Groupe de travail Réseau
Request for Comments : 3261
Rendue obsolète : 2543
Catégorie : Norme
juin 2002

Traduction Claude Brière de L'Isle
septembre 2006

J. Rosenberg ; dynamicsoft
H. Schulzrinne ; Columbia U.
G. Camarillo ; Ericsson
A. Johnston ; WorldCom
J. Peterson ; Neustar
R. Sparks ; dynamicsoft
M. Handley ; ICIR
E. Schooler ;AT&T

SIP : Protocole d'initialisation de session

Statut du présent mémoire
La présente RFC spécifie un protocole de normalisation pour la communauté Internet et appelle à des discussions et suggestions pour son amélioration. Prière de se reporter à l'édition en cours des "Internet Official Protocol Standards" (*normes officielles du protocole Internet*) (STD 1) pour connaître l'état de la normalisation et le statut du présent protocole. La distribution du présent mémo n'est pas soumise à restriction.

Résumé
Le présent document décrit le Protocole d'initialisation de session (SIP, *Session Initiation Protocol*), protocole de commande de couche d'application (signalisation) pour la création, modification, et terminaison de sessions avec un ou plusieurs participants. Ces sessions incluent les appels téléphoniques sur l'Internet, la distribution multimédia, et les conférences multimédia.

Les invites SIP utilisées pour créer des sessions portent des descriptions de session qui permettent aux participants de se mettre d'accord sur un ensemble de types de supports compatibles. SIP utilise des éléments appelés serveurs mandataires pour aider à acheminer les demandes jusqu'à la localisation actuelle de l'utilisateur, authentifier et autoriser les utilisateurs pour les services, mettre en œuvre les politiques d'acheminement d'appel du fournisseur, et fournir les caractéristiques aux utilisateurs. SIP fournit aussi une fonction d'enregistrement qui permet aux utilisateurs de télécharger leur configuration actuelle pour qu'elle soit utilisée par les serveurs mandataires. SIP passe sur plusieurs protocoles de transport différents.

Table des matières

SIP : Protocole d'initialisation de session....1
1 Introduction....4
2 Généralités sur les fonctions de SIP....4
3 Terminologie....5
4 Généralités sur le fonctionnement....5
5 Structure du protocole....9
6 Définitions....10
7 Messages de SIP....14
7.1 Demandes....14
7.2 Réponses....15
7.3 Champs d'en-tête....15
7.4 Corps....17
7.5 Tramage des messages SIP....18
8 Comportement général d'agent utilisateur....18
8.1 Comportement d'UAC....19
8.2 Comportement de l'UAS....25
8.3 Serveurs de redirection....28
9 Annulation d'une demande....29
9.1 Comportement du client....30
9.2 Comportement du serveur....30
10 Enregistrements....31
10.1 Généralités....31
10.2 Construction de la demande REGISTER....31
11 Interrogation des capacités....36

11.1 Construction d'une demande OPTIONS....37
11.2 Traitement des demandes OPTIONS....37
12 Dialogues....38
12.1 Création d'un dialogue....39
12.2 Demandes au sein d'un dialogue....40
12.3 Terminaison d'un dialogue....43
13 Initialisation d'une session....43
13.1 Généralités....43
13.2 Traitement de l'UAC....43
13.3 Traitement de l'UAS....46
14 Modification d'une session existante....48
14.1 Comportement de l'UAC....48
14.2 Comportement de l'UAS....49
15 Terminaison d'une session....50
15.1 Terminaison d'une session avec une demande BYE....50
16 Comportement du mandataire....51
16.2 Mandataire à états pleins....51
16.3 Validation de la demande....52
16.4 Traitement des informations d'acheminement....53
16.5 Détermination des cibles de la demande....54
16.6 Transmission de la demande....55
16.7 Traitement des réponses....59
16.8 Traitement du temporisateur C....62
16.9 Traitement des erreurs de transport....63
16.10 Traitement de CANCEL....63
16.11 Mandataire sans état....63
16.12 Résumé du traitement d'acheminement de mandataire....64
17 Transactions....67
17.1 Transaction client....68
17.2 Transaction de serveur....72
18 Transport....75
18.1 Clients....75
18.2 Serveurs....77
18.3 Tramage....78
18.4 Traitement d'erreurs....78
19 Composants de message communs....78
19.1 Indicateurs de ressource uniformes SIP et SIPS....78
19.2 Étiquettes d'option....84
19.3 Étiquettes....84
20 Champs d'en-tête....85
20.1 Accept....86
20.2 Accept-Encoding....87
20.3 Accept-Language....87
20.4 Alert-Info....87
20.5 Allow....87
20.6 Authentication-Info....88
20.7 Authorization....88
20.8 Call-ID....88
20.9 Call-Info....88
20.10 Contact....89
20.11 Content-Disposition....89
20.12 Content-Encoding....90
20.13 Langage-du-contenu....90
20.14 Content-Length....90
20.15 Content-Type....91
20.16 CSeq....91
20.17 Date....91
20.18 Error-Info....91
20.19 Expires....92
20.20 From....92
20.21 In-Reply-To....92
20.22 Max-Forwards....92
20.23 Min-Expires....93
20.24 MIME-Version....93

20.25 Organization....93
20.26 Priority....93
20.27 Proxy-Authenticate....93
20.28 Proxy-Authorization....94
20.29 Proxy-Require....94
20.30 Record-Route....94
20.31 Reply-To....94
20.32 Require....94
20.33 Retry-After....95
20.34 Route....95
20.35 Server....95
20.36 Subject....95
20.37 Supported....95
20.38 Timestamp....95
20.39 To....96
20.40 Unsupported....96
20.41 User-Agent....96
20.42 Via....96
20.43 Warning....97
20.44 WWW-Authenticate....98
21 Codes de réponse....98
21.1 1xx Provisoire....98
21.2 2xx Réussite....99
21.3 3xx Réorientation....99
21.4 4xx Défaillance de la demande....100
22 Usage de l'authentification HTTP....106
22.1 Cadre de travail....106
22.2 Authentification d'utilisateur à utilisateur....107
22.3 Authentification de mandataire à utilisateur....108
22.4 Schéma d'authentification Digest....109
23 S/MIME....110
23.1 Certificats S/MIME....110
23.2 Échange de clé S/MIME....111
23.3 Sécurisation des corps MIME....112
23.4 Confidentialité et intégrité des en-têtes SIP en utilisant S/MIME : Tunnelage de SIP....113
24 Exemples....117
24.1 Enregistrement....117
24.2 Établissement de session....117
25 BNF augmenté pour le protocole SIP....120
25.1 Règles de base....120
26 Considérations sur la sécurité : Modèle de menace et recommandations d'utilisation de la sécurité....128
26.1 Modèles d'attaques et de menaces....129
26.2 Mécanismes de sécurité....131
26.3 Mise en œuvre des mécanismes de sécurité....133
26.4 Limitations....137
26.5 Confidentialité....139
27 Considérations relatives à l'IANA....140
27.1 Étiquettes d'option....140
27.2 Codes d'avertissement....140
27.3 Noms de champs d'en-tête....140
27.4 Codes de méthode et de réponse....141
27.5 Type MIME "message/sip"....141
27.6 Enregistrement de paramètres de nouveau contenu-disposition....141
28 Changements par rapport à la RFC 2543....142
28.1 Changements fonctionnels majeurs....142
28.2 Changements fonctionnels mineurs....144
29 Références normatives....144
30 Références informatives....145
Tableau des valeurs des temporisateurs....145
Remerciements....146

1 Introduction

De nombreuses applications de l'Internet demandent la création et la gestion d'une session, où une session est considérée comme un échange de données entre une association de participants. La mise en œuvre de ces applications est compliquée par les pratiques des participants : les usagers peuvent se déplacer entre les points d'extrémité, ils peuvent être désignés par plusieurs noms, et ils peuvent communiquer sur plusieurs supports différents - parfois simultanément. De nombreux protocoles ont été rédigés qui portent diverses formes de données de session multimédia en temps réel, telles que la voix, la vidéo, ou des messages textuels. Le Protocole d'initialisation de session (SIP) travaille de concert avec des protocoles en permettant aux points de terminaison Internet (appelés agents utilisateurs) de se découvrir l'un l'autre et de se mettre d'accord sur une caractérisation d'une session qu'ils aimeraient partager. Pour localiser les participants potentiels à une session, et pour d'autres fonctions, SIP permet la création d'une infrastructure d'hôtes de réseau (appelés serveurs mandataires) auxquels les agents utilisateurs peuvent envoyer leurs enregistrements, invitations aux sessions, et autres requêtes. SIP est un outil souple et général pour créer, modifier, et terminer les sessions, qui travaille indépendamment des protocoles de transport sous jacents et sans être dépendant du type de session établie.

2 Généralités sur les fonctions de SIP

SIP est un protocole de commande de couche application qui peut établir, modifier et terminer des sessions multimédia (conférences) telles que des communications téléphoniques par l'Internet. SIP peut aussi inviter des participants à des sessions déjà existantes, telles que des conférences en multidiffusion. Des supports peuvent être ajoutés (et retirés) à une session existante. SIP prend en charge de façon transparente la transposition de nom et les services de redirection, ce qui sert de support à la mobilité personnelle [27] – les utilisateurs peuvent conserver une identification unique vue de l'extérieur, indépendamment de leur localisation dans le réseau.

SIP prend en charge cinq facettes de l'établissement et de la terminaison de communications multimédia :
- Localisation de l'utilisateur : détermination du système terminal à utiliser pour la communication ;
- Disponibilité de l'utilisateur : détermination de la volonté de l'appelé à s'engager dans une communication ;
- Capacités de l'utilisateur : détermination du support et des paramètres de support à utiliser ;
- Etablissement de session : "sonnerie", établissement des paramètres de session à la fois chez l'appelant et l'appelé ;
- Gestion de session : y compris le transfert et la terminaison des sessions, la modification des paramètres de session, et l'invocation des services.

SIP n'est pas un système de communications intégré verticalement. SIP est plutôt un composant qui peut être utilisé avec d'autres protocoles de l'IETF pour construire une architecture multimédia complète. Normalement, ces architectures vont inclure des protocoles tels que le protocole de transport en temps réel (RTP) (RFC 1889 [28]) pour le transport en temps réel de données et la fourniture d'informations en retour sur la qualité de service, le protocole à défilement continu en temps réel (RTSP, *Real-Time streaming protocol*) (RFC 2326 [29]) pour le contrôle de livraison de supports à défilement continu, le protocole de commande de passerelle de support (MEGACO, *Media Gateway Control Protocol*) (RFC 3015 [30]) pour le contrôle des passerelles vers le réseau téléphonique public commuté (RTPC), et le protocole de description de session (SDP, *Session Description Protocol*) (RFC 2327 [1]) pour la description des sessions multimédia. Donc, SIP devrait être utilisé en conjonction avec les autres protocoles afin de fournir des services complets aux utilisateurs. Cependant, la fonction et le fonctionnement de base de SIP ne dépendent d'aucun de ces protocoles.

SIP ne fournit pas de services. Plutôt, SIP fournit des primitives qui peuvent être utilisées pour mettre en œuvre différents services. Par exemple, SIP peut localiser un utilisateur et livrer un objet opaque à l'endroit où il se trouve. Si cette primitive est utilisée pour délivrer une description de session écrite, par exemple, en SDP, les points de terminaison peuvent se mettre d'accord sur les paramètres d'une session. Si la même primitive est utilisée pour livrer une photo de l'appelant aussi bien que la description de session, un service d'"ID d'appelant" peut facilement être mis en œuvre. Comme le montre cet exemple, une seule primitive est normalement utilisée pour fournir plusieurs services différents.

SIP n'offre pas de services de contrôle de conférence du genre de la commande de salle ou des votes et n'a aucune exigence sur la façon dont une conférence doit être gérée. SIP peut être utilisé pour initialiser une session qui utilise un autre protocole de contrôle de conférence. Comme les messages SIP et les sessions qu'ils établissent peuvent passer à travers des réseaux entièrement différents, SIP ne peut pas fournir, et ne fournit pas, de capacités de réservation de ressources de réseau d'aucune sorte.

La nature des services fournis rend la sécurité particulièrement importante. A cette fin, SIP fournit une série de services de sécurité, qui comporte la prévention du déni de service, l'authentification (à la fois d'usager à usager et de

mandataire à usager), la protection de l'intégrité, et de services de chiffrement et de confidentialité.

SIP travaille aussi bien avec IPv4 qu'avec IPv6.

3 Terminologie

Dans le présent document, les mots clé "DOIT", "NE DOIT PAS", "EXIGE", "DEVRAIT" NE DEVRAIT PAS", "RECOMMANDE", "NON RECOMMANDE", "PEUT", et "FACULTATIF" doivent être interprétés comme décrit dans le BCP 14, RFC 2119 [2] et indiquent les niveaux d'exigence pour les mises en œuvres conformes à SIP.

4 Généralités sur le fonctionnement

La présente section introduit les opérations de base de SIP en utilisant des exemples simples. Cette section est didactique par nature et ne contient aucune déclaration normative.

Le premier exemple montre les fonctions de base de SIP : localisation d'un point de terminaison, signal d'un désir de communiquer, négociation des paramètres de session pour établir la session, et suppression de la session une fois établie.

La Figure 1 montre un exemple typique d'échange de messages SIP entre deux utilisateurs, Alice et Bob. (Chaque message est étiqueté avec la lettre "F" et un numéro pour référence par le texte.) Dans cet exemple, Alice utilise une application SIP sur son micro ordinateur (auquel on se réfère comme téléphone logiciel) pour appeler Bob sur son téléphone SIP sur l'Internet. Deux serveurs mandataires SIP sont également montrés, qui agissent au nom d'Alice et de Bob pour faciliter l'établissement de la session. Cet arrangement typique est souvent désigné sous le nom "trapézoïde de SIP" comme le montre la forme géométrique de lignes pointillées dans la Figure 1.

Alice "appelle" Bob en utilisant son identité SIP, un type d'identifiant de ressource universel (URI, *Uniform Resource Identifier*) appelé un URI SIP. Les URI SIP sont définis au paragraphe 19.1. Ils ont une forme similaire à celle d'une adresse de messagerie électronique, contenant normalement un nom d'utilisateur et un nom d'hôte. Dans ce cas, c'est sip:bob@biloxi.com, où biloxi.com est le domaine du fournisseur de service SIP de Bob. Alice a un URI SIP qui est sip:alice@atlanta.com. Alice pourrait avoir tapé l'URI de Bob ou peut être cliqué sur un lien hypertexte ou sur une entrée dans un carnet d'adresses. SIP fournit aussi des URI sécurisés, appelés URI SIPS. Ce serait par exemple sips:bob@biloxi.com. Un appel fait à un URI SIPS garantit qu'un transport sécurisé, chiffré (à savoir TLS) est utilisé pour porter tous les messages SIP de l'appelant au domaine de l'appelé. A partir de là, la demande est envoyée en toute sécurité à l'appelé, mais avec des mécanismes de sécurité qui dépendent de la politique du domaine de l'appelé.

SIP se fonde sur un modèle de transaction de demande/réponse du style HTTP. Chaque transaction consiste en une demande qui implique une méthode ou fonction particulière sur le serveur, et au moins une réponse. Dans cet exemple, la transaction débute avec l'envoi par le téléphone logiciel d'Alice d'une demande INVITE adressée à l'URI SIP de Bob. INVITE est un exemple de méthode SIP qui spécifie l'action que le demandeur (Alice) veut que fasse le serveur (Bob). La demande INVITE contient un certain nombre de champs d'en-tête. Les champs d'en-tête sont des attributs désignés qui fournissent des informations supplémentaires sur un message. Ceux qui sont présents dans un INVITE incluent un identifiant unique pour l'appel, l'adresse de destination, l'adresse d'Alice, et des informations sur le type de session qu'Alice souhaite établir avec Bob. L'INVITE (message F1 dans la Figure 1) pourrait ressembler à :

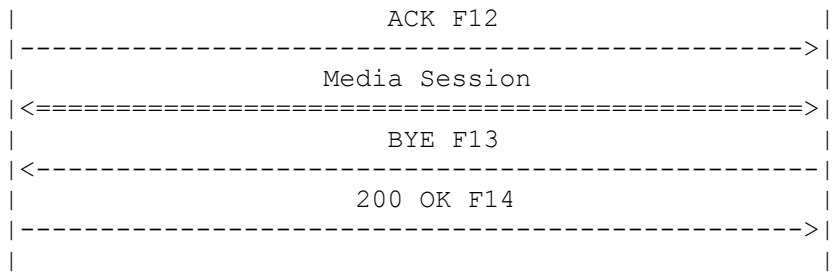


Figure 1 : Exemple d'établissement de session SIP avec le trapézoïde SIP

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
(Le SDP d'Alice n'est pas montré)

La première ligne du message en texte codé contient le nom de la méthode (INVITE). Les lignes qui suivent sont une liste de champs d'en-tête. Cet exemple contient l'ensemble minimum requis. Les champs d'en-tête sont brièvement décrits ci-dessous :

Via contient l'adresse (pc33.atlanta.com) à laquelle Alice s'attend à recevoir des réponses à cette demande. Il contient aussi un paramètre de branche qui identifie cette transaction.

To *(à)* contient un nom d'affichage (Bob) et un URI SIP ou SIPS (sip:bob@biloxi.com) sur lequel la demande a été dirigée à l'origine. Les noms d'affichage sont décrit dans la RFC 2822 [3].

From *(de)* contient aussi un nom d'affichage (Alice) et un URI SIP ou SIPS (sip:alice@atlanta.com) qui indique la personne à l'origine de la demande. Ce champs d'en-tête a aussi un paramètre d'étiquette contenant une chaîne aléatoire (1928301774) qui a été ajoutée à l'URI par le téléphone logiciel. Elle sert pour des besoins d'identification.

Call-ID *(identifiant d'appel)* contient un identifiant unique au monde pour cet appel, généré par la combinaison d'une chaîne aléatoire et du nom d'hôte ou de l'adresse IP du téléphone logiciel. La combinaison de l'étiquette To, de l'étiquette From, et du Call-ID *(identifiant d'appel)* définit de façon complète une relation SIP d'homologue à homologue entre Alice et Bob qu'on appelle un dialogue.

CSeq ou Command Sequence *(séquence de commande)* contient un entier et un nom de méthode. Le nombre CSeq est incrémenté pour chaque nouvelle demande d'un dialogue et c'est un numéro de séquence traditionnel.

Contact contient un URI SIP ou SIPS qui représente une route directe pour contacter Alice, habituellement composé d'un nom d'utilisateur sur un nom de domaine pleinement qualifié (FQDN, *fully qualified domain name*). Alors qu'un FQDN est préféré, de nombreux systèmes d'extrémité n'ont pas de noms de domaine enregistrés, de sorte que les adresses IP sont permises. Alors que le champs d'en-tête Via indique les autres éléments sur le lieu où envoyer la réponse, le champs d'en-tête Contact indique d'autres éléments sur où envoyer des demandes futures.

Max-Forwards *(retransmissions maximales)* sert à limiter le nombre de sauts que peut faire une demande sur son chemin vers sa destination. Il consiste en un entier qui est décrémenté d'un à chaque saut.

Content-Type *(type de contenu)* contient une description du corps de message (non montré).

Content-Length *(longueur du contenu)* contient un compte d'octets du corps de message.

L'ensemble complet des champs d'en-tête SIP est défini à la Section 20.

Les détails de la session, comme le type de support, codec, ou débit d'échantillonnage, ne sont pas décrits à l'aide de SIP. Le corps d'une message SIP contient plutôt une description de la session, codée dans un autre format de protocole.

Un de ces formats est le protocole de description de session (SDP, *Session Description Protocol*) (RFC 2327 [1]). Ce message SDP (non montré dans l'exemple) est porté par le message SIP d'une façon analogue à celle dont un document joint est porté par un message électronique, ou dont une page web est portée par un message HTTP.

Comme le téléphone logiciel ne connaît pas la localisation de Bob ou le serveur SIP dans le domaine biloxi.com, le téléphone logiciel envoie l'INVITE au serveur SIP qui dessert le domaine d'Alice, atlanta.com. L'adresse du serveur SIP atlanta.com pourrait avoir été configurée dans le téléphone logiciel d'Alice, ou elle aurait pu être découverte par DHCP, par exemple.

Le serveur SIP atlanta.com est un type de serveur SIP connu sous le nom de serveur mandataire. Un serveur mandataire reçoit des demandes SIP et les transmet au nom du demandeur. Dans cet exemple, le serveur mandataire reçoit la demande INVITE et renvoie une réponse 100 (En essai) au téléphone logiciel d'Alice. La réponse 100 (En essai) indique que l'INVITE a été reçu et que le mandataire travaille en son nom pour acheminer l'INVITE à la destination. Les réponses dans SIP utilisent un code à trois chiffres suivi par une phrase descriptive. Cette réponse contient les mêmes paramètres To, From, Call-ID, CSeq et branch dans le Via que dans l'INVITE, ce qui permet au téléphone logiciel d'Alice de corréler cette réponse avec l'INVITE envoyé. Le serveur mandataire atlanta.com localise le serveur mandataire à biloxi.com, éventuellement en effectuant un type particulier de recherche de DNS (service de nom de domaine) pour trouver le serveur SIP qui dessert le domaine biloxi.com. Ceci est décrit en [4]. En résultat, il obtient l'adresse IP du serveur mandataire biloxi.com et y transmet ou mandate la demande INVITE. Avant de transmettre la demande, le serveur mandataire atlanta.com ajoute une valeur de champ d'en-tête Via supplémentaire qui contient sa propre adresse (l'INVITE contient déjà l'adresse d'Alice dans le premier Via). Le serveur mandataire biloxi.com reçoit l'INVITE et répond par une réponse 100 (En essai) au serveur mandataire atlanta.com pour indiquer qu'il a reçu l'INVITE et traite la demande. Le serveur mandataire consulte une base de données, appelée de façon générique un service de localisation, qui contient l'adresse IP courante de Bob. (Nous verrons dans la prochaine section comment on peut remplir cette base de données.) Le serveur mandataire biloxi.com ajoute une autre valeur de champ d'en-tête Via avec sa propre adresse pour l'INVITE et l'envoie par procuration au téléphone SIP de Bob.

Le téléphone SIP de Bob reçoit l'INVITE et alerte Bob de l'appel entrant provenant d'Alice, de sorte que Bob peut décider s'il va répondre à l'appel, c'est à dire, le téléphone de Bob sonne. Le téléphone SIP de Bob indique cela dans une réponse 180 (Sonnerie), qui est acheminée en retour aux deux mandataires dans les directions opposées. Chaque mandataire utilise le champ d'en-tête Via pour déterminer où il faut envoyer la réponse et retire sa propre adresse du haut. En résultat, bien que les examens DNS et de service de localisation soient nécessaires pour acheminer l'INVITE initiale, la réponse 180 (Sonnerie) peut être retournée à l'appelant sans boucles ou sans que des états soient maintenus dans les mandataires. Cela a aussi la propriété avantageuse que chaque mandataire qui voit l'INVITE va aussi voir toutes les réponses à l'INVITE.

Lorsque le téléphone logiciel d'Alice reçoit la réponse 180 (Sonnerie), il passe cette information à Alice, peut-être en utilisant une tonalité de retour d'appel audio ou en affichant un message sur l'écran d'Alice.

Dans cet exemple, Bob décide de répondre à l'appel. Lorsqu'il décroche le combiné, son téléphone SIP envoie une réponse 200 (OK) pour indiquer que l'appel a reçu une réponse. Le 200 (OK) contient un corps de message avec la description de support SIP du type de session que Bob veut établir avec Alice. En résultat, il y a un échange de messages SDP en deux phases : Alice en envoie un à Bob, et Bob en renvoie un à Alice. Cet échange en deux phases donne une capacité de négociation de base et il est fondé sur un modèle simple d'offre/réponse d'échange SDP. Si Bob ne souhaite pas répondre à l'appel ou qu'il est occupé sur une autre ligne, une réponse d'erreur aurait été envoyée à la place du 200 (OK), et il en serait résulté qu'aucune session de support n'aurait été établie. La liste complète des codes de réponse SIP figure à la Section 21. Le 200 (OK) (message F9 dans la Figure 1) pourrait ressembler à cela car Bob a envoyé :

SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bKnashds8;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131
(le SDP de Bob n'est pas montré)

La première ligne de la réponse contient le code de réponse (200) et la phrase de cause (OK). Les lignes restantes

contiennent des champs d'en-tête. Les champs d'en-tête Via, To, From, Call-ID, et CSeq sont copiés de la demande INVITE. (Il y a trois valeurs de champ d'en-tête Via - une ajoutée par le téléphone SIP d'Alice, une ajoutée par le mandataire atlanta.com, et une ajoutée par le mandataire biloxi.com.) Le téléphone SIP de Bob a ajouté un paramètre d'étiquette au champs d'en-tête. Cette étiquette sera incorporée dans le dialogue par les deux points d'extrémité et sera incluse dans toutes les futures demandes et réponses de cet appel. Le champ d'en-tête Contact contient un URI auquel Bob peut être atteint directement sur son téléphone SIP. Le Content-Type (*type de contenu*) et Content-Length (*longueur du contenu*) se réfèrent au corps de message (non montré) qui contient les informations de support SDP de Bob.

En plus de l'examen de DNS et de service de localisation montré dans cet exemple, les serveurs mandataires peuvent faire des "décisions d'acheminement" souples pour décider d'envoyer une demande. Par exemple, si le téléphone SIP de Bob a retourné une réponse 486 (Occupé ici), le serveur mandataire biloxi.com pourrait mandater l'INVITE au serveur de messagerie vocale de Bob. Un serveur mandataire peut aussi envoyer un INVITE à un certain nombre de localisations en même temps. Ce type de recherche parallèle est connu sous le nom de fourchage *(forking)*.

Dans ce cas, le 200 (OK) est réacheminé à travers les deux mandataires et est reçu par le téléphone logiciel d'Alice, qui arrête alors la tonalité de retour d'appel et indique que l'appel a reçu une réponse. Finalement, le téléphone logiciel d'Alice envoie un message d'accusé de réception, ACK, au téléphone SIP de Bob pour confirmer la réception de la réponse finale (200 (OK)). Dans cet exemple, le ACK est envoyé directement du téléphone logiciel d'Alice au téléphone SIP de Bob, outrepassant les deux mandataires. Ceci survient parce que les points d'extrémité ont appris leurs adresses respectives d'après les champs d'en-tête Contact à travers l'échange INVITE/200 (OK), qui n'étaient pas connues lorsque l'INVITE initial a été envoyé. Les examens effectués par les deux mandataires ne sont plus nécessaires, de sorte que les mandataires abandonnent le flux d'appel. Cela termine la prise de contact à trois INVITE/200/ACK utilisée pour établir les sessions SIP. Tous les détails sur l'établissement de session figurent dans la Section 13.

La session de support d'Alice et Bob a maintenant commencé, et ils envoient des paquets de support en utilisant le format sur lequel ils se sont mis d'accord dans l'échange de SDP. En général, les paquets de support de bout en bout prennent un chemin différent de celui des messages de signalisation SIP.

Durant la session, Alice ou Bob peut décider de changer les caractéristiques de la session de support. Ceci est accompli en envoyant un re-INVITE contenant une nouvelle description de support. Ce re-INVITE fait référence au dialogue existant de sorte que l'autre partie sait que c'est pour modifier une session existante et non pour établir un nouvelle session. L'autre partie envoie un 200 (OK) pour accepter le changement. Le demandeur répond au 200 (OK) par un ACK. Si l'autre partie n'accepte pas le changement, elle envoie une réponse d'erreur comme 488 (Non acceptable ici), qui reçoit aussi un ACK. Cependant, l'échec du re-INVITE ne cause pas de défaillance de l'appel existant - la session continue en utilisant les caractéristiques précédemment négociées. Tous les détails sur la modification de session figurent à la Section 14.

A la fin de l'appel, Bob déconnecte (raccroche) d'abord et génère un message BYE. Ce BYE est acheminé directement au téléphone logiciel d'Alice, outrepassant à nouveau les mandataires. Alice confirme la réception du BYE par une réponse 200 (OK), qui termine la session et la transaction BYE. Aucun ACK n'est envoyé - un ACK n'est envoyé qu'en réponse à une réponse à une demande INVITE. Les raisons de ce traitement spécial pour INVITE seront exposées plus loin mais se rapportent à des mécanismes de fiabilité dans SIP, à la durée qu'il faut pour répondre à un téléphone qui sonne et au fourchage. Pour cette raison, le traitement de la demande dans SIP est souvent classé comme INVITE ou non-INVITE, se référant à toutes les autres à côté de INVITE. Tous les détails sur la fin de session figurent dans la Section 15.

La Section 24.2 décrit complètement les messages montrés à la Figure 1.

Dans certains cas, il peut être utile pour les mandataires qui sont sur le chemin de signalisation SIP de voir tout l'échange de messages entre les points d'extrémité pour la durée de la session. Par exemple, si le serveur mandataire biloxi.com souhaite rester dans le chemin d'échange de messages SIP au-delà de l'INVITE initial, il ajouterait à l'INVITE un champs d'en-tête d'acheminement exigé connu sous le nom de Record-Route (*chemin enregistré*) qui contient un URI qui se résout par le nom d'hôte ou l'adresse IP du mandataire. Ces informations seraient reçues à la fois par le téléphone SIP de Bob et (du fait du champ d'en-tête Record-Route qui a été renvoyé dans le 200 (OK)) au téléphone logiciel d'Alice et mémorisé pour la durée du dialogue. Le serveur mandataire biloxi.com recevrait alors et mandaterait les ACK, BYE, et 200 (OK) au BYE. Chaque mandataire peut décider indépendamment de recevoir les messages suivants, et ces messages passeront à travers tous les mandataires qui ont choisi de le recevoir. Cette capacité est fréquemment utilisée pour les mandataires qui fournissent des dispositifs de milieu d'appel.

L'enregistrement est une autre opération commune dans SIP. L'enregistrement est une façon qu'a le serveur biloxi.com

d'apprendre la localisation actuelle de Bob. À l'initialisation, et à des intervalles périodiques, le téléphone SIP de Bob envoie des messages REGISTER à un serveur dans le domaine biloxi.com connu comme un registre SIP. Les messages REGISTER associent l'URI SIP ou SIPS de Bob (sip:bob@biloxi.com) à la machine dans laquelle il est hébergé actuellement (convoyé comme un URI SIP ou SIPS dans le champ d'en-tête Contact). Le registre inscrit cette association, aussi appelée un lien, dans une base de données, appelée le service de localisation , où elle peut être utilisée par le mandataire dans le domaine biloxi.com. Souvent, un serveur registre pour un domaine est co-localisé avec le mandataire pour ce domaine. Cette distinction entre types de serveurs SIP est un concept logique important, mais il n'est pas physique.

Bob n'est pas limité à s'enregistrer à partir d'un seul appareil. Par exemple, aussi bien son téléphone SIP chez lui que celui de son bureau peuvent envoyer des enregistrements. Ces informations sont mémorisées ensemble dans le service de localisation et permettent au mandataire d'effectuer différents types de recherches pour localiser Bob. De même, plus d'un utilisateur peut être enregistré sur un seul appareil au même moment.

Le service de localisation est un simple concept abstrait. Il contient généralement des informations qui permettent à un mandataire d'entrer dans un URI et de recevoir un ensemble de zéro ou plusieurs URI qui disent au mandataire où envoyer la demande. Les enregistrements sont une façon de créer ces informations, mais ne sont pas la seule façon. Des fonctions de transposition arbitraires peuvent être configurées à la discrétion de l'administrateur.

Finalement, il est important de noter que dans SIP, l'enregistrement est utilisé pour acheminer les demandes SIP entrantes et qu'il ne joue pas de rôle dans l'autorisation des demandes sortantes. L'autorisation et l'authentification sont traitées dans SIP, soit demande par demande avec un mécanisme d'interrogation/réponse, soit en utilisant un schéma de couche inférieure comme exposé à la Section 26.

L'ensemble complet des détails des messages SIP pour cet exemple d'enregistrement figure au paragraphe 24.1.

Des opérations supplémentaires dans SIP, telles que les questions sur les capacités d'un serveur ou client SIP utilisant OPTIONS, ou annulant une demande en cours en utilisant CANCEL, seront introduites dans des sections ultérieures.

5 Structure du protocole

SIP est structuré comme un protocole en couches, ce qui signifie que son comportement est décrit en termes d'ensembles de stades de traitement très indépendants avec seulement un couplage lâche entre chaque stade. Le comportement du protocole est décrit en couches pour les besoins de la présentation, permettant la description de fonctions communes à travers des éléments dans une seule section. Cela n'impose en aucune façon une quelconque mise en œuvre. Lorsque nous disons qu'un élément "contient" une couche, nous voulons dire qu'il est conforme à l'ensemble des règles définies par cette couche.

Tous les élément spécifiés par le protocole ne contiennent pas toutes les couches. De plus, les éléments spécifiés par SIP sont des éléments logiques, non pas physiques. Une réalisation physique peut choisir d'agir comme différents éléments logiques, peut-être même transaction par transaction.

La plus basse couche de SIP est sa syntaxe et son codage. Son codage est spécifié en utilisant une grammaire Backus-Naur Form (BNF) augmentée. Le BNF complet est spécifié à la Section 25 ; on trouvera une vue générale de la structure du message SIP à la Section 7.

La seconde couche est la couche de transport. Elle définit comment un client envoie des demandes et reçoit les réponses et comment un serveur reçoit les demandes et envoie les réponses sur le réseau. Tous les éléments SIP contiennent une couche transport. La couche transport est décrite à la Section 18.

La troisième couche est la couche transaction. Les transactions sont un composant fondamental de SIP. Une transaction est une demande envoyée par un client de transaction (utilisant la couche transport) à un serveur de transaction, avec toutes les réponses à cette demande renvoyées depuis le serveur de transaction au client. La couche de transaction traite les retransmissions de couche application, appariant les réponses aux demandes, et les temporisations de couche application. Toutes les tâches qu'accomplit un client d'agent d'utilisateur (UAC, *user agent client*) ont lieu en utilisant une série de transactions. L'exposé sur les transactions figure à la Section 17. Les agents d'utilisateur contiennent une couche transaction, tout comme les mandataires à états pleins *(stateful)*. Les mandataires sans état ne contiennent pas de couche de transaction. La couche de transaction a un composant client (désigné comme transaction de client) et un composant serveur (désigné comme transaction de serveur), chacun d'eux étant représenté par une machine à états finis qui est constituée pour traiter une demande particulière.

La couche au-dessus de la couche de transaction est appelée utilisateur de transaction (TU, *transaction user*). Chacune des entités de SIP, excepté les mandataires sans état, est un utilisateur de transaction. Lorsqu'un TU souhaite envoyer une demande, il crée une instance de transaction de client et lui passe la demande avec l'adresse IP de destination, le port, et le transport, auxquels envoyer la demande. Un TU qui créée une transaction de client peut aussi l'annuler. Lorsqu'un client annule une transaction, il demande que le serveur arrête le traitement, revienne à l'état qui existait avant l'initialisation de la transaction, et génère une réponse d'erreur spécifique de cette transaction. Ceci est fait avec une demande CANCEL, ce qui constitue une transaction en soi, mais se réfère à la transaction à annuler (Section 9).

Les éléments SIP, qui sont les clients et serveurs d'agent d'utilisateur, les mandataires sans état et à états pleins et les registres, contiennent un noyau qui les distingue les uns des autres. Les noyaux *(Cores)*, sauf pour le mandataires sans état, sont des utilisateurs de transaction. Alors que le comportement des noyaux des UAC et UAS dépend de la méthode, il y a certaines règles communes pour toutes les méthodes (Section 8). Pour un UAC, ces règles gouvernent la construction d'une demande; pour un UAS, elles gouvernent le traitement d'une demande et la génération d'une réponse. Comme les enregistrements jouent un rôle important dans SIP, un UAS qui tient un REGISTER reçoit le nom spécial de registre. La Section 10 décrit le comportement des noyaux d'UAC et d'UAS pour la méthode REGISTER. La Section 11 décrit le comportement des noyaux d'UAC et d'UAS pour la méthode OPTIONS, utilisée pour déterminer les capacités d'un UA.

Certaines autres demandes sont envoyées au sein d'un dialogue. Un dialogue est une relation SIP d'homologue à homologue entre deux agents d'utilisateur qui persiste pendant un certain temps. Le dialogue facilite le séquençage des messages et l'acheminement appropriés des demandes entre les agents d'utilisateur. La méthode INVITE est la seule façon définie dans la présente spécification pour établir un dialogue. Lorsqu'un UAC envoie une demande qui est dans le contexte d'un dialogue, il suit les règles communes d'UAC telles qu'exposées à la Section 8 mais aussi les règles pour les demandes de mi-dialogue. La Section 12 expose les dialogues et présente les procédures pour leur construction et leur maintenance, en plus de la construction de demandes au sein d'un dialogue.

La plus importante méthode dans SIP est la méthode INVITE, qui est utilisée pour établir une session entre des participants. Une session est une collection de participants, et des flux de support entre eux, pour les besoins d'une communication. La Section 13 expose comment les sessions sont initialisées, résultant en un ou plusieurs dialogues SIP. La Section 14 expose comment modifier les caractéristiques de cette session par l'utilisation d'une demande INVITE au sein d'un dialogue. Finalement, la Section 15 expose comment terminer une session.

Les procédures des Sections 8, 10, 11, 12, 13, 14, et 15 traitent entièrement du noyau d'UA (la Section 9 décrit l'annulation, qui s'applique au noyau d'UA et au noyau de mandataire La Section 16 discute de l'élément mandataire, qui facilite l'acheminement de messages entre les agents d'utilisateur.

