultativement être utilisé pour un hôte avec des pilotes de disques. Pour un hôte sans disque, le processus d'initialisation est appelé "amorçage réseau" et est commandé par un programme d'amorçage localisé dans une mémoire en lecture seule d'amorçage.

Pour initialiser un hôte sans disque à travers le réseau, il y a deux phases distinctes :

(1) Configurer la couche IP :
Les machines sans disque n'ont souvent pas de mémorisation permanente dans laquelle stocker les informations de configuration du réseau, de sorte que des informations de configuration suffisantes doivent être obtenues de façon dynamique pour prendre en charge la phase de chargement qui suit. Ces informations doivent inclure au moins les adresses IP de l'hôte et du serveur d'amorçage. Pour prendre en charge l'amorçage à travers un routeur, le gabarit d'adresse et une liste des routeurs par défaut sont aussi nécessaires.

(2) Charger le code du système d'hôte.
Durant la phase de chargement, un protocole de transfert de fichiers approprié est utilisé pour copier le code système à travers le réseau à partir du serveur d'amorçage.

Un hôte avec un disque peut effectuer la première étape de configuration dynamique. C'est important pour les micro-ordinateurs, dont les disquettes permettent aux informations de configuration d'être frauduleusement dupliquées sur plus d'un hôte. Aussi, l'installation de nouveaux hôtes est beaucoup plus simple si ils obtiennent automatiquement leurs

informations de configuration à partir d'un serveur central, épargnant le temps de l'administrateur et diminuant ainsi la probabilité d'erreurs.

6.2.2 Exigences

6.2.2.1 Configuration dynamique

Un certain nombre de dispositions du protocole ont été prises pour la configuration dynamique.

- Messages de demande/réponse d'informations ICMP
  Cette paire de messages obsolète était destinée à permettre à un hôte de trouver le numéro du réseau sur lequel il est. Malheureusement, il n'était utile que si l'hôte connaissait déjà la partie numéro d'hôte de son adresse IP, information que les hôtes qui requièrent une configuration dynamique ont rarement.

- Protocole de résolution inverse d'adresse (RARP, *Reverse Address Resolution Protocol*) [BOOT:4]
  RARP est un protocole de couche de liaison pour un support de diffusion qui permet à un hôte de trouver son adresse IP connaissant son adresse de couche de liaison. Malheureusement, RARP ne fonctionne pas à travers les routeurs IP et exige donc un serveur RARP sur chaque réseau. De plus, RARP ne fournit aucune autre information de configuration.

- Messages de demande/réponse de gabarit d'adresse ICMP
  Ces messages ICMP permettent à un hôte d'apprendre le gabarit d'adresse pour une interface réseau particulière.

- Protocole BOOTP [BOOT:2]
  Ce protocole permet à un hôte de déterminer les adresses IP de l'hôte local et du serveur d'amorçage, le nom d'un fichier d'amorçage approprié, et facultativement le gabarit d'adresse et la liste des routeurs par défaut. Pour localiser un serveur BOOTP, l'hôte diffuse une demande BOOTP en utilisant UDP. Des extensions de routeurs ad hoc ont été utilisées pour transmettre la diffusion BOOTP à travers les routeurs, et à l'avenir la facilité de diffusion groupée IP fournira un mécanisme normalisé pour répondre à ce besoin.

  L'approche suggérée de configuration dynamique est d'utiliser le protocole BOOTP avec les extensions définies dans la RFC 1084 "Extensions d'informations de fabricant BOOTP" [BOOT:3]. La RFC 1084 définit quelques extensions générales importantes (non spécifiques du fabricant). En particulier, ces extensions permettent que le gabarit d'adresse soit fourni dans BOOTP ; il est RECOMMANDÉ que le gabarit d'adresse soit fourni de cette manière.

Discussion :
Historiquement, la mise en sous-réseau a été définie longtemps après IP, et donc un mécanisme distinct (les messages ICMP de gabarit d'adresse) a été conçu pour fournir le gabarit d'adresse à un hôte. Cependant, gabarit d'adresse IP et adresse IP correspondante forment une paire conceptuelle, et pour la simplicité du fonctionnement ils devraient être définis en même temps et par les mêmes mécanismes, que ce soit un fichier de configuration ou un mécanisme dynamique comme BOOTP.

Noter que BOOTP n'est pas suffisamment général pour spécifier les configurations de tous les interfaces d'un hôte à rattachements multiples. Un hôte à rattachements multiples doit soit utiliser BOOTP séparément pour chaque interface, soit configurer une interface utilisant BOOTP pour effectuer le chargement, et effectuer ultérieurement l'initialisation complète à partir d'un fichier.

Les informations de configuration de la couche d'application sont supposées obtenues à partir des fichiers après le chargement du code système.