6 Définitions

Les termes suivants ont une signification particulière pour SIP.

Adresse d'enregistrement *(*AOR, *Address-of-Record)* : Une adresse d'enregistrement est un URI SIP ou SIPS qui pointe sur un domaine avec un service de localisation qui peut transposer l'URI en un autre URI où l'utilisateur pourrait être disponible. Normalement, le service de localisation est rempli au moyen des enregistrements. Un AOR est fréquemment vu comme l'"adresse publique" de l'utilisateur.

Agent d'utilisateur de boucle locale (B2BUA, *Back-to-Back User Agent*) : Un agent d'utilisateur de boucle locale est une entité logique qui reçoit une demande et la traite comme une serveur d'agent d'utilisateur (UAS). Afin de déterminer comment il devrait être répondu à la demande, il agit comme un client d'agent d'utilisateur (UAC) et génère des demandes. A la différence d'un serveur mandataire, il maintient l'état du dialogue et doit participer à toutes les demandes envoyées dans les dialogues qu'il a établi. Comme c'est un enchaînement d'UAC et d'UAS, aucune définition explicite n'est nécessaire pour son comportement.

Appel *(call)* : Appel est un terme informel qui se réfère à une certaine communication entre homologues, généralement établie pour les besoins d'une conversation multimédia.

Partie d'appel *(Call Leg)* : Autre nom pour un dialogue [31] ; n'est plus utilisé dans la présente spécification.

Appel à états pleins *(Call Stateful)* : Un mandataire est en appel à états pleins lorsqu'il retient les états d'un dialogue depuis l'INVITE d'initialisation jusqu'à la demande BYE de terminaison. Un mandataire en appels à états plein est toujours à état pleins de transaction, mais la réciproque n'est pas nécessairement vraie.

Client : Un client est tout élément de réseau qui envoie une demande SIP et reçoit des réponses SIP. Les clients peuvent ou non interagir directement avec un utilisateur humain. Les clients et mandataires d'agent d'utilisateur sont des clients.

Conférence : Session multimédia (voir ci-dessous) qui contient plusieurs participants.

Noyau *(Core)* : Désigne les fonctions spécifiques d'un type particulier d'entité SIP, par exemple, spécifiques d'un mandataire à états pleins ou sans état, d'un agent d'utilisateur ou d'un registre. Tous les noyaux, sauf pour les mandataires sans état, sont des utilisateurs de transactions.

Dialogue : Un dialogue est une relation SIP d'homologue à homologue qui persiste pendant un certain temps entre deux agents d'utilisateur. Un dialogue est établi par les messages SIP, comme une réponse 2xx à une demande INVITE. Un dialogue est identifié par un identifiant d'appel, une étiquette locale, et une étiquette distante. Un dialogue était précédemment appelé partie d'appel *(call leg)* dans la RFC 2543.

Aval *(Downstream)* : Direction de la transmission d'un message au sein d'une transaction qui se réfère à la direction des flux de demandes du client d'agent d'utilisateur au serveur d'agent d'utilisateur.

Réponse finale : Réponse qui termine une transaction SIP, par opposition à une réponse provisoire, qui ne le fait pas. Toutes les réponses 2xx, 3xx, 4xx, 5xx et 6xx sont finales.

En-tête : Un en-tête est un composant d'un message SIP qui convoie des informations sur le message. Il est structuré comme une séquence de champs d'en-tête.

Champ d'en-tête : Un champ d'en-tête est un composant de l'en-tête de message SIP. Un champ d'en-tête peut apparaître comme une ou plusieurs rangées de champs d'en-tête. Les rangées de champs d'en-tête consistent en un nom de champ d'en-tête et zéro ou plusieurs valeurs de champs d'en-tête. Plusieurs valeurs de champs d'en-tête sur une rangée donnée de champs d'en-tête sont séparées par des virgules. Certains champs d'en-tête peuvent avoir une seule valeur de champ d'en-tête, et par conséquent apparaître toujours comme une seule rangée de champs d'en-tête.

Valeur de champ d'en-tête : C'est une seule valeur ; un champ d'en-tête consiste en zéro ou plusieurs valeurs de champ d'en-tête.

Domaine de rattachement : C'est le domaine qui fournit le service à un utilisateur SIP. Normalement, c'est le domaine présent dans l'URI dans l'adress-of-record d'un enregistrement.

Réponse d'information : Identique à réponse provisoire.

Initiateur, partie appelante, appelant : C'est la partie qui initialise une session (et un dialogue) par une demande INVITE. Un appelant conserve ce rôle depuis le moment où il envoie l'INVITE initial qui établit un dialogue jusqu'à la fin de ce dialogue.

Invitation : Une demande INVITE.

Invité, utilisateur invité, partie appelée, appelé : C'est la partie qui reçoit une demande INVITE pour les besoins de l'établissement d'une nouvelle session. Un appelé conserve ce rôle depuis le moment où il reçoit l'INVITE jusqu'à la fin du dialogue établi par cet INVITE.

Service de localisation : Un service de localisation est utilisé par un redirigeur SIP ou serveur mandataire pour obtenir des informations sur la ou les localisations possibles d'un appelé. Il contient une liste de liens de clés d'address-of-record pour zéro ou plusieurs adresses de contact. Les liens peuvent être créés et retirés de nombreuses façons ; la présente spécification définit une méthode REGISTER qui met à jour les liens.

Boucle : Demande qui arrive à un mandataire, est transmise, et revient ultérieurement à ce même mandataire. Lorsqu'elle arrive pour la seconde fois, son URI-de-demande est identique à celui de la première fois et les autres champs d'en-tête qui affectent le fonctionnement du mandataire sont inchangés, de sorte que le mandataire prendrait le même décision de traitement sur la demande que celle de la première fois. Les demandes en boucle sont des erreurs, et les procédures pour les détecter et les traiter sont décrites dans le protocole.

Acheminement lâche *(Loose Routing)* : Un mandataire est dit à acheminement lâche si il suit les procédures définies dans la présente spécification pour le traitement des champs d'en-tête Route. Ces procédures séparent la destination de

la demande (présente dans l'URI-de-demande) de l'ensemble des mandataires qui doivent être visités le long de l'acheminement (présent dans le champs d'en-tête Route). Un mandataire conforme à ces mécanismes est aussi connu sous le nom de routeur souple.

Message : Données envoyées entre des éléments SIP au titre du protocole. Les messages SIP sont des demandes ou des réponses.

Méthode : C'est la principale fonction qu'une demande vise à invoquer sur un serveur. La méthode est portée dans le message de demande lui-même. Des exemples de méthodes sont INVITE et BYE.

Mandataire de sortie : Mandataire qui reçoit des demandes de la part d'un client, même si il peut n'être pas le serveur donné par la résolution de l'URI-de-demande. Normalement un agent d'utilisateur est configuré manuellement avec un mandataire de sortie, ou il peut en acquérir la connaissance au moyen de protocoles d'autoconfiguration.

Recherche parallèle : Dans une recherche parallèle, un mandataire produit plusieurs demandes à des localisations d'utilisateur possibles à réception d'une demande entrante. Plutôt que de produire une seule demande et d'attendre une réponse finale avant de produire la demande suivante comme dans une recherche séquentielle, une recherche parallèle produit des demandes sans attendre le résultat des demandes précédentes.

Réponse provisoire : Réponse utilisée par le serveur pour indiquer l'avancement, mais qui ne termine pas une transaction SIP. Les réponses 1xx sont provisoires, les autres réponses sont considérées comme finales.

Mandataire, serveur mandataire : Entité intermédiaire qui agit à la fois comme un serveur et comme un client pour les besoins de l'élaboration de demandes au nom des autres clients. Un serveur mandataire joue principalement un rôle d'acheminement, ce qui signifie que sa tâche est de s'assurer qu'une demande est envoyée à une autre entité "plus proche" de l'utilisateur cible. Les mandataires sont aussi utiles pour mettre en application la politique (par exemple, s'assurer qu'un utilisateur est autorisé à effectuer un appel). Un mandataire interprète, et, si nécessaire, réécrit des parties spécifiques d'un message de demande avant de le retransmettre.

Récurrence : Un client revient sur une réponse 3xx lorsqu'il génère une nouvelle demande à un ou plusieurs des URI dans le champ d'en-tête Contact dans la réponse.

Serveur de redirection : C'est un serveur d'agent d'utilisateur (UAS) qui génère des réponses 3xx aux demandes qu'il reçoit, amenant le client à contacter un ensemble d'URI de remplacement.

Registre : C'est un serveur qui accepte les demandes REGISTER et qui place les informations qu'il reçoit dans ces demandes dans le service de localisation pour le domaine qu'il gère.

Transaction régulière : Toute transaction ayant une méthode autre que INVITE, ACK, ou CANCEL.

Demande : Message SIP envoyé d'un client à un serveur, pour les besoins de l'invocation d'une opération particulière.

Réponse : Message SIP envoyé d'un serveur à un client, pour indiquer l'état d'une demande envoyée du client au serveur.

Retour d'appel : C'est la tonalité de signalisation produite par l'application de la partie appelante qui indique que l'appelé est en train d'être alerté (par la sonnerie).

Ensemble de routes : C'est une collection d'URI SIP ou SIPS ordonnée qui représente une liste de mandataires qui doivent être traversés lors de l'envoi d'une demande particulière. Un ensemble de routes peut être appris, par l'intermédiaire des en-têtes comme Record-Route, ou il peut être configuré.

Serveur : C'est un élément de réseau qui reçoit des demandes afin de les servir et qui renvoie des réponses à ces demandes. Des exemples de serveur sont les mandataires, les serveurs d'agent d'utilisateur, les serveurs de redirection et les registres.

Recherche séquentielle : Dans une recherche séquentielle, un serveur mandataire essaye chaque adresse de contact à la suite, ne passant à la suivante qu'après que la précédente a généré une réponse finale. Une réponse finale de classe 2xx ou 6xx termine toujours une recherche séquentielle.

Session : D'après la spécification SDP : "Une session multimédia est un ensemble d'envoyeurs et de receveurs

multimédia et les flux de données qui s'écoulent des envoyeurs vers les receveurs. Une conférence multimédia est un exemple de session multimédia." (RFC 2327 [1]) (Une session telle que définie pour SDP peut comprendre une ou plusieurs sessions RTP.) Ainsi définie, un appelé peut être invité plusieurs fois, par des appels différent, à la même session. Si SDP est utilisé, une session est définie par l'enchaînement du nom d'utilisateur SDP, de l'identifiant de session, du type de réseau, du type d'adresse, et des éléments d'adresse dans le champ origine.

Transaction SIP : Une transaction SIP survient entre un client et un serveur et comprend tous les messages depuis la première demande envoyée du client au serveur jusqu'à la réponse finale (non-1xx) envoyée du serveur au client. Si la demande est INVITE et la réponse finale est un non-2xx, la transaction comporte aussi un ACK à la réponse. Le ACK pour une réponse 2xx à une demande INVITE est une transaction distincte.

Spirale : Une spirale est une demande SIP qui est acheminée vers un mandataire, transmise, et revient encore à ce mandataire, mais elle diffère cette fois d'une façon qui a pour résultat une décision de traitement différente de celle de la demande d'origine. Normalement, cela signifie que l'URI-de-demande de la demande diffère de celui de l'arrivée précédente. Une spirale n'est pas une condition d'erreur, à la différence d'une boucle. Une cause typique en est le renvoi d'appel. Un utilisateur appelle joe@example.com. Le mandataire example.com le transmet au microordinateur de Joe, qui à son tour, le transmet à bob@example.com. La demande est remandatée au mandataire example.com. Cependant, ceci n'est pas une boucle. Comme la demande est ciblée sur un utilisateur différent, c'est considéré comme une spirale, qui est une condition valide.

Mandataire à états pleins : C'est une entité logique qui maintient les machines d'état de transaction de client et de serveur définies par la présente spécification durant le traitement d'une demande, et est aussi connue sous le nom de mandataire de transaction à états pleins. Le comportement d'un mandataire à états pleins est défini plus complètement à la Section 16. Un mandataire (de transaction) à états pleins n'est pas la même chose qu'un mandataire d'appel à états pleins.

Mandataire sans état : C'est une entité logique qui ne maintient pas les machines d'état de transaction de client ou de serveur définies dans la présente spécification lorsqu'elle traite les demandes. Un mandataire sans état transmet chaque demande qu'il reçoit en aval et chaque réponse qu'il reçoit en amont.

Acheminement strict : Un mandataire est dit à acheminement strict si il suit les règles de traitement d'acheminement de la RFC 2543 et de nombreux travaux antérieurs en cours dans les versions de cette RFC. Cette règle amenait les mandataires à détruire le contenu de l'URI-de-demande lorsqu'un champs d'en-tête Route était présent. Le comportement d'acheminement strict n'est pas utilisé dans la présente spécification, en faveur d'un comportement d'acheminement lâche. Les mandataires qui effectuent un acheminement strict sont aussi appelés routeurs stricts.

Demande de rafraîchissement de cible : C'est une demande envoyée au sein d'un dialogue et se définit comme une demande qui peut modifier la cible distante du dialogue.

Utilisateur de transaction (TU, *Transaction User*) : C'est la couche de traitement du protocole qui réside au dessus de la couche de transaction. Les utilisateurs de transaction incluent le noyau d'UAC, le noyau d'UAS, et le noyau de mandataire.

Amont : Direction de transmission de message au sein d'une transaction qui se réfère à la direction des flux de réponses du serveur d'agent utilisateur en retour vers le client d'agent utilisateur.

Codé en URL (*URL-encoded*) : Chaîne de caractères codée conformément au paragraphe 2.4 de la RFC 2396 [5].

Client d'agent d'utilisateur (UAC, *User Agent Client*) : C'est une entité logique qui crée une nouvelle demande, puis utilise la machinerie d'état de transaction de client pour l'envoyer. Le rôle de l'UAC ne dure que pendant le temps de cette transaction. En d'autres termes, si un logiciel initialise une demande, elle agit comme UAC pour la durée de cette transaction. Si elle reçoit plus tard une demande, elle assume le rôle de serveur d'agent d'utilisateur pour le traitement de cette transaction.

Noyau d'UAC : Ensemble des fonctions de traitement exigées d'un UAC qui réside au-dessus des couches de transaction et de transport.

Serveur d'agent d'utilisateur (UAS, *User Agent Server*) : C'est une entité logique qui génère une réponse à une demande SIP. La réponse accepte, rejette, ou redirige la demande. Ce rôle ne dure que pendant le temps de cette transaction. En d'autres termes, si un logiciel répond à une demande, elle agit comme UAS pour la durée de cette transaction. Si elle génère plus tard une demande, elle assume le rôle d'un client d'agent d'utilisateur pour le traitement de cette transaction.

Noyau d'UAS : Ensemble des fonctions de traitement exigées d'un UAS qui réside au-dessus des couches de transaction et de transport.

Agent d'utilisateur (UA, *User Agent*) : Entité logique qui peut agir à la fois comme client d'agent d'utilisateur et comme serveur d'agent d'utilisateur.

Le rôle d'UAC et d'UAS, ainsi que ceux de serveur mandataire et redirecteur, sont définis transaction par transaction. Par exemple, l'agent d'utilisateur qui initialise un appel agit comme UAC lors de l'envoi de la demande INVITE initiale et comme UAS lors de la réception d'une demande BYE de la part de l'appelé. De même, le même logiciel peut agir comme serveur mandataire pour une demande et comme serveur redirecteur pour la demande suivante.

Les serveurs mandataires, de localisation, et de registre définis ci-dessus sont des entités logiques ; les mises en œuvre PEUVENT les combiner en une seule application.

7 Messages de SIP

SIP est un protocole à base de texte qui utilise l'ensemble de caractères UTF-8 (RFC 2279 [7]). Un message SIP est une demande d'un client à un serveur, ou une réponse d'un serveur à un client.

Les deux messages de demande (section 7.1) et de réponse (section 7.2) utilisent le format de base de la RFC 2822 [3], même si la syntaxe diffère dans l'ensemble de caractères et dans la syntaxe spécifiques. (SIP permet des champs d'en-tête qui ne seraient pas des champs d'en-tête valides dans la RFC 2822, par exemple.) Les deux types de messages consistent en une ligne de début, un ou plusieurs champs d'en-tête, une ligne vide indiquant la fin des champs d'en-tête, et un corps de message facultatif.

```
Message-générique  =  ligne de début
                      *en-tête-de-message
                      CRLF
                      [ corps-de-message ]
ligne-de-début        =  Ligne-de-demande / Ligne-d'état
```

La ligne-de-début, la ligne de chaque en-tête de message, et la ligne vide DOIVENT être terminées par une séquence de remplissage de ligne de retour chariot (CRLF, *carriage-return line-feed sequence*). Noter que la ligne vide DOIT être présente même si le corps de message ne l'est pas.

Excepté pour la différence ci-dessus dans les ensembles de caractères, une grande partie de la syntaxe de message et de champs d'en-tête de IP est identique à celle de HTTP/1.1. Plutôt que de répéter ici la syntaxe et la sémantique, on utilise [HX.Y] pour se référer à la Section X.Y de la spécification HTTP/1.1 en cours (RFC 2616 [8]).

Cependant, SIP n'est pas une extension de HTTP.

7.1 Demandes

Les demandes SIP se distinguent en ayant une Ligne-de-demande *(Request-Line)* comme ligne de début (*start-line*). Une Ligne-de-demande contient un nom de méthode, un URI-de-demande, et la version du protocole, séparés par un caractère d'un seul espace (SP, *single space*).

La Ligne-de-demande se termine par un CRLF. Aucun CR *(retour chariot)* ni LF (remplissage de ligne) n'est admis sauf dans la séquence de CRLF de fin de ligne. Aucun espace blanc linéaire (LWS, *linear whitespace*) n'est admis dans un des éléments.

Ligne-de-demande = Méthode SP URI-de-demande SP Version-SIP CRLF

Méthode : La présente spécification définit six méthodes : REGISTER pour enregistrer les informations de contact, INVITE, ACK, et CANCEL pour établir les sessions, BYE pour terminer les sessions, et OPTIONS pour interroger les serveurs sur leurs capacités. Les extensions à SIP, documentées dans les RFC normalisées, peuvent définir des méthodes supplémentaires.

URI-de-demande : L'URI-de-demande est un URI SIP ou SIPS comme décrit au paragraphe 19.1 ou un URI général (RFC 2396 [5]). Il indique l'utilisateur ou le service auquel cette demande est adressée. L'URI-de-demande NE DOIT PAS contenir d'espaces non formatés (*unescaped spaces*) ou de caractères de contrôle et NE DOIT PAS être inséré entre des crochets "<>".

Les éléments SIP PEUVENT prendre en charge des URI-de-demande avec des schémas autres que "sip" et "sips", par exemple le schéma d'URI "tel" de la RFC 2806 [9]. Les éléments SIP PEUVENT traduire des URI non-SIP en utilisant tout mécanisme à leur disposition, qui résultent en URI SIP, en URI SIPS, ou tout autre schéma.

Version-SIP : Les messages de demande et de réponse incluent tous deux la version de SIP en cours, et suivent [H3.1] (avec HTTP remplacé par SIP, et HTTP/1.1 remplacé par SIP/2.0) en ce qui concerne l'ordre des versions, les exigences de conformité, et la mise à niveau des numéros de version. Pour être conforme à la présente spécification, les applications qui envoient des messages SIP DOIVENT inclure un Version-SIP de "SIP/2.0". La chaîne Version-SIP est insensible à la casse, mais les mises en œuvre DOIVENT envoyer des majuscules.

A la différence de HTTP/1.1 SIP traite le numéro de version comme une chaîne littérale. En pratique, il ne devrait pas y avoir de différence.

7.2 Réponses

Les réponses SIP se distinguent des demandes par la Ligne-d'état *(Status-Line)* en ligne-de-début. Une Ligne-d'état comporte la version du protocole suivie par un Code-d'état (*Status-Code*) numérique et sa phrase textuelle associée, chaque élément étant séparé par un seul caractère SP *(espace simple)*.

Aucun CR *(retour chariot)* ou LF *(remplissage de ligne)* n'est admis sauf dans la séquence CRLF finale.

Ligne-d'état = Version-SIP SP Code-d'état SP Phrase-de-cause CRLF

Le Code-d'état est un code de résultat entier de trois chiffres qui indique le résultat d'une tentative de comprendre et satisfaire une demande. La Phrase-de-cause est destinée à donner une brève description textuelle du Code-d'état. Le Code-d'état est destiné à être utilisé par un automate, alors que la Phrase-de-cause est destinée à l'utilisateur humain. Un client n'est pas obligé d'examiner ou afficher la Phrase-de-cause.

Bien que la présente spécification suggère une formulation spécifique des phrases de cause, les mises en œuvre PEUVENT choisir d'autres textes, par exemple, dans la langue indiquée par le champs d'en-tête Langage-accepté de la demande.

Le premier chiffre du Code-d'état définit la classe de réponse. Les deux derniers chiffres n'ont pas de rôle spécifique en tant que catégorie. Pour cette raison, toute réponse avec un code d'état entre 100 et 199 est appelée "réponse 1xx", toute réponse avec un code d'état entre 200 et 299 une "réponse 2xx", et ainsi de suite. SIP/2.0 permet six valeurs pour le premier chiffre :
1xx : Provisoire -- demande reçue, poursuite du traitement de la demande ;
2xx : Réussite -- action bien reçue, comprise, et acceptée ;
3xx : Redirection – d'autres actions doivent être entreprise afin de mener la demande à bien ;
4xx : Erreur Client -- la demande contient une mauvaise syntaxe ou ne peut être satisfaite à ce serveur ;
5xx : Erreur Serveur -- le serveur a échoué à satisfaire une demande apparemment valide ;
6xx : Défaillance Globale -- la demande ne peut être satisfaite par aucun serveur.

La Section 21 définit ces classes et décrit les codes individuels.

7.3 Champs d'en-tête

Les champs d'en-tête SIP sont similaires aux champs d'en-tête HTTP en ce qui concerne la syntaxe et la sémantique. En particulier, les champs d'en-tête SIP suivent les définitions [H4.2] de syntaxe d'en-tête de message et les règles pour l'extension des champs d'en-tête sur plusieurs lignes. Cependant, ces dernières sont spécifiées dans HTTP avec un espace blanc et un retour à la ligne *(folding)* implicites. La présente spécification se conforme à la RFC 2234 [10] et utilise seulement les espaces blancs et les retours à la ligne explicites comme partie intégrante de la grammaire.

[H4.2] spécifie aussi que plusieurs champs d'en-tête du même nom de champ dont la valeur est une liste dont les éléments sont séparés par des virgules peuvent être combinés en un seul champ d'en-tête. Cela s'applique aussi à SIP, mais la règle spécifique est différente à cause des grammaires différentes. Précisément, tout en-tête SIP dont la grammaire est de la forme

en-tête = "nom-d'en-tête" HCOLON valeur-d'en-tête *(COMMA valeur-d'en-tête)

permet de combiner des champs d'en-tête du même nom dans une liste séparée par des virgules (*comma*). Le champ d'en-tête Contact permet une liste séparée par des virgules sauf si la valeur du champ d'en-tête valeur est "*".

7.3.1 Format de champ d'en-tête

Les champs d'en-tête suivent le même format générique d'en-tête que celui donné au paragraphe 2.2 de la RFC 2822 [3]. Chaque champ d'en-tête consiste en un nom de champ suivi par deux points (":") et la valeur du champ.

Nom-de-champ: valeur-de-champ

La grammaire formelle d'un en-tête de message spécifié à la Section 25 permet une quantité arbitraire d'espaces blancs d'un côté ou de l'autre des deux points ;cependant, les mises en œuvre devraient éviter les espaces entre le nom de champ et les deux points et utiliser un espace simple (SP) entre les deux points et la valeur-de-champ.

Sujet: déjeuner
Sujet : déjeuner
Sujet :déjeuner
Sujet: déjeuner

Et donc les quatre formes ci-dessus sont valides et équivalentes, mais la dernière est la forme préférée.

Les champs d'en-tête peuvent s'étendre sur plusieurs lignes en faisant précéder chaque ligne supplémentaire par au moins un SP ou une tabulation horizontale (HT). Le saut de ligne et l'espace blanc au début de la ligne suivante sont traités comme un seul caractère SP. Et donc, ce qui sui est équivalent :

Sujet: Je sais que tu es là, décroche le téléphone et réponds moi!
Sujet: Je sais que tu es là,
décroche le téléphone
et réponds moi!

L'ordre relatif des champs d'en-tête avec des noms de champ différents n'est pas significatif. Cependant, il est RECOMMANDÉ que les champs d'en-tête qui sont nécessaires pour le traitement par les mandataires (Via, Route, Record-Route, Proxy-Require, Max-Forwards, et Proxy-Authorization, par exemple) apparaissent vers le haut du message pour faciliter une analyse grammaticale rapide. L'ordre relatif des rangées de champs d'en-tête avec le même nom de champ est important. Plusieurs rangées de champs d'en-tête avec le même nom-de-champ PEUVENT être présentes dans un message si et seulement si la valeur-de-champ entière est définie pour ce champ d'en-tête comme une liste d'éléments séparés par des virgules (c'est-à-dire, si elle suit la grammaire définie au paragraphe 7.3). Il DOIT être possible de combiner les multiples rangées de champs d'en-tête en une paire "nom-de-champ: valeur-de-champ", sans changer la sémantique du message, en ajoutant chaque valeur-de-champ suivante à la première, chacune séparée par une virgule. Les exceptions à cette règle sont les champs d'en-tête WWW-Authenticate, Authorization, Proxy-Authenticate, et Proxy-Authorization. Plusieurs rangées de champs d'en-tête avec ces noms PEUVENT être présentes dans un message, mais comme leur grammaire ne suit pas la forme générale figurant au paragraphe 7.3, ils NE DOIVENT PAS être combinés en une seule rangée de champs d'en-tête.

Les mises en œuvre DOIVENT être capables de traiter plusieurs rangées de champs d'en-tête avec le même nom dans toute combinaison de formes de valeur à une seule valeur par ligne ou séparées par des virgules.

Les groupes suivants de rangées de champs d'en-tête sont valides et équivalentes :

Route: <sip:alice@atlanta.com>
Sujet: Déjeuner
Route: <sip:bob@biloxi.com>
Route: <sip:carol@chicago.com>
Route: <sip:alice@atlanta.com>, <sip:bob@biloxi.com>

Route: <sip:carol@chicago.com>
Sujet: Déjeuner
Sujet: Déjeuner
Route: <sip:alice@atlanta.com>,<sip:bob@biloxi.com>,<sip:carol@chicago.com>

Chacun des blocs suivants est valide mais pas équivalent aux autres :

Route: <sip:alice@atlanta.com>
Route: <sip:bob@biloxi.com>
Route: <sip:carol@chicago.com>
Route: <sip:bob@biloxi.com>
Route: <sip:alice@atlanta.com>
Route: <sip:carol@chicago.com>
Route: <sip:alice@atlanta.com>,<sip:carol@chicago.com>,<sip:bob@biloxi.com>

Le format d'une valeur-de-champ-d'en-tête est défini par nom-d'en-tête. Il sera toujours une séquence opaque d'octets de TEXT-UTF8, ou une combinaison d'espaces blancs, de jetons, séparateurs, et chaînes entre guillemets. De nombreux champs d'en-tête existants vont adhérer à la forme générale d'une valeur suivie par une séquence de paires de nom-de-paramètre, valeur-de-paramètre séparés par deux points :

Nom-de-champ: valeur-de-champ *(;nom-de-paramètre=valeur-de-paramètre)

Bien qu'un nombre arbitraire de paires de paramètres puissent être attachées à une valeur de champ d'en-tête, aucun nom-de-paramètre donné NE DOIT apparaître plus d'une fois.

Lorsqu'on compare les champs d'en-tête, les noms des champs sont toujours insensibles à la casse. Sauf mention contraire dans la définition d'un champ d'en-tête particulier, les valeurs de champ, noms de paramètre, et valeurs de paramètre sont insensibles à la casse. Les jetons sont toujours insensibles à la casse. Sauf spécification contraire, les valeurs exprimées comme chaînes entre guillemets sont sensibles à la casse. Par exemple,
Contact: <sip:alice@atlanta.com>;expires=3600
est équivalent à
CONTACT: <sip:alice@atlanta.com>;ExPiReS=3600
et
Content-Disposition: session;handling=optional
est équivalent à
content-disposition: Session;HANDLING=OPTIONAL
Les deux champs d'en-tête suivants ne sont pas équivalents :
Warning: 370 devnull "Prend un plus gros tuyau"
Warning: 370 devnull "PREND UN PLUS GROS TUYAU"

7.3.2 Classification des champs d'en-tête

Certains champs d'en-tête n'ont de sens que dans des demandes ou des réponses. On les appelle respectivement champs d'en-tête de demande et champs d'en-tête de réponse. Si un champ d'en-tête apparaît dans un message comme ne correspondant pas à sa catégorie (comme un champ d'en-tête de demande dans une réponse), il DOIT être ignoré. La Section 20 définit la classification de chaque champ d'en-tête.

7.3.3 Forme compacte

SIP fournit un mécanisme pour représenter les noms communs de champ d'en-tête en forme abrégée. Elle peut être utile lorsque autrement les messages deviendraient trop longs pour être acheminés sur le transport disponible pour ce faire (excédant l'unité maximum de transmission (MTU) lors de l'utilisation d'UDP, par exemple). Ces formes compactes sont définies à la Section 20. Une forme compacte PEUT être substituée à la forme longue d'un nom de champ d'en-tête à tout moment sans changer la sémantique du message. Un nom de champ d'en-tête PEUT apparaître dans les deux formes longue et courte au sein du même message. Les mises en œuvre DOIVENT accepter les deux formes longue et courte de chaque nom d'en-tête.

7.4 Corps

Les demandes, y compris les nouvelles demandes qui seront définies dans les extensions à la présente spécification, PEUVENT contenir des corps de message sauf mention contraire. L'interprétation du corps dépend de la méthode de la demande.

Pour les messages de réponse, la méthode de demande et le code d'état de la réponse déterminent le type et l'interprétation de tout corps de message. Toute réponses PEUT inclure un corps.

7.4.1 Type de corps de message

Le type de support Internet du corps de message DOIT être donné par le champ d'en-tête Content-Type. Si le corps n'a subi aucun codage du type compression, cela DOIT alors être indiqué par le champ d'en-tête Content-Encoding ; autrement, Content-Encoding DOIT être omis. Si applicable, l'ensemble de caractères du corps de message est indiqué au titre de la valeur du champ d'en-tête Content-Type.

Le type MIME "multiparti" défini dans la RFC 2046 [11] PEUT être utilisé au sein du corps du message. Les mises en œuvre qui envoient des demandes contenant des corps de message multiparties DOIVENT envoyer une description de session comme corps de message non multipartis si la mise en oeuvre distante le requiert au moyen d'un champ d'en-tête Accept qui ne contient pas de multiparti.

Les messages SIP PEUVENT contenir des corps binaires ou des parties de corps. Lorsque aucun paramètre explicite de charset n'est fourni par l'expéditeur, des sous-types de support du type "texte" sont définis comme ayant une valeur de charset par défaut de "UTF-8".

7.4.2 Longueur de corps de message

La longueur en octets du corps est fournie par le champ d'en-tête Content-Length. La Section 20.14 décrit en détail le contenu nécessaire pour ce champ d'en-tête.

Le codage de transfert "fragmenté" de HTTP/1.1 NE DOIT PAS être utilisé pour SIP. (Note : Le codage fragmenté modifie le corps d'un message afin de le transférer comme série de fragments, ayant chacun son propre indicateur de taille.)

7.5 Tramage des messages SIP

A la différence de HTTP, les mises en œuvre SIP peuvent utiliser UDP ou un autre protocole de datagrammes non fiable. Chacun de ces datagrammes porte une demande ou réponse. Voir à la Section 18 les contraintes sur l'usage de transports non fiables.

Les mises en œuvre qui traitent les messages SIP sur des transports orientés flux DOIVENT ignorer tout CRLF apparaissant avant la ligne-de-début [H4.1].

La valeur du champ d'en-tête Content-Length est utilisée pour localiser la fin de chaque message SIP dans un flux. Elle sera toujours présente lorsque les messages SIP sont envoyés sur des transports orientés flux.

8 Comportement général d'agent utilisateur

Un agent d'utilisateur représente un système de terminaison. Il contient un client d'agent d'utilisateur (UAC), qui génère des demandes, et un serveur d'agent d'utilisateur (UAS), qui leur répond. Un UAC est capable de générer une demande sur la base de certains stimulus externes (l'utilisateur clique sur un bouton, ou un signal sur une ligne RTPC) et de traiter une réponse. Un UAS est capable de recevoir une demande et de générer une réponse sur la base d'une action d'usager, d'un stimulus externe, du résultat de l'exécution d'un programme, ou de quelque autre mécanisme.

Lorsqu'un UAC envoie une demande, celle-ci passe à travers un certain nombre de serveurs mandataires, qui transmettent la demande à l'UAS. Lorsque l'UAS génère une réponse, la réponse est transmise à l'UAC.

Les procédures d'UAC et d'UAS dépendent fortement de deux facteurs. Premièrement de savoir si la demande ou réponse est à l'intérieur ou à l'extérieur d'un dialogue, et deuxièmement, de savoir la méthode d'une demande. Les dialogues sont exposés en détail à la Section 12 ; ils représentent une relation d'homologue à homologue entre les agents d'utilisateur et sont établis par des méthodes spécifiques de SIP, telles que INVITE.

Dans cette section, nous discutons des règles indépendantes de la méthode pour le comportement de l'UAC et de l'UAS lors du traitement des demandes qui sont en dehors d'un dialogue. Cela inclut, bien sûr, les demandes qui établissent elles-mêmes un dialogue.

Les procédures de sécurité pour des demandes et réponses en-dehors d'un dialogue sont décrites à la Section 26. Il existe des mécanismes spécifiques pour l'authentification mutuelle de l'UAS et de l'UAC. Un ensemble limité de dispositifs de confidentialité est également pris en charge par le chiffrement des corps en utilisant S/MIME.

8.1 Comportement d'UAC

Ce paragraphe traite du comportement d'UAC en-dehors d'un dialogue.

8.1.1 Générer la demande

Une demande SIP valide formulée par un UAC DOIT, au minimum, contenir les champs d'en-tête suivants : To, From, CSeq, Call-ID, Max-Forwards, et Via ; tous ces champs d'en-tête sont obligatoires dans toutes les demandes SIP. Ces six champs d'en-tête sont les blocs de construction fondamentaux d'un message SIP, car il fournissent ensemble la plus grande partie des services d'acheminement de message critiques, y compris l'adressage des messages, l'acheminement des réponses, la limitation de la propagation de message, l'ordonnancement des messages, et l'identification univoque des transactions. Ces champs d'en-tête s'ajoutent à la ligne de demande obligatoire, qui contient la méthode, l'URI-de-demande et la version-SIP.

Des exemples de demandes envoyées en–dehors d'un dialogue incluent un INVITE pour établir une session (Section 13) et un OPTIONS pour interroger sur les capacités (Section 11).

8.1.1.1 URI-de-demande

L'URI-de-demande initial du message DEVRAIT être réglé à la valeur de l'URI du champ To. Une exception notable est la méthode REGISTER ; le comportement pour régler l'URI-de-demande de REGISTER est donné à la Section 10. Il peut aussi n'être pas souhaitable pour des raisons de confidentialité ou de convenance de régler ces champs à la même valeur (particulièrement si l'UA générateur s'attend à ce que l'URI-de-demande soit changé durant le transit).

Dans certaines circonstances particulières, la présence d'un ensemble de routes pré allouées peut affecter l'URI-de-demande du message. Un ensemble préexistant de routes est un ensemble ordonné d'URI qui identifie une chaîne de serveurs, auxquels un UAC va envoyer des demandes sortantes qui sont en-dehors d'un dialogue. Communément, elles sont configurées manuellement sur l'UA par un utilisateur ou fournisseur de service, ou par quelque autre mécanisme non-SIP. Lorsqu'un fournisseur souhaite configurer un UA avec un mandataire extérieur, il est RECOMMANDÉ que cela soit fait en le configurant (l'approvisionnant) avec un ensemble de routes préexistant avec un seul URI, celui du mandataire extérieur.

Lorsqu'un ensemble de routes préexistant est présent, les procédures de remplissage des champs d'en-tête URI-de-demande et Route détaillées au paragraphe 12.2.1.1 DOIVENT être suivies (même s'il n'y a pas de dialogue), en utilisant l'URI-de-demande désiré comme URI cible distant.

8.1.1.2 To

Le champ d'en-tête To spécifie avant tout le receveur "logique" souhaité de la demande, ou l'adresse-d'enregistrement de l'utilisateur ou de la ressource qui est la cible de cette requête. Ce peut être ou pas le receveur ultime de la demande. Le champ d'en-tête To PEUT contenir un URI SIP ou SIPS, mais il peut aussi faire usage d'autres schémas d'URI (l'URL tel (RFC 2806 [9]), par exemple) où c'est approprié. Toutes les mises en œuvre de SIP DOIVENT prendre en charge le schéma d'URI SIP. Toute mise en œuvre qui prend en charge TLS DOIT prendre en charge le schéma d'URI SIPS. Le champ d'en-tête To permet un nom d'affichage.

Un UAC peut apprendre à remplir le champ d'en-tête To pour une demande particulière de nombreuses façons. Habituellement, l'utilisateur va suggérer le champ d'en-tête To par l'intermédiaire d'une interface humaine, peut être en entrant manuellement l'URI ou en le choisissant d'une sorte de carnet d'adresses. Souvent, l'utilisateur ne va pas entrer un URI complet, mais plutôt une chaîne de chiffres ou lettres (par exemple, "bob"). C'est à la discrétion de l'UA de choisir comment interpréter cette entrée. Utiliser la chaîne pour former la partie utilisateur d'un URI de SIP implique que l'UA souhaite que le nom soit résolu dans le domaine figurant à droite (RHS, *right hand side*) du signe à (@) dans l'URI de SIP (par exemple, sip:bob@exemple.com). Utiliser la chaîne pour former la partie utilisateur d'un URI SIPS implique que l'UA souhaite communiquer en toute sécurité, et que le nom est à résoudre dans le domaine à droite du signe à. Le RHS sera fréquemment le domaine de rattachement (*home domain*) du demandeur, ce qui permet au domaine de rattachement de traiter les demandes sortantes. Ceci est utile pour des dispositifs comme la "numérotation rapide" qui exige une interprétation de la part de l'utilisateur dans le domaine de rattachement. L'URL tel peut être utilisé lorsque l'UA ne souhaite pas spécifier le domaine que devrait interpréter un numéro de téléphone qui a été entré par l'utilisateur. Plutôt, chaque domaine à travers lequel passe la demande aura cette opportunité. A titre d'exemple, un utilisateur dans un aéroport pourrait se connecter et envoyer des demandes à travers un mandataire externe situé dans l'aéroport. S'il entre "411" (qui est le numéro de téléphone de l'assistance locale à l'annuaire aux USA), cela doit être interprété et traité par le mandateur externe dans l'aéroport, et non le domaine de rattachement de l'utilisateur. Dans ce cas, tel:411 serait le bon choix.

Une requête en dehors d'un dialogue NE DOIT PAS contenir d'étiquette To ; l'étiquette dans le champ To d'une requête identifie l'homologue du dialogue. Comme il n'est pas établi de dialogue, il n'y a pas d'étiquette.

Pour des informations complémentaires sur le champ d'en-tête To, voir le paragraphe 20.39. Ce qui suit est un exemple de champ d'en-tête To valide :

To: Carol <sip:carol@chicago.com>

8.1.1.3 From

Le champ d'en-tête From indique l'entité logique de l'initiateur de la demande, qui peut être l'adresse-d'enregistrement de l'utilisateur. Comme le champ d'en-tête To, il contient un URI et facultativement un nom d'affichage. Il est utilisé par les éléments SIP pour déterminer quelles règles de traitement appliquer à une requête (par exemple, rejet automatique d'appel). Comme tel, il est très important que l'URI From ne contienne pas d'adresses IP ou le FQDN de l'hôte sur lequel fonctionne l'UA, car ce ne sont pas des noms logiques.

Le champ d'en-tête From permet un nom d'affichage. Un UAC DEVRAIT utiliser le nom d'affichage "Anonymous", avec un URI syntaxiquement correct, mais sans autre signification particulière (comme sip:thisis@anonymous.invalid), si l'identité du client doit rester cachée.

Habituellement, la valeur qui remplit le champ d'en-tête From dans des demandes générées par un UA particulier est préprovisionnée par l'utilisateur ou par les administrateurs du domaine local de l'utilisateur. Si un UA particulier est utilisé par plusieurs usagers, il peut avoir des profils commutables qui incluent un URI correspondant à l'identité de l'usager profilé. Les receveurs des demandes peuvent authentifier le générateur d'une requête afin de s'assurer qu'ils sont bien ce que leur champ d'en-tête From prétend qu'ils sont (voir la Section 22 pour des précisions sur l'authentification).

Le champ From DOIT contenir un nouveau paramètre "étiquette", choisi par l'UAC. Voir au paragraphe 19.3 les détails sur le choix d'une étiquette. Pour des informations complémentaires sur le champ d'en-tête From, voir au paragraphe 20.20.

Exemples :
From: "Bob" <sips:bob@biloxi.com> ;tag=a48s
From: sip:+12125551212@phone2net.com;tag=887s
From: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8

8.1.1.4 Call-ID

Le champ d'en-tête Call-ID *(identifiant d'appel)* agit comme un identifiant unique pour grouper ensemble une série de messages. Il DOIT être le même pour toutes les demandes et réponses envoyées par l'un ou l'autre UA dans un dialogue. Il DEVRAIT être le même dans chaque enregistrement provenant d'un UA.

Dans une nouvelle demande créée par un UAC en dehors de tout dialogue, le champ d'en-tête Call-ID DOIT être choisi par l'UAC comme un identifiant unique au monde sur l'espace et le temps à moins qu'il ne soit outrepassé par un comportement spécifique de la méthode. Tous les UA de SIP doivent avoir un moyen de garantir que le champ d'en-tête

Call-ID qu'ils produisent ne va pas être généré par inadvertance par un autre UA. Noter que lorsque des demandes sont reprises après des réponses d'échec certain qui sollicitent des amendements à une demande (par exemple, une confrontation pour l'authentification), ces reprises des demandes ne sont pas considérées comme de nouvelles demandes, et n'ont donc pas besoins de nouveau champ d'en-tête Call-ID; voir le paragraphe 8.1.3.5.

L'utilisation d'identifiants cryptographiquement aléatoires (RFC 1750 [12]) est RECOMMANDÉE pour la génération des Call-ID. Les mises en œuvre PEUVENT utiliser la forme "localid@host". Les Call-ID sont sensibles à la casse et sont simplement comparés octet par octet.

L'utilisation d'identifiants cryptographiquement aléatoires procure une certaine protection contre la capture de session et réduit la vraisemblance de collisions de Call-ID non intentionnelles.

Aucun approvisionnement ni aucune interface humaine n'est exigé pour le choix de la valeur de champ d'en-tête Call-ID pour une demande.

Pour des informations complémentaires sur le champ d'en-tête Call-ID, voir au paragraphe 20.8.

Exemple :

Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@foo.bar.com

8.1.1.5 CSeq

Le champ d'en-tête CSeq sert de moyen d'identifier et ordonner les transactions. Il consiste en un numéro de séquence et une méthode. La méthode DOIT correspondre à celle de la demande. Pour des demandes non-REGISTER en-dehors d'un dialogue, la valeur du numéro de séquence est arbitraire. La valeur du numéro de séquence DOIT pouvoir être exprimée comme un entier non signé de 32 bits et DOIT être inférieure à 2**31. Tant qu'il suit les lignes directrices ci-dessus, un client peut utiliser tout mécanisme qu'il veut pour choisir les valeurs de champs d'en-tête CSeq.

Le paragraphe 12.2.1.1 expose la construction de CSeq pour des demandes au sein d'un dialogue.

Exemple :
CSeq: 4711 INVITE

8.1.1.6 Max-Forwards

Le champ d'en-tête Max-Forwards sert à limiter le nombre de sauts par lesquels une demande peut transiter sur le chemin menant à sa destination. Il consiste en un entier qui est décrémenté de un à chaque saut. Si la valeur de Max-Forwards atteint 0 avant que la demande n'atteigne sa destination, elle sera rejetée avec une réponse d'erreur 483(Trop de bonds).

Un UAC DOIT insérer un champ d'en-tête Max-Forwards dans chaque demande qu'il génère avec une valeur qui DEVRAIT être 70. Ce nombre a été chois pour être suffisamment grand pour garantir qu'une demande ne sera abandonnée sur aucun réseau SIP lorsqu'il n'y a pas de boucle, mais pas assez grand pour consommer les ressources des mandataires lorsque survient une boucle. Ces valeurs devraient être utilisées avec prudence et seulement dans des réseaux dont la topologie est connue de l'UA.

8.1.1.7 Via

Le champ d'en-tête Via indique le transport utilisé pour la transaction et identifie la localisation où la réponse doit être envoyée. Une valeur de champ d'en-tête Via n'est ajoutée qu'après que le transport qui sera utilisé pour atteindre le prochain saut a été choisi (ce qui peut implique l'usage de la procédures décrite dans [4]).

Lorsque l'UAC crée une demande, il DOIT insérer un Via dans cette demande. Le nom du protocole et la version du protocole dans le champ d'en-tête DOIVENT être respectivement SIP et 2.0. La valeur du champ d'en-tête Via DOIT contenir un paramètre de branche. Ce paramètre est utilisé pour identifier la transaction créée par cette demande. Ce paramètre est utilisé par le client et par le serveur.

La valeur du paramètre de branche DOIT être unique à travers l'espace et le temps pour toutes les demandes envoyées par l'UA. Les exceptions à cette règle sont CANCEL et ACK pour les réponses non-2xx. Comme exposé ci-dessous, une demande CANCEL aura la même valeur de paramètre de branche que la demande qu'il annule. Comme exposé au paragraphe 17.1.1.3, un ACK pour une réponse non-2xx aura aussi le même ID de branche que l'INVITE de la réponse

dont il accuse réception.

La propriété d'unicité du paramètre ID de branche, pour faciliter son utilisation comme ID de transaction, n'était pas incluse dans la RFC 2543.

L'ID de branche inséré par un élément conforme à la présente spécification DOIT toujours commencer par les caractères "z9hG4bK". Ces 7 caractères sont utilisés comme cookie magique (7 est réputé suffisant pour garantir qu'une mise en œuvre plus ancienne de la RFC 2543 ne prendrait pas une telle valeur), de sorte que les serveurs recevant la demande puissent déterminer que l'ID de branche a été construit de la façon décrite par la présente spécification (c'est-à-dire, unique au monde). Au-delà de cette exigence, le format précis du jeton de branche est défini par la mise en œuvre.

Les composants maddr, ttl, et send-by d'en-tête Via seront réglés lors du traitement de la demande par la couche transport (Section 18).

Le traitement de Via par les mandataires est décrit au paragraphe 16.6, item 8, et au paragraphe 16.7, item 3.

8.1.1.8 Contact

Le champ d'en-tête Contact fournit un URI SIP ou SIPS qui peut être utilisé pour contacter cette instance spécifique de l'UA pour des demandes ultérieures. Le champ d'en-tête Contact DOIT être présent et contenir exactement un URI SIP ou SIPS dans toute demande qui peut résulter en l'établissement d'un dialogue. Pour les méthodes définies dans la présente spécification, cela n'inclut que les demandes INVITE. Pour ces demandes, la portée de Contact est mondiale. C'est-à-dire que la valeur du champ d'en-tête Contact contient l'URI auquel l'UA aimerait recevoir les demandes, et cet URI DOIT être valide même s'il est utilisé dans des demandes ultérieures en dehors de tout dialogue.

Si l'URI-de-demande ou la valeur du champ d'en-tête Route de tête contient un URI SIPS URI, le champ d'en-tête Contact DOIT aussi contenir un URI SIPS.

Pour des informations complémentaires sur le champ d'en-tête Contact, voir au paragraphe 20.10.

8.1.1.9 Supported et Required

Si l'UAC prend en charge les extensions à SIP qui peuvent être appliquées par le serveur à la réponse, l'UAC DEVRAIT inclure un champ d'en-tête Supported *(pris en charge)* dans la demande faisant la liste des étiquettes d'option (paragraphe 19.2) pour ces extensions.

Les étiquettes d'option figurant sur la liste DOIVENT seulement se référer aux extensions définies dans les RFC normalisées. Ceci pour empêcher que les serveurs insistent pour que les clients mettent en œuvre des caractéristiques non standard définies par les fabricants afin de recevoir du service. Les extensions définies par les RFC expérimentales et informatives sont explicitement exclues de l'utilisation avec le champ d'en-tête Supported dans une demande, car elles sont trop souvent utilisées pour exposer des extensions définies par des fabricants.

Si l'UAC souhaite insister pour qu'un UAS comprenne une extension que l'UAC va appliquer à la demande afin de traiter la demande, il DOIT insérer un champ d'en-tête Require dans la demande qui fait la liste des étiquettes d'option pour cette extension. Si l'UAC souhaite appliquer une extension à la demande et insiste pour que tout mandataire traversé comprenne cette extension, il DOIT insérer un champ d'en-tête Proxy-Require dans la demande listant les étiquettes d'option pour cette extension.

Comme avec le champ d'en-tête Supported, les étiquettes d'option dans les champs d'en-tête Require et Proxy-Require DOIVENT seulement se référer aux extensions définies dans les RFC normalisées *(standards-track)*.

8.1.1.10 Composants de message supplémentaires

Après la création d'une nouvelle demande, et que les champs d'en-tête décrits ci-dessus ont été correctement construits, tous les champs d'en-tête facultatifs supplémentaires sont ajoutés, comme le sont tous les champs d'en-tête spécifiques de la méthode.

Les demandes SIP PEUVENT contenir un corps de message codé en MIME. Sans considération du type de corps que contient une demande, certains champs d'en-tête doivent être formulés pour caractériser le contenu de ce corps. Pour des informations complémentaires sur ces champs d'en-tête, voir des paragraphes 20.11 à 20.15.

8.1.2 Envoi de la demande

La destination est alors calculée pour la demande. Sauf si des politiques locales spécifient autre chose, la destination DOIT être déterminée en appliquant les procédures DNS décrites en [4] comme suit. Si le premier élément dans l'ensemble de routes indique un routage strict (résultant en la formation de la demande comme décrit au paragraphe 12.2.1.1), les procédures DOIVENT être appliquées à l'URI-de-demande de la demande. Autrement, les procédures sont appliquées à la première valeur de champ d'en-tête Route dans la demande (s'il en existe une), ou à l'URI-de-demande de la demande si aucun champ d'en-tête Route n'est présent. Ces procédures donnent un ensemble ordonné d'adresse, port, et transports à essayer. Indépendamment de l'URI utilisé en entrée aux procédures de [4], si l'URI-de-demande spécifie une ressource SIPS, l'UAC DOIT suivre les procédures de [4] comme si l'URI d'entrée était un URI SIPS.

La politique locale PEUT spécifier un ensemble de destinations de remplacement pour la tentative. Si l'URI-de-demande contient un URI SIPS, toute destination de remplacement DOIT être contactée avec TLS. Au-delà de cela, il n'y a pas de restriction sur les destinations de remplacement si la demande ne contient pas de champ d'en-tête Route. Cela donne une simple alternative à l'ensemble de routes préexistant en tant que moyen de spécifier un mandataire de sortie. Cependant, cette approche de la configuration d'un mandataire de sortie N'EST PAS RECOMMANDÉE ; un ensemble préexistant de routes avec un seul URI DEVRAIT être utilisé. Si la demande contient un champ d'en-tête Route, la demande DEVRAIT être envoyée aux localisations déduites de sa valeur la plus élevée, mais PEUT être envoyée à tout serveur dont l'UA est certain qu'il honorera les politiques de Route et d'URI-de-demande spécifiées dans le présent document (par opposition à celles de la RFC 2543). En particulier, un UAC configuré avec un mandataire de sortie DEVRAIT tenter d'envoyer la demande à la localisation indiquée dans la première valeur du champ d'en-tête Route au lieu d'adopter la politique d'envoi de tous les messages au mandataire de sortie.

Ceci garantit que les mandataires de sortie qui n'ajoutent pas de valeurs de champ d'en-tête Record-Route seront éliminés du chemin des demandes ultérieures. Cela permet aux points d'extrémité qui ne peuvent pas résoudre l'URI du premier Route de déléguer cette tâche à un mandataire de sortie.

L'UAC DEVRAIT suivre les procédures définies en [4] pour les éléments à états pleins, en essayant chaque adresse jusqu'à ce qu'un serveur soit contacté. Chaque essai constitue une nouvelle transaction, et donc, chacune porte une valeur supérieure de champ d'en-tête Via différente avec un nouveau paramètre de branche. De plus, la valeur du transport dans le champ d'en-tête Via est réglé au transport déterminé pour le serveur cible, quel qu'il soit.

8.1.3 Traitement des réponses

Les réponses sont d'abord traitées par la couche transport puis passées à la couche transaction. La couche transaction effectue son traitement puis passe la réponse au TU. La plus grande partie du traitement de la réponse dans le TU est spécifique de la méthode. Cependant, il y a certains comportements généraux qui sont indépendants de la méthode.

8.1.3.1 Erreurs de couche de transaction

Dans certains cas, la réponse retournée par la couche transaction ne sera pas un message SIP, mais plutôt une erreur de couche transaction. Lorsque une erreur de fin de temporisation est reçue de la couche transaction, elle DOIT être traitée comme si un code d'état 408 (Fin de temporisation de demande) avait été reçu. Si une erreur fatale de transport est rapportée par la couche transport (généralement due à des erreurs fatales d'ICMP dans UDP ou des défaillances de connexion dans TCP), la condition DOIT être traitée comme un code d'état 503 (Service indisponible).

8.1.3.2 Réponses non reconnues

Un UAC DOIT traiter toute réponse finale qu'il ne reconnaît pas comme étant équivalente au code de réponse x00 de cette classe, et DOIT être capable de traiter le code de réponse x00 pour toutes les classes. Par exemple, si un UAC reçoit un code de réponse non reconnu de 431, il peut en toute sécurité supposer qu'il y avait quelque chose d'erroné dans sa demande et traiter la réponse comme s'il avait reçu un code de réponse 400 (Mauvaise demande). Un UAC DOIT traiter toute réponse provisoire différente de 100 qu'il ne reconnaît pas comme 183 (Session en cours). Un UAC DOIT être capable de traiter les réponses 100 et 183.

8.1.3.3 Via

Si plus d'une valeur de champ d'en-tête Via est présente dans une réponse, l'UAC DEVRAIT éliminer le message.

La présence de valeurs de champ d'en-tête Via supplémentaires qui précèdent l'origine de la demande suggère que le message a été mal routé ou peut-être corrompu.

8.1.3.4 Traitement des réponses 3xx

A réception d'une réponse de redirection (par exemple, un code d'état de réponse 301), les clients DEVRAIENT utiliser le ou les URI dans les champs d'en-tête Contact pour formuler une ou plusieurs nouvelles demandes sur la base de la demande redirigée. Ce processus est similaire à celui d'un mandataire recourant à une réponse de classe 3xx, comme détaillé aux paragraphes 16.5 et 16.6. Un client commence avec un ensemble initial de cibles contenant exactement un URI, l'URI-de-demande de la demande originelle. Si un client souhaite formuler de nouvelles demandes sur la base d'une réponse de classe 3xx à cette demande, il place les URI à essayer dans l'ensemble cible. Sous réserve des restrictions de la présente spécification, un client peut choisir quels URI Contact il place dans l'ensemble cible. Comme avec la récurrence du mandataire, un client traitant des réponses de classe 3xx NE DOIT ajouter aucun URI donné plus d'une fois à l'ensemble cible. Si la demande originelle a un URI SIPS dans l'URI-de-demande, le client PEUT choisir de recourir à un URI non-SIPS, mais DEVRAIT informer l'utilisateur de la redirection sur un URI non sécurisé.

Toute nouvelle demande peut recevoir des réponses 3xx contenant elles-mêmes l'URI d'origine comme contact. Deux localisations peuvent être configurées pour se rediriger l'une sur l'autre. Placer une seule fois tout URI donné dans l'ensemble cible empêche des boucles de redirection infinies.

Avec la croissance de l'ensemble cible, le client PEUT générer de nouvelles demandes aux URI dans n'importe quel ordre. Un mécanisme courant est d'ordonner l'ensemble par valeur du paramètre "q" à partir des valeurs de champ d'en-tête Contact. Les demandes aux URI PEUVENT être générées en série ou en parallèle. Une approche est de traiter les groupes de valeurs q décroissantes en série et de traiter les URI dans chaque groupe de valeur q en parallèle. Une autre est d'effectuer seulement des traitements en série en ordre décroissant des valeur q, en choisissant arbitrairement entre les contacts de valeur q égale.

Si le contact d'une adresse de la liste résulte en une défaillance, comme défini au paragraphe suivant, l'élément passe à l'adresse suivante de la liste, jusqu'à épuisement de la liste. Si la liste est épuisée, la demande a échoué.

Les défaillances DEVRAIENT être détectées au moyen des codes de réponse d'échec (codes supérieurs à 399) ; pour les erreurs réseau, la transaction client va rapporter toute défaillance de couche transport à l'utilisateur de transaction. Noter que certains codes de réponse (détaillés en 8.1.3.5) indiquent que la demande peut être réessayée ; des demandes qui sont réessayées ne devraient pas être considérées comme des défaillances.

Lorsqu'une défaillance est reçue pour une adresse de contact particulière, le client DEVRAIT essayer l'adresse de contact suivante. Ceci impliquera de créer une nouvelle transaction client pour délivrer une nouvelle demande.

Afin de créer une demande sur la base d'une adresse de contact dans une réponse 3xx, un UAC DOIT copier l'URI entier de l'ensemble cible dans l'URI-de-demande, excepté les paramètres d'URI "method-param" et "header" (voir au paragraphe 19.1.1 la définition de ces paramètres). Il utilise les paramètres "header" pour créer des valeurs de champ d'en-tête pour la nouvelle demande, supplantant les valeurs de champ d'en-tête associées à la demande redirigée conformément aux lignes directrices du paragraphe 19.1.5.

Noter que dans certains cas, les champs d'en-tête qui ont été communiqués dans l'adresse de contact peuvent au lieu de cela s'ajouter aux champs d'en-tête de la demande existante dans la demande originelle redirigée. En règle générale, si les champs d'en-tête peuvent accepter une liste de valeurs séparées par des virgules, la nouvelle valeur de champ d'en-tête PEUT être ajoutée à toutes valeurs existantes dans la demande redirigée originelle. Si les champs d'en-tête n'acceptent pas plusieurs valeurs, la valeur dans la demande originelle redirigée PEUT être supplantée par la valeur de champ d'en-tête communiquée dans l'adresse de contact. Par exemple, si une adresse de contact est retournée avec la valeur suivante :

sip:user@host?Subject=foo&Call-Info=<http://www.foo.com>

tout champ d'en-tête Subject dans la demande originelle redirigée est supplantée, mais l'URL HTTP est simplement ajouté à toute valeur existante de champ d'en-tête Call-Info.

Il est RECOMMENDÉ que l'UAC réutilise les mêmes To, From, et Call-ID qu'utilisés dans la demande redirigée

originelle, mais l'UAC PEUT aussi choisir de mettre à jour la valeur du champ d'en-tête Call-ID pour des demandes nouvelles, par exemple.

Finalement, une fois que la nouvelle demande a été construite, elle est envoyée en utilisant une nouvelle transaction client, et donc, elle DOIT avoir un nouvel ID de branche dans le champ Via de tête comme exposé au paragraphe 8.1.1.7.

A tous les autres égards, des demandes envoyées à réception d'une réponse redirigée DEVRAIENT réutiliser les champs d'en-tête et corps de la demande originelle.

Dans certains cas, des valeurs de champ d'en-tête Contact peuvent être mises en mémoire cache à l'UAC temporairement ou de façon permanente en fonction du code d'état reçu et de la présence d'un intervalle d'expiration ; voir aux paragraphes 21.3.2 et 21.3.3.

8.1.3.5 Traitement des réponses 4xx

Certains codes de réponse 4xx requièrent un traitement d'UA spécifique, indépendant de la méthode.

Si une réponse 401 (Non autorisée) ou 407 (Authentification de mandataire exigée) est reçue, l'UAC DEVRAIT suivre les procédures d'autorisation du paragraphe 22.2 et du paragraphe 22.3 pour réessayer la demande avec des accréditations.

Si une réponse 413 (Entité de demande trop grande) est reçue (paragraphe 21.4.11), la demande contenait un corps qui était plus long que ce que l'UAS ne veut accepter. Si possible, l'UAC DEVRAIT ressayer la demande, soit en omettant le corps soit en en utilisant un plus court.

Si une réponse 415 (Type de support non accepté) est reçue (paragraphe 21.4.13), la demande contenait des types de support non acceptés par l'UAS. L'UAC DEVRAIT réessayer l'envoi de la demande, cette fois en utilisant seulement un contenu avec les types listés dans le champ d'en-tête Accept dans la réponse, avec les codages listés dans le champ d'en-tête Accept-Encoding dans la réponse, et avec les langages listés dans le Accept-Language dans la réponse.

Si une réponse 416 (Schéma d'URI non accepté) est reçue (paragraphe 21.4.14), l'URI-de-demande a utilisé un schéma d'URI non accepté par le serveur. Le client DEVRAIT réessayer la demande, cette fois, en utilisant un URI SIP.

Si une réponse 420 (Mauvaise extension) est reçue (paragraphe 21.4.15), la demande contenait un champ d'en-tête Require ou Proxy-Require faisant la liste des étiquettes d'option pour un dispositif non accepté par un mandataire ou UAS. L'UAC DEVRAIT réessayer la demande, cette fois en omettant toutes les extensions listées dans le champ d'en-tête Unsupported dans la réponse.

Dans tous les cas ci-dessus, la demande est réessayée en créant une nouvelle demande avec les modifications appropriées. Cette nouvelle demande constitue une nouvelle transaction et DEVRAIT avoir la même valeur de Call-ID, To, et From que la demande précédente, mais le CSeq devrait contenir un nouveau numéro de séquence supérieur de un au précédent.

Avec les autres réponses 4xx, y compris celle restant à définir, un réessai peut être ou non possible selon la méthode et le cas d'utilisation.

8.2 Comportement de l'UAS

Lorsqu'une demande en dehors d'un dialogue est traitée par un UAS, il y a un ensemble de règles de traitement qui sont suivies, indépendamment de la méthode. La Section 12 donne des lignes directrices sur la façon dont un UAS peut dire si une demande est en dehors ou dans un dialogue.

Noter que le traitement d'une demande est individuel. Si une demande est acceptée, tous les changements d'état qui lui sont associés DOIVENT être effectués. Si elle est rejetée, aucun changement d'état NE DOIT être effectué.

Les UAS DEVRAIENT traiter les demandes dans l'ordre des étapes qui suivent dans la présente section (c'est-à-dire, en commençant par l'authentification, puis en inspectant la méthode, les champs d'en-tête, et ainsi de suite tout au long du reste de cette section).

8.2.1 Inspection de la méthode

Une fois qu'une demande est authentifiée (ou que l'authentification est sautée), l'UAS DOIT inspecter la méthode de la demande. Si l'UAS reconnaît la méthode d'une demande mais ne la prend pas en charge, il DOIT générer une réponse 405 (Méthode non admise). Les procédures de génération des réponses sont décrites au paragraphe 8.2.6. L'UAS DOIT aussi ajouter un champ d'en-tête Allow (*permis*) à la réponse 405 (Méthode non admise). Le champ d'en-tête Allow DOIT faire la liste de l'ensemble des méthodes prises en charge par l'UAS qui génère le message. Le champ d'en-tête Allow est présenté au paragraphe 20.5. Si la méthode est une de celles prises en charge par le serveur, le traitement continue.

8.2.2 Inspection d'en-tête

Si un UAS ne comprend pas un champ d'en-tête dans une demande (c'est-à-dire, le champ d'en-tête n'est pas défini dans la présente spécification ou dans aucune extension prise en charge) il DOIT ignorer ce champ d'en-tête et continuer à traiter le message. Un UAS DEVRAIT ignorer tout champ d'en-tête mal formé qui n'est pas nécessaire pour le traitement des demandes.

8.2.2.1 To et URI de demande

Le champ d'en-tête To identifie le receveur originel de la demande désigné par l'utilisateur identifié dans le champ From. Le receveur originel peut être ou pas l'UAS traitant la demande, du fait d'un renvoi d'appel ou d'autres opérations de mandataire. Un UAS PEUT appliquer toute politique qu'il souhaite pour déterminer s'il accepte des demandes lorsque le champ d'en-tête To n'est pas l'identité de l'UAS. Cependant, il est RECOMMANDÉ qu'un UAS accepte des demandes même si il ne reconnaît pas le schéma d'URI (par exemple, un URI tel:) dans le champ d'en-tête To, ou si le champ d'en-tête To ne s'adresse pas à un utilisateur connue ou habituel de cet UAS. Si, d'un autre côte, l'UAS décide de rejeter la demande, il DEVRAIT générer une réponse avec un code d'état 403 (Interdit) et la passer au serveur de transaction pour transmission.

Cependant, l'URI-de-demande identifie l'UAS qui va traiter la demande. Si l'URI-de-demande utilise un schéma non pris en charge par l'UAS, il DEVRAIT rejeter la demande avec une réponse 416 (Schéma d'URI non pris en charge). Si l'URI-de-demande n'identifie pas une adresse pour laquelle l'UAS veut accepter des demandes, il DEVRAIT rejeter la demande avec une réponse 404 (Non trouvée). Normalement, un UA qui utilise la méthode REGISTER pour lier son adresse-d'enregistrement à une adresse de contact spécifique verra des demandes dont l'URI-de-demande est égal à l'adresse de contact. D'autres sources potentielles d'URI-de-demandes reçus incluent le champ d'en-tête Contact des demandes et réponses envoyées par l'UA qui établissent ou rafraîchissent les dialogues.

8.2.2.2 Demandes amalgamées

Si la demande n'a pas d'étiquette dans le champ d'en-tête To, le noyau d'UAS DOIT vérifier la demande par rapport aux transactions en cours. Si l'étiquette From, le Call-ID, et CSeq correspondent exactement à ceux associés à une transaction en cours, mais que la demande ne correspond pas à cette transaction (sur la base des règles de correspondance du paragraphe 17.2.3), le noyau d'UAS DEVRAIT générer une réponse 482 (Boucle détectée) et la passer au serveurde transaction.

La même demande est arrivée plus d'une fois à l'UAS, en suivant des chemins différents, le plus vraisemblablement à cause d'un fourchement. L'UAS traite la première de ces demandes reçue et répond par un 482 (Boucle détectée) au reste d'entre elles.

8.2.2.3 Require

En supposant que l'UAS décide qu'il est l'élément approprié pour traiter la demande, il examine le champ d'en-tête Require, s'il est présent.

Le champ d'en-tête Require est utilisé par un UAC pour dire à l'UAS quelles extensions SIP l'UAC s'attend à ce que l'UAS prenne en charge pour traiter correctement la demande. Son format est décrit au paragraphe 20.32. Si un UAS ne comprend pas une étiquette d'option figurant sur la liste d'un champ d'en-tête Require, il DOIT répondre en générant une réponse avec le code d'état 420 (Mauvaise extension). L'UAS DOIT ajouter un champ d'en-tête Unsupported, et

faire la liste des options qu'il ne comprend pas parmi celles du champ d'en-tête Require de la demande.

Noter que Require et Proxy-Require NE DOIVENT PAS être utilisés dans une requête SIP CANCEL, ou dans une requête ACK envoyée pour une réponse non 2xx. Ces champs d'en-tête DOIVENT être ignorés s'ils sont présents dans ces demandes.

Une requête ACK pour une réponse 2xx DOIT contenir seulement les valeurs Require et Proxy-Require qui étaient présentes dans la demande initiale.

Exemple :

UAC → UAS : INVITE sip:watson@bell-telephone.com SIP/2.0
Require: 100rel
UAS → UAC : SIP/2.0 420 Bad Extension
Unsupported: 100rel

Ce comportement assure que l'interaction client-serveur va se faire sans délai lorsque toutes les options seront comprises par les deux côtés, et être seulement un peu ralentie si des options ne sont pas comprises (comme dans l'exemple ci-dessus). Pour une paire client-serveur bien assortie, l'interaction se fait rapidement, économisant un aller-retour souvent rendu nécessaire par les mécanismes de négociation. De plus, cela supprime aussi des ambiguïtés lorsque le client demande des caractéristiques que le serveur ne comprend pas. Certaines caractéristiques, telles que les champs de traitement d'appel, n'ont d'intérêt que pour les systèmes de terminaison.

8.2.3 Traitement du contenu

En supposant que l'UAS comprend toutes les extensions demandées par le client, l'UAS examine le corps du message, et les champs d'en-tête qui le décrivent. S'il y a des corps dont le type (indiqué par le Type-de-contenu), la langue (indiquée par le Langage-de-contenu) ou le codage (indiqué par le Codage-de-contenu) ne sont pas compris, et que cette partie du corps n'est pas facultative (comme indiqué par le champ d'en-tête Disposition-du-contenu) l'UAS DOIT rejeter la demande avec une réponse 415 (Type de support non accepté). La réponse DOIT contenir un champ d'en-tête Accept qui fait la liste des types de tous les corps qu'il comprend, dans l'éventualité où la demande contiendrait des corps de types non acceptés par l'UAS. Si la demande contenait des codages de contenu non compris par l'UAS, la réponse DOIT contenir un champ d'en-tête Accept-Encoding faisant la liste des codages compris de l'UAS. Si la demande contenait un contenu dans des langues non comprises de l'UAS, la réponse DOIT contenir un champ d'en-tête Accept-Encoding indiquant les langues comprises de l'UAS. Au-delà de ces vérifications, le traitement des corps dépend de la méthode et du type. Pour des informations complémentaires sur le traitement des champs d'en-tête spécifiques du contenu, voir au paragraphe 7.4 ainsi que des paragraphes 20.11 à 20.15.

8.2.4 Application des extensions

Un UAS qui souhaite appliquer certaines extension lors de la génération de la réponse NE DOIT PAS faire ainsi sauf si la prise en charge de cette extension est indiquée dans le champ d'en-tête Supported dans la demande. Si l'extension désirée n'est pas prise en charge, le serveur DEVRAIT se replier sur le SIP de base et sur toutes autres extensions prises en charge par le client. Dans des circonstances exceptionnelles, lorsque le serveur ne peut pas traiter la demande sans l'extension, le serveur PEUT envoyer une réponse 421 (Extension demandée). Cette réponse indique que la réponse appropriée ne peut être générée sans prise en charge d'une extension spécifique. La ou les extensions nécessaires DOIVENT être incluses dans un champ d'en-tête Require dans la réponse. Ce comportement N'EST PAS RECOMMANDÉ, car il va généralement interrompre l'interopérabilité.

Toutes extensions s'appliquant à une réponse non 421 DOIT être listée dans un champ d'en-tête Require inclus dans la réponse. Bien sûr, le serveur NE DOIT PAS appliquer d'extensions non listées dans le champ d'en-tête Supported dans la demande. Il en résulte que le champ d'en-tête Require dans une réponse ne contiendra jamais que les étiquettes d'option définies dans les RFC normalisées.

8.2.5 Traitement de la demande

En supposant que toutes les vérifications du paragraphe précédent sont réussis, le traitement de l'UAS devient spécifique de la méthode. La Section 10 traite la requête REGISTER, la Section 11 traite la requête OPTIONS, la Section 13 traite la requête INVITE, et la Section 15 traite la requête BYE.

8.2.6 Génération de la réponse

Lorsqu'un UAS souhaite construire une réponse à une demande, il suit les procédures générales détaillées dans les paragraphes suivants. Des comportements supplémentaire spécifiques du code de réponse en question, qui ne sont pas précisés dans la présente section, peuvent aussi être nécessaires.

Une fois que toutes les procédures associées à la création d'une réponse ont été achevées, l'UAS renvoie la réponse au serveur de transaction duquel il avait reçu la demande.

8.2.6.1 Envoi d'une réponse provisoire

Une ligne directrice non spécifique de la méthode pour la génération des réponses est que les UAS NE DEVRAIENT PAS produire de réponse provisoire pour une demande non-INVITE. Au contraire, les UAS DEVRAIENT générer aussitôt que possible une réponse finale à une demande non-INVITE.

Lorsque une réponse 100 (Essai en cours) est générée, tout champ d'en-tête Horodatage présent dans la demande DOIT être copié dans cette réponse 100 (Essai en cours). S'il y a un délais dans la génération de la réponse, l'UAS DEVRAIT ajouter une valeur de délais dans la valeur d'horodatage dans la réponse. Cette valeur DOIT contenir la différence entre l'heure d'envoi de la réponse et celle de réception de la demande, mesurée en secondes.

8.2.6.2 En-têtes et étiquettes

Le champ From de la réponse DOIT être égal au champ d'en-tête From de la demande. Le champ d'en-tête Call-ID de la réponse DOIT être égal au champ d'en-tête Call-ID de la demande. Le champ d'en-tête CSeq de la réponse DOIT être égal au champ CSeq de la demande. Les valeurs du champ d'en-tête Via dans la réponse DOIVENT être égales aux valeurs de champ d'en-tête Via dans la demande et DOIVENT conserver le même ordre.

Si une demande contient une étiquette To dans la demande, le champ d'en-tête To dans la réponse DOIT être égal à celui de la demande. Cependant, si le champ d'en-tête To dans la demande ne contenait pas d'étiquette, l'URI dans le champ d'en-tête To dans la réponse DOIT être égal à l'URI dans le champ d'en-tête To ; de plus, l'UAS DOIT ajouter une étiquette au champ d'en-tête To dans la réponse (sauf pour la réponse 100 (En essai), dans laquelle une étiquette PEUT être présente). Cela sert à identifier l'UAS qui répond, ce qui peut avoir pour résultat un composant d'ID de dialogue. La même étiquette DOIT être utilisée pour toutes les réponses à cette demande, à la fois finales et provisoires (toujours avec l'exception de 100 (En essai)). Les procédures pour la génération des étiquettes sont définies au paragraphe 19.3.

8.2.7 Comportement d'UAS sans état

Un UAS sans état est un UAS qui ne conserve pas l'état de la transaction. Il répond normalement à des demandes, mais élimine tout état qui devrait ordinairement être conservé par un UAS après l'envoi d'une réponse. Si un UAS sans état reçoit la retransmission d'une demande, il régénère la réponse et la renvoie, juste comme s'il répondait à la première instance de la demande. Un UAS ne peut être sans état à moins que le traitement de la demande pour cette méthode ne résulte toujours en la même réponse si les demandes sont identiques. Cela élimine les registres sans état, par exemple. Les UAS sans état n'utilisent pas de couche de transaction ; ils reçoivent des demandes directement de la couche transport et envoient les réponses directement à la couche transport.