6.2.2.2 Phase de chargement

Une approche suggérée pour la phase de chargement est d'utiliser TFTP [BOOT:1] entre les adresses IP établies par BOOTP.

TFTP NE DEVRAIT PAS être utilisé vers une adresse en diffusion, pour les raisons expliquées au paragraphe 4.2.3.4.

6.3 Gestion distante

6.3.1 Introduction

La communauté de l'Internet a fait récemment des efforts considérables pour le développement des protocoles de gestion de réseau. Il en résulte une approche dans deux directions [MGT:1], [MGT:6] : le protocole simple de gestion de réseau (SNMP, *Simple Network Management Protocol*) [MGT:4] et le protocole commun d'informations de gestion sur TCP (CMOT, *Common Management Information Protocol over TCP*) [MGT:5].

Pour être géré en utilisant SNMP ou CMOT, un hôte aura besoin de mettre en œuvre un agent de gestion approprié. Un hôte Internet DEVRAIT inclure un agent pour SNMP ou pour CMOT.

SNMP et CMOT fonctionnent tous deux sur une base d'informations de gestion (MIB, *Management Information Base*) qui définit une collection de valeurs de gestion. En lisant et en établissant ces valeurs, une application distante peut interroger l'état du système géré et le changer.

Une MIB normalisée [MGT:3] a été définie pour être utilisée par les deux protocoles de gestion, avec des types de données définis par la structure des informations de gestion (SMI, *Structure of Management Information*) définie dans [MGT:2]. Des variables de MIB supplémentaires peuvent être introduites sous les sous-arborescences "enterprises" et "experimental" de l'espace de dénomination de la MIB [MGT:2].

Tout module de protocole de l'hôte DEVRAIT mettre en œuvre les variables de MIB pertinentes. Un hôte DEVRAIT mettre en œuvre les variables de MIB comme défini dans la norme de MIB la plus récente, et PEUT mettre en œuvre d'autres variables de MIB quand c'est approprié et utile.

6.3.2 Survol du protocole

La MIB est destinée à couvrir à la fois les hôtes et les routeurs, bien qu'il puisse y avoir des différences de détail dans les applications de MIB entre les deux cas. Ce paragraphe contient l'interprétation appropriée de la MIB pour les hôtes. Il est vraisemblable que les prochaines versions de MIB incluront plus d'entrées pour la gestion d'hôte.

Un hôte géré doit mettre en œuvre les groupes suivants de définitions d'objet de MIB : Système, Interfaces, Traduction d'adresse, IP, ICMP, TCP, et UDP.

Les interprétations spécifiques suivantes s'appliquent aux hôtes :

- ipInHdrErrors
  Noter que l'erreur "temps de vie dépassé" ne peut survenir dans un hôte que lorsqu'il transmet un datagramme à acheminement de source.

- ipOutNoRoutes
  Cet objet compte les datagrammes éliminés parce qu'aucune route n'a pu être trouvée. Cela peut arriver à un hôte si tous les routeurs par défaut de la configuration de l'hôte sont hors service.

- ipFragOKs, ipFragFails, ipFragCreates
  Un hôte qui ne met pas en œuvre la fragmentation intentionnelle (voir au paragraphe "Fragmentation" de [INTRO:1]) DOIT retourner la valeur zéro pour ces trois objets.

- icmpOutRedirects
  Pour un hôte, cet objet DOIT toujours être à zéro, car les hôtes n'envoient pas de Redirection.

- icmpOutAddrMaskReps
  Pour un hôte, cet objet DOIT toujours être à zéro, sauf si l'hôte est une source d'autorisation des informations de gabarit d'adresse.

- ipAddrTable
  Pour un hôte, l'objet "Tableau d'adresse IP" est effectivement un tableau des interfaces logiques.

- ipRoutingTable
  Pour un hôte, l'objet "Tableau d'acheminement IP" est effectivement une combinaison de l'anté-mémoire d'acheminement de l'hôte et du tableau des routes statiques décrit au paragraphe "Acheminement des datagrammes sortants" de [INTRO:1].

Au sein de chaque ipRouteEntry, ipRouteMetric1...4 n'aura normalement aucune signification pour un hôte et DEVRAIT toujours être -1, alors que ipRouteType devra normalement avoir la valeur "distant".

Si les destinations sur le réseau connecté n'apparaissent pas dans l'anté-mémoire d'acheminement (voir au paragraphe "Acheminement des datagrammes sortants" de [INTRO:1], il n'y aura aucune entrée avec ipRouteType "direct".

Discussion : La MIB actuelle ne comporte pas de Type-de-Service dans une ipRouteEntry, mais une future révision prévue fera cet ajout.

On prévoit aussi que la MIB soit étendue pour permettre la gestion à distance des applications (par exemple, la capacité de reconfigurer partiellement les systèmes de messagerie). Les applications de service réseau telles que les systèmes de messagerie devraient donc être écrites avec les accroches pour la gestion à distance.