Le rôle de l'UAS sans état est principalement utile pour traiter des demandes d'appel non authentifiées pour lesquelles une demande de réponse est produite. Si des demandes non authentifiées étaient traitées en état plein, des flux malveillants de demandes non authentifiées pourraient créer des masses d'état de transaction qui pourraient ralentir ou complètement arrêter le traitement d'appel dans un UAS, créant effectivement une condition de déni de service ; pour des informations complémentaires, voir au paragraphe 26.1.5.

Les comportements les plus importants d'un UAS sans état sont les suivants :
- Un UAS sans état NE DOIT PAS envoyer de réponses provisoires (1xx).
- Un UAS sans état NE DOIT PAS retransmettre de réponses.
- Un UAS sans état DOIT ignorer les demandes d'accusé de réception (ACK).
- Un UAS sans état DOIT ignorer les demandes d'annulation (CANCEL).

o Les étiquettes d'en-tête To DOIVENT être générées pour les réponses sans état, d'une façon qui génère la même étiquette pour la même demande systématiquement. Pour plus d'informations sur la construction des étiquettes, voir au paragraphe 19.3.

Sous tous les autres aspects, un UAS sans état se comporte de la même façon qu'un UAS à états pleins. Un UAS peut fonctionner en mode à état plein ou en mode sans état pour chaque nouvelle requête.

8.3 Serveurs de redirection

Dans certaines architectures, il peut être souhaitable de réduire la charge de traitement des serveurs mandataires qui sont chargés de l'acheminement des demandes, et d'améliorer la robustesse des chemins de signalisation, en s'appuyant sur la redirection.

La redirection permet aux serveurs de repousser les informations d'acheminement pour une requête dans la réponse au client, se sortant par là de la boucle d'autres échanges de messages pour cette transaction tout en continuant à aider à la localisation de la cible de la demande. Lorsque le générateur de la demande reçoit la redirection, il va envoyer une nouvelle demande sur la base de l'URI ou des URI qu'il a reçu. En propageant les URI depuis le cœur du réseau vers sa périphérie, la redirection permet de considérables économies d'échelle pour le réseau.

Un serveur redirecteur est logiquement constitué d'un serveur de couche de transaction et d'un utilisateur de transaction qui a accès à un service de localisation de quelque sorte (voir la Section 10 pour plus d'informations sur les registres et les services de localisation). Ce service de localisation est effectivement une base de données qui contient des correspondances entre un seul URI et un ensemble d'une ou plusieurs localisations de remplacement auxquelles la cible de cet URI peut être trouvée.

Un serveur redirecteur ne produit aucune demande SIP de lui-même. Après réception d'une requête autre que CANCEL, le serveur refuse la demande ou rassemble la liste des localisations de remplacement provenant du service de localisation et retourne une réponse finale de classe 3xx. Pour des demandes CANCEL bien formées, il DEVRAIT retourner une réponse 2xx. Cette réponse termine la transaction SIP. Le serveur de redirection maintient l'état de la transaction pendant une transaction SIP entière. Il est de la responsabilité des clients de détecter les boucles de transmission entre les serveurs de redirection.

Lorsqu'un serveur redirecteur retourne une réponse 3xx à une demande, il remplit la liste de (une ou plusieurs) localisations de remplacement dans le champ d'en-tête Contact. Une valeur de paramètre "expires" des valeurs de champ d'en-tête Contact peut aussi être fournie pour indiquer la durée de vie des données de Contact.

Le champ d'en-tête Contact contient des URI qui donnent les nouvelles localisations ou noms d'utilisateur à essayer, ou peuvent simplement spécifier des paramètres de transport supplémentaires. Une réponse 301 (Déplacement permanent) ou 302 (Déplacement temporaire) peut aussi donner la même localisation et nom d'utilisateur qui était ciblée par la demande initiale mais spécifier des paramètres de transport supplémentaires tels qu'un serveur différent ou une adresse multidiffusion à essayer, ou un changement de transport SIP d'UDP à TCP ou vice versa.

Cependant, les serveurs redirecteurs NE DOIVENT PAS rediriger une demande sur un URI égal à celui de l'URI-de-demande ; au lieu de cela, pourvu que l'URI ne pointe pas sur lui-même, le serveur PEUT mandater la demande à l'URI de destination, ou PEUT la rejeter avec une 404.

Si un client utilise un mandataire externe, et que ce mandataire redirige en réalité les demandes, il y a un potentiel de boucles infinies de redirection.

Noter qu'une valeur de champ d'en-tête Contact PEUT aussi se référer à une ressource différente de celle appelée à l'origine. Par exemple, un appel SIP connecté à une passerelle RTPC peut avoir besoin de délivrer une annonce d'informations spéciale telle que "Le numéro que vous avez composé a été changé."

Un champ d'en-tête de réponse Contact peut contenir tout URI convenable qui indique où peut être atteint le demandé, sans être limité aux URI SIP. Par exemple, il pourrait contenir des URI pour des téléphones, fax, ou irc (si ils sont définis) ou un mailto : (RFC 2368 [32]) URL. Le paragraphe 26.4.4 expose les implications et limitations de la redirection d'URI SIPS sur un URI non SIPS.

Le paramètre "expires" d'une valeur de champ d'en-tête Contact indique la durée de validité de l'URI. La valeur du paramètre est un nombre qui indique des secondes. Si ce paramètre n'est pas fourni, la valeur du champ d'en-tête

Expires détermine la durée de validité de l'URI. Les valeurs mal formées DEVRAIENT être traitées comme équivalentes à 3600.

Ceci fournit un modeste niveau de rétrocompatibilité avec la RFC 2543, qui permet l'affichage du temps absolu dans ces champs d'en-tête. Si un temps absolu est reçu, il devra être traité comme mal formé, et revenir par défaut à 3600.

Les serveurs redirecteurs DOIVENT ignorer les caractéristiques qui ne sont pas comprises (y compris les champs d'en-tête non reconnus, toute étiquette d'option inconnue dans Require, ou même dans les noms de méthode) et continuer avec la redirection de la demande en question.

9 Annulation d'une demande

La section précédente a discuté du comportement général d'UA pour créer des demandes et traiter des réponses pour des demandes de toutes les méthodes. Dans la présente section, on expose une méthode de propos général, appelée CANCEL.

La demande CANCEL, comme l'implique son nom, sert à annuler une demande précédente envoyée par un client. Précisément, elle demande à l'UAS de cesser de traiter la demande et de générer une réponse d'erreur à cette demande. CANCEL n'a pas d'effet sur une demande à laquelle un UAS a déjà donné une réponse finale. A cause de cela, il est très utile d'annuler des demandes pour lesquelles il peut prendre à un serveur beaucoup de temps pour répondre. Pour cette raison, CANCEL est meilleur pour des demandes INVITE, qui peuvent nécessiter un long temps pour générer une réponse. Dans cette utilisation, un UAS qui reçoit une requête CANCEL pour un INVITE, mais n'a pas encore envoyé de réponse finale, "arrêterait de sonner", puis répondrait à l'INVITE par une réponse d'erreur spécifique (un 487).

Les demandes CANCEL peuvent être construites et envoyées aussi bien par les mandataires que par les clients d'agent d'utilisateur. La Section 15 expose dans quelles conditions un UAC annulerait une demande INVITE, et le paragraphe 16.10 expose l'utilisation de CANCEL par un mandataire.

Un mandataire à états pleins répond à CANCEL, plutôt que de simplement transmettre une réponse qu'il recevrait d'un élément aval. Pour cette raison, on se réfère à CANCEL comme à une demande "saut par saut", car il y est répondu à chaque saut de mandataire à état plein.

9.1 Comportement du client

Une demande CANCEL NE DEVRAIT PAS être envoyée pour annuler une demande autre que INVITE.

Comme il est répondu immédiatement aux demandes autres que INVITE, l'envoi de CANCEL pour une demande non INVITE créerait toujours une condition de conflit.

Les procédures suivantes sont utilisées pour construire une demande CANCEL. L'URI-de-demande, le Call-ID, To, la partie numérique de CSeq, et le champ d'en-tête From dans la demande CANCEL DOIVENT être identiques à ceux de la demande en cours d'annulation, y compris les étiquettes. Un CANCEL construit par un client DOIT avoir une seule valeur de champ d'en-tête Via correspondant à la valeur Via de tête dans la demande en cours d'annulation. Utiliser les mêmes valeurs pour ces champs d'en-tête permet de faire correspondre CANCEL avec la demande qu'il annule (le paragraphe 9.2 indique comment survient une telle correspondance). Cependant, la partie méthode du champ d'en-tête CSeq DOIT avoir une valeur de CANCEL. Cela lui permet d'être identifié et traité comme une transaction de plein droit (Voir la Section 17).

Si la demande en cours d'annulation contient un champ d'en-tête Route, la demande CANCEL DOIT inclure cette valeur de champ d'en-tête Route.

Ceci est nécessaire de façon que les mandataires sans état soient capable d'acheminer correctement des demandes CANCEL.

La demande CANCEL NE DOIT PAS contenir de champs d'en-tête Require ou Proxy-Require.

Une fois que CANCEL est construit, le client DEVRAIT vérifier si il a reçu une réponse (provisoire ou finale) pour la demande en cours d'annulation (qu'on appelle ici la "demande originelle").

Si aucune réponse provisoire n'a été reçue, la demande CANCEL NE DOIT PAS être envoyée ; le client DOIT plutôt attendre l'arrivée d'une réponse provisoire avant d'envoyer la demande. Si la demande originelle a généré une réponse finale, le CANCEL NE DEVRAIT PAS être envoyé, car c'est un no-op effectif, dans la mesure où CANCEL n'a pas d'effet sur des demandes qui ont déjà généré une réponse finale. Lorsque le client décide d'envoyer le CANCEL, il crée une transaction client pour le CANCEL et lui passe la demande CANCEL avec l'adresse, le port et le transport de destination. L'adresse, le port et le transport de destination pour le CANCEL DOIVENT être identiques à ceux utilisés pour l'envoi de la demande originelle.

Si il était admis d'envoyer le CANCEL avant de recevoir une réponse pour la demande précédente, le serveur pourrait recevoir le CANCEL avant la demande originelle.

Noter que les deux transactions correspondant à la demande originelle et la transaction CANCEL vont se terminer indépendamment. Cependant, un UAC qui annule une requête ne peut pas se contenter de recevoir une réponse 487 (Requête terminée) pour la demande originelle, car un UAS conforme à la RFC 2543 ne va pas générer une telle réponse. Si il n'y a pas de réponse finale pour la demande originelle en 64*T1 secondes (T1 est défini au paragraphe 17.1.1.1), le client DEVRAIT considérer alors la transaction originelle comme annulée et DEVRAIT détruire la transaction client traitant la demande originelle.

9.2 Comportement du serveur

La méthode CANCEL demande que le TU du côté serveur annule une transaction en cours. Le TU détermine que la transaction doit être annulée en prenant la demande CANCEL, et en supposant alors que méthode de la demande est tout sauf CANCEL ou ACK et en appliquant la procédure de correspondance de transaction du paragraphe 17.2.3. La transaction qui correspond est celle qui doit être annulée.

Le traitement d'une demande CANCEL à un serveur dépend du type de serveur. Un mandataire sans état va la transmettre, un mandataire à états pleins peut y répondre et générer des demandes CANCEL de lui-même, et un UAS va y répondre. Voir au paragraphe 16.10 le traitement de CANCEL par un mandataire.

Un UAS traite d'abord la demande CANCEL conformément au traitement général d'UAS décrit au paragraphe 8.2. Cependant, comme les demandes CANCEL sont saut par saut et ne peuvent pas être resoumises, elles ne peuvent pas être vérifiées par le serveur afin de d'obtenir des accréditifs appropriés dans un champ d'en-tête Authorization. Noter aussi que les demandes CANCEL ne contiennent pas de champ d'en-tête Require.

Si l'UAS n'a pas trouvé de transaction correspondante pour le CANCEL conformément à la procédure ci-dessus, il DEVRAIT répondre au CANCEL par un 481 (La branche d'appel/transaction n'existe pas). Si la transaction pour la demande originelle existe toujours, le comportement de l'UAS à réception d'une demande CANCEL dépend de si il a déjà envoyé une réponse finale pour la demande originelle. Si il l'a fait, la demande CANCEL n'a pas d'effet sur le traitement de la demande originelle, pas d'effet sur un état de session, et pas d'effet sur les réponses générées pour la demande originelle. Si l'UAS n'a pas fourni de réponse finale pour la demande originelle, son comportement dépend de la méthode de la demande originelle. Si la demande originelle était un INVITE, l'UAS DEVRAIT immédiatement répondre à l'INVITE par un 487 (Requête terminée). Une demande CANCEL n'a pas d'impact sur le traitement des transactions avec toute autre méthode définie dans la présente spécification.

Sans considération de la méthode de la demande originelle, tant que le CANCEL correspond à une transaction existante, l'UAS répond lui-même à la demande CANCEL par une réponse 200 (OK). Cette réponse est construite en suivant les procédures décrites au paragraphe 8.2.6, en notant que l'étiquette To de la réponse à CANCEL et l'étiquette To dans la réponse à la demande originelle DEVRAIENT être les mêmes. La réponse à CANCEL est passée au serveur de transaction pour transmission.

10 Enregistrements

10.1 Généralités

SIP offre une capacité de découverte. Si un utilisateur veut initialiser une session avec un autre utilisateur, SIP doit découvrir le ou les hôtes auxquels l'utilisateur de destination est couramment joignable. Ce processus de découverte est fréquemment accompli par les éléments de réseau SIP tels que des serveurs mandataires et les serveurs redirecteurs qui

sont chargés de recevoir une demande, de déterminer où l'envoyer sur la base de la connaissance de la localisation de l'utilisateur, et ensuite de l'y envoyer. Pour ce faire, les éléments de réseau SIP consultent un service abstrait connu sous le nom de service de localisation, qui fournit les liens d'adresse pour un domaine particulier. Ces liens d'adresse transposent un URI SIP ou SIPS entrant, sip:bob@biloxi.com, par exemple, en un ou plusieurs URI qui sont en quelque sorte "plus proches" de l'utilisateur désiré, sip:bob@engineering.biloxi.com, par exemple. Finalement, un mandataire va consulter un service de localisation qui fait correspondre un URI reçu à l'agent ou aux agents d'utilisateur dans lesquels le receveur désiré résident actuellement.

L'enregistrement crée des liens dans un service de localisation pour un domaine particulier qui associe un URI d'adresse-d'enregistrement avec une ou plusieurs adresses de contact. Et donc, lorsqu'un mandataire pour ce domaine reçoit une requête dont l'URI-de-demande correspond à l'adresse-d'enregistrement, le mandataire va transmettre la demande aux adresses de contact enregistrées à cette adresse-d'enregistrement. Généralement, enregistrer une adresse-d'enregistrement à un service de localisation d'un domaine n'a de sens que lorsque des demandes pour cette adresse-d'enregistrement devraient être acheminées sur ce domaine. Dans la plupart des cas, cela signifie que le domaine d'enregistrement aura besoin de correspondre au domaine dans l'URI de l'adresse-d'enregistrement.

Il y a de nombreuses façons d'établir le contenu du service de localisation. Une de ces façons est administrative. Dans l'exemple ci-dessus, Bob est connu pour être un membre du département d'ingénierie grâce à l'accès à une base de données professionnelle. Cependant, SIP fournit un mécanisme permettant à un UA de créer explicitement un lien. Ce mécanisme est appelé enregistrement.

L'enregistrement a pour conséquence l'envoi d'une requête REGISTER à un type particulier d'UAS connu sous le nom de registraire. Un registraire agit comme extrémité frontale du service de localisation pour un domaine, lisant et écrivant les transpositions sur la base du contenu des demandes REGISTER. Ce service de localisation est alors normalement consulté par un serveur mandataire qui est chargé d'acheminer les demandes pour ce domaine.

Une illustration du processus général d'enregistrement est donnée à la Figure 2. Noter que le registraire et le serveur mandataire sont des rôles logiques qui peuvent être joués par un seul appareil dans un réseau ; pour la clarté de l'exposé, les deux sont séparés sur cette figure. Noter aussi que les UA peuvent envoyer des demandes à travers un serveur mandataire afin d'atteindre un registraire si les deux sont des éléments séparés.

SIP ne recommande pas de mécanisme particulier pour la mise en œuvre du service de localisation. La seule exigence est qu'un registraire pour un domaine DOIT être capable de lire et d'écrire des données sur le service de localisation, et un mandataire ou serveur redirecteur pour ce domaine DOIT être capable de lire ces mêmes données. Un registraire PEUT être co-localisé avec un serveur mandataire SIP particulier pour le même domaine.

10.2 Construction de la demande REGISTER

Les demandes REGISTER ajoutent, retirent et interrogent des liens. Une demande REGISTER peut ajouter un nouveau lien entre une adresse-d'enregistrement et une ou plusieurs adresses de contact. L'enregistrement au nom d'une adresse-d'enregistrement particulière peut être effectuée par un tiers convenablement autorisé. Un client peut aussi retirer des liens antérieurs ou faire une interrogation pour déterminer quels liens sont actuellement en place pour une adresse-d'enregistrement.

Sauf notation contraire, la construction de la demande REGISTER et le comportement des clients qui envoient une demande REGISTER sont identiques au comportement général d'UAC décrits aux paragraphes 8.1 et 17.1.

Une demande REGISTER n'établit pas un dialogue. Un UAC PEUT inclure un champ d'en-tête Route dans une demande REGISTER sur la base d'un ensemble de routes préexistant comme décrit au paragraphe 8.1. Le champ d'en-tête Record-Route n'a pas de signification dans des demandes ou réponses REGISTER, et DOIT être ignoré s'il est présent. En particulier, l'UAC NE DOIT PAS créer un nouvel ensemble de routes sur la base de la présence ou l'absence d'un champ d'en-tête Record-Route dans toute réponse à une demande REGISTER.

Les champs d'en-tête suivants, excepté Contact, DOIVENT être inclus dans une demande REGISTER. Un champ d'en-tête Contact PEUT être inclus.

URI-de-demande : URI-de-demande nomme le domaine du service de localisation pour lequel l'enregistrement est voulu (par exemple, "sip:chicago.com"). Les composants "userinfo" et "@" de l'URI SIP NE DOIVENT PAS être présents.

To : Le champ d'en-tête To contient l'adresse d'enregistrement dont l'enregistrement est à créer, interroger ou modifier. Le champ d'en-tête To et le champ URI-de-demande sont normalement différent, car le premier contient un nom d'utilisateur. Cette adresse-d'enregistrement DOIT être un URI SIP ou un URI SIPS.

From : Le champ d'en-tête From contient l'adresse-d'enregistrement de la personne responsable de l'enregistrement. La valeur est la même que celle du champ d'en-tête To à moins que la demande soit un enregistrement par un tiers.

Call-ID : Tous les enregistrements provenant d'un UAC DEVRAIENT utiliser la même valeur de champ d'en-tête Call-ID pour les enregistrements envoyés à un registraire particulier. Si le même client devait utiliser des valeurs de Call-ID différentes, un registraire ne pourrait détecter si une demande REGISTER retardée pourrait arriver en désordre.

Cseq : La valeur CSeq valeur garantie un ordre approprié des demandes REGISTER. Un UA DOIT incrémenter la valeur CSeq de un pour chaque demande REGISTER avec le même Call-ID.

Contact : Les demandes REGISTER PEUVENT contenir un champ d'en-tête Contact avec zéro ou plus valeurs contenant des liens d'adresse.

Les UA NE DOIVENT PAS envoyer un nouvel enregistrement (c'est-à-dire, contenant de nouvelles valeurs de champ d'en-tête Contact, par opposition à une retransmission) jusqu'à ce qu'ils aient reçu une réponse finale de la part du registraire pour le précédent ou que la précédente demande REGISTER soit arrivée en fin de temporisation.

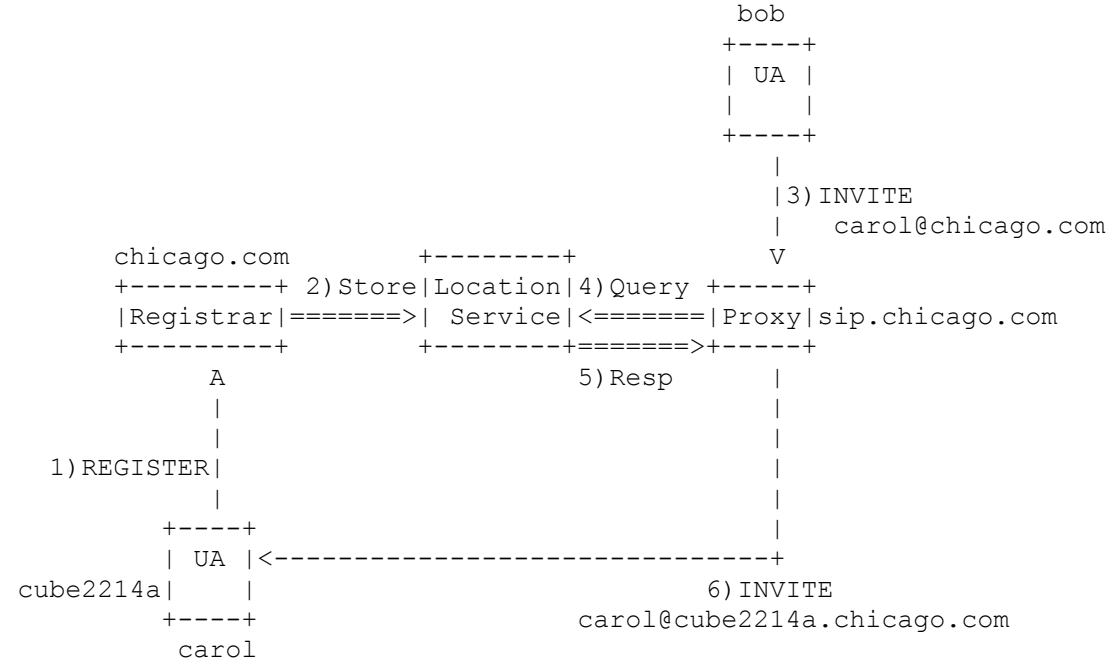


Figure 2 : Exemple de REGISTER

Les paramètres d'en-tête Contact suivants ont une signification particulière dans les demandes REGISTER :

action : Le paramètre "action" de la RFC 2543 est déconseillé. Les UAC NE DEVRAIENT PAS utiliser le paramètre "action".

expires : Le paramètre "expires" indique la durée pendant laquelle l'UA aimerait que le lien soit valide. La valeur est un nombre qui indique des secondes. Si ce paramètre n'est pas fourni, la valeur du champ d'en-tête Expires est utilisée à la place. Les mises en œuvre PEUVENT traiter les valeurs supérieures à 2**32-1 (4 294 967 295 secondes ou 136 ans) comme équivalentes à 2**32-1. Les valeurs mal formées DEVRAIENT être traitées comme équivalentes à 3 600.

10.2.1 Ajout des liens

La demande REGISTER envoyée à un registraire inclut la ou les adresses de contact auxquelles devraient être envoyées les demandes SIP pour l'adresse-d'enregistrement. L'adresse-d'enregistrement est incluse dans le champ d'en-tête To de la demande REGISTER.

Les valeurs de champ d'en-tête Contact de la demande consistent normalement en URI SIP ou SIPS qui identifient des points d'extrémité SIP particuliers (par exemple, "sip:carol@cube2214a.chicago.com"), mais ils PEUVENT utiliser tous schémas d'URI. Un UA SIP peut choisir d'enregistrer des numéros de téléphone (avec l'URL tel, de la RFC 2806 [9]) ou des adresses de messagerie électronique (avec un URL mailto URL, de la RFC 2368 [32]) comme Contacts pour une adresse-d'enregistrement, par exemple.

Par exemple, Carol, avec l'adresse-d'enregistrement "sip:carol@chicago.com", s'enregistrerait auprès du registraire SIP du domaine chicago.com. Ses enregistrements seraient alors utilisés par un serveur mandataire dans le domaine chicago.com pour acheminer des demandes pour l'adresse-d'enregistrement de Carol à son point d'extrémité SIP.

Une fois qu'un client a établi des liens chez un registraire, il PEUT envoyer des enregistrements ultérieurs contenant de nouveaux liens ou des modifications à des liens existants en tant que de besoin. La réponse 2xx à la demande REGISTER contiendra, dans un champ d'en-tête Contact, une liste complète des liens qui ont été enregistrés pour cette adresse-d'enregistrement auprès de ce registraire.

Si l'adresse-d'enregistrement dans le champ d'en-tête To d'une demande REGISTER est un URI SIPS, toutes les valeurs de champ d'en-tête Contact dans la demande DEVRAIT alors aussi être des URL SIPS. Les clients devraient seulement enregistrer des URI non SIPS sous une adresse-d'enregistrement SIPS lorsque la sécurité de la ressource représentée par l'adresse de contact est garantie par d'autres moyens. Ceci peut être applicable aux URI qui impliquent des protocoles autres que SIP, ou des appareils SIP sécurisés par des protocoles autres que TLS.

Les enregistrements n'ont pas besoin de mettre à jour tous les liens. Normalement, un UA ne met à jour que ses propres adresses de contact.

10.2.1.1 Réglage de l'intervalle d'expiration des adresses de contact

Lorsqu'un client envoie une demande REGISTER, il PEUT suggérer un intervalle d'expiration qui indique la durée pendant laquelle le client aimerait que l'enregistrement soit valide. (Comme décrit au paragraphe 10.3, le registraire choisit l'intervalle de temps réel sur la base de sa politique locale.)

Il y a deux moyens par lesquels un client peut suggérer un intervalle d'expiration pour un lien : au moyen d'un champ d'en-tête Expires ou d'un paramètre d'en-tête Contact "expires". Ce dernier permet aux intervalles d'expiration d'être suggérés lien par lien lorsque plus d'un lien est donné dans une seule demande REGISTER, tandis que l'autre suggère un intervalle d'expiration pour toutes les valeurs de champ d'en-tête Contact qui ne contiennent pas le paramètre "expires".

Si aucun des mécanismes d'expression d'une heure d'expiration suggérée n'est présent dans REGISTER, le client indique son désir que le serveur fasse le choix.

10.2.1.2 Préférences parmi les adresses de contact

Si plus d'un Contact est envoyé dans une demande REGISTER, l'UA enregistreur a l'intention d'associer tous les URI dans ces valeurs de champ d'en-tête Contact avec l'adresse-d'enregistrement présente dans le champ To. Cette liste peut être priorisée avec le paramètre "q" dans le champ d'en-tête Contact. Le paramètre "q" indique une préférence relative pour la valeur du champ d'en-tête Contact particulier par rapport à d'autres liens pour cette adresse-d'enregistrement. Le paragraphe 16.6 décrit comment un serveur mandataire utilise cette indication de préférence.

10.2.2 Retrait des liens

Les enregistrements sont à état conditionnel et arrivent à expiration s'ils ne sont pas rafraîchis, mais peuvent aussi être explicitement retirés. Un client peut essayer d'influencer l'intervalle d'expiration choisi par le registraire comme indiqué au paragraphe 10.2.1. Un UA demande le retrait immédiat d'un lien en spécifiant un intervalle d'expiration de "0" pour cette adresse de contact dans une demande REGISTER. Les UA DEVRAIENT prendre en charge ce mécanisme de telle sorte que les liens puissent être retirés avant l'échéance de leur intervalle d'expiration.

La valeur de champ d'en-tête Contact spécifique de REGISTER de "*" s'applique à tous les enregistrements, mais elle NE DOIT PAS être utilisée si le champ d'en-tête Expires n'est pas présent avec une valeur de "0".

L'utilisation de la valeur "*" de champ d'en-tête Contact permet à un UA enregistreur de retirer tous les liens associés à une adresse-d'enregistrement sans connaître leur valeurs précises.

10.2.3 Aller chercher les liens

Une réponse de succès à toute demande REGISTER contient la liste complète des liens existants, que la demande contienne ou non un champ d'en-tête Contact. Si aucun champ d'en-tête Contact n'est présent dans une demande REGISTER, la liste des liens reste inchangée.

10.2.4 Rafraîchissement des liens

Chaque UA est responsable du rafraîchissement des liens qu'il a établis antérieurement. Un UA NE DEVRAIT PAS rafraîchir des liens établis par d'autres UA.

La réponse 200 (OK) du registraire contient une liste des champs Contact qui énumère tous les liens en cours. L'UA compare chaque adresse de contact pour voir si c'est lui qui a créé cette adresse de contact, en utilisant les règles de comparaison du paragraphe 19.1.4. Si c'est lui, il met à jour l'intervalle de temps d'expiration conformément au paramètre expires ou, s'il est absent, à la valeur du champ Expires. L'UA produit alors une demande REGISTER pour chacun de ses liens avant l'écoulement de l'intervalle d'expiration. Il PEUT combiner plusieurs mises à jour dans une seule demande REGISTER.

Un UA DEVRAIT utiliser le même Call-ID pour tous les enregistrements durant un seul cycle d'amorçage. Les rafraîchissements d'enregistrement DEVRAIENT être envoyés à la même adresse réseau que celle de l'enregistrement d'origine, sauf en cas de redirection.

10.2.5 Réglage de l'horloge interne

Si la réponse à une demande REGISTER contient un champ d'en-tête Date, le client PEUT utiliser ce champ d'en-tête pour connaître l'heure exacte afin de régler toutes ses horloges internes.

10.2.6 Découverte d'un registre

Les UA peuvent utiliser trois façons de déterminer l'adresse à laquelle envoyer les enregistrements : par configuration, en utilisant l'adresse-d'enregistrement, et en multidiffusion. Un UA peut être configuré, par des moyens qui sortent du domaine d'application de la présente spécification, avec une adresse de registraire. S'il n'y a pas d'adresse de registraire configurée, l'UA DEVRAIT utiliser la partie hôte de l'adresse-d'enregistrement comme URI-de-demande et adresser là la demande, en utilisant les mécanismes normaux de localisation de serveur SIP [4]. Par exemple, l'UA pour l'utilisateur "sip:carol@chicago.com" adresse la demande REGISTER à "sip:chicago.com".

Finalement, un UA peut être configuré pour utiliser la multidiffusion. Les enregistrements en multidiffusion sont adressés à l'adresse multidiffusion bien connue de tous les serveurs SIP "sip.mcast.net" (224.0.1.75 pour IPv4). Il n'a pas été alloué d'adresse multidiffusion bien connue pour IPv6 ; une telle allocation devra être documentée séparément en temps utile. Les UA SIP PEUVENT surveiller cette adresse et l'utiliser pour se mettre au courant de la localisation d'autres utilisateurs locaux (voir [33]) ; cependant, ils ne répondent pas à la demande.

L'enregistrement multidiffusion peut être inapproprié dans certains environnements, par exemple, si plusieurs entreprises partagent le même réseau de zone locale.

10.2.7 Transmission d'une demande

Une fois que la méthode REGISTER a été construite, et la destination du message identifiée, les UAC suivent les procédures décrites au paragraphe 8.1.2 pour passer le REGISTER à la couche transaction.

Si la couche transaction retourne une erreur de temporisation parce que le REGISTER ne donne pas de réponse, l'UAC NE DEVRAIT PAS immédiatement réessayer un enregistrement sur le même registraire.

Un réessai immédiat va vraisemblablement aussi arriver en fin de temporisation. Attendre que s'écoule un délai raisonnable pour que les conditions qui causent la fin de temporisation soient corrigées réduit les charges inutiles sur le réseau. Il n'est pas spécifié d'intervalle particulier.

10.2.8 Réponses d'erreur

Si un UA reçoit une réponse 423 (Intervalle trop bref), il PEUT réessayer l'enregistrement après avoir rendu l'intervalle d'expiration de toutes les adresses de contact dans la demande REGISTER égales ou supérieures à l'intervalle d'expiration au sein du champ d'en-tête Min-Expires de la réponse 423 (Intervalle trop bref).

10.3 Traitement des demandes REGISTER

Un registraire est un UAS qui répond à des demandes REGISTER et maintient une liste des liens qui sont accessibles aux serveurs mandataires et aux serveurs redirecteurs au sein de son domaine administratif. Un registraire traite des demandes conformément aux paragraphes 8.2 et 17.2, mais il n'accepte que des demandes REGISTER. Un registraire NE DOIT PAS générer de réponses 6xx.

Un registraire PEUT rediriger des demandes REGISTER comme il convient. Une utilisation courante serait qu'un registraire écoute sur une interface multidiffusion pour rediriger des demandes REGISTER multidiffusion sur sa propre interface monodiffusion avec une réponse 302 (Déplacement temporaire).

Les registraires DOIVENT ignorer le champ d'en-tête Record-Route s'il est inclus dans une demande REGISTER. Les registraires NE DOIVENT PAS inclure un champ d'en-tête Record-Route dans une réponse à une demande REGISTER.

Un registraire peut recevoir une requête qui a traversé un mandataire qui traite REGISTER comme une requête inconnue et a ajouté une valeur de champ d'en-tête Record-Route.

Un registraire a à connaître (par exemple, au moyen de la configuration) l'ensemble du ou des domaines pour lesquels il maintient des liens. Des demandes REGISTER DOIVENT être traitées par un registraire dans l'ordre dans lequel elles ont été reçues. Des demandes REGISTER DOIVENT aussi être traitées une par une, c'est-à-dire qu'une demande REGISTER particulière est, soit traitée complètement, soit pas du tout. Chaque message REGISTER message DOIT être traité indépendamment de tout autre changement d'enregistrement ou de lien.

Lors de la réception d'une demande REGISTER, un registraire suit ces étapes :

1. Le registraire inspecte l'URI-de-demande pour déterminer si il a accès aux liens pour le domaine identifié dans l'URI-de-demande. Sinon, et si le serveur agit aussi comme un serveur mandataire, le serveur DEVRAIT transmettre la demande au domaine figurant dans l'adresse, suivant le comportement général pour le mandatement de messages décrit à la Section 16.

2. Pour garantir que le registraire prend en charge toutes les extensions nécessaires, le registraire DOIT traiter les valeurs de champ d'en-tête Require comme décrit pour les UAS au paragraphe 8.2.2.

3. Un registraire DEVRAIT authentifier l'UAC. Les mécanismes pour l'authentification des agents d'utilisateur SIP sont décrits à la Section 22. Le comportement d'enregistrement n'outrepasse en aucune façon le cadre générique d'authentification de SIP. Si aucun mécanisme d'authentification n'est disponible, le registraire PEUT prendre l'adresse From comme identité certifiée de l'origine de la demande.

4. Le registraire DEVRAIT déterminer si l'utilisateur authentifié est autorisé à modifier les enregistrements pour cette adresse-d'enregistrement. Par exemple, un registraire peut consulter une base de données d'autorisation qui transpose les noms d'utilisateurs en une liste d'adresses-d'enregistrement dont cet utilisateur a l'autorisation de modifier les liens. Si l'utilisateur authentifié n'est pas autorisé à modifier les liens, le registraire DOIT retourner un 403 (Interdit) et sauter les étapes restantes. Dans les architectures qui prennent en charge l'enregistrement par un tiers, une entité peut être responsable de la mise à jour des enregistrements associés à des adresses-d'enregistrement multiples.

5. Le registraire extrait l'adresse-d'enregistrement du champ d'en-tête To de la demande. Si l'adresse-d'enregistrement n'est pas valide pour le domaine dans l'URI-de-demande, le registraire DOIT envoyer une réponse 404 (Non trouvé) et sauter les étapes restantes. L'URI DOIT alors être converti en une forme canonique. Pour faire cela, tous les paramètres d'URI DOIVENT être retirés (y compris les paramètres d'utilisateur), et tout caractère formaté DOIT être converti en sa forme non formatée. Le résultat sert d'index dans la liste des liens.

6. Le registraire vérifie si la demande contient le champ d'en-tête Contact. Si elle ne le contient pas, il saute à la dernière étape. Si le champ d'en-tête Contact est présent, le registraire vérifie si il y a une valeur de champ Contact qui

contient la valeur particulière "*" et un champ Expires. Si la demande a des champs Contact additionnels ou un temps d'expiration autre que zéro, la demande est invalide, et le serveur DOIT retourner un 400 (Requête invalide) et sauter les étapes suivantes. Si non, le registraire vérifie si le Call-ID est en accord avec la valeur mémorisée pour chaque lien. Sinon, il DOIT retirer le lien. S'il n'est pas d'accord, il ne DOIT retirer le lien que si le CSeq dans la demande est supérieur à la valeur mémorisée pour ce lien. Autrement, la mise à jour DOIT être avortée et la demande échoue.

7. Le registraire traite ensuite à tour de rôle chaque adresse de contact dans le champ d'en-tête Contact. Pour chaque adresse, il détermine l'intervalle d'expiration comme suit :
- Si la valeur du champ a un paramètre "expires", cette valeur DOIT être prise comme l'expiration demandée.
- Si il n'y a pas de tel paramètre, mais que la demande a un champ d'en-tête Expires, cette valeur DOIT être prise comme l'expiration demandée.
- S'il n'y a ni l'un ni l'autre, une valeur par défaut configurée localement DOIT être prise comme l'expiration demandée.

Le registraire PEUT choisir une expiration plus courte que l'intervalle d'expiration demandé. Si et seulement si l'intervalle d'expiration demandé est supérieur à zéro ET plus petit qu'une heure ET inférieur à un minimum configuré par le registraire, le registraire PEUT rejeter l'enregistrement avec une réponse de 423 (Intervalle trop bref). Cette réponse DOIT contenir un champ d'en-tête Min-Expires qui établit l'intervalle d'expiration minimum que le registraire veut honorer. Il saute alors les étapes restantes.

Permettre au registraire de régler l'intervalle d'enregistrement le protège contre des rafraîchissements d'enregistrement excessivement fréquents tout en limitant l'état où il doit les maintenir et diminue la probabilité d'avoir des enregistrements obsolètes. L'intervalle d'expiration d'un enregistrement est fréquemment utilisé dans la création de services. Un exemple est un service de renvoi d'appel, où l'utilisateur peut seulement être disponible à un terminal pour une brève période. Donc, les registraires devraient accepter des enregistrements brefs ; une requête ne devrait être rejetée que si l'intervalle est si court que le rafraîchissement dégraderait les performances du registraire.

Pour chaque adresse, le registraire cherche ensuite la liste des liens en cours en utilisant les règles de comparaisons d'URI. Si le lien n'existe pas, il fait une tentative d'ajout. Si le lien existe, le registraire vérifie la valeur de Call-ID. Si la valeur de Call-ID dans le lien existant diffère de la valeur de Call-ID dans la demande, le lien DOIT être retiré si le délai d'expiration est zéro, et mis à jour autrement. Si elles sont les mêmes, le registraire compare la valeur de CSeq. Si la valeur est supérieure à celle du lien existant, il DOIT mettre à jour ou retirer le lien comme ci-dessus. Sinon, la mise à jour DOIT être abandonnée et la demande échoue.

Cet algorithme assure que les demandes qui ne sont pas dans l'ordre en provenance du même UA sont ignorées.

Chaque enregistrement de lien enregistre les valeurs de Call-ID et de CSeq à partir de la demande.

La mise à jour de lien DOIT être engagée (c'est-à-dire, rendue visible au mandataire ou serveur redirecteur) si et seulement si toutes les mises à jour et ajouts de liens réussissent. Si l'une d'elles échoue (par exemple, à cause de la défaillance de l'engagement de la base de données arrière), la demande DOIT échouer avec une réponse 500 (Erreur du serveur) et toutes les tentatives de mise à jour de lien DOIVENT être retirées.

8. Le registraire retourne une réponse 200 (OK). La réponse DOIT contenir les valeurs de champ d'en-tête Contact qui énumèrent tous les liens en cours. Chaque valeur Contact DOIT afficher un paramètre "expires" qui indique son intervalle d'expiration choisi par le registraire. La réponse DEVRAIT inclure un champ d'en-tête Date.

11 Interrogation des capacités

La méthode SIP OPTIONS permet à un agent d'utilisateur d'interroger un autre UA ou un serveur mandataire sur ses capacités. Ceci permet à un client de découvrir des informations sur les méthodes prises en charge, les types de contenu, les extensions, les codecs, etc. sans "appeler" l'autre partie. Par exemple, avant qu'un client n'insère un champ d'en-tête Require dans une liste INVITE une option dont il n'est pas certain que l'UAS de destination la prenne en charge, le client peut interroger l'UAS de destination avec une OPTIONS pour voir si cette option est retournée dans un champ d'en-tête Supported. Tous les agents d'utilisateur DOIVENT prendre en charge la méthode OPTIONS.

La cible de la demande OPTIONS est identifiée par l'URI-de-demande, qui pourrait identifier un autre agent d'utilisateur ou un Serveur SIP. Si OPTIONS est adressé à un serveur mandataire, l'URI-de-demande est établi sans partie utilisateur, de la même façon que l'URI-de-demande est établi pour une demande REGISTER.

Autrement, un serveur qui reçoit une demande OPTIONS avec une valeur de champ d'en-tête Max- Forwards de 0

PEUT répondre à la demande sans considération de l'URI-de-demande.

Ce comportement est commun avec HTTP/1.1. Il peut être utilisé comme une fonctionnalité "traceroute" pour vérifier les capacités des serveurs individuels de bond en envoyant une série de demandes OPTIONS avec des valeurs de Max-Forwards incrémentées.

Comme c'est le cas pour le comportement général d'agent d'utilisateur, la couche transaction peut retourner une erreur de fin de temporisation si OPTIONS ne donne pas de réponse. Cela peut indiquer que la cible n'est pas joignable et donc indisponible.

Une demande OPTIONS PEUT être envoyée au titre d'un dialogue établi pour interroger l'homologue sur les capacités qui peuvent être utilisées ultérieurement dans le dialogue.

11.1 Construction d'une demande OPTIONS

Une demande OPTIONS est construite en utilisant les règles standard pour une demande SIP comme exposé au paragraphe 8.1.1.

Un champ d'en-tête Contact PEUT être présent dans une demande OPTIONS.

Un champ d'en-tête Accept DEVRAIT être inclus pour indiquer le type de corps de message que l'UAC souhaite recevoir dans la réponse. Normalement, il est réglé à un format qui est utilisé pour décrire les capacités de support d'un agent d'utilisateur, comme SDP (application/sdp).

La réponse à une demande OPTIONS est supposée être adressée à l'URI-de-demande dans la demande originelle. Cependant, c'est seulement lorsqu'une OPTIONS est envoyée au titre d'un dialogue établi qu'il est garanti que les demandes futures seront reçues par le serveur qui a généré la réponse OPTIONS.

Exemple de demande OPTIONS :

```
        OPTIONS sip:carol@chicago.com SIP/2.0
        Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKhjhs8ass877
Max-Forwards: 70
To: <sip:carol@chicago.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:alice@pc33.atlanta.com>
        Accept: application/sdp
Content-Length: 0
```

11.2 Traitement des demandes OPTIONS

La réponse à OPTIONS est construite en utilisant les règles standard pour une réponse SIP, comme exposé au paragraphe 8.2.6. Le code de réponse choisi DOIT être le même que celui qui aurait été choisi si la demande avait été INVITE. C'est à dire, un 200 (OK) serait retourné si l'UAS est prêt à accepter un appel, un 486 (Occupé) serait retourné si l'UAS est occupé, etc. Cela permet à une demande OPTIONS d'être utilisée pour déterminer l'état de base d'un UAS, qui peut être une indication sur l'acceptation d'une demande INVITE par l'UAS.

Une demande OPTIONS reçue au sein d'un dialogue génère une réponse 200 (OK) qui est identique à celle construite en-dehors d'un dialogue et n'a aucun impact sur le dialogue.

Cette utilisation d'OPTIONS a des limitations dues aux différences dans le traitement des demandes OPTIONS et INVITE par le mandataire. Alors qu'un INVITE "fourchu" peut avoir pour résultat le retour de plusieurs réponses 200 (OK), un OPTIONS "fourchu" n'aura pour résultat qu'une seule réponse 200 (OK), car il est traité par les mandataires en utilisant le traitement non-INVITE. Voir au paragraphe 16.7 les détails normatifs.

Si la réponse à un OPTIONS est générée par un serveur mandataire, le mandataire retourne un 200 (OK), en faisant la liste des capacités du serveur. La réponse ne contient pas de corps de message.

Les champs d'en-tête Accept, Accept-Encoding, Accept-Language, et Supported DEVRAIENT être présents dans une réponse 200 (OK) à une demande OPTIONS. Si la réponse est générée par un mandataire, le champ d'en-tête Allow DEVRAIT être omis comme étant ambigu car un mandataire est ignorant de la méthode. Le champ d'en-tête Contact PEUT être présent dans une réponse 200 (OK) et avoir la même sémantique que dans une réponse 3xx. C'est-à-dire qu'ils peuvent faire une liste de l'ensemble des noms et méthodes de remplacement pour atteindre l'utilisateur. Un champ d'en-tête Warning PEUT être présent.

Un corps de message PEUT être envoyé, dont le type est déterminé par le champ d'en-tête Accept dans la demande OPTIONS (application/sdp est par défaut si le champ d'en-tête Accept n'est pas présent). Si les types en incluent un qui peut décrire les capacités de support, l'UAS DEVRAIT inclure un corps dans la réponse à ce sujet. Des précisions sur la construction d'un tel corps dans le cas d'application/sdp sont données en [13].

Exemple de réponse OPTIONS générée par un UAS (correspondant à la demande du paragraphe 11.1) :