6.3.3 Résumé des exigences de gestion

Caractéristique	Paragraphe	Doit	Devrait	Peut
Accepte l'agent SNMP ou CMOT	6.3.1		x	
Met en œuvre les objets spécifiés dans la MIB standard	6.3.1		x	

7 Références

La présente section donne la liste des principales références avec lesquelles tout développeur doit être étroitement familier. Elle donne aussi la liste de quelques références secondaires dont la lecture est suggérée.

Références introductives :

[INTRO:1] "Exigences pour les hôtes Internet – Couches de communication", Groupe de travail Exigences d'hôtes de l'IETF, R. Braden, éditeur, RFC-1122, octobre 1989.

[INTRO:2] "DDN Protocol Handbook", NIC-50004, NIC-50005, NIC-50006, (trois volumes), SRI International, décembre 1985.

[INTRO:3] "Protocoles officiels de l'Internet", J. Reynolds et J. Postel, RFC-1011, mai 1987. Ce document est republié périodiquement avec un nouveau numéro de RFC ; on doit utiliser la dernière version.

[INTRO:4] "Informations pour commander les documents de protocole de l'Internet", O. Jacobsen et J. Postel, RFC-980, mars 1986.

[INTRO:5] "Numéros alloués," J. Reynolds et J. Postel, RFC-1010, mai 1987.
Ce document est republié périodiquement avec un nouveau numéro de RFC ; on doit utiliser la dernière version.

Références Telnet :

[TELNET:1] "Spécification du protocole Telnet", J. Postel et J. Reynolds, RFC-854, mai 1983.

[TELNET:2] "Spécification des options Telnet", J. Postel et J. Reynolds, RFC-855, mai 1983.

[TELNET:3] "Transmission binaire Telnet", J. Postel et J. Reynolds, RFC-856, mai 1983.

[TELNET:4] "Option Echo Telnet", J. Postel et J. Reynolds, RFC-857, mai 1983.

[TELNET:5] "Option Suppress Go Ahead Telnet", J. Postel et J. Reynolds, RFC-858, mai 1983.

[TELNET:6] "Option Status Telnet", J. Postel et J. Reynolds, RFC- 859, mai 1983.

[TELNET:7] "Option Timing Mark Telnet", J. Postel et J. Reynolds, RFC-860, mai 1983.

[TELNET:8] "Liste étendue d'options Telnet" J. Postel et J. Reynolds, RFC-861, mai 1983.

[TELNET:9] "Option Telnet End-Of-Record" J. Postel, RFC-855, décembre 1983.

[TELNET:10] "Option Telnet Type de terminal" J. VanBokkelen, RFC-1091, février 1989. Remplace la RFC-930.

[TELNET:11] "Option Telnet taille de fenêtre" D. Waitzman, RFC-1073, octobre 1988.

[TELNET:12] "Option Telnet mode de ligne" D. Borman, RFC-1116, août 1989.

[TELNET:13] "OPtion Telnet vitesse du terminal" C. Hedrick, RFC-1079, décembre 1988.

[TELNET:14] "Option Telnet commande de flux à distance" C. Hedrick, RFC- 1080, novembre 1988.

Références Telnet secondaires :

[TELNET:15] "Protocole Telnet" MIL-STD-1782, U.S. Department of Defense, mai 1984.
Ce document est destiné à décrire le même protocole que la RFC-854. En cas de conflit, la RFC-854 prend le pas, et le présent document prend le pas sur les deux.

[TELNET:16] "Protocole SUPDUP" M. Crispin, RFC-734, octobre 1977.

[TELNET:17] "Option Telnet SUPDUP" M. Crispin, RFC-736, octobre 1977.

[TELNET:18] "Option Terminal d'entrée de données" J. Day, RFC-732, juin 1977.

[TELNET:19] "Option Telnet Terminal d'entrée de données – Mise en œuvre DODIIS," A. Yasuda et T. Thompson, RFC-1043, février 1988.

Références pour FTP :

[FTP:1] "Protocol de transfert de données" J. Postel et J. Reynolds, RFC- 959, octobre 1985.

[FTP:2] "Normes de format de fichiers de document" J. Postel, RFC-678, décembre1974.

[FTP:3] "Protocole de transfert de fichiers" MIL-STD-1780, U.S. Department of Defense, mai 1984.
Ce document se fonde sur une version antérieure de la spécification FTP (RFC-765) et est obsolète.

Références pour TFTP :

[TFTP:1] "Protocole TFTP, révision 2," K. Sollins, RFC-783, juin 1981.

Références pour la messagerie :

[SMTP:1] "Protocole simple de transfert de messagerie," J. Postel, RFC-821, août 1982.