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKhjhs8ass877;received=192.0.2.4
To: <sip:carol@chicago.com>;tag=93810874
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:carol@chicago.com>
Contact: <mailto:carol@chicago.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
Accept: application/sdp
Accept-Encoding: gzip
Accept-Language: en
Supported: foo
Content-Type: application/sdp
Content-Length: 274
(SDP non indiqué)
```

12 Dialogues

Un concept clé pour un agent d'utilisateur est celui de dialogue. Un dialogue représente une relation SIP d'homologue à homologue qui dure un certain temps entre deux agents d'utilisateur. Le dialogue facilite le séquençage des messages entre les agents d'utilisateur et un acheminement appropriés des demandes deux à deux. Le dialogue représente un contexte dans lequel interpréter les messages SIP. La section 8 expose le traitement d'agent d'utilisateur indépendant de la méthode pour des demandes et réponses en dehors d'un dialogue. La présente section expose la façon dont des demandes et réponses sont utilisées pour construite un dialogue, et dont des demandes et réponses ultérieures sont envoyées au sein d'un dialogue.

Un dialogue est identifié à chaque agent d'utilisateur avec un identifiant de dialogue, qui consiste en une valeur d'identifiant d'appel, une étiquette locale et une étiquette distante. L'ID de dialogue à chaque agent d'utilisateur impliqué dans le dialogue n'est pas le même. Précisément, l'étiquette locale à un agent d'utilisateur est identique à l'étiquette distante à l'agent d'utilisateur homologue. Les étiquettes sont des jetons opaques qui facilitent la création d'ID de dialogue univoques.

Un ID de dialogue est aussi associé à toutes les réponses et à toute demande qui contient une étiquette dans le champ To. Les règles pour le calcul de l'ID de dialogue d'un message dépendent de si l'élément SIP est un UAC ou un UAS. Pour un UAC, la valeur Call-ID de l'ID de dialogue est réglée au Call-ID du message, l'étiquette distante est réglée à l'étiquette dans le champ To du message, et l'étiquette locale est réglée à l'étiquette du champ From du message (ces règles s'appliquent à la fois aux demandes et aux réponses). Comme on pourrait s'y attendre pour un UAS, la valeur de Call-ID de l'ID de dialogue est réglée au Call-ID du message, l'étiquette distante est réglée à l'étiquette du champ From du message, et l'étiquette locale est réglée à l'étiquette du champ To du message.

Un dialogue contient certains éléments d'état nécessaires pour les retransmissions ultérieures du message au sein du dialogue. Cet état consiste en l'ID de dialogue, un numéro de séquence local (utilisé pour ordonner les demandes provenant de l'agent d'utilisateur vers son homologue), un numéro de séquence distant (utilisé pour ordonner les demandes provenant de l'homologue vers l'agent d'utilisateur), un URI local, un URI distant, une cible distante, une étiquette booléenne appelée "secure"*(sécurisé)*, et l'ensemble des routes, qui est une liste ordonnée d'URI. L'ensemble des routes est la liste des serveurs qui doivent être traversés pour envoyer une demande à l'homologue. Un dialogue

peut aussi être dans le stade "early" *(précoce)*, qui survient lorsqu'il est créé avec une réponse provisoire, et passe ensuite à l'état "confirmed" *(confirmé)*lorsque une réponse finale 2xx arrive. Pour les autres réponses, ou si aucune réponse n'arrive du tout pour ce dialogue, le dialogue précoce se termine.

12.1 Création d'un dialogue

Les dialogues sont créés avec des méthodes spécifiques au moyen de réponses de non-échec à des demandes. Dans la présente spécification, seules les réponses 2xx et 101-199 avec une étiquette To, lorsque la demande était INVITE, vont établir un dialogue. Un dialogue établi par une réponse non finale à une demande est dans l'état "précoce" et est appelée un dialogue précoce. Des extensions PEUVENT définir d'autres moyens de création de dialogues. La Section 13 donne des précisions qui sont spécifiques de la méthode INVITE. On décrit ici le processus de création de l'état de dialogue qui ne dépend pas de la méthode.

Les agents d'utilisateur DOIVENT allouer des valeurs aux composants d'ID de dialogue comme décrit ci-dessous.

12.1.1 Comportement de l'UAS

Lorsqu'un UAS répond à une demande avec une réponse qui établit un dialogue (comme une 2xx à INVITE), l'UAS DOIT copier toutes les valeurs de champs d'en-tête Record-Route provenant de la demande dans la réponse (y compris les URI, les paramètres d'URI, et tous paramètres de champs d'en-tête Record-Route, qu'ils soient connus ou inconnus de l'UAS) et DOIT maintenir l'ordre de ces valeurs. L'UAS DOIT ajouter un champ d'en-tête Contact à la réponse. Le champ d'en-tête Contact contient une adresse à laquelle l'UAS aimerait être contacté pour des demandes ultérieures dans le dialogue (qui incluent le ACK pour une réponse 2xx dans le cas d'un INVITE). Généralement, la portion hôte de cet URI est l'adresse IP ou FQDN de l'hôte. L'URI fourni dans le champ d'en-tête Contact DOIT être un URI SIP ou SIPS. Si la demande qui a initialisé le dialogue contenait un URI SIPS dans l'URI-de-demande ou dans la plus haute valeur de champ d'en-tête Record-Route, s'il en est une, ou dans le champ d'en-tête Contact s'il n'y avait pas de champ d'en-tête Record-Route, le champ d'en-tête Contact dans la réponse DOIT être un URI SIPS. L'URI DEVRAIT avoir une portée mondiale (c'est-à-dire, que le même URI peut être utilisé dans des messages en-dehors de ce dialogue). De la même façon, la portée de l'URI dans le champ d'en-tête Contact de INVITE n'est pas limité non plus à ce dialogue. Il peut donc être utilisé dans des messages à l'UAC même en-dehors de ce dialogue.

L'UAS construit ensuite l'état du dialogue. Cet état DOIT être maintenu pour la durée du dialogue.

Si la demande est arrivée sur TLS, et si l'URI-de-demande contenait un URI SIPS, le fanion "secure" est établi à VRAI.

L'ensemble des routes DOIT être mis dans la liste des URI dans le champ d'en-tête Record-Route provenant de la demande, pris dans l'ordre et préservant tous les paramètres d'URI. Si aucun champ d'en-tête Record-Route n'est présent dans la demande, l'ensemble des routes DOIT être mis à l'ensemble vide. Cet ensemble des routes, même vide, prend le pas sur tout ensemble des routes préexistant pour les demandes à venir dans ce dialogue. La cible distante DOIT être réglée à l'URI provenant du champ d'en-tête Contact de la demande.

Le numéro de séquence distant DOIT être mis à la valeur du numéro de séquence du champ d'en-tête CSeq de la demande. Le numéro de séquence local DOIT être vide. Le composant d'identifiant d'appel de l'ID de dialogue DOIT être mis à la valeur du Call-ID de la demande. Le composant d'étiquette locale de l'ID de dialogue DOIT être réglé à l'étiquette dans le champ To dans la réponse à la demande (qui inclut toujours une étiquette), et le composant d'étiquette distante de l'ID de dialogue DOIT être réglé à l'étiquette provenant du champ From dans la demande. Un UAS DOIT être prêt à recevoir une demande sans étiquette dans le champ From, auquel cas l'étiquette est considérée comme ayant une valeur de NULL.

Ceci est destiné à maintenir la compatibilité amont avec la RFC 2543, qui ne prévoyait pas d'étiquettes From.

L'URI distant DOIT être établi à l'URI dans le champ From, et l'URI local DOIT être établi à l'URI dans le champ To.

12.1.2 Comportement de l'UAC

Lorsqu'un UAC envoie une demande qui peut établir un dialogue (comme un INVITE) il DOIT fournir un URI SIP ou SIPS avec portée mondiale (par exemple, le même URI SIP peut être utilisé dans des messages en-dehors de ce dialogue) dans le champ d'en-tête Contact de la demande. Si la demande a une valeur de champ d'en-tête URI-de-demande ou une Route préférée avec un URI SIPS, le champ d'en-tête Contact DOIT contenir un URI SIPS.

Lorsqu'un UAC reçoit une réponse qui établit un dialogue, il construit l'état du dialogue. Cet état DOIT être maintenu pour la durée du dialogue.

Si la demande a été envoyée sur TLS, et si l'URI-de-demande contenait un URI SIPS, le fanion "secure" est mis à VRAI.

L'ensemble des routes DOIT être la liste des URI du champ d'en-tête Record-Route provenant de la réponse, prise dans l'ordre inverse et en préservant tous les paramètres d'URI. Si aucun champ d'en-tête Record-Route n'est présent dans la réponse, l'ensemble des routes DOIT être réglé à l'ensemble vide. Cet ensemble des routes, même vide, prend le pas sur tout ensemble des routes préexistant pour des demandes ultérieures dans ce dialogue. La cible distante DOIT être réglée à l'URI provenant du champ d'en-tête Contact de la réponse.

Le numéro de séquence local DOIT être mis à la valeur du numéro de séquence dans le champ d'en-tête CSeq de la demande. Le numéro de séquence distant DOIT être vide (il est établi lorsque l'agent d'utilisateur distant envoie une demande au sein du dialogue). Le composant d'identifiant d'appel de l'ID de dialogue DOIT être réglé à la valeur du Call-ID qui est dans la demande. Le composant d'étiquette locale de l'ID de dialogue DOIT être réglé à l'étiquette dans le champ From qui est dans la demande, et le composant d'étiquette distante de l'ID de dialogue DOIT être réglé à l'étiquette dans le champ To de la réponse. Un UAC DOIT être prêt à recevoir une réponse sans étiquette dans le champ To, auquel cas l'étiquette est considérée comme ayant une valeur nulle.

Ceci est pour le maintien de la compatibilité amont avec la RFC 2543, qui ne prévoyait pas d'étiquettes To.

L'URI distant DOIT être réglé à l'URI dans le champ To, et l'URI local DOIT être réglé à l'URI dans le champ From.

12.2 Demandes au sein d'un dialogue

Une fois qu'un dialogue a été établi entre deux agents d'utilisateur, l'un d'eux PEUT initialiser les nouvelles transactions nécessitées par le dialogue. L'agent d'utilisateur qui envoie la demande tiendra le rôle d'UAC pour la transaction. L'agent d'utilisateur recevant la demande jouera le rôle de l'UAS. Noter que ces rôles peuvent être différents des rôles que jouent les agents d'utilisateur durant la transaction qui a établi le dialogue.

Les demandes dans un dialogue PEUVENT contenir des champs d'en-tête Record-Route et Contact. Cependant, ces demandes ne causent pas la modification de l'ensemble des routes du dialogue, bien qu'elles puissent modifier l'URI de la cible distante. Précisément, les demandes qui ne sont pas des demandes de rafraîchissement de la cible ne modifient pas l'URI de la cible distante du dialogue, et des demandes qui sont des demandes de rafraîchissement de la cible le font. Pour les dialogues qui ont été établis avec un INVITE, la seule demande de rafraîchissement de la cible définie est re-INVITE (voir à la section 14). D'autres extensions peuvent définir des demandes différentes de rafraîchissement de la cible pour les dialogues établis d'autres façons.

Noter qu'un ACK N'EST PAS une demande de rafraîchissement de la cible.

Les demandes de rafraîchissement de la cible ne mettent à jour que l'URI de la cible distante du dialogue, et pas l'ensemble des routes formées à partir de Record-Route. La mise à jour de ce dernier introduirait de sévères problèmes de compatibilité amont avec les systèmes conformes à la RFC 2543.

12.2.1 Comportement de l'UAC

12.2.1.1 Génération de la demande

Une demande dans un dialogue est construite en utilisant de nombreux composants de l'état mémorisé au titre du dialogue.

L'URI dans le champ To de la demande DOIT être réglé à l'URI distant provenant de l'état du dialogue. L'étiquette dans le champ d'en-tête To de la demande DOIT être réglé à l'étiquette distante de ID de dialogue. L'URI From de la demande DOIT être réglé à l'URI local provenant de l'état du dialogue. L'étiquette dans le champ d'en-tête From de la demande DOIT être réglée à l'étiquette locale de l'ID de dialogue. Si la valeur des étiquettes distante ou locale est nulle, le paramètre d'étiquette DOIT être omis respectivement du champ d'en-tête To ou From.

L'utilisation de l'URI provenant des champs To et From dans la demande originelle au sein des demandes ultérieures est faite pour la compatibilité amont avec la RFC 2543, qui utilisait l'URI pour l'identification du dialogue. Dans la présente spécification, seules les étiquettes sont utilisées pour l'identification du dialogue. On s'attend à ce que l'obligation de refléter les URI d'origine de To et From dans les demandes à mi-dialogue soit déconseillé dans une révision future de la présente spécification.

Le Call-ID de la demande DOIT être réglé au Call-ID du dialogue. Au sein d'un dialogue, les demandes DOIVENT contenir des numéros de séquence CSeq à accroissement strictement monotone et contigus (par pas de un) dans chaque direction (excepté ACK et CANCEL, bien sûr, dont les numéros sont égaux à la demande dont on accuse réception ou qu'on annule). Donc, si le numéro de séquence n'est pas vide, la valeur du numéro de séquence local DOIT être incrémentée de un, et cette valeur DOIT être placée dans le champ d'en-tête CSeq. Si le numéro de séquence local est vide, une valeur initiale DOIT être choisie en utilisant les lignes directrices du paragraphe 8.1.1.5. Le champ méthode dans la valeur de champ d'en-tête CSeq DOIT correspondre à la méthode de la demande.

Avec une longueur de 32 bits, un client pourrait générer, au sein d'un seul appel, une demande par seconde pendant environ 136 ans avant de devoir revenir à zéro. La valeur initiale du numéro de séquence est choisie de telle sorte que des demandes ultérieures dans le même appel ne puissent faire le tour du compteur. Une valeur initiale différente de zéro permet aux clients d'utiliser un numéro de séquence initial fondé sur l'heure. Un client peut, par exemple, choisir les 31 bits de plus fort poids d'une seconde d'horloge de 32 bits comme numéro de séquence initial.

L'UAC utilise la cible distante et l'ensemble des routes pour construire les champs d'en-tête URI-de-demande et Route de la demande.

Si l'ensemble des routes est vide, l'UAC DOIT placer l'URI de la cible distante dans l'URI-de-demande. L'UAC NE DOIT PAS ajouter un champ d'en-tête Route à la demande.

Si l'ensemble des routes n'est pas vide, et si le premier URI dans l'ensemble des routes contient le paramètre lr (voir au paragraphe 19.1.1), l'UAC DOIT placer l'URI de la cible distante dans l'URI-de-demande et DOIT inclure un champ d'en-tête Route contenant les valeurs de l'ensemble des routes, dans l'ordre, en incluant tous les paramètres.

Si l'ensemble des routes n'est pas vide, et si son premier URI ne contient pas le paramètre lr, l'UAC DOIT placer le premier URI provenant de l'ensemble des routes dans l'URI-de-demande, en écartant tout paramètre non autorisé dans un URI-de-demande. L'UAC DOIT ajouter un champ d'en-tête Route contenant le reste des valeurs de l'ensemble des routes, dans l'ordre, en incluant tous les paramètres. L'UAC DOIT alors placer l'URI de la cible distante dans le champ d'en-tête Route en dernière valeur.

Par exemple, si la cible distante est sip:user@remoteua et si l'ensemble des routes contient :

<sip:proxy1>,<sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>

La demande sera formée avec les champs d'en-tête URI-de-demande et Route suivants :

METHOD sip:proxy1
Route: <sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>,<sip:user@remoteua>

Si le premier URI de l'ensemble des routes ne contient pas le paramètre lr, le mandataire indiqué ne comprendra pas les mécanismes d'acheminement décrits dans ce document et agira comme spécifié dans la RFC 2543, remplaçant l'URI-de-demande par la première valeur de champ d'en-tête Route qu'il reçoit lors de la retransmission du message. En plaçant l'URI-de-demande à la fin du champ d'en-tête Route, on préserve les informations contenues dans cet URI-de-demande à travers les routeurs stricts (elles sont retournées dans l'URI-de-demande lorsque la demande atteint un routeur souple).

Un UAC DEVRAIT inclure un champ d'en-tête Contact dans toute demande de rafraîchissement de la cible au sein d'un dialogue, et à moins qu'il n'y ait besoin de le changer, l'URI DEVRAIT être le même que celui utilisé dans les demandes précédentes au sein du dialogue. Si le fanion "secure" est mis à vrai, cet URI DOIT être un URI SIPS.
Comme exposé au paragraphe 12.2.2, un champ d'en-tête Contact dans une demande de rafraîchissement de la cible met à jour l'URI de la cible distante. Cela permet à un agent d'utilisateur de fournir une nouvelle adresse de contact, même si cette adresse change pendant la durée du dialogue.

Cependant, les demandes qui ne sont pas des demandes de rafraîchissement de cible n'affectent pas l'URI de la cible distante pour le dialogue.

Le reste de la demande est formé comme indiqué au paragraphe 8.1.1.

Une fois que la demande a été construite, l'adresse du serveur est calculée et la demande envoyée, en utilisant les mêmes procédures que pour des demandes en-dehors d'un dialogue (paragraphe 8.1.2).

Les procédures du paragraphe 8.1.2 vont normalement avoir pour résultat l'envoi de la demande à l'adresse indiquée par la valeur de champ d'en-tête Route la plus élevée ou par l'URI-de-demande si aucun champ d'en-tête Route n'est présent. Sous réserve de certaines restrictions, elles permettent à la demande d'être envoyée à une adresse de remplacement (telle qu'un mandataire extérieur par défaut non représenté dans l'ensemble des routes).

12.2.1.2 Traitement des réponses

L'UAC va recevoir des réponses à la demande de la part de la couche transaction. Si le client de transaction retourne un dépassement de temporisation, c'est traité comme une réponse 408 (Fin de temporisation de la demande).

Le comportement d'un UAC qui reçoit une réponse 3xx pour une demande envoyée au sein d'un dialogue est le même que si la demande avait été envoyée en-dehors d'un dialogue. Ce comportement est décrit au paragraphe 8.1.3.4.

Noter cependant, que lorsque l'UAC essaye des localisations de remplacement, il utilise toujours l'ensemble des routes pour le dialogue pour construire l'en-tête Route de la demande.

Lorsqu'un UAC reçoit une réponse 2xx à une demande de rafraîchissement de la cible, il DOIT remplacer l'URI de cible distante du dialogue par l'URI provenant du champ d'en-tête Contact de cette réponse, si elle est présente.

Si la réponse à une demande au sein d'un dialogue est une 481 (L'appel/transaction n'existe pas) ou une 408 (Fin de temporisation de demande), l'UAC DEVRAIT terminer le dialogue. Un UAC DEVRAIT aussi terminer un dialogue si aucune réponse n'est reçue pour la demande (le client de transaction devrait informer l'utilisateur de transaction de cette fin de temporisation.)

Pour les dialogues initialisés par INVITE, la terminaison du dialogue consiste en l'envoi d'un BYE.

12.2.2 Comportement de l'UAS

Les demandes envoyées au sein d'un dialogue, comme toutes les autres demandes, sont non sécables. Si une demande particulière est acceptée par l'UAS, tous les changements d'état qui lui sont associés sont effectués. Si la demande est rejetée, aucun changement d'état n'est effectué.

Noter que certaines des demandes, telles que les INVITE, affectent plusieurs états.

L'UAS va recevoir la demande de la couche transaction. Si la demande a une étiquette dans le champ d'en-tête To, le cœur de l'UAS calcule l'identifiant de dialogue correspondant à la demande et le compare avec les dialogues existants. Si il y a une correspondance, c'est une demande de mi-dialogue. Dans ce cas, l'UAS applique d'abord les mêmes règles de traitement que pour des demandes en-dehors d'un dialogue, exposées au paragraphe 8.2.

Si la demande a une étiquette dans le champ d'en-tête To, mais que l'identifiant de dialogue ne correspond à aucun des dialogues existants, l'UAS peut avoir connu une défaillance et s'être redémarré, ou il peut avoir reçu une demande pour un UAS différent (qui a pu échouer) (les UAS peuvent construire les étiquettes To de telle sorte qu'un UAS puisse identifier que l'étiquette était pour un UAS pour lequel il fournit la récupération). Une autre possibilité est que la demande entrante ait été simplement mal acheminée. Sur la base de l'étiquette To, l'UAS PEUT accepter ou rejeter la demande. Accepter la demande pour les étiquettes To acceptables donne de la robustesse, de sorte que les dialogues peuvent persister même en cas de défaillance. Les agents d'utilisateur qui souhaitent prendre en charge cette capacité doivent tenir compte de quelques problèmes comme le choix de numéros de séquence CSeq à accroissement monotone même à travers les réamorçages, la reconstruction de l'ensemble des routes, et l'acceptation des horodatages et numéros de séquence RTP hors gamme.

Si l'UAS souhaite rejeter la demande parce qu'il ne souhaite pas recréer le dialogue, il DOIT répondre à la demande par un code d'état 481 (Appel/Transaction n'existe pas) et passer cela au serveur de transaction.

Les demandes qui ne changent en aucune façon l'état d'un dialogue peuvent être reçues au sein d'un dialogue (par

exemple, une demande OPTIONS). Elles sont traitées comme si elles avaient été reçues en-dehors du dialogue.

Si le numéro de séquence distant est vide, il DOIT être réglé à la valeur du numéro de séquence dans la valeur de champ d'en-tête CSeq dans la demande. Si le numéro de séquence n'était pas vide, mais que le numéro de séquence de la demande est inférieur au numéro de séquence distant, la demande est déclassée et DOIT être rejetée avec une réponse 500 (Erreur interne du serveur). Si le numéro de séquence distant n'était pas vide, et si le numéro de séquence de la demande est supérieur au numéro de séquence distant, la demande est en ordre. Il est possible que le numéro de séquence CSeq soit supérieur au numéro de séquence distant de plus de un. Ce n'est pas une condition d'erreur, et un UAS DEVRAIT être prêt à recevoir et traiter des demandes avec des valeurs de CSeq supérieures de plus de un à la demande reçue précédente. L'UAS DOIT alors mettre le numéro de séquence distant à la valeur du numéro de séquence de la valeur de champ d'en-tête CSeq de la demande.

Si un mandataire conteste une demande générée par l'UAC, l'UAC doit resoumettre la demande avec des accréditifs. La demande resoumise aura un nouveau numéro de CSeq. L'UAS ne verra jamais la première demande, et donc, il remarquera un trou dans l'espace des numéros de CSeq. Un tel trou ne représente aucune condition d'erreur.

Lorsqu'un UAS reçoit une demande de rafraîchissement de la cible, il DOIT remplacer l'URI de cible distante du dialogue par l'URI provenant du champ d'en-tête Contact de cette demande, si il est présent.

12.3 Terminaison d'un dialogue

Indépendamment de la méthode, si une demande en-dehors d'un dialogue génère une réponse finale non-2xx, tout dialogue précoce créé au moyen de réponses provisoires à cette demande est terminé. Le mécanisme pour terminer les dialogues confirmés est spécifique de la méthode. Dans la présente spécification, la méthode BYE termine une session et le dialogue qui lui est associé. Voir les précisions à la Section 15.

13 Initialisation d'une session

13.1 Généralités

Lorsqu'un client d'agent d'utilisateur désire initialiser une session (par exemple, audio, vidéo, ou un jeu), il formule une demande INVITE. La demande INVITE demande au serveur d'établir une session. Cette demande peut être transmise par des mandataires, arrivant finalement à un ou plusieurs UAS qui peuvent accepter l'invitation. Ces UAS auront fréquemment besoin d'interroger l'utilisateur pour savoir s'il accepte l' invitation. Après un certain temps, ces UAS peuvent accepter l'invitation (ce qui signifie l'établissement de la session) en envoyant une réponse 2xx.
Si l'invitation n'est pas acceptée, une réponse 3xx, 4xx, 5xx ou 6xx est envoyée, selon la raison du rejet. Avant d'envoyer une réponse finale, l'UAS peut aussi envoyer des réponses provisoires (1xx) pour aviser l'UAC des progrès du contact avec l'utilisateur appelant.

Après avoir pu recevoir une ou plusieurs réponses provisoires, l'UAC va obtenir une ou plusieurs réponses 2xx ou une réponse finale non-2xx. A cause de la durée prolongée que peut prendre la réception des réponses finales à INVITE, les mécanismes de fiabilité des transactions INVITE diffèrent de celles des autres demandes (comme OPTIONS). Une fois qu'il reçoit une réponse finale, l'UAC doit envoyer un ACK pour chaque réponse finale qu'il reçoit. La procédure d'envoi de cet ACK dépend du type de réponse. Pour les réponses finales entre 300 et 699, le traitement de l'ACK est effectué dans la couche transaction et suit un ensemble de règles (voir la Section 17). Pour les réponses 2xx, le ACK est généré par l'UAC central.

Une réponse 2xx à un INVITE établit une session, et elle crée aussi un dialogue entre l'agent d'utilisateur qui a produit l'INVITE et l'agent d'utilisateur qui a généré la réponse 2xx. Donc, lorsque plusieurs réponses 2xx sont reçues en provenance de différents agents d'utilisateur distants (à cause d'un INVITE "fourchu"), chaque 2xx établit un dialogue différent. Tous ces dialogues font partie du même appel.

La présente section donne des précisions sur l'établissement d'une session en utilisant INVITE. Un agent d'utilisateur qui prend en charge INVITE DOIT aussi prendre en charge ACK, CANCEL et BYE.

13.2 Traitement de l'UAC

13.2.1 Création de l'INVITE initial

Comme l'INVITE initial représente une demande en dehors d'un dialogue, sa construction suit les procédures du paragraphe 8.1.1. Un traitement supplémentaire est nécessaire pour le cas spécifique d'INVITE.

Un champ d'en-tête Allow (paragraphe 20.5) DEVRAIT être présent dans INVITE. Il indique quelles méthodes peuvent être invoquées dans un dialogue, lorsque l'agent d'utilisateur envoie l'INVITE, pour la durée du dialogue. Par exemple, un agent d'utilisateur capable de recevoir des demandes INFO au sein d'un dialogue [34] DEVRAIT inclure un champ d'en-tête Allow faisant la liste de la méthode INFO.

Un champ d'en-tête Supported (paragraphe 20.37) DEVRAIT être présent dans INVITE. Il énumère toutes les extensions comprises par l'UAC.

Un champ d'en-tête Accept (paragraphe 20.1) PEUT être présent dans INVITE. Il indique quels types de contenu sont acceptables pour l'agent d'utilisateur, à la fois dans la réponse qu'il reçoit, et dans toutes demandes ultérieures qui lui sont envoyées au sein des dialogues établis par l'INVITE. Le champ d'en-tête Accept est particulièrement utile pour indiquer la prise en charge des divers formats de description de session.

L'UAC PEUT ajouter un champ d'en-tête Expires (paragraphe 20.19) pour limiter la validité de l'invitation. Si l'heure indiquée dans le champ d'en-tête Expires est atteint et qu'il n'a pas été reçu de réponse finale à l'INVITE, l'UAC central DEVRAIT générer une demande CANCEL pour l'INVITE, conformément à la Section 9.

Un UAC PEUT aussi trouver utile d'ajouter, entre autres, les champs d'en-tête Subject (paragraphe 20.36), Organization (paragraphe 20.25) et User-Agent (paragraphe 20.41). Ils contiennent tous des informations qui se rapportent à INVITE.

L'UAC PEUT choisir d'ajouter un corps de message à l'INVITE. Le paragraphe 8.1.1.10 traite de la façon de construire les champs d'en-tête -- Content-Type entre autres – nécessaires pour décrire le corps de message.

Il y a des règles particulières pour les corps de message qui contiennent une description de session - leur Content-Disposition correspondant est "session". SIP utilise un modèle d'offre/réponse où un agent d'utilisateur envoie une description de session, appelée l'offre, qui contient une proposition de description de la session. L'offre indique les moyens de communication désirés (audio, vidéo, jeux), les paramètres de ces moyens (tels que les types de codec) et les adresses pour les supports de réception de la part de celui qui répond. L'autre agent d'utilisateur répond par une autre description de session, appelée la réponse, qui indique quels moyens de communication sont acceptés, les paramètres qui s'appliquent à ces moyens, et les adresses pour les supports de réception de la part de l'offreur. Un échange offre/réponse se fait dans le contexte d'un dialogue, de sorte que si une INVITE SIP résulte en plusieurs dialogues, chacun est un échange offre/réponse séparé. Le modèle offre/réponse définit des restrictions sur le moment où peuvent être faites les offres et les réponses (par exemple, on ne peut pas faire une nouvelle offre tant qu'il y en a une en cours). Il en résulte des restrictions sur l'endroit où les offres et les réponses peuvent apparaître dans les messages SIP. Dans la présente spécification, les offres et les réponses ne peuvent apparaître que dans des demandes et réponses INVITE, et ACK. L'utilisation des offres et réponses est encore limitée. Pour la transaction initiale INVITE, les règles sont :

- L'offre initiale DOIT être dans une INVITE ou, si elle n'y est pas, dans le premier message de non échec fiable provenant de l'UAS en retour à l'UAC. Dans la présente spécification, c'est la réponse finale 2xx.
- Si l'offre initiale est une INVITE, la réponse DOIT être dans un message de non échec fiable de UAS en retour à l'UAC qui est corrélé à cette INVITE. Pour la présente spécification, c'est seulement la réponse finale 2xx à cette INVITE. La même réponse exacte PEUT aussi être placée dans toute réponse provisoire envoyée avant la réponse. L'UAC DOIT traiter la première description de session qu'il reçoit comme la réponse, et DOIT ignorer toute description de session dans les réponses suivantes à l'INVITE initiale.
- Si l'offre initiale est dans le premier message de non échec fiable de l'UAS en retour à l'UAC, la réponse DOIT être dans l'accusé de réception à ce message (dans la présente spécification, ACK pour une réponse 2xx).
- Après avoir envoyé ou reçu une réponse à la première offre, l'UAC PEUT générer des offres ultérieures dans des demandes sur la base des règles spécifiées pour cette méthode, mais seulement si il a reçu des réponses à toute offre précédente, et n'a pas envoyé d'autre offres pour laquelle il n'ait pas reçu de réponse.
- Une fois que l'UAS a envoyé ou reçu une réponse à l'offre initiale, il NE DOIT PAS générer d'offre ultérieure dans une réponse à l'INVITE initiale. Cela signifie qu'un UAS se fondant sur la présente spécification seule ne peut jamais générer d'offre ultérieure jusqu'à l'achèvement de la transaction initiale.

Concrètement, les règles ci-dessus spécifient deux échanges pour les agents d'utilisateur conformes à la présente spécification seule – l'offre est dans l'INVITE, et la réponse dans le 2xx (et peut aussi être dans une 1xx, avec la même

valeur), ou l'offre est dans la 2xx, et la réponse est dans le ACK. Tous les agent d'utilisateurs qui prennent en charge INVITE DOIVENT prendre en charge ces deux échanges.

Le protocole de description de session (SDP) (RFC 2327 [1]) DOIT être pris en charge par tous les agents d'utilisateur comme moyen de décrire les sessions, et son utilisation pour la construction des offres et réponses DOIT suivre les procédures définies en [13].

Les restrictions au modèle d'offre-réponse décrites ci-dessus ne s'appliquent qu'aux objets dont la valeur de champ d'en-tête Disposition-de-contenu est "session". Donc, il est possible que l'INVITE et l'ACK contiennent tous deux un corps de message (par exemple, l'INVITE porte une photo (Disposition-de-contenu : rendu) et l'ACK une description de session (Disposition-de-contenu : session)).

Si le champ d'en-tête Disposition-de-contenu manque, les objets de Type-de-contenu application/sdp impliquent la disposition "session", alors que les autres types de contenu impliquent "rendu".

Une fois que l'INVITE a été créé, l'UAC suit les procédures définies pour l'envoi des demandes en-dehors d'un dialogue (Section 8). Il en résulte une construction de transaction client qui va finalement envoyer la demande et livrer les réponses à l'UAC.

13.2.2 Traitement des réponses à INVITE

Une fois que l'INVITE a été passée au client de transaction INVITE, l'UAC attend les réponses à l'INVITE. Si le client de transaction INVITE retourne une fin de temporisation plutôt qu'une réponse, le TU agit comme si une réponse 408 (Demande hors limite) avait été reçue, comme indiqué au paragraphe 8.1.3.

13.2.2.1 Réponses 1xx

Zéro, une ou plusieurs réponses provisoires peuvent arriver avant qu'une ou plusieurs réponses finales ne soient reçues. Les réponses provisoires à une demande INVITE peuvent créer des "dialogues précoces". Si une réponse provisoire a une étiquette dans le champ To, et si l'identifiant de dialogue de la réponse ne correspond pas à un dialogue existant, il en est construit une en utilisant les procédures définies au paragraphe 12.1.2.

Le dialogue précoce ne sera nécessaire que si l'UAC a besoin d'envoyer une demande à son homologue au sein du dialogue avant l'achèvement de la transaction INVITE initiale. Les champs d'en-tête présents dans une réponse provisoire sont applicables tant que le dialogue est dans l'état précoce (par exemple, un champ d'en-tête Allow dans une réponse provisoire contient les méthodes qui peuvent être utilisées dans le dialogue alors qu'il est dans l'état précoce).

13.2.2.2 Réponses 3xx

Une réponse 3xx peut contenir une ou plusieurs valeurs de champ d'en-tête Contact fournissant les nouvelles adresses où l'appelé pourrait être joint. Selon le code d'état de la réponse 3xx (voir au paragraphe 21.3), l'UAC PEUT choisir d'essayer les nouvelles adresses.

13.2.2.3 Réponses 4xx, 5xx et 6xx

Une seule réponse finale non-2xx peut être reçue pour l'INVITE. Des réponses 4xx, 5xx et 6xx peuvent contenir une valeur de champ d'en-tête Contact qui indique la localisation où des informations supplémentaires sur l'erreur peuvent être trouvées. Des réponses fiables ultérieures (qui n'arriveraient que dans des conditions d'erreur) DOIVENT être ignorées.

Tous les dialogues précoces sont considérés comme terminés à réception de la réponse finale non-2xx.

Après avoir reçu la réponse finale non-2xx, l'UAC central considère que la transaction INVITE est terminée. Le client de transaction INVITE traite la création des ACK pour la réponse (voir au paragraphe 17).

13.2.2.4 Réponses 2xx

Plusieurs réponses 2xx peuvent arriver à l'UAC pour une seule demande INVITE du fait du mandataire "fourchu". Chaque réponse est distinguée par le paramètre d'étiquette dans le champ d'en-tête To, et chacune représente un dialogue distinct, avec un identifiant de dialogue distinct.

Si l'identifiant de dialogue dans la réponse 2xx correspond à l'identifiant de dialogue pour un dialogue existant, le dialogue DOIT être passé à l'état "confirmé", et l'ensemble des routes pour le dialogue DOIT être recalculé sur la base de la réponse 2xx en utilisant les procédures du paragraphe 12.2.1.2. Autrement, un nouveau dialogue dans l'état "confirmé" DOIT être construit en utilisant les procédures du paragraphe 12.1.2.

Noter que seulement une partie des états qui sont recalculés est l'ensemble des routes. Les autres parties de l'état, comme le plus haut numéro de séquences (distant et local) envoyées au sein du dialogue, ne sont pas recalculées. L'ensemble des routes seul est recalculé pour la compatibilité amont. La RFC 2543 ne rendait pas obligatoire la réflexion du champ d'en-tête Record-Route dans une réponse 1xx, et seulement dans une réponse 2xx. Cependant, on ne peut pas mettre à jour la totalité de l'état du dialogue car les demandes de mi-dialogue peuvent avoir été envoyées au sein du dialogue précoce, modifiant le numéro de séquences, par exemple.

L'UAC central DOIT générer une demande ACK pour chaque réponse 2xx reçue de la part de la couche transaction. Le champ d'en-tête de l'ACK est construit de la même façon que toute demande envoyée au sein d'un dialogue (voir à la Section 12) avec l'exception de CSeq et des champs d'en-tête se rapportant à l'authentification. Le numéro de séquence du champ d'en-tête CSeq DOIT être le même que celui de l'INVITE dont il est accusé réception, mais la méthode CSeq DOIT être ACK. Le ACK DOIT contenir les mêmes accréditifs que l'INVITE. Si le 2xx contient une offre (sur la base des règles ci-dessus), le ACK DOIT porter une réponse dans son corps. Si l'offre dans la réponse 2xx n'est pas acceptable, l'UAC central DOIT générer une réponse valide dans le ACK et ensuite envoyer immédiatement un BYE.

Une fois que le ACK a été construit, les procédures de [4] sont utilisées pour déterminer l'adresse de destination, le port et le transport. Cependant, la demande est passée à la couche transport directement pour transmission, plutôt qu'à la transaction client. Cela parce que l'UAC central traite les retransmissions de l'ACK, et non la couche transaction. L'ACK DOIT être passé au transport client chaque fois qu'une retransmission de la réponse 2xx finale est déclanchée à l'arrivée de l'ACK.

L'UAC central considère les transactions INVITE comme achevées 64*T1 secondes après la réception de la première réponse 2xx. A ce point des dialogues précoces qui ne sont pas passés à l'établissement de dialogues, ces dialogues sont terminés. Une fois que la transaction INVITE est considérée comme terminée à l'UAC central, il n'est attendu aucune autre réponse 2xx nouvelle.

Si, après l'accusé de réception de toute réponse 2xx à un INVITE, l'UAC ne veut pas continuer ce dialogue, l'UAC DOIT alors terminer le dialogue en envoyant une demande BYE comme indiqué au paragraphe 15.

13.3 Traitement de l'UAS

13.3.1 Traitement de INVITE

L'UAS central va recevoir des demandes INVITE provenant de la couche transaction. Il effectue d'abord les procédures de traitement de la demande du paragraphe 8.2, qui sont appliquées aussi bien pour les demandes à l'intérieur et à l'extérieur d'un dialogue. En supposant que ces étapes de traitement soient achevées sans avoir généré une réponse, l'UAS central effectue les étapes de traitement additionnelles :

1. Si la demande est un INVITE qui contient un champ d'en-tête Expires, l'UAS central établit un temporisateur pour le nombre de secondes indiqué dans la valeur de champ d'en-tête. Lorsque le temporisateur arrive à expiration, l'invitation est considérée comme expirée. Si l'invitation expire avant que l'UAS ait généré une réponse finale, une réponse 487 (Demande terminée) DEVRAIT être générée.
2. Si la demande est demande de mi-dialogue, le traitement indépendant de la méthode décrit au paragraphe 12.2.2 est appliqué d'abord. Il pourrait aussi modifier la session ; la Section 14 fournit des précisions.
3. Si la demande a une étiquette dans le champ d'en-tête To mais que l'identifiant de dialogue ne correspond à aucun des dialogues existants, l'UAS peut avoir eu une panne et avoir redémarré, ou peut avoir reçu une demande pour un UAS différent (qui a pu échouer). Le paragraphe 12.2.2 donne des lignes directrices pour obtenir un comportement fiable dans une telle situation.

Le traitement à partir d'ici suppose que l'INVITE est en-dehors d'un dialogue, et est donc destiné à l'établissement d'une nouvelle session.

L'INVITE peut contenir une description de session, auquel cas l'UAS se présente avec une offre pour cette session. Il est possible que l'utilisateur soit déjà un participant à cette session, bien que l'INVITE soit en-dehors d'un dialogue. Ceci peut survenir lorsqu'un usager est invité à la même conférence multidiffusion par plusieurs autres participants. Si

c'est souhaité, l'UAS PEUT utiliser des identifiants au sein de la description de session pour détecter cette duplication. Par exemple, SDP contient un identifiant de session et un numéro de version dans le champ d'origine (o). Si l'utilisateur est déjà membre de la session, et si les paramètres de session contenus dans la description de session n'ont pas changé, l'UAS PEUT accepter en silence l'INVITE (c'est-à-dire, envoyer une réponse 2xx sans invite à l'utilisateur).

Si l'INVITE ne contient pas de description de session, il est demandé à l'UAS de participer à une session, et l'UAC a demandé que l'UAS fournisse l'offre de session. Il DOIT fournir l'offre dans son premier message fiable de non-échec retourné à l'UAC. Dans la présente spécification, c'est une réponse 2xx à l'INVITE.

L'UAS peut indiquer l'avancement, l'acceptation, la redirection, ou le rejet de l'invitation. Dans tous ces cas, il formule une réponse en utilisant les procédures décrites au paragraphe 8.2.6.

13.3.1.1 Avancement

Si l'UAS n'est pas capable de répondre immédiatement à l'invitation, il peut choisir d'indiquer une forme d'avancement à l'UAC (par exemple, une indication de sonnerie d'un téléphone). Ceci est accompli au moyen d'une réponse provisoire entre 101 et 199. Ces réponses provisoires établissent des dialogues précoces et donc suivent les procédures du paragraphe 12.1.1 en plus de celles du paragraphe 8.2.6. Un UAS PEUT envoyer autant de réponses provisoires qu'il veut. Chacune d'elles DOIT indiquer le même ID de dialogue. Cependant, leur livraison ne sera pas fiable.

Si l'UAS désire une période étendue pour répondre à l'INVITE, il aura besoin de demander une "extension" afin d'empêcher les mandataires d'annuler la transaction. Un mandataire a la faculté d'annuler une transaction lorsqu'il y a un trou de 3 minutes entre les réponses dans une transaction. Pour empêcher l'annulation, l'UAS DOIT envoyer une réponse provisoire non-100 toutes les minutes, pour faire face à la possibilité de réponses provisoires perdues.

Une transaction INVITE peut continuer sur des extensions de durée lorsque l'utilisateur est placé en garde, ou lors d'une interaction avec des systèmes RTPC qui permettent à des communications d'avoir lieu sans qu'il soit répondu à l'appel. Cette dernière est courante dans les systèmes de réponse vocale interactive (IVR, *Interactive Voice Response*).

13.3.1.2 INVITE est redirigé

Si l'UAS décide de rediriger l'appel, une réponse 3xx est envoyée. Une réponse 300 (Choix multiples), 301 (Déplacement permanent) ou 302 (Déplacement temporaire) DEVRAIT contenir un champ d'en-tête Contact contenant un ou plusieurs URI de nouvelles adresses à essayer. La réponse est passée au serveur de transaction INVITE, qui va s'occuper de sa retransmission.

13.3.1.3 INVITE est rejeté

Un scénario courant survient lorsque l'appelé ne veut pas ou n'est pas en mesure d'accepter des appels supplémentaires à ce système de terminaison. Un 486 (Occupé) DEVRAIT être retourné dans un tel scénario. Si l'UAS sait qu'aucun autre système de terminaison ne sera capable d'accepter cet appel, une réponse 600 (Occupé partout) DEVRAIT être envoyée à la place. Cependant, il est peu vraisemblable qu'un UAS soit en général capable de savoir cela, et donc cette réponse ne sera habituellement pas utilisée. La réponse est passée au serveur de transaction de l'INVITE, qui va s'occuper de sa retransmission.

Un UAS rejetant une offre contenant un INVITE DEVRAIT retourner une réponse 488 (Non acceptable ici). Une telle réponse DEVRAIT inclure une valeur de champ d'en-tête Warning expliquant pourquoi l'offre a été rejetée.

13.3.1.4 INVITE est accepté

L'UAS central génère une réponse 2xx. Cette réponse établit un dialogue, et donc suit les procédures du paragraphe 12.1.1 en plus de celles du paragraphe 8.2.6.

Une réponse 2xx à un INVITE DEVRAIT contenir le champ d'en-tête Allow et le champ d'en-tête Supported, et PEUT contenir le champ d'en-tête Accept. Inclure ces champs d'en-tête permet à l'UAC de déterminer les caractéristiques et les extensions acceptées par l'UAS pour la durée de l'appel, sans vérification.

Si la demande INVITE contenait une offre, et si l'UAS n'a pas encore envoyé de réponse, la 2xx DOIT contenir une réponse. Si l'INVITE ne contenait pas d'offre, la 2xx DOIT contenir une offre si l'UAS n'a pas encore envoyé une offre.

Une fois que la réponse a été construite, elle est passée au serveur de transaction d'INVITE. Noter, cependant, que le serveur de transaction d'INVITE sera détruit aussitôt qu'il aura reçu cette réponse finale et l'aura passée au transport. Donc, il est nécessaire de passer périodiquement la réponse directement au transport jusqu'à ce qu'arrive le ACK. La réponse 2xx est passée au transport avec un intervalle qui débute à T1 secondes et double à chaque retransmission jusqu'à ce qu'il atteigne T2 secondes (T1 et T2 sont définis à la Section 17). Les retransmissions de réponses cessent lorsqu'une demande ACK est reçue pour la réponse. Ceci est indépendant des protocoles de transport utilisés pour envoyer la réponse.

Comme 2xx est retransmis de bout en bout, il peut y avoir des sauts entre UAS et UAC qui soient UDP. Pour assurer une livraison fiable à travers ces bonds, la réponse est retransmise périodiquement même si le transport est fiable à l'UAS.

Si le serveur retransmet la réponse 2xx pendant 64*T1 secondes sans recevoir un ACK, le dialogue est confirmé, mais la session DEVRAIT être terminée. Ceci s'accomplit avec un BYE, comme indiqué à la Section 15.

14 Modification d'une session existante

Une demande INVITE réussie (voir à la Section 13) établit à la fois un dialogue entre deux agents utilisateurs et une session en utilisant le modèle offre-réponse. La Section 12 explique comment modifier un dialogue existant en utilisant une demande de rafraîchissement de la cible (par exemple, en changeant l'URI de la cible distante du dialogue). La présente section décrit comment modifier la session réelle. Cette modification peut impliquer de changer les adresses ou les accès, d'ajouter un flux de support, de supprimer un flux de support, et ainsi de suite. Ceci est fait par l'envoi d'une nouvelle demande INVITE au sein même du dialogue qui a établi la session. Une demande INVITE envoyée au sein d'un dialogue existant est appelée une re-INVITE.

Noter qu'une seule re-INVITE peut modifier en même temps le dialogue et les paramètres de la session.

L'appelant ou l'appelé peuvent tous deux modifier une session existante.

Le comportement d'un agent utilisateur à détection d'une défaillance du support est une affaire de politique locale. Cependant, la génération automatique de re-INVITE ou BYE N'EST PAS RECOMMANDÉE pour éviter de submerger le réseau avec du trafic en cas d'encombrement. Dans tous les cas, si ces messages sont envoyés automatiquement, ils DEVRAIENT être envoyés après un certain intervalle aléatoire.

Noter que le paragraphe ci-dessus se réfère à des BYE et re-INVITE générés automatiquement. Si l'utilisateur raccroche en présence d'une défaillance du support, l'agent utilisateur enverra une demande BYE comme d'ordinaire.

14.1 Comportement de l'UAC

Le même modèle d'offre-réponse qui s'applique aux descriptions de session dans des INVITE (paragraphe 13.2.1) s'applique aux re-INVITE. Il en résulte qu'un UAC qui veut ajouter un flux de support va, par exemple, créer une nouvelle offre qui contient ce flux de support, et l'envoyer dans une demande INVITE à son homologue. Il est important de noter que la description complète de session est envoyée, et non simplement les modifications. Cela prend en charge le traitement de session sans état dans divers éléments, et les capacités de récupération sur défaillance. Bien sûr, un UAC PEUT envoyer un re-INVITE sans description de session, auquel cas la première réponse fiable de non-échec au re-INVITE contiendra l'offre (dans la présente spécification, c'est une réponse 2xx).

Si le format de description de session dispose de la capacité de traiter le numéro de version, l'offreur DEVRAIT indiquer que la version de la description de session a changé.

Les To, From, Call-ID, CSeq, et URI-de-demande d'un re-INVITE sont établis suivant les mêmes règles que pour les demandes régulières au sein d'un dialogue existant, décrites à la Section 12.

Un UAC PEUT choisir de ne pas ajouter de champ d'en-tête Alert-Info ou un corps avec la disposition de contenu "alert" pour les re-INVITE parce que les UAS n'alertent normalement pas l'utilisateur à réception d'un re-INVITE.

A la différence d'un INVITE, qui peut être "fourchu", un re-INVITE ne sera jamais "fourchu", et donc ne générera

jamais qu'une seule réponse finale. La raison pour laquelle un re-INVITE n'est jamais "fourchu" est que l'URI-de-demande identifie la cible comme instance d'agent utilisateur avec lequel il a établi le dialogue, plutôt que d'identifier une adresse-d'enregistrement pour l'utilisateur.

Noter qu'un UAC NE DOIT PAS initialiser une nouvelle transaction INVITE au sein d'un dialogue alors qu'une autre transaction INVITE est en cours dans l'une ou l'autre direction.

1. S'il y a une transaction client INVITE en cours, le TU DOIT attendre que la transaction atteigne l'état achevé ou terminé avant d'initialiser le nouvel INVITE.
   S'il y a un serveur de transaction INVITE en cours, le TU DOIT attendre que la transaction atteigne l'état confirmé ou terminé avant d'initialiser le nouvel INVITE.

Cependant, un agent utilisateur PEUT initialiser une transaction régulière alors qu'une transaction INVITE est en cours. Un agent utilisateur PEUT aussi initialiser une transaction INVITE alors qu'une transaction régulière est en cours.

Si un agent utilisateur reçoit une réponse finale non-2xx à un re-INVITE, les paramètres de session DOIVENT rester inchangés, comme si aucun re-INVITE n'avait été produit. Noter que, comme établit au paragraphe 12.2.1.2, si la réponse finale non-2xx est un 481 (L'appel/transaction n'existe pas), ou un 408 (Demande hors limite) ou si aucune réponse n'est reçue du tout pour le re-INVITE (c'est-à-dire qu'une fin de temporisation est retournée par la transaction client INVITE), l'UAC va terminer le dialogue.

Si un UAC reçoit une réponse 491 à un re-INVITE, il DEVRAIT lancer un temporisateur avec une valeur T choisie comme suit :

2. Si l'UAC est le propriétaire du Call-ID de l'ID de dialogue (signifiant qu'il a généré la valeur), T a une valeur choisie de façon aléatoire entre 2,1 et 4 secondes en unités de 10 ms.
   Si l'UAC n'est pas le propriétaire du Call-ID de l'ID de dialogue, T a une valeur choisie de façon aléatoire entre 0 et 2 secondes en unités de 10 ms.

Lorsque le temporisateur arrive à expiration, l'UAC DEVRAIT essayer le re-INVITE une fois de plus, s'il désire toujours qu'intervienne cette modification de session. Par exemple, si l'appel a déjà été raccroché avec un BYE, le re-INVITE n'aura pas lieu.

Les règles d'émission d'un re-INVITE et de génération d'un ACK pour une réponse 2xx à re-INVITE sont les mêmes que pour l'INVITE initial (paragraphe 13.2.1).

14.2 Comportement de l'UAS

Le paragraphe 13.3.1 décrit la procédure pour distinguer les re-INVITE entrants des INVITE entrants initiaux et le traitement d'un re-INVITE pour un dialogue existant.

Un UAS qui reçoit un second INVITE avant qu'il n'envoie la réponse finale à un premier INVITE avec un numéro de séquence CSeq inférieur sur le même dialogue DOIT retourner une réponse 500 (Erreur de serveur interne) au second INVITE et DOIT inclure un champ d'en-tête Retry-After avec une valeur choisie de façon aléatoire entre 0 et 10 secondes.

Un UAS qui reçoit un INVITE sur un dialogue alors qu'un INVITE qu'il a envoyé sur ce dialogue est en cours DOIT retourner une réponse 491 (Demande en cours) à l'INVITE reçu.

Si un agent utilisateur reçoit un re-INVITE pour un dialogue existant, il DOIT vérifier tous les identifiants de version dans la description de session ou, s'il n'y a pas d'identifiant de version, le contenu de la description de session pour voir si il a changé. Si la description de session a changé, l'UAS DOIT ajuster les paramètres de session en conséquence, éventuellement après avoir demandé confirmation à l'utilisateur.

La numérotation des versions de la description de session peut être utilisée pour s'accommoder des capacités de nouveaux arrivants à une conférence, ajouter ou supprimer des supports, ou passer d'une conférence monodiffusion à une conférence multidiffusion.

Si la nouvelle description de session n'est pas acceptable, l'UAS peut la rejeter en retournant une réponse 488 (Non acceptable ici) pour le re-INVITE. Cette réponse DEVRAIT inclure un champ d'en-tête Warning.

Si un UAS génère une réponse 2xx et ne reçoit jamais d'ACK, il DEVRAIT générer un BYE pour terminer le dialogue.

Un UAS PEUT choisir de ne pas générer de réponse 180 (Sonnerie) pour un re-INVITE parce que normalement les UAC ne restituent pas ces informations à l'utilisateur. Pour la même raison, les UAS PEUVENT choisir de ne pas utiliser un champ d'en-tête Alert-Info ou un corps avec une disposition de contenu "alert" dans les réponses à un re-INVITE.

Un UAS qui fournit une offre dans une 2xx (parce que l'INVITE ne contenait pas d'offre) DEVRAIT construire l'offre comme si l'UAS faisait un nouvel appel, sous réserve des contraintes de l'envoi d'une offre qui met à jour une session existante, comme décrit en [13] dans le cas de SDP. Précisément, cela signifie qu'il DEVRAIT inclure autant de formats et types de support que l'agent utilisateur veut en prendre en charge. L'UAS DOIT s'assurer que la description de session se recoupe avec la précédente description de session en formats de support, transports, ou autres paramètres qui exigent la prise en charge par l'homologue. Ceci est pour éviter que l'homologue ne soit obligé de rejeter la description de session. Si, cependant, c'est inacceptable pour l'UAC, il DEVRAIT générer une réponse avec une description de session valide, et envoyer ensuite un BYE pour terminer la session.

15 Terminaison d'une session

La présente section décrit les procédures pour terminer une session établie par SIP. L'état de la session et l'état du dialogue sont en relation très étroite. Lorsqu'une session est initialisée avec un INVITE, chaque réponse 1xx ou 2xx provenant d'un UAS distinct crée un dialogue, et si cette réponse achève l'échange offre/réponse, elle crée aussi une session. Il en résulte que chaque session est "associée" à un seul dialogue – celui qui a eu pour résultat sa création. Si un INVITE initial génère une réponse finale non-2xx, cela termine toutes les sessions (s'il en est) et tous les dialogues (s'il en est) qui ont été créés à travers les réponses à la demande. Grâce à l'achèvement de la transaction, une réponse finale non-2xx empêche aussi la création de sessions ultérieures par suite de l'INVITE. La demande BYE est utilisée pour terminer une session spécifique ou une tentative de session. Dans ce cas, la session spécifique est celle avec l'agent utilisateur homologue sur l'autre côté du dialogue. Lorsqu'un BYE est reçu sur un dialogue, toute session associée à ce dialogue DEVRAIT se terminer. Un agent utilisateur NE DOIT PAS envoyer un BYE en-dehors d'un dialogue. L'agent utilisateur de l'appelant PEUT envoyer un BYE soit pour des dialogues précoces, soit pour des dialogues confirmés et l'agent utilisateur de l'appelé PEUT envoyer un BYE sur les dialogues confirmés, mais NE DOIT PAS envoyer de BYE sur des dialogues précoces.

Cependant, l'agent utilisateur de l'appelé NE DOIT PAS envoyer de BYE sur un dialogue confirmé jusqu'à ce qu'il ait reçu un ACK pour sa réponse 2xx ou jusqu'à ce que le serveur de transaction arrive en fin de temporisation. Si aucune extension SIP n'a défini d'autres états de couche d'application associés au dialogue, le BYE termine aussi le dialogue.

L'impact d'une réponse finale non-2xx à INVITE sur des dialogues et sessions rend intéressante l'utilisation de CANCEL. CANCEL essaye de forcer une réponse non-2xx à l'INVITE (en particulier, une 487). Donc, si un UAC souhaite abandonner entièrement sa tentative d'appel, il peut envoyer un CANCEL. Si l'INVITE a eu pour résultat une ou des réponses finales 2xx à l'INVITE, cela signifie qu'un UAS a accepté l'invitation alors que le CANCEL était en cours. L'UAC PEUT continuer les sessions établies par toute réponse 2xx, ou PEUT les terminer avec BYE.

La notion de "raccroché" n'est pas bien définie dans SIP. C'est spécifique d'une interface d'utilisateur particulière, quoique commune. Normalement, lorsque l'utilisateur raccroche, il indique un désir de terminer la tentative d'établissement d'une session, et de terminer toutes sessions déjà créées. Pour l'agent utilisateur de l'appelant, cela impliquerait une demande CANCEL si l'INVITE initial n'avait pas généré une réponse finale, et un BYE pour tous les dialogues confirmés après une réponse finale. Pour l'agent utilisateur de l'appelé, cela impliquerait normalement un BYE ; vraisemblablement, lorsque l'utilisateur a décroché le téléphone, une 2xx a été générée, et donc le raccroché devrait résulter en un BYE après la réception du ACK. Cela ne signifie pas qu'un utilisateur ne puisse pas raccrocher avant la réception du ACK, cela signifie simplement que le logiciel de son téléphone a besoin de maintenir l'état pendant un court instant afin de faire le ménage. Si l'UI particulier permet à l'utilisateur de rejeter un appel avant qu'il y soit répondu, une réponse 403 (Interdit) est une bonne façon d'exprimer cela. Conformément aux règles ci-dessus, un BYE ne peut pas être envoyé.

15.1 Terminaison d'une session avec une demande BYE

15.1.1 Comportement de l'UAC

Une demande BYE est construite comme le serait toute autre demande au sein d'un dialogue, comme indiqué à la Section 12.

Une fois que le BYE est construit, l'UAC central crée une nouvelle transaction client non-INVITE, et la passe à la demande BYE. L'UAC DOIT considérer la session comme terminée (et donc arrêter d'envoyer ou d'écouter les supports) aussitôt que la demande BYE est passée à la transaction client. Si la réponse au BYE est une 481 (L'appel/transaction n'existe pas) ou une 408 (Fin de temporisation de la demande) ou qu'aucune réponse n'est reçue du tout pour le BYE (c'est-à-dire qu'une fin de temporisation est retournée par la transaction client), l'UAC DOIT considérer la session et le dialogue comme terminés.

15.1.2 Comportement de l'UAS

Un UAS traite d'abord la demande BYE conformément au traitement général d'UAS décrit au paragraphe 8.2. Un UAS central recevant une demande BYE vérifie si elle correspond à un dialogue existant. Si le BYE ne correspond pas à un dialogue existant, l'UAS central DEVRAIT générer une réponse 481 (L'appel/transaction n'existe pas) et passer cela au serveur de transaction.

Cette règle signifie qu'un BYE envoyé sans étiquette par un UAC devra être rejeté. Ceci est un changement par rapport à la RFC 2543, qui permettait BYE sans étiquette.

Un UAS central qui reçoit une demande BYE pour un dialogue existant DOIT suivre les procédures du paragraphe 12.2.2 pour traiter la demande. Une fois que c'est fait, l'UAS DEVRAIT terminer la session (et donc arrêter d'envoyer et d'écouter les supports). Le seul cas où il peut choisir de ne pas le faire est celui des sessions multidiffusion, où la participation est possible même si d'autres participants au dialogue ont terminé leur engagement dans la session. Qu'il ait ou non terminé sa participation à la session, l'UAS central DOIT générer une réponse 2xx au BYE, et DOIT passer cela au serveur de transaction pour transmission.

L'UAS DOIT encore répondre à toutes demandes reçues pour ce dialogue. Il est RECOMMANDÉ qu'une réponse 487 (Demande terminée) soit générée sur les demandes en cours.

16 Comportement du mandataire

16.1 Généralités

Les mandataires SIP sont des éléments qui acheminent les demandes SIP au serveur d'agents utilisateurs et les réponses SIP aux clients d'agents utilisateurs. Une demande peut traverser plusieurs mandataires sur son chemin vers un UAS. Chacun prendra des décisions d'acheminement, modifiant la demande avant de la transmettre au prochain élément. Les réponses seront acheminées par le même ensemble de mandataires traversés par la demande dans l'ordre inverse.

Etre un mandataire est un rôle logique pour un élément SIP. Lorsqu'une demande arrive, un élément qui peut jouer le rôle de mandataire décide d'abord s'il a besoin de répondre de lui-même à la demande. Par exemple, la demande peut être malformée ou l'élément peut avoir besoin d'accréditifs de la part du client avant d'agir comme mandataire. L'élément PEUT répondre avec tout code d'erreur approprié. Lorsqu'il répond directement à une demande, l'élément joue le rôle d'un UAS et DOIT se comporter comme indiqué au paragraphe 8.2.

Un mandataire peut fonctionner soit en mode à états pleins soit en mode sans état pour chaque nouvelle demande. Lorsqu'il est sans état, un mandataire agit comme un simple élément de transmission. Il transmet chaque demande vers l'aval à un seul élément déterminé en prenant une décision de ciblage et d'acheminement sur la base de la demande. Il transmet simplement chaque réponse qu'il reçoit vers l'amont. Un mandataire sans état élimine les informations sur un message une fois que le message a été transmis. Un mandataire à états pleins se souvient des informations (précisément, de l'état de la transaction) sur chaque demande entrante et chacune des demandes qu'il envoie en résultat du traitement de la demande entrante. Il utilise ces informations pour affecter le traitement des messages futurs associés à cette demande. Un mandataire à états pleins PEUT choisir de "fourcher" une demande, l'acheminant sur plusieurs destinations. Toute demande qui est transmise à plus d'une localisation DOIT être traitée à états pleins.

Dans certaines circonstances, un mandataire PEUT faire suivre des demandes en utilisant des transports à états pleins (tels que TCP) sans être à état pleins au niveau des transactions. Par exemple, un mandataire PEUT transmettre une demande provenant d'une connexion TCP à une autre transaction sans état pourvu qu'il place suffisamment d'informations dans le message pour être capable de transmettre la réponse sur la même connexion que celle sur laquelle la demande est arrivée. Les demandes transmises entre différents types de transports où le TU du mandataire doit jouer un rôle actif pour s'assurer d'une livraison fiable sur un des transports DOIVENT être transmises avec des transactions à états pleins.

Un mandataire à états pleins PEUT passer à un fonctionnement sans état à tout moment durant le traitement d'une demande, tant qu'il ne fait rien qui l'empêcherait autrement d'être initialement sans état (fourchement, par exemple, ou

génération d'une réponse 100). Lorsqu'il effectue une telle transition, tous les états sont simplement éliminés. Le mandataire NE DEVRAIT PAS initialiser une demande CANCEL.

La plus grande partie du traitement impliqué pour une demande lors de l'action à état plein ou sans état est identique. Les paragraphes suivants sont écrits du point de vue d'un mandataire à états pleins. Le dernier paragraphe souligne les endroits où un mandataire sans état se comporte différemment.

16.2 Mandataire à états pleins

Lorsqu'il est à états pleins, un mandataire est uniquement un moteur de traitement de transaction SIP. Son comportement est modélisé ici en termes de transactions serveur et client définies à la Section 17. Un mandataire à états pleins a un serveur de transaction associé à une ou plusieurs transactions client par un composant de traitement de mandataire de couche supérieure (voir la Figure 3), connu sous le nom de mandataire central. Une demande entrante est traitée par un serveur de transaction. Les demandes provenant du serveur de transaction sont passées à un mandataire central. Le mandataire central détermine où acheminer la demande, choisissant une ou plusieurs localisations pour le prochain saut. Une demande sortante pour chaque localisation de saut suivant est traitée par sa propre transaction client associée. Le mandataire central collecte les réponses provenant des transactions client et les utilise pour envoyer les réponses au serveur de transaction.

Un mandataire à états pleins crée un nouveau serveur de transaction pour chaque nouvelle demande reçue. Toute retransmission de la demande sera alors traitée par ce serveur de transaction selon la Section 17. Le mandataire central DOIT se comporter comme un UAS par rapport à l'envoi d'une réponse provisoire immédiate sur ce serveur de transaction (comme 100 En essai) comme indiqué au paragraphe 8.2.6. Et donc, un mandataire à états pleins NE DEVRAIT PAS générer de réponses 100 (En essai) à des demandes non INVITE.

Ceci est un modèle de comportement de mandataire, non de logiciel. Une mise en œuvre est libre de prendre toute approche qui reproduit le comportement externe défini par ce modèle.

Pour toutes les nouvelles demandes, y compris celles avec des méthodes inconnues, un élément qui désire agir comme mandataire vis à vis de la demande DOIT :

1. Valider la demande (paragraphe 16.3)
2. Prétraiter les informations d'acheminement (paragraphe 16.4)
3. Déterminer la ou les cibles pour la demande (paragraphe 16.5)

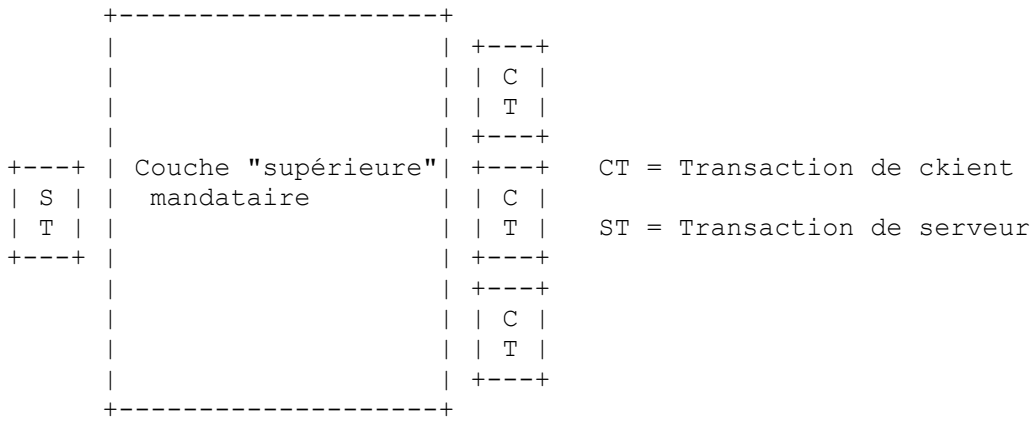


Figure 3 : Modèle de mandataire à états pleins

4. Transmettre la demande à chaque cible (paragraphe 16.6)
5. Traiter toutes les réponses (paragraphe 16.7)

16.3 Validation de la demande

Avant qu'un élément puisse agir comme mandataire pour une demande, il DOIT vérifier la validité du message. Un message valide doit franchir les étapes de vérification suivantes :

1. Une syntaxe raisonnable
2. Le schéma d'URI
3. Le nombre maximum de retransmissions (Max-Forwards)
4. La détection de boucle (facultative)
5. Des exigences pour le mandataire (Proxy-Require)
6. Une autorisation du mandataire (Proxy-Authorization)

Si une de ces vérifications échoue, l'élément DOIT se comporter comme un serveur d'agent utilisateur (voir au paragraphe 8.2) et répondre par un code d'erreur.

Remarquer qu'un mandataire n'est pas obligé de détecter des demandes fusionnées et NE DOIT PAS traiter des demandes fusionnées comme condition d'erreur. Les points d'extrémité qui reçoivent les demandes résoudront la fusion comme indiqué au paragraphe 8.2.2.2.

1. Vérification d'une syntaxe raisonnable
La demande DOIT être suffisamment bien formée pour être traitée avec une transaction de serveur. Tous les composants impliqués dans le reste de ces étapes de validation de demande ou du paragraphe sur la transmission des demandes DOIVENT être bien formées. Tous les autres composants, bien formés ou non, DEVRAIENT être ignorés et rester inchangés lorsque le message est transmis. Par exemple, un élément ne devrait pas rejeter une demande à cause d'un champ d'en-tête Date mal formé. Vraisemblablement, un mandataire ne retirerait pas un champ d'en-tête Date mal formé avant de transmettre une demande.

Ce protocole est conçu pour avoir des extensions Les futures extensions pourront définir à tout moment de nouvelles méthodes et champs d'en-tête. Un élément NE DOIT PAS refuser d'agir comme mandataire pour une demande parce qu'elle contient une méthode ou champ d'en-tête dont il ne sait rien.

2. Vérification du schéma d'URI
Si l'URI-de-demande a un URI dont le schéma n'est pas compris par le mandataire, le mandataire DEVRAIT rejeter la demande avec une réponse 416 (Schéma d'URI non accepté).

3. Vérification de Max-Forwards
Le champ d'en-tête Max-Forwards (paragraphe 20.22) sert à limiter le nombre d'éléments qu'une demande SIP peut traverser.
Si la demande ne contient pas un champ d'en-tête Max-Forwards, cette vérification est réussie.
Si la demande contient un champ d'en-tête Max-Forwards avec une valeur de champ supérieure à zéro, la vérification est réussie.
Si la demande contient un champ d'en-tête Max-Forwards avec une valeur de champ de zéro (0), l'élément NE DOIT PAS transmettre la demande. Si la demande était pour OPTIONS, l'élément PEUT agir comme receveur final et répondre selon la Section 11. Autrement, l'élément DOIT retourner une réponse 483 (Trop de sauts).

4. Vérification facultative de détection de boucle
Un élément PEUT rechercher des boucles de transmission avant de transmettre une demande. Si la demande contient un champ d'en-tête Via avec une valeur émise qui égale une valeur placée dans des demandes précédentes par le mandataire, la demande a été transmise auparavant par cet élément. La demande passe en boucle ou est dans une spirale légitime autour de cet élément. Pour déterminer si la demande est en boucle, l'élément PEUT effectuer le calcul de paramètre de branche décrit à l'étape 8 du paragraphe 16.6, sur ce message, et le comparer au paramètre reçu dans ce champ d'en-tête Via. Si les paramètres correspondent, la demande a effectué une boucle. S'ils diffèrent, la demande effectue une spirale, et le traitement continue. Si une boucle est détectée, l'élément PEUT retourner une réponse 482 (Détection de boucle).

5. Vérification de Proxy-Require
Des extensions du présent protocole pourront à l'avenir introduire des caractéristiques qui exigent un traitement particulier de la part des mandataires. Les points d'extrémité devront inclure un champ d'en-tête Proxy-Require dans des demandes qui utilisent ces caractéristiques, pour dire au mandataire de ne pas traiter la demande à moins que la caractéristique ne soit comprise.
Si la demande contient un champ d'en-tête Proxy-Require (paragraphe 20.29) avec une ou plusieurs étiquettes d'options que cet élément ne comprend pas, l'élément DOIT retourner une réponse 420 (Mauvaise extension). La réponse DOIT inclure un champ d'en-tête Unsupported (paragraphe 20.40) faisant la liste des étiquettes d'option que l'élément n'a pas compris.

6. Vérification de Proxy-Authorization
Si un élément exige des accréditifs avant de transmettre une demande, la demande DOIT être inspectée comme indiqué au paragraphe 22.3. Cette section définit aussi ce que doit faire l'élément si l'inspection échoue.

16.4 Traitement des informations d'acheminement

Le mandataire DOIT inspecter l'URI-de-demande de la demande. Si l'URI-de-demande de la demande contient une

valeur que ce mandataire avait précédemment placée dans un champ d'en-tête Record-Route (voir au paragraphe 16.6 item 4), le mandataire DOIT replacer l'URI-de-demande dans la demande avec la dernière valeur provenant du champ d'en-tête Route, et retirer cette valeur du champ d'en-tête Route. Le mandataire DOIT alors procéder comme si il avait reçu cette demande modifiée.

Cela n'arrivera seulement que lorsque l'élément qui envoie la demande au mandataire (qui peut avoir été un point d'extrémité) est un routeur strict. Cette réécriture à réception est nécessaire pour activer la compatibilité vers l'amont avec ces éléments. Elle permet aussi à des éléments qui suivent la présente spécification de préserver l'URI-de-demande à travers des mandataires à acheminement strict (voir au paragraphe 12.2.1.1).

Cette exigence n'oblige pas un mandataire à garder l'état afin de détecter les URI qu'il avait précédemment placés dans le champ d'en-tête Record-Route. Au lieu de cela, un mandataire a juste besoin de placer des informations suffisantes dans ces URI pour les reconnaître comme des valeurs qu'il a fournies lorsqu'elles apparaissent ultérieurement.

Si l'URI-de-demande contient un paramètre maddr, le mandataire DOIT vérifier pour voir si sa valeur est dans l'ensemble des adresses ou domaines où le mandataire est configuré comme responsable. Si l'URI-de-demande a un paramètre maddr avec une valeur dont le mandataire est responsable, et si la demande a été reçue en utilisant l'accès et le transport indiqués (explicitement ou par défaut) dans l'URI-de-demande, le mandataire DOIT ôter le paramètre maddr et tout paramètre d'accès ou de transport qui n'est pas par défaut et continuer le traitement comme si ces valeurs n'avaient pas été présentes dans la demande.

Une demande peut arriver avec un maddr correspondant au mandataire, mais sur un accès ou transport différent de ceux indiqués dans l'URI. Une telle demande doit être transmise au mandataire en utilisant l'accès et le transport indiqués.

Si la première valeur dans le champ d'en-tête Route indique ce mandataire, le mandataire DOIT retirer cette valeur de la demande.

16.5 Détermination des cibles de la demande

Ensuite, le mandataire calcule la ou les cibles de la demande. L'ensemble des cibles devra être prédéterminé par le contenu de la demande ou devra être obtenu d'un service de localisation abstrait. Chaque cible de l'ensemble est représentée comme un URI.

Si l'URI-de-demande de la demande contient un paramètre maddr, l'URI-de-demande DOIT être placé dans l'ensemble cible comme le seul URI cible, et le mandataire DOIT procéder comme indiqué au paragraphe 16.6.

Si le domaine de l'URI-de-demande indique un domaine dont cet élément n'est pas responsable, l'URI-de-demande DOIT être placé dans l'ensemble cible comme la seule cible, et l'élément DOIT procéder à la tâche de transmission de la demande (paragraphe 16.6).

Il y a de nombreuses circonstances dans lesquelles un mandataire peut recevoir une demande pour un domaine dont il n'est pas responsable. Un mandataire pare-feu traitant des appels sortants (de la façon dont les mandataires HTTP traitent les demandes sortantes) est un exemple de l'endroit où cela a des chances de se produire.

Si l'ensemble cible pour la demande n'a pas été prédéterminé comme décrit ci-dessus, cela implique que l'élément est responsable pour le domaine dans l'URI-de-demande, et l'élément PEUT utiliser le mécanisme qu'il veut pour déterminer où envoyer la demande. Tous ces mécanismes peuvent être modélisés comme accédant à un service de localisation abstrait. Cela peut consister à obtenir des informations de la part d'un service de localisation créé par un registraire SIP, à lire une base de données, à consulter un serveur de présence, à utiliser d'autres protocoles, ou simplement à effectuer une substitution algorithmique sur l'URI-de-demande. Lors de l'accès au service de localisation construit par un registraire, l'URI-de-demande DOIT d'abord être canonisé comme indiqué au paragraphe 10.3 avant d'être utilisé comme indice. Le résultat de ces mécanismes est utilisé pour construire l'ensemble cible.

Si l'URI-de-demande ne fournit pas d'informations suffisantes pour que le mandataire détermine l'ensemble cible, il DEVRAIT retourner une réponse 485 (Ambigu). Cette réponse DEVRAIT contenir un champ d'en-tête Contact contenant les URI des nouvelles adresses à essayer. Par exemple, un INVITE à sip:John.Smith@company.com peut être ambigu pour un mandataire dont le service de localisation a plusieurs John Smith sur ses listes. Voir des précisions au paragraphe 21.4.23.

Toute information dans ou sur la demande ou l'environnement actuel de l'élément PEUT être utilisée dans la

construction de l'ensemble cible. Par exemple, différents ensembles peuvent être construits selon le contenu ou la présence des champs d'en-tête et corps, l'heure du jour de l'arrivée de la demande, l'interface sur lequel arrive la demande, l'échec des demandes précédentes, ou même du niveau courant d'utilisation de l'élément.

Comme les cibles potentielles sont situées à travers ces services, leurs URI sont ajoutés à l'ensemble cible. Les cibles peuvent seulement être placées une fois dans l'ensemble cible. Si un URI cible est déjà présent dans l'ensemble (sur la base de la définition d'égalité du type d'URI), il NE DOIT PAS être ajouté à nouveau.

Un mandataire NE DOIT PAS ajouter de cibles supplémentaires à l'ensemble cible si l'URI-de-demande de la demande originelle n'indique pas une ressource dont ce mandataire est responsable.

Un mandataire peut seulement changer l'URI-de-demande d'une demande durant sa transmission si il est responsable de cet URI. Si le mandataire n'est pas responsable de cet URI, il ne reviendra pas sur des réponses 3xx ou 416 comme décrit ci-dessous.

Si l'URI-de-demande de la demande originelle indique une ressource dont ce mandataire est responsable, le mandataire PEUT continuer à ajouter des cibles à l'ensemble après le début de la transmission de la demande. Il PEUT utiliser toute information obtenue durant ce traitement pour déterminer de nouvelles cibles. Par exemple, un mandataire peut choisir d'incorporer des contacts obtenus dans une réponse redirect (3xx) dans l'ensemble cible. Si un mandataire utilise une source d'informations dynamique alors qu'il est en train de construire l'ensemble cible (par exemple, si il consulte un registraire SIP), il DEVRAIT surveiller cette source pendant la durée du traitement de la demande. De nouvelles localisations DEVRAIENT être ajoutées à l'ensemble cible au fur et à mesure qu'elles deviennent disponibles. Comme ci-dessus, aucun URI donné NE DOIT être ajouté plus d'une fois à l'ensemble.

Permettre à un URI d'être ajouté une seule fois à l'ensemble réduit le trafic réseau inutile, et dans le cas de l'incorporation de contacts provenant de la redirection des demandes, empêche les récurrences infinies.

Par exemple, un service de localisation commun est un "no-op", où l'URI cible est égal à l'URI de la demande entrante. La demande est envoyée à un mandataire spécifique du prochain saut pour le traitement ultérieur. Durant la transmission de demande du paragraphe 16.6, item 6, l'identité de ce prochain saut, exprimé comme un URI SIP ou SIPS, est inséré comme valeur la plus élevée de champ d'en-tête Route dans la demande.

Si l'URI-de-demande indique une ressource qui n'existe pas à ce mandataire, le mandataire DOIT retourner une réponse 404 (Non trouvée).

Si l'ensemble cible reste vide après application de ce qui précède, le mandataire DOIT retourner une réponse d'erreur, qui DEVRAIT être la réponse 480 (Temporairement indisponible).

16.6 Transmission de la demande

Sitôt que l'ensemble cible est non vide, un mandataire PEUT commencer à transmettre la demande. Un mandataire à états pleins PEUT traiter l'ensemble dans n'importe quel ordre. Il PEUT traiter plusieurs cibles en série, permettant à chaque transaction client de se terminer avant de commencer la suivante. Il PEUT commencer des transactions client avec chaque cible en parallèle. Il PEUT aussi diviser arbitrairement l'ensemble en groupes, traitant les groupes en série et traitant les cibles dans chaque groupe en parallèle.

Un mécanisme d'ordonnancement courant est d'utiliser le paramètre qvaleur des cibles obtenues des champs d'en-tête Contact (voir au paragraphe 20.10). Les cibles sont traitées de la plus forte qvaleur à la plus faible. Les cibles avec des qvaleur égales peuvent être traitées en parallèle.

Un mandataire à états pleins doit avoir un mécanisme de maintenance de l'ensemble cible lorsque les réponses sont reçues et associer les réponses à chaque demande transmise avec la demande originelle. Pour les besoins du modèle, ce mécanisme est un "contexte de réponse" créé par la couche mandataire avant la transmission de la première demande.

Pour chaque cible, le mandataire transmet la demande en suivant ces étapes :
1. Faire une copie de la demande reçue
2. Mettre à jour l'URI-de-demande
3. Mettre à jour le champ d'en-tête Max-Forwards
4. Ajouter facultativement une valeur de champ d'en-tête Record-route
5. Ajouter facultativement un champ d'en-tête supplémentaire

6. Post-traiter les informations d'acheminement
7. Déterminer l'adresse, port, et transport du prochain saut
8. Ajouter une valeur de champ d'en-tête Via
9. Ajouter si nécessaire un champ d'en-tête Content-Length
10. Transmettre la nouvelle demande
11. Etablir le temporisateur C

Chacune de ces étapes est détaillée ci-dessous.

1. Copie de la demande
Le mandataire commence par une copie de la demande reçue. La copie DOIT initialement contenir tous les champs d'en-tête provenant de la demande reçue. Les champs non détaillés dans le traitement décrit ci-dessous NE DOIVENT PAS être retirés. La copie DEVRAIT conserver l'ordre des champs d'en-tête comme dans la demande reçue. Le mandataire NE DOIT PAS réordonner les valeurs de champs avec un nom de champ commun (voir au paragraphe 7.3.1). Le mandataire NE DOIT PAS ajouter au corps de message, ni le modifier ou le retirer.

Une mise en œuvre réelle n'a pas besoin d'effectuer une copie ; la principale exigence est que le traitement pour chaque prochain saut commence avec la même demande.

2. URI-de-demande
L'URI-de-demande dans la ligne de début de la copie DOIT être remplacé par l'URI pour cette cible. Si l'URI contient des paramètres non autorisés dans un URI-de-demande, ils DOIVENT être retirés.
Ceci est l'essence du rôle du mandataire. C'est le mécanisme par lequel un mandataire achemine une demande vers sa destination.
Dans certaines circonstances, l'URI-de-demande reçu est placé dans l'ensemble cible sans être modifié. Pour cette cible, le remplacement ci-dessus est effectivement une no-op.

3. Max-Forwards
Si la copie contient un champ d'en-tête Max-Forwards, le mandataire DOIT diminuer sa valeur de un (1).
Si la copie ne contient pas de champ d'en-tête Max-Forwards, le mandataire DOIT en ajouter un avec une valeur de champ qui DEVRAIT être 70.
Certains agents d'utilisateur existants ne fourniront pas de champ d'en-tête Max-Forwards dans une demande.

4. Record-Route
Si ce mandataire souhaite rester sur le chemin des demandes futures dans un dialogue créé par cette demande (en supposant que la demande crée un dialogue), il DOIT insérer une valeur de champ d'en-tête Record-Route dans la copie avant toute valeur de champ d'en-tête Record-Route existant, même si un champ d'en-tête Route est déjà présent.
Les demandes qui établissent un dialogue peuvent contenir un champ d'en-tête Route préchargé.
Si cette demande fait déjà partie d'un dialogue, le mandataire DEVRAIT insérer une valeur de champ d'en-tête Record-Route s'il souhaite rester sur le chemin des demandes futures dans le dialogue. Dans le fonctionnement normal de point d'extrémité, comme indiqué au paragraphe 12, ces valeurs de champs d'en-tête Record-Route n'auront aucun effet sur l'ensemble des routes utilisées par les points d'extrémité.

Le mandataire va rester sur le chemin s'il choisit de ne pas insérer de valeur de champ d'en-tête Record-Route dans des demandes qui font déjà partie d'un dialogue. Cependant, il serait retiré du chemin lorsqu'un point d'extrémité ayant subi un échec reconstitue le dialogue.

Un mandataire PEUT insérer une valeur de champ d'en-tête Record-Route dans toute demande. Si la demande n'initialise pas un dialogue, les points d'extrémité ignoreront la valeur. Voir à la Section 12 les précisions sur la façon dont les points d'extrémité utilisent les valeurs de champ d'en-tête Record-Route pour construire les champs d'en-tête Route.

Chaque mandataire sur le chemin d'une demande choisit indépendamment s'il ajoute une valeur de champ d'en-tête Record-Route - la présence d'un champ d'en-tête Record-Route dans une demande n'oblige pas ce mandataire à ajouter une valeur.

L'URI placé dans la valeur de champ d'en-tête Record-Route DOIT être un URI SIP ou SIPS. Cet URI DOIT contenir un paramètre lr (voir au paragraphe 19.1.1). Cet URI PEUT être différent pour chaque destination à laquelle la demande est transmise. L'URI NE DEVRAIT PAS contenir le paramètre de transport sauf si le mandataire a connaissance (comme dans le cas d'un réseau privé) que le prochain élément vers l'aval qui sera dans le chemin des demandes ultérieures prend en charge ce transport.

L'URI que fournit ce mandataire sera utilisé par un autre élément pour prendre une décision d'acheminement. Ce mandataire, en général, n'a aucun moyen de savoir les capacités de cet élément, de sorte qu'il doit se restreindre aux éléments obligatoires d'une mise en œuvre SIP : les URI SIP et les transports TCP ou UDP.

L'URI placé dans le champ d'en-tête Record-Route DOIT se résoudre aux élément qui l'insèrent (ou un remplaçant convenable) lorsque les procédures de localisation de serveur de [4] lui sont appliquées, de façon que les demandes ultérieures atteignent le même élément SIP. Si l'URI-de-demande contient un URI SIPS, ou si la valeur de champ d'en-tête Route la plus élevée (après le post traitement du point 6) contient un URI SIPS, l'URI placé dans le champ d'en-tête Record-Route DOIT être un URI SIPS. De plus, si la demande n'a pas été reçue sur TLS, le mandataire DOIT insérer un champ d'en-tête Record-Route. De façon similaire, un mandataire qui reçoit une demande sur TLS, mais génère une demande sans URI SIPS dans l'URI-de-demande ou la plus forte valeur de champ d'en-tête Route (après le post traitement du point 6), DOIT insérer un champ d'en-tête Record-Route qui n'est pas un URI SIPS.

Un mandataire sur un périmètre de sécurité doit rester sur ce périmètre tout au long du dialogue.

Si l'URI placé dans le champ d'en-tête Record-Route doit être réécrit lorsqu'il repasse à travers une réponse, l'URI DOIT être suffisamment distinct pour être localisé à ce moment. (La demande peut être en train de faire une spirale passant par ce mandataire, ce qui a pour résultat d'ajouter plus d'une valeur de champ d'en-tête Record-Route). Le point 8 du paragraphe 16.7 recommande un mécanisme pour rendre l'URI suffisamment distinct.

Le mandataire PEUT inclure des paramètres dans la valeur de champ d'en-tête Record-Route. Il leur sera fait écho dans certaines réponses à la demande comme les réponses 200 (OK) à INVITE. De tels paramètres peuvent être utiles pour la conservation de l'état dans le message plutôt que chez le mandataire.

Si un mandataire a besoin d'être sur le chemin d'un type de dialogue quelconque (tel que celui pour enjamber un pare-feu), il DEVRAIT ajouter une valeur de champ d'en-tête Record-Route à chaque demande ayant une méthode qu'il ne comprend pas car cette méthode peut avoir une sémantique de dialogue.

L'URI qu'un mandataire place dans un champ d'en-tête Record-Route n'est valide que pour la durée de vie de tout dialogue créé par la transaction dans laquelle il survient. Un mandataire de dialogue à états pleins, par exemple, PEUT refuser d'accepter les futures demandes avec cette valeur dans l'URI-de-demande après la fin du dialogue. Les mandataires de dialogue à état plein, bien sûr, n'ont aucune idée du moment où le dialogue se termine, mais ils PEUVENT coder suffisamment d'informations dans la valeur pour la comparer avec l'identifiant de dialogue des futures demandes et PEUT rejeter des demandes qui ne correspondent pas à cette information. Les points d'extrémité NE DOIVENT PAS utiliser un URI obtenu d'un champ d'en-tête Record-Route en-dehors du dialogue dans lequel il a été fourni. Voir à la Section 12 des informations complémentaires sur l'utilisation par les point d'extrémité du champ d'en-tête Record-Route.

Le Record-route peut être nécessaire à certains services où le mandataire a besoin d'observer tous les messages d'un dialogue. Cependant, cela ralentit le traitement et obère l'extensibilité et donc les mandataires ne devraient utiliser Record-route que si c'est demandé pour un service précis.

Le processus Record-Route est conçu pour fonctionner avec toute demande SIP qui initialise un dialogue. INVITE est la seule demande de cette sorte dans la présente spécification, mais les extensions au protocole PEUVENT en définir d'autres.

5. Ajout de champs d'en-tête supplémentaires
Le mandataire PEUT ajouter tout autre champ d'en-tête approprié à la copie à ce moment.

6. Post-traitement des informations d'acheminement
Un mandataire PEUT avoir une politique locale qui ordonne qu'une demande visite un ensemble spécifique de mandataires avant d'être livrée à sa destination. Un mandataire DOIT s'assurer que tous ces mandataires sont des routeurs souples. Généralement, ceci ne peut être connu avec certitude que si les mandataires sont au sein du même domaine administratif. Cet ensemble de mandataires est représenté par un ensemble d'URI (dont chacun contient le paramètre lr). Cet ensemble DOIT être poussé dans le champ d'en-tête Route de la copie avant toutes les valeurs existantes, s'il en est. Si le champ d'en-tête Route est absent, il DOIT être ajouté, contenant cette liste des URI.

Si le mandataire a une politique locale qui oblige que la demande visite un mandataire spécifique, une alternative à l'introduction d'une valeur de Route dans le champ d'en-tête Route est de sauter la logique de transmission du point 10 ci-dessous, et à la place d'envoyer simplement la demande à l'adresse, port, et transport de ce mandataire spécifique. Si

la demande a un champ d'en-tête Route, cette alternative NE DOIT PAS être utilisée à moins qu'il soit connu que le prochain mandataire de saut est un routeur souple. Autrement, cette approche PEUT être utilisée, mais le mécanisme d'insertion de Route ci-dessus est préférable pour sa robustesse, sa souplesse, son caractère général et sa cohérence de fonctionnement. De plus, si l'URI-de-demande contient un URI SIPS, TLS DOIT être utilisé pour communiquer avec ce mandataire.

Si la copie contient un champ d'en-tête Route, le mandataire DOIT inspecter l'URI dans sa première valeur. Si cet URI ne contient pas de paramètre lr, le mandataire DOIT modifier la copie comme suit :
- Le mandataire DOIT placer l'URI-de-demande dans le champ d'en-tête Route comme dernière valeur.
- Le mandataire DOIT ensuite placer la première valeur de champ d'en-tête Route dans l'URI-de-demande et retirer cette valeur du champ d'en-tête Route.

Ajouter l'URI-de-demande au champ d'en-tête Route fait partie d'un mécanisme utilisé pour passer des informations dans cet URI-de-demande à travers des éléments à acheminement strict. "Sauter" la première valeur de champ d'en-tête Route dans l'URI-de-demande formate le message de la façon dont un élément à acheminement strict s'attend à le recevoir (avec son propre URI dans l'URI-de-demande et la prochaine localisation à visiter dans la première valeur de champ d'en-tête Route).

7. Déterminer l'adresse, port, et transport du prochain saut
Le mandataire PEUT avoir une politique locale d'envoi de la demande à une adresse, port, et transport IP spécifiques, indépendamment des valeurs de Route et URI-de-demande. Une telle politique NE DOIT PAS être utilisée si le mandataire n'est pas certain que l'adresse, port, et transport IP correspondent à un serveur qui soit un routeur souple. Cependant, ce mécanisme d'envoi de la demande à travers un prochain saut spécifique N'EST PAS RECOMMANDÉ ; au lieu de cela, un champ d'en-tête Route devrait être utilisé à cette fin comme décrit ci-dessus.
En l'absence d'un tel mécanisme de substitution, le mandataire applique comme suit les procédures dont la liste figure en [4] pour déterminer où envoyer la demande. Si le mandataire a reformaté la demande pour l'envoyer à un élément à acheminement strict, comme décrit à l'étape 6 ci-dessus, le mandataire DOIT appliquer ces procédures à l'URI-de-demande de la demande. Autrement, le mandataire DOIT appliquer les procédures à la première valeur dans le champ d'en-tête Route, s'il est présent, et autrement à l'URI-de-demande. Les procédures produiront un ensemble ordonné de tuplets (adresse, port, transport). Indépendamment de l'URI qui est utilisé comme entrée à ces procédures de [4], si l'URI-de-demande spécifie une ressource SIPS, le mandataire DOIT suivre les procédures de [4] comme si l'URI d'entrée était un URI SIPS.

Comme décrit en [4], le mandataire DOIT essayer de livrer le message au premier tuplet de cet ensemble, et continuer à travers l'ensemble dans l'ordre jusqu'à la réussite de la livraison.
Pour chaque tuplet essayé, le mandataire DOIT formater le message conformément au tuplet et envoyer la demande en utilisant une nouvelle transaction client comme précisé aux étapes de 8 à 10.

Comme chaque tentative utilise une nouvelle transaction client, elle représente une nouvelle branche. Et donc, le paramètre branche fourni avec le champ d'en-tête Via inséré dans l'étape 8 DOIT être différent à chaque tentative.

Si la transaction client rapporte l'échec de l'envoi de la demande ou une fin de temporisation de la part de l'automate à états, le mandataire continue avec l'adresse suivante de l'ensemble ordonné. Si l'ensemble ordonné est épuisé, la demande ne peut être transmise à cet élément dans l'ensemble cible. Le mandataire n'a pas besoin de placer quelque chose dans le contexte de réponse, mais agit comme si cet élément de l'ensemble cible avait retourné une réponse finale 408 (Délai de demande expiré).

8. Ajout d'une valeur de champ d'en-tête Via
Le mandataire DOIT insérer une valeur de champ d'en-tête Via dans la copie avant les valeurs de champ d'en-tête Via existantes. La construction de cette valeur suit les mêmes lignes directrices qu'au paragraphe 8.1.1.7. Ceci implique que le mandataire va calculer son propre paramètre de branche, qui devra être mondialement unique pour cette branche, et contenir le témoin de connexion requis. Noter que ceci implique que le paramètre de branche soit différent pour les différentes instances d'une demande spiralée ou bouclée à travers un mandataire.

Les mandataires qui choisissent de détecter les boucles ont une contrainte supplémentaire par la valeur qu'ils utilisent pour la construction du paramètre de branche. Un mandataire qui choisit de détecter les boucles DEVRAIT créer un paramètre de branche séparable en deux parties par la mise en œuvre. La première partie DOIT satisfaire aux contraintes du paragraphe 8.1.1.7 comme décrit ci-dessus. La seconde sert à effectuer la détection de boucle et distinguer les boucles des spirales.

La détection de boucle est effectuée en vérifiant que, lorsqu'une demande retourne au mandataire, les champs qui ont un impact sur le traitement de la demande n'ont pas changé. La valeur placée dans cette partie du paramètre de branche