[SMTP:2] "Norme pour le format des messages de texte de ARPA Internet" D. Crocker, RFC-822, août 1982.
Ce document a rendu obsolète une spécification antérieure, la RFC-733.

[SMTP:3] "Acheminement de messagerie et système des domaines" C. Partridge, RFC-974, janvier 1986.
Cette RFC décrit l'utilisation des enregistrements MX, extension obligatoire au processus de livraison de messages.

[SMTP:4] "Messages dupliqués et SMTP," C. Partridge, RFC-1047, février 1988.

[SMTP:5a] "Transposition entre X.400 et la RFC 822," S. Kille, RFC-987, juin 1986.

[SMTP:5b] "Addendum à la RFC-987," S. Kille, RFC-1026, septembre 1987.

Les deux RFC précédentes définissent une proposition de norme pour le routage de messagerie entre l'Internet et les environnements X.400.

[SMTP:6] "Protocole simple de transfert de messagerie" MIL-STD-1781, U.S. Department of Defense, mai 1984.
Cette spécification est destinée à décrire le même protocole que la RFC-821. Cependant, MIL-STD-1781 est incomplète ; en particulier, elle n'inclut pas les enregistrements MX [SMTP:3].

[SMTP:7] "Champ Content-Type pour les messages Internet" M. Sirbu, RFC-1049, mars 1988.

Références du système des noms de domaines :

[DNS:1] "Noms de domaines - Concepts et facilités," P. Mockapetris, RFC-1034, novembre 1987.
Ce document et le suivant rendent obsolète les RFC-882, RFC-883 et la RFC-973.

[DNS:2] "Noms de domaines – Mise en œuvre et spécification," RFC-1035, P. Mockapetris, novembre 1987.

[DNS:3] "Routage de messagerie et système des domaines" C. Partridge, RFC-974, janvier 1986.

[DNS:4] "Spécification du tableau d'hôte Internet du DoD" K. Harrenstein, RFC-952, M. Stahl, E. Feinler, octobre 1985.

Références secondaires du DNS :

[DNS:5] "Serveur de nom d'hôte" K. Harrenstein, M. Stahl, E. Feinler, RFC-953, octobre 1985.

[DNS:6] "Guide des administrateurs de domaine" M. Stahl, RFC-1032, novembre 1987.

[DNS:7] "Guide du fonctionnement des administrateurs de domaine" M. Lottor, RFC-1033, novembre 1987.

[DNS:8] "Manuel du système des noms de domaine" Vol. 4 du Manuel des protocoles Internet, NIC 50007, SRI Network Information Center, août 1989.

Références d'initialisation du système :

[BOOT:1] "Chargement de Bootstrap à l'aide de TFTP," R. Finlayson, RFC-906, juin 1984.

[BOOT:2] "Protocole Bootstrap (BOOTP)," W. Croft et J. Gilmore, RFC-951, septembre 1985.

[BOOT:3] "Extensions Informations de fabricant BOOTP" J. Reynolds, RFC-1084, décembre 1988. Note : cette RFC révise et rend obsolète la RFC-1048.

[BOOT:4] "Protocole de résolution d'adresse inverse" R. Finlayson, T. Mann, J. Mogul et M. Theimer, RFC-903, juin 1984.

Références de gestion :

[MGT:1] "Recommandations de l'IAB pour le développement des normes de gestion de réseau de l'Internet" V. Cerf, RFC-1052, avril 1988.

[MGT:2] "Structure et identification des informations de gestion pour les internets fondés sur TCP/IP" M. Rose et K. McCloghrie, RFC-1065, août 1988.

[MGT:3] "Base d'informations de gestion pour la gestion de réseau des internets fondés sur TCP/IP" M. Rose et K. McCloghrie, RFC-1066, août 1988.

[MGT:4] "Protocole simple de gestion de réseau" J. Case, M. Fedor, M. Schoffstall et C. Davin, RFC-1098, avril 1989.

[MGT:5] "Services et protocoles courants d'informations de gestion sur TCP/IP," U. Warrier et L. Besaw, RFC-1095, avril 1989.

[MGT:6] "Rapport du second groupe ad hoc de révision de la gestion de réseau" V. Cerf, RFC-1109, août 1989.

Considérations pour la sécurité

Il y a de nombreux problèmes de sécurité dans les programmes d'application et de soutien de logiciel d'hôte, mais un exposé complet sort du domaine d'application de la présente RFC. Les questions en rapport avec la sécurité sont mentionnées dans les sections concernant TFTP (paragraphes 4.2.1, 4.2.3.4, 4.2.3.5) les commandes SMTP VRFY et EXPN (paragraphe 5.2.3) la commande SMTP HELO (5.2.5) et la commande SMTP DATA (paragraphe 5.2.8).

Adresse de l'auteur

Robert Braden
USC/Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
téléphone : (213) 822 1511
mél : Braden@ISI.EDU