DEVRAIT refléter tous ces champs (y compris tout champs d'en-tête Route, Proxy-Require et Proxy-Authorization). Ceci est destiné à s'assurer que si la demande est renvoyée au mandataire et qu'un de ces champs change, elle est traitée comme une spirale et non une boucle (voir au paragraphe 16.3). Une façon commune de créer cette valeur est de calculer un hachage cryptographique de l'étiquette To, de l'étiquette From, du champ d'en-tête Call-ID, de l'URI-de-demande de la demande reçue (avant traduction), de l'en-tête Via le plus élevé, et du numéro de séquence provenant du champ d'en-tête CSeq, en plus de tout champ d'en-tête Proxy-Require et Proxy-Authorization qui pourrait être présent. L'algorithme utilisé pour calculer le hachage dépend de la mise en œuvre, mais MD5 (RFC 1321 [35]), exprimé en hexadécimal, est un choix raisonnable. (La Base64 n'est pas admissible pour un jeton.)

Si un mandataire souhaite détecter les boucles, le paramètre "branch" qu'il fournit DOIT dépendre de toutes les informations qui affectent les traitement d'une demande, y compris l'URI-de-demande entrant et tout champ d'en-tête affectant l'admission ou l'acheminement de la demande. Ceci est nécessaire pour distinguer les demandes en boucle des demandes dont les paramètres d'acheminement ont changé avant de revenir à ce serveur.

La méthode de la demande NE DOIT PAS être incluse dans le calcul du paramètre branch. En particulier, les demandes CANCEL et ACK (pour les réponses non 2xx) DOIVENT avoir la même valeur branch que la demande correspondante qu'elles annulent ou dont elles accusent réception. Le paramètre branch est utilisé pour corréler ces demandes au serveur qui les traite (voir au paragraphes 17.2.3 et 9.2).

9. Ajout d'un champ d'en-tête Content-Length si nécessaire
Si la demande va être envoyée au prochain saut en utilisant un transport fondé sur le flux et que la copie ne contient aucun champ d'en-tête Content-Length, le mandataire DOIT en insérer un avec la valeur correcte pour le corps de la demande (voir au paragraphe 20.14).

10. Transmission de la demande
Un mandataire à états pleins DOIT créer une nouvelle transaction client pour cette demande comme indiqué au paragraphe 17.1 et ordonner à la transaction d'envoyer la demande en utilisant l'adresse, le port et le transport déterminés dans l'étape 7.

11. Etablir le temporisateur C
Afin de traiter le cas où une demande INVITE ne génère jamais une réponse finale, le TU utilise un temporisateur qu'on appelle temporisateur C. Le temporisateur C DOIT être établi pour chaque transaction client lorsqu'une demande INVITE est passée par un mandataire. Le temporisateur DOIT être supérieur à 3 minutes. Le paragraphe 16.7, point 2 discute de la façon dont ce temporisateur est mis à jour avec les réponses provisoires, et le paragraphe 16.8 discute du traitement lorsqu'il arrive à expiration.

16.7 Traitement des réponses

Lorsqu'une réponse est reçue par un élément, il essaye d'abord de localiser une transaction client (paragraphe 17.1.3) correspondant à la réponse. S'il n'en trouve aucune, l'élément DOIT traiter la réponse (même si c'est une réponse d'information) comme un mandataire sans état (décrit ci-dessous). S'il trouve une correspondance, la réponse est passée à la transaction client.

Transmettre des réponses pour lesquelles on ne trouve pas de transaction client (ou de façon plus générale on n'a pas de connaissance d'avoir envoyé une demande associée) met à l'épreuve la robustesse. En particulier, cela assure que les réponses 2xx "tardives" à des demandes INVITE sont correctement transmises.

Lorsque les transactions client passent les réponses à la couche mandataire, le processus suivant DOIT avoir lieu :

1. Trouver le contexte de réponse approprié
2. Mettre le temporisateur C à jour pour les réponses provisoires
3. Retirer le Via le plus élevé
4. Ajouter la réponse au contexte de réponse
5. Vérifier si cette réponse devrait être transmise immédiatement
6. Lorsque nécessaire, choisir la meilleure réponse finale d'après le contexte de réponse
   Si aucune réponse finale n'a été transmise après que chaque transaction client associée au contexte de réponse a été terminée, le mandataire doit choisir et transmettre la "meilleure" réponse parmi celles qu'il a vu jusque là. Le traitement suivant DOIT être effectué sur chaque réponse transmise. Il est vraisemblable que plus d'une réponse à chaque demande sera transmise : au moins toutes les provisoires et une réponse finale.
7. Agréger les valeurs de champs d'en-tête d'autorisation si nécessaire
8. Facultativement réécrire les valeurs de champs d'en-tête Record-Route
9. Transmettre la réponse
10. Générer toutes les demandes CANCEL nécessaires.

Chacune des étapes ci-dessus est détaillée ci-dessous :

1. Trouver le contexte
   Le mandataire localise le "contexte de réponse" qu'il a créé avant de transmettre la demande originelle en utilisant la clé décrite au paragraphe 16.6. Les étapes de traitement restantes prennent place dans ce contexte.

2. Mettre à jour le temporisateur C pour les réponses provisoires
   Pour une transaction INVITE, si la réponse est une réponse provisoire avec code d'états 101 à 199 inclus (par exemple, tout sauf 100), le mandataire DOIT remettre le temporisateur C à zéro pour cette transaction client. Le temporisateur PEUT être remis à une valeur différente, mais cette valeur DOIT être supérieure à 3 minutes.

3. Via
   Le mandataire retire la valeur de champ d'en-tête Via la plus élevée de la réponse.
   S'il ne reste aucune valeur de champ d'en-tête Via dans la réponse, la réponse était destinée à cet élément et NE DOIT PAS être transmise. Le reste du processus décrit dans ce paragraphe n'est pas effectué sur de message, et à la place, on suit les règles de traitement d'UAC décrites au paragraphe 8.1.3 (le traitement de couche transport a déjà été effectué). Cela va arriver, par exemple, lorsque l'élément génère les demandes CANCEL comme indiqué section 10.

4. Ajouter la réponse au contexte
   Les réponses finales reçues sont mémorisées dans le contexte de réponse jusqu'à ce qu'une réponse finale soit générée sur le transaction de serveur associée à ce contexte. La réponse peut être une candidate pour la meilleure réponse finale qui sera retournée sur cette transaction de serveur. Les informations tirées de cette réponse peuvent être nécessaires pour former la meilleure réponse, même si cette réponse n'est pas choisie. Si le mandataire choisit de revenir sur un des contacts contenus dans une réponse 3xx en les ajoutant à l'ensemble cible, il DOIT les retirer de la réponse avant d'ajouter la réponse au contexte de réponse. Cependant, un mandataire NE DEVRAIT PAS revenir sur un URI non-SIPS si l'URI-de-demande de la demande originelle était un URI SIPS. Si le mandataire revient sur tous les contacts contenus dans une réponse 3xx, le mandataire NE DEVRAIT PAS ajouter la réponse sans contact résultante au contexte de réponse. Retirer le contact avant d'ajouter la réponse au contexte de réponse empêche le prochain élément amont de réessayer une localisation que ce mandataire a déjà essayée.

   Les réponses 3xx peuvent contenir une mixture d'URI SIP, SIPS, et non-SIP. Un mandataire peut choisir de revenir sur les URI SIP et SIPS et placer le reste dans le contexte de réponse à retourner, potentiellement dans la réponse finale. Si un mandataire reçoit une réponse 416 (Schéma d'URI non accepté) à une demande dont le schéma d'URI-de-demande n'était pas SIP, mais si le schéma dans la demande originale reçue était SIP ou SIPS (c'est-à-dire que le mandataire a changé le schéma de SIP ou SIPS à quelque chose d'autre lorsqu'il a traité une demande), le mandataire DEVRAIT ajouter un nouvel URI à l'ensemble cible. Cet URI DEVRAIT être une version d'URI SIP de l'URI non-SIP qui vient d'être essayé. Dans le cas de l'URL tel, ceci est accompli en plaçant la partie abonné téléphonique de l'URL tel dans la partie utilisateur de l'URI SIP, et en établissant la partie hôte au domaine auquel la demande précédente avait été envoyée. Voir au paragraphe 19.1.6 les détails complémentaires sur la formation des URI SIP à partir des URL tel.

   Comme avec une réponse 3xx, si un mandataire "revient" sur la réponse 416 en essayant à la place un URI SIP ou SIPS, la réponse 416 NE DEVRAIT PAS être ajoutée au contexte de réponse.

5. Vérifier la réponse pour la transmission
   Jusqu'à ce qu'une réponse finale ait été envoyée sur le transaction de serveur, les réponses suivantes DOIVENT être transmises immédiatement :
   - toute réponse provisoire autre que 100 (En essai)
   - toute réponse 2xx.

   Si une réponse 6xx est reçue, elle n'est pas immédiatement transmise, mais le mandataire à états pleins DEVRAIT annuler toutes les transactions client en cours comme indiqué à la Section 10, et il NE DOIT PAS créer de nouvelles branches dans ce contexte.
   Ceci est un changement par rapport à la RFC 2543, qui obligeait que le mandataire transmette immédiatement la réponse 6xx . Pour une transaction INVITE, cette approche posait le problème qu'une réponse 2xx pouvait arriver sur une autre branche, auquel cas le mandataire aurait à transmettre la réponse 2xx. Il en résultait que l'UAC pouvait recevoir une réponse 6xx suivie par une réponse 2xx, ce qui ne devrait jamais être permis. Avec les nouvelles règles, à réception d'une 6xx, un mandataire va produire une demande CANCEL, dont il résultera en général une réponse 487 de la part de toutes les transactions client en instance, et c'est ensuite à ce moment que la 6xx est transmise vers l'amont.
   Après qu'une réponse finale a été envoyée sur le transaction de serveur, les réponses suivantes DOIVENT être

transmises immédiatement :

- Toute réponse 2xx à une demande INVITE

Un mandataire à états pleins NE DOIT PAS transmettre immédiatement toute autre réponse. En particulier un mandataire à états pleins NE DOIT transmettre aucune réponse 100 (En essai). Ces réponses qui sont candidates pour être transmises ultérieurement comme "meilleure" réponse ont été rassemblées comme décrit à l'étape "Ajouter la réponse au contexte".

Toute réponse choisie pour transmission immédiate DOIT être traitée comme décrit de l'étape "Agréger les valeurs de champ d'en-tête d'autorisation" à l'étape "Record-Route".

Cette étape, combinée à la suivante, garantit qu'un mandataire à états pleins transmettra exactement une réponse finale à une demande non-INVITE, et exactement une réponse non-2xx ou une ou plusieurs réponses 2xx à une demande INVITE.

6. Choisir la meilleure réponse

Un mandataire à états pleins DOIT envoyer une réponse finale à la transaction de serveur d'un contexte de réponse si aucune réponse finale n'a été immédiatement transmise selon les règles ci-dessus et si toutes les transactions client dans ce contexte de réponse ont été terminées. Le mandataire à états pleins DOIT choisir la "meilleure" réponse finale parmi celles reçues et mémorisées dans le contexte de réponse.

Si il n'y a aucune réponse finale dans le contexte, le mandataire DOIT envoyer une réponse 408 (Expiration du délai de réponse) à la transaction de serveur.

Autrement, le mandataire DOIT transmettre une réponse à partir des réponses mémorisées dans le contexte de réponse. Il DOIT choisir à partir des réponse de classe 6xx s'il en existe dans le contexte. Si aucune réponse de classe 6xx n'est présente, le mandataire DEVRAIT choisir à partir de la plus basse classe de réponse mémorisée dans le contexte de réponse. Le mandataire PEUT sélectionner toute réponse au sein de la classe choisie. Le mandataire DEVRAIT donner la préférence aux réponses qui fournissent des informations qui affectent la re-soumission de cette demande, telles que 401, 407, 415, 420, et 484 si la classe 4xx est choisie.

Un mandataire qui reçoit une réponse 503 (Service indisponible) NE DEVRAIT PAS la transmettre vers l'amont s'il ne peut pas déterminer qu'une des demandes ultérieures pour laquelle il pourrait être mandataire génèrera aussi une réponse 503. En d'autres termes, retransmettre une 503 signifie que le mandataire sait qu'il ne peut servir aucune des demandes, et pas seulement celle pour l'URI-de-demande est l'URI dans la demande qui a généré la réponse 503. Si la seule réponse reçue est une 503, le mandataire DEVRAIT générer une réponse 500 et la transmettre vers l'amont.

La réponse transmise DOIT être traitée comme décrit à l'étape "Agréger les valeurs de champ d'en-tête d'autorisation" à travers le "Record-Route".

Par exemple, si un mandataire transmet une demande à 4 localisations, et reçoit des réponses 503, 407, 501, et 404, il peut choisir de transmettre la réponse 407 (Authentification du mandataire exigée).

Les réponses 1xx et 2xx peuvent être impliquées dans l'établissement des dialogues. Lorsqu'une demande ne contient pas d'étiquette To, l'étiquette To de la réponse est utilisée par l'UAC pour distinguer plusieurs réponses à un dialogue créant une demande. Un mandataire NE DOIT PAS insérer une étiquette dans le champ d'en-tête To d'une réponse 1xx ou 2xx si la demande n'en contenait pas. Un mandataire NE DOIT PAS modifier l'étiquette dans le champ d'en-tête To d'une réponse 1xx ou 2xx.

Comme un mandataire peut ne pas insérer d'étiquette dans le champ d'en-tête To d'une réponse 1xx à une demande qui n'en contenait pas, il ne peut pas produire de réponses non-100 provisoires de lui même. Cependant, il peut brancher la demande sur un UAS partageant le même élément que le mandataire. Cet UAS peut retourner ses propres réponses provisoires, entrant dans un dialogue précoce avec l'initiateur de la demande. L'UAS n'a pas à être un processus discret de la part du mandataire. Il pourrait être un UAS virtuel mis en œuvre dans le même espace de code que le mandataire.

Les réponses 3-6xx sont livrées saut par saut. Lors de la production d'une réponse 3-6xx, l'élément est agit effectivement comme un UAS, produisant sa propre réponse, habituellement fondée sur les réponses reçues de la part des éléments aval. Un élément DEVRAIT préserver l'étiquette To lorsqu'il transmet simplement une réponse 3-6xx à une demande qui ne contenait pas d'étiquette To.

Un mandataire NE DOIT PAS modifier l'étiquette To dans une réponse transmise à une demande qui contient une étiquette To.

Alors qu'il ne fait aucune différence pour les éléments amont que le mandataire remplace l'étiquette To dans une réponse 3-6xx transmise, préserver l'étiquette originale peut aider au débogage. Lorsque le mandataire agrège les informations provenant de plusieurs réponses, le choix d'une étiquette To parmi elles est arbitraire, et générer une nouvelle étiquette To peut rendre le débogage plus facile. Cela arrive, par exemple, lors du conflit de la combinaison de 401 (Non autorisé) et 407 (Authentification du mandataire exigée), ou de la combinaison des valeurs Contact provenant de réponses 3xx non chiffrées et non authentifiées.

7. Agréger les valeurs de champ d'en-tête d'autorisation

Si la réponse choisie est une 401 (Non autorisé) ou 407 (Authentification du mandataire exigée), le mandataire

DOIT collecter toutes les valeurs de champs d'en-tête WWW-Authenticate et Proxy-Authenticate provenant de toutes les autres réponses 401 (Non autorisé) et 407 (Authentification du mandataire exigée) reçues jusque là dans ce contexte de réponse et les ajouter à cette réponse sans modification avant l'envoi. La réponse 401 (Non autorisé) ou 407 (Authentification du mandataire exigée) qui en résulte pourrait avoir plusieurs valeurs de champ d'en-tête WWW-Authenticate ET Proxy-Authenticate.

Ceci est nécessaire parce que une ou toutes les destinations auxquelles la demande était retransmise peuvent avoir demandé des accréditifs. Le client a besoin de recevoir toutes ces sollicitations et fournir les accréditifs pour chacun d'eux lorsqu'il réessaye la demande. Les motifs de ce comportement sont fournis à la Section 26.

8. Record-Route

Si la réponse choisie contient une valeur de champ d'en-tête Record-Route fournie à l'origine par ce mandataire, le mandataire PEUT choisir de réécrire la valeur avant de transmettre la réponse. Ceci permet au mandataire de fournir différents URI pour lui-même pour les prochains éléments amont et aval. Un mandataire peut choisir d'utiliser ce mécanisme pour n'importe quelle raison. Par exemple, il est utile pour les hôtes multi-domiciles.

Si le mandataire a reçu la demande sur TLS, et l'a renvoyée sur une connexion non-TLS, le mandataire DOIT réécrire l'URI dans le champ d'en-tête Record-Route pour que ce soit un URI SIPS. Si le mandataire a reçu la demande sur une connexion non-TLS, et l'a renvoyée sur TLS, le mandataire DOIT réécrire l'URI dans le champ d'en-tête Record-Route pour qu'il soit un URI SIP.

Le nouvel URI fourni par le mandataire DOIT satisfaire aux mêmes contraintes sur les URI placées dans le champ d'en-tête Record-Route dans des demandes (voir l'étape 4 du paragraphe 16.6) avec les modifications suivantes :

L'URI NE DEVRAIT contenir le paramètre transport QUE si le mandataire a connaissance que le prochain élément amont (par opposition à aval) qui sera sur le chemin des demandes ultérieures prend en charge ce transport.

Lorsqu'un mandataire décide de modifier le champ d'en-tête Record-Route dans la réponse, une des opérations qu'il effectue est de localiser la valeur Record-Route qu'il a insérée. Si la demande est en spirale, et si le mandataire a inséré une valeur Record-Route dans chaque itération de la spirale, la localisation de la valeur correcte dans la réponse (qui doit être la bonne itération dans la direction inverse) est ardu. Les règles ci-dessus recommandent qu'un mandataire qui souhaite réécrire les valeurs de champ d'en-tête Record-Route insère des URI suffisamment distincts dans le champ d'en-tête Record-Route de sorte que le bon puisse être sélectionné pour la réécriture. Un mécanisme RECOMMANDÉ pour y arriver est que le mandataire ajoute un identifiant univoque pour l'instance de mandataire à la partie utilisateur de l'URI.

Lorsque la réponse arrive, le mandataire modifie le premier Record-Route dont l'identifiant correspond à l'instance de mandataire. La modification a pour résultat un URI sans cette portion de données ajoutées à la partie utilisateur de l'URI. Lors de la prochaine itération, le même algorithme (trouver la valeur de champ d'en-tête Record-Route la plus élevée avec le paramètre) extraira correctement la prochaine valeur de champ d'en-tête Record-Route insérée par ce mandataire.

Toutes les réponses à une demande à laquelle un mandataire ajoute une valeur de champ d'en-tête Record-Route ne contiendront pas un champ d'en-tête Record-Route. Si la réponse contient un champ d'en-tête Record-Route, il contiendra la valeur ajoutée par le mandataire.

9. Transmettre la réponse

Après avoir effectué le traitement décrit aux étapes "Agréger les valeurs de champ d'en-tête d'autorisation" jusqu'à "Record-Route", le mandataire PEUT effectuer toute manipulation spécifique sur la réponse choisie. Le mandataire NE DOIT PAS ajouter au corps de message, le modifier, ou le retirer. Sauf spécification contraire, le mandataire NE DOIT PAS retirer d'autres valeurs de champ d'en-tête que la valeur de champ d'en-tête Via exposée au point 3 du paragraphe 16.7. En particulier, le mandataire NE DOIT PAS retirer de paramètre "received" qu'il pourrait avoir ajouté à la prochaine valeur de champ d'en-tête Via lors du traitement de la demande associée à cette réponse. Le mandataire DOIT passer la réponse à la transaction de serveur associée au contexte de réponse. Il en résultera que la réponse sera envoyée à la localisation qui est maintenant indiquée dans la valeur de champ d'en-tête Via la plus élevée. Si la transaction de serveur n'est plus disponible pour traiter la transmission, l'élément DOIT transmettre la réponse sans état en l'envoyant au transport du serveur. La transaction de serveur peut indiquer l'échec de l'envoi de la réponse ou signaler une fin de temporisation dans son automate à états. Ces erreurs devraient être enregistrée de façon appropriée pour l'établissement du diagnostic, mais le protocole ne requiert pas d'action corrective de la part du mandataire.

Le mandataire DOIT maintenir le contexte de réponse jusqu'à ce que toutes ses transactions associées aient été terminées, même après avoir transmis une réponse finale.

10. Générer les CANCEL

Si la réponse transmise était une réponse finale, le mandataire DOIT générer une demande CANCEL pour toutes

les transactions client associées à ce contexte de réponse. Un mandataire DEVRAIT aussi générer une demande CANCEL pour toutes les transactions client en cours associées à ce contexte de réponse lorsqu'il reçoit une réponse 6xx. Une transaction client en cours est celle qui a reçu une réponse provisoire, mais pas de réponse finale (elle est dans l'état en cours de traitement) et n'a pas de CANCEL associé généré pour elle. La génération des demandes CANCEL est décrite au paragraphe 9.1.

L'exigence de l'annulation par CANCEL des transactions client en cours à la transmission d'une réponse finale ne garantit pas qu'un point d'extrémité ne va par recevoir de multiples réponses 200 (OK) à un INVITE. Les réponses 200 (OK) sur plus d'une branche peuvent être générées avant que les demandes CANCEL puissent être envoyées et traitées. De plus, il est raisonnable de s'attendre à ce qu'une future extension se substitue à l'exigence de produire les demandes CANCEL.

16.8 Traitement du temporisateur C

Si le temporisateur C arrive à expiration, le mandataire DOIT soit remettre le temporisateur à zéro avec toute valeur qu'il choisit, soit terminer la transaction client. Si la transaction client a reçu une réponse provisoire, le mandataire DOIT générer une demande CANCEL correspondant à cette transaction. Si la transaction client n'a pas reçu de réponse provisoire, le mandataire DOIT se comporter comme si la transaction avait reçu une réponse 408 (Expiration du délai de réponse).

Autoriser le mandataire à remettre le temporisateur à zéro lui permet d'étendre de façon dynamique la durée de vie de la transaction sur la base des conditions actuelles (comme l'utilisation) lorsque le temporisateur arrive à expiration.

16.9 Traitement des erreurs de transport

Si la couche transport notifie une erreur au mandataire lorsqu'il essaye de transmettre une demande (voir au paragraphe 18.4), le mandataire DOIT se comporter comme si la demande transmise avait reçu une réponse 503 (Service indisponible).

Si le mandataire reçoit une notification d'erreur lors de la transmission d'une réponse, il abandonne la réponse. Le mandataire NE DEVRAIT PAS annuler de transaction client en cours associée à ce contexte de réponse à cause de cette notification.

Si un mandataire annule ses transactions client en cours, un seul client malveillant ou se comportant mal peut provoquer l'échec de toutes les transactions avec son champ d'en-tête Via.

16.10 Traitement de CANCEL

Un mandataire à états pleins PEUT générer une demande CANCEL pour tout autre demande qu'il a généré à tout moment (sous réserve d'avoir reçu une réponse provisoire à cette demande comme décrit au paragraphe 9.1). Un mandataire DOIT annuler toute transaction client en cours associée à un contexte de réponse lorsqu'il reçoit une demande CANCEL correspondante.

Un mandataire à états pleins PEUT générer des demandes CANCEL pour des transactions client INVITE en cours sur la base de la période spécifiée dans le délai d'écoulement du champs d'en-tête Expires de INVITE. Cependant, ceci n'est généralement pas nécessaire car les points d'extrémité impliqués auront soin de signaler la fin de la transaction.

Alors qu'une demande CANCEL est traitée dans un mandataire à états pleins par sa propre transaction de serveur, il n'est pas créé de nouveau contexte de réponse pour elle. A la place, la couche mandataire cherche ses contextes de réponse existants pour la transaction de serveur qui traite la demande associée à ce CANCEL. Si un contexte de réponse correspondant est trouvé, l'élément DOIT immédiatement retourner une réponse 200 (OK) à la demande CANCEL. Dans ce cas, l'élément agit comme un serveur d'agent utilisateur comme défini au paragraphe 8.2. De plus, l'élément DOIT générer les demandes CANCEL pour toutes les transactions client en cours dans le contexte, comme indiqué au paragraphe 16.7, étape 10.

Si un contexte de réponse n'est pas trouvé, l'élément n'a aucune connaissance de la demande à laquelle s'applique le CANCEL. Il DOIT transmettre sans état la demande CANCEL (il peut avoir précédemment transmis sans état la demande associée).

16.11 Mandataire sans état

Lorsqu'il agit sans état, un mandataire est un simple transmetteur de message. Une grande partie du processus effectué lorsqu'il agit sans état est la même que lorsqu'il agit à états pleins. Les différences sont détaillées ci-après.

Un mandataire sans état n'a aucune notion d'une transaction, ou du contexte de la réponse utilisé pour décrire le comportement de mandataire à états pleins. Au lieu de cela, le mandataire sans état prend les messages, à la fois demandes et réponses, directement à la couche transport (voir la section 18). Il en résulte que les mandataires sans état ne retransmettent pas les messages de leur propre chef. Il effectuent, cependant, toutes les retransmissions de ce qu'ils reçoivent (ils n'ont pas la capacité de distinguer une retransmission du message original). De plus, lorsqu'il traite une demande sans état, un élément NE DOIT PAS générer sa propre 100 (En essai) ou toute autre réponse provisoire.

Un mandataire sans état DOIT valider une demande comme indiqué au paragraphe 16.3.

Un mandataire sans état DOIT suivre les étapes de traitement de la demande décrites aux paragraphes 16.4 à 16.5 avec l'exception suivante :

- Un mandataire sans état DOIT choisir une seule cible dans l'ensemble cible. Ce choix DOIT s'appuyer seulement sur les champs contenus dans le message et les propriétés indépendantes du temps du serveur. En particulier, une demande retransmise DOIT être transmise à la même destination chaque fois qu'elle est traitée. De plus, les demandes CANCEL et ACK non routées DOIVENT générer le même choix que leur INVITE associée.

Un mandataire sans état DOIT suivre les étapes de traitement de la demande décrites au paragraphe 16.6 avec les exceptions suivantes :

- L'exigence des identifiants uniques de branche dans l'espace et le temps s'appliquent aussi aux mandataires sans état. Cependant, un mandataire sans état ne peut pas simplement utiliser un générateur de nombres aléatoires pour calculer le premier composant de l'ID de branche, comme indiqué point 8 du paragraphe 16.6. Cela à cause du fait que les retransmissions d'une demande doivent avoir la même valeur, et un mandataire sans état ne peut pas distinguer une retransmission de la demande originelle. Donc, le composant du paramètre branch qui la rend unique DOIT être le même chaque fois qu'une demande retransmise est envoyée. Et donc, pour un mandataire sans état, le paramètre branch DOIT être calculé comme une fonction combinatoire des paramètres invariants en retransmission du message.
  Le mandataire sans état PEUT utiliser toutes les technique qu'il veut pour garantir l'unicité de ses ID de branche dans les transactions. Cependant, la procédure qui suit est RECOMMANDÉE. Le mandataire examine l'ID de branche dans le champ d'en-tête Via le plus élevé de la demande. Si il commence par le cookie magique, le premier composant de l'ID de branche de la demande sortante est calculé comme un hachage de l'ID de branche reçu. Autrement, le premier composant de l'ID de branche est calculé comme un hachage du champ d'en-tête Via le plus élevé, de l'étiquette dans le champ d'en-tête To, de l'étiquette dans le champ d'en-tête From, du champ d'en-tête Call-ID, du numéro CSeq (mais pas la méthode), et de l'URI-de-demande tirés de la demande reçue. Un de ces champs variera toujours d'une transaction à l'autre.

- Toutes les autres transformations de message spécifiée au paragraphe 16.6 DOIVENT avoir pour résultat la même transformation d'une demande retransmise. En particulier, si le mandataire insère une valeur Record-Route ou pousse des URI dans le champ d'en-tête Route, il DOIT placer les mêmes valeurs dans les retransmissions de la demande. Comme pour le paramètre de branche Via, ceci implique que les transformations DOIVENT être fondées sur une configuration invariante dans le temps ou des propriétés de la demande invariantes selon la retransmission.

- Un mandataire sans état détermine où transmettre la demande comme décrit pour les mandataires à état pleins au point 10 du paragraphe 16.6. La demande est envoyée directement à la couche transport au lieu de passer par une transaction client.
  Comme un mandataire sans état doit transmettre les demandes retransmises à la même destination et ajouter des paramètres de branche identiques à chacun d'eux, il peut seulement utiliser les informations provenant du message lui-même et des données de configuration invariantes dans le temps pour ces calculs. Si l'état de configuration n'est pas invariant dans le temps (par exemple, si un tableau d'acheminement est mis à jour) toutes les demandes qui pourraient être affectées par le changement peuvent n'être pas transmises sans état durant un intervalle égal à la fenêtre de temporisation de la transaction avant ou après le changement. La méthode de traitement des demandes affectées pendant cet intervalle est à la décision de la mise en œuvre. Une solution habituelle est de les transmettre comme transactions à états pleins.

Les mandataires sans état NE DOIVENT PAS effectuer de traitement spécial pour les demandes CANCEL. Elles sont traitées selon les règles ci-dessus comme toutes les autres demandes. En particulier, un mandataire sans état applique les mêmes traitements de champ d'en-tête Route aux demandes CANCEL que ceux qu'il applique à toute autre demande.

Le traitement des réponses comme indiqué au paragraphe 16.7 ne s'applique pas à un mandataire qui se comporte sans état. Lorsqu'une réponse arrive au mandataire sans état, le mandataire DOIT inspecter la valeur send-by dans la première (la plus élevée) valeur de champs d'en-tête Via. Si cette adresse correspond au mandataire, (elle égale une valeur que ce mandataire a inséré dans des demandes précédentes) le mandataire DOIT retirer cette valeur de champ d'en-tête de la réponse et transmettre le résultat à la localisation indiquée dans la prochaine valeur de champ d'en-tête Via. Le mandataire NE DOIT PAS ajouter au corps de message, ni le modifier ou le retirer. Sauf spécification contraire, le mandataire NE DOIT PAS retirer d'autres valeurs de champ d'en-tête. Si l'adresse ne correspond pas au mandataire, le message DOIT être éliminé en silence.

16.12 Résumé du traitement d'acheminement de mandataire

En l'absence de politique locale disant le contraire, le traitement qu'effectue un mandataire sur une demande contenant un champ d'en-tête Route peut être résumé par les étapes suivantes.

1. Le mandataire va inspecter l'URI-de-demande. Si il indique une ressource qui lui appartient, le mandataire va le remplacer par le résultat du service de localisation. Autrement, le mandataire ne changera pas l'URI-de-demande.

2. Le mandataire va inspecter l'URI dans la valeur de champs d'en-tête Route la plus élevée. Si elle indique ce mandataire, il la retire du champ d'en-tête Route(ce nœud d'acheminement a déjà été atteint).

3. Le mandataire va transmettre la demande aux ressources indiquées par l'URI dans la valeur de champ d'en-tête Route la plus élevée ou dans l'URI-de-demande si aucun champ d'en-tête Route n'est présent. Le mandataire détermine les adresse, port et transport à utiliser lorsqu'il retransmet la demande en appliquant les procédures de [4] à cet URI.

Si aucun élément à acheminement strict n'est rencontré sur le chemin de la demande, l'URI-de-demande indiquera toujours la cible de la demande.

16.12.1 Exemples

16.12.1.1 Trapézoïde SIP de base

Ce scénario est le trapézoïde SIP de base, U1 -> P1 -> P2 -> U2, avec les deux Record-Route mandataires. Le flux est le suivant :

U1 envoie :

INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com

à P1. P1 est un mandataire extérieur. P1 n'est pas responsable de domain.com, et donc il le recherche dans DNS et l'y envoie. Il ajoute aussi une valeur de champ d'en-tête Record-Route :

INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p1.example.com;lr>

P2 reçoit cela. Il est responsable pour domain.com et donc il fait tourner un service de localisation et réécrit l'URI-de-demande. Il ajoute aussi une valeur de champ d'en-tête Record-Route. Il n'y a pas de champ d'en-tête Route, et donc il résout le nouvel URI-de-demande pour déterminer où envoyer la demande :

INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>

L'appelé à u2.domain.com reçoit cela et répond avec une 200 OK :

SIP/2.0 200 OK
Contact: sip:callee@u2.domain.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>

L'appelé à u2 met aussi son URI de cible distante d'état de dialogue à sip:caller@u1.example.com et son ensemble des routes à :
(<sip:p2.domain.com;lr>,<sip:p1.example.com;lr>)

Ce qui est transmis par P2 de P1 à U1 comme normal. Maintenant, U1 met son URI de cible distante d'état de dialogue à sip:callee@u2.domain.com et son ensemble de routes à :
(<sip:p1.example.com;lr>,<sip:p2.domain.com;lr>)

Comme tous les éléments de l'ensemble des routes contiennent le paramètre lr, U1 construit la demande BYE suivante :

BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p1.example.com;lr>,<sip:p2.domain.com;lr>

Comme le ferait tout autre élément (y compris les mandataires), il résout l'URI dans la valeur de champ d'en-tête Route la plus élevée en utilisant DNS pour déterminer où envoyer la demande. Cela va à P1. P1 remarque qu'il n'est pas responsable des ressources indiquées dans l'URI-de-demande et donc il ne le change pas. Il voit qu'il n'est pas la première valeur dans le champ d'en-tête Route, et donc il retire cette valeur, et transmet la demande à P2 :

BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p2.domain.com;lr>

P2 remarque aussi qu'il n'est pas responsable des ressources indiquées par l'URI-de-demande (il est responsable pour domain.com, et non pour u2.domain.com), et donc il ne le change pas. Il ne se voit pas dans la première valeur de champ d'en-tête Route, et donc il la retire et transmet ce qui suit à u2.domain.com sur la base d'une recherche DNS par rapport à l'URI-de-demande :

BYE sip:callee@u2.domain.com SIP/2.0

16.12.1.2 Traversée d'un mandataire à acheminement strict

Dans ce scénario, un dialogue est établi à travers quatre mandataires, chacun d'eux ajoutant des valeurs de champ d'en-tête Record-Route. Le troisième mandataire met en œuvre les procédures d'acheminement strict spécifiées dans la RFC 2543 et de nombreux travaux en cours

U1->P1->P2->P3->P4->U2

L'INVITE arrivant à U2 contient :
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p4.domain.com;lr>
Record-Route: <sip:p3.middle.com>
Record-Route: <sip:p2.example.com;lr>
Record-Route: <sip:p1.example.com;lr>

Auquel U2 répond avec une 200 OK. Plus tard, U2 envoie la demande BYE suivante à P4 sur la base de la première valeur de champ d'en-tête Route.

BYE sip:caller@u1.example.com SIP/2.0
Route: <sip:p4.domain.com;lr>
Route: <sip:p3.middle.com>
Route: <sip:p2.example.com;lr>
Route: <sip:p1.example.com;lr>

P4 n'est pas responsable des ressources indiquées dans l'URI-de-demande et donc il va le laisser comme il est. Il remarque qu'il est l'élément dans la première valeur de champ d'en-tête Route et donc il la retire. Il se prépare alors à envoyer la demande sur la base de ce qui est maintenant la première valeur de champ d'en-tête Route de sip:p3.middle.com, mais il remarque que cet URI ne contient pas le paramètre lr, et donc, avant l'envoi, il reformate la

demande pour qu'elle soit :

BYE sip:p3.middle.com SIP/2.0
Route: <sip:p2.example.com;lr>
Route: <sip:p1.example.com;lr>
Route: <sip:caller@u1.example.com>

P3 est un routeur strict, il transmet donc ce qui suit à P2 :

BYE sip:p2.example.com;lr SIP/2.0
Route: <sip:p1.example.com;lr>
Route: <sip:caller@u1.example.com>

P2 voit que l'URI-de-demande est une valeur qu'il a placé dans un champ d'en-tête Record-Route, et donc, avant tout autre traitement, il réécrit la demande pour qu'elle devienne :

BYE sip:caller@u1.example.com SIP/2.0
Route: <sip:p1.example.com;lr>

P2 n'est pas responsable de u1.example.com, et donc il envoie la demande à P1 sur la base de la résolution de la valeur de champ d'en-tête Route.

P1 note qu'il figure lui-même dans la valeur de champ d'en-tête Route la plus élevée, et donc, il la retire, et il en résulte :

BYE sip:caller@u1.example.com SIP/2.0

Comme P1 n'est pas responsable de u1.example.com et qu'il n'y a pas de champ d'en-tête Route, P1 va transmettre la demande à u1.example.com sur la base de l'URI-de-demande.

16.12.1.3 Réécriture des valeurs de champ d'en-tête Record-Route

Dans ce scénario, U1 et U2 sont dans des espaces de nom privés différents et ils entrent dans un dialogue à travers un mandataire P1, qui agit comme une passerelle entre les espaces de nom.

U1->P1->U2
U1 envoie :

INVITE sip:callee@gateway.leftprivatespace.com SIP/2.0
Contact: <sip:caller@u1.leftprivatespace.com>

P1 utilise son service de localisation et envoie ce qui suit à U2 :

INVITE sip:callee@rightprivatespace.com SIP/2.0
Contact: <sip:caller@u1.leftprivatespace.com>
Record-Route: <sip:gateway.rightprivatespace.com;lr>

U2 renvoie cette 200 (OK) à P1 :

SIP/2.0 200 OK
Contact: <sip:callee@u2.rightprivatespace.com>
Record-Route: <sip:gateway.rightprivatespace.com;lr>

P1 réécrit son paramètre d'en-tête Record-Route pour fournir une valeur que U1 puisse utiliser, et envoie à U1 :

SIP/2.0 200 OK
Contact: <sip:callee@u2.rightprivatespace.com>
Record-Route: <sip:gateway.leftprivatespace.com;lr>

Plus tard, U1 envoie la demande BYE suivante à P1 :

BYE sip:callee@u2.rightprivatespace.com SIP/2.0
Route: <sip:gateway.leftprivatespace.com;lr>

que P1 transmet à U2 comme :

BYE sip:callee@u2.rightprivatespace.com SIP/2.0

17 Transactions

SIP est un protocole transactionnel : les interactions entre composants ont lieu dans une série d'échanges de messages indépendants. Précisément, une transaction SIP consiste en une seule demande et un nombre quelconque de réponses à cette demande, ce qui inclut zéro, une ou plusieurs réponses provisoires et une ou plusieurs réponses finales. Dans le cas d'une transaction où la demande était une INVITE (connue sous le nom de transaction INVITE), la transaction n'inclut aussi le ACK que si la réponse finale n'était pas une réponse 2xx. Si la réponse était une 2xx, le ACK n'est pas considéré comme faisant partie de la transaction.

La raison de cette séparation prend sa source dans l'importance de la livraison de toutes les réponses 200 (OK) à une INVITE à l'UAC. Pour les livrer toutes à l'UAC, l'UAS prend seul la responsabilité de les retransmettre (voir au paragraphe 13.3.1.4), et l'UAC prend seul la responsabilité pour d'en accuser réception avec ACK (voir au paragraphe 13.2.2.4). Comme cet ACK n'est retransmis que par l'UAC, il est effectivement considéré comme sa propre transaction.

Les transactions ont un côté client et un côté serveur. Le côté client est appelé une transaction client et le côté serveur une transaction de serveur. La transaction client envoie la demande, et la transaction de serveur envoie la réponse. Les transactions client et serveur sont des fonctions logiques qui sont incorporées dans un nombre quelconque d'éléments. Précisément, elles existent au sein des agents utilisateurs et des serveurs mandataires à états pleins. Considérons l'exemple de la Section 4. Dans cet exemple, l'UAC exécute la transaction client, et son mandataire extérieur exécute la transaction de serveur. Le mandataire extérieur exécute aussi une transaction client, qui envoie la demande à une transaction de serveur dans le mandataire interne. Ce mandataire exécute aussi une transaction client, qui à son tour envoie la demande à une transaction de serveur dans l'UAS. Ceci est montré à la Figure 4.

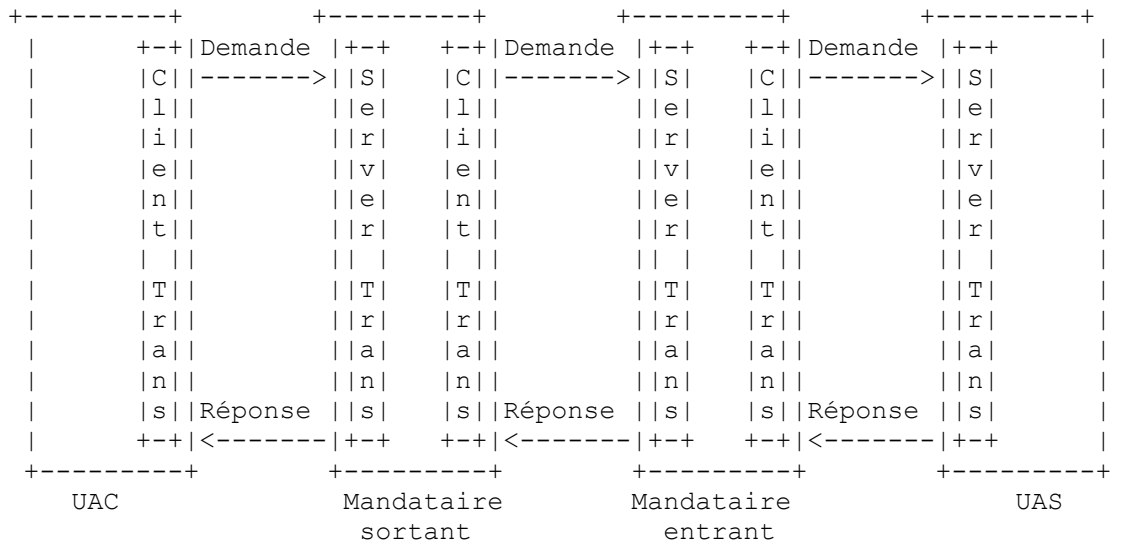


Figure 4 : Relations de transaction

Un mandataire sans état ne contient pas de transaction client ou serveur. La transaction existe entre l'agent utilisateur ou mandataire à états pleins d'un côté, et l'agent utilisateur ou mandataire à états pleins de l'autre côté. Pour autant que soient concernées les transactions SIP, les mandataires sans état sont effectivement transparents. L'objet de la transaction client est de recevoir une demande de la part de l'élément dans lequel le client est incorporé (qu'on appelle "Utilisateur de transaction" ou TU ; il peut être un agent utilisateur ou un mandataire à états pleins), et de livrer de façon fiable la demande à une transaction de serveur.

La transaction client est aussi responsable de la réception des réponses et de leur livraison au TU, en filtrant toutes les retransmissions de réponses ou de réponses de désaccord (comme une réponse à ACK). De plus, dans le cas d'une demande INVITE, la transaction client est responsable de générer la demande ACK pour toute réponse finale acceptant une réponse 2xx.

De la même façon, l'objet de la transaction de serveur est de recevoir des demandes de la part de la couche transport et de les livrer au TU. La transaction de serveur filtre toutes les retransmissions de demandes provenant du réseau. La transaction de serveur accepte les réponses provenant du TU et les livre à la couche transport pour transmission sur le réseau. Dans le cas d'une transaction INVITE, elle absorbe la demande ACK pour toutes les réponses finales excepté une réponse 2xx.

La réponse 2xx et son ACK reçoivent un traitement spécial. Cette réponse n'est retransmise que par un UAS, et son

ACK n'est généré que par l'UAC. Ce traitement d'extrémité à extrémité est nécessaire pour qu'un appelant connaisse l'ensemble complet des usagers qui ont accepté l'appel. A cause de ce traitement spécial, les retransmissions de la réponse 2xx sont prises en main par l'agent utilisateur central, et la couche transaction. De la même façon, la génération de l'ACK pour la 2xx est traité par l'agent utilisateur central. Chaque mandataire le long du chemin fait simplement suivre la réponse 2xx à INVITE et son ACK correspondant.

17.1 Transaction client

La transaction client fournit ses fonctionnalités par la maintenance d'un automate à états.

Le TU communique avec la transaction client à travers une simple interface. Lorsque le TU souhaite initialiser une nouvelle transaction, il crée une transaction client et la passe dans la demande SIP pour envoi avec une adresse, un port, et un transport auxquels l'envoyer. La transaction client commence l'exécution de son automate à états. Les réponses valides sont passées au TU à partir de la transaction client.

Il y a deux types d'automates à état de transaction client, selon la méthode de la demande passée par le TU. L'une traite les transactions client pour des demandes INVITE. Ce type de machine est désigné comme transaction client INVITE. Un autre type traite les transactions client pour toutes les demandes sauf INVITE et ACK. Celui-ci est désigné comme transaction client non-INVITE. Il n'y a pas de transaction client pour ACK. Si le TU souhaite envoyer un ACK, il le passe directement à la couche transport pour transmission.

La transaction INVITE est différente de celle des autres méthodes à cause de sa durée étendue. Normalement, une entrée humaine est nécessaire afin de répondre à une INVITE. Les longs délais supposés pour l'envoi d'une réponse plaident en faveur d'une prise de contact à trois. D'un autre côté, des demandes d'autres méthodes sont supposées se terminer rapidement. A cause de la fiabilité de la transaction non-INVITE sur une prise de contact à deux, les TU DEVRAIENT répondre immédiatement à des demandes non-INVITE.

17.1.1 Transaction INVITE du client

17.1.1.1 Généralités sur la transaction INVITE

La transaction INVITE consiste en une prise de contact à trois. La transaction client envoie un INVITE, la transaction de serveur envoie les réponses, et la transaction client envoie un ACK. Pour les transports non fiables (tels que UDP), la transaction client retransmet des demandes à un intervalle qui débute à T1 secondes et double après chaque retransmission. T1 est une estimation de délai d'aller-retour (RTT, *round-trip time*), et vaut par défaut 500 ms. Presque tous les temporisateurs de transaction décrits ici sont étalonnés avec T1, et changer T1 ajuste leur valeur. La demande n'est pas retransmise sur des transports fiables. Après avoir reçu une réponse 1xx, toutes les retransmissions cessent ensemble, et le client attend les réponses suivantes. La transaction de serveur peut envoyer des réponses 1xx supplémentaires, qui ne sont pas transmise de façon fiable par la transaction de serveur. Finalement, la transaction de serveur décide d'envoyer une réponse finale. Pour les transports non fiables, cette réponse est retransmise périodiquement, et pour les transports fiables, elle est envoyée une seule fois. Pour chaque réponse finale qui est reçue à la transaction client, celle-ci envoie un ACK, dont l'objet est de faire cesser les retransmissions de la réponse.

17.1.1.2 Description formelle

L'automate à états pour la transaction client INVITE est montré à la Figure 5. L'état initial, "calling", DOIT être entré lorsque le TU initialise une nouvelle transaction client avec une demande INVITE. La transaction client DOIT passer la demande à la couche transport pour transmission (voir au paragraphe 18). Si un transport non fiable est utilisé, la transaction client DOIT lancer le temporisateur A avec une valeur de T1. Si un transport fiable est utilisé, la transaction client NE DEVRAIT PAS lancer le temporisateur A (le temporisateur A commande les retransmissions de demandes). Pour tout transport, la transaction client DOIT lancer le temporisateur B avec une valeur de 64*T1 secondes (le temporisateur B commande le délai d'expiration des transactions).

Lorsque le temporisateur A arrive à expiration, la transaction client DOIT retransmettre la demande en la passant à la couche transport, et DOIT remettre le temporisateur à zéro avec une valeur de 2*T1. La définition formelle de la retransmission dans le contexte de la couche transaction est de prendre le message précédemment envoyé à la couche transport et de le passer une fois de plus à la couche transport.

Lorsque le temporisateur A arrive à expiration 2*T1 secondes plus tard, la demande DOIT être retransmise à nouveau (en supposant que la transaction client soit toujours dans cet état). Ce processus DOIT continuer de sorte que la

demande soit retransmise avec des intervalles qui doublent après chaque transmission. Ces retransmissions DEVRAIENT n'être faites que lorsque la transaction client est dans l'état "calling".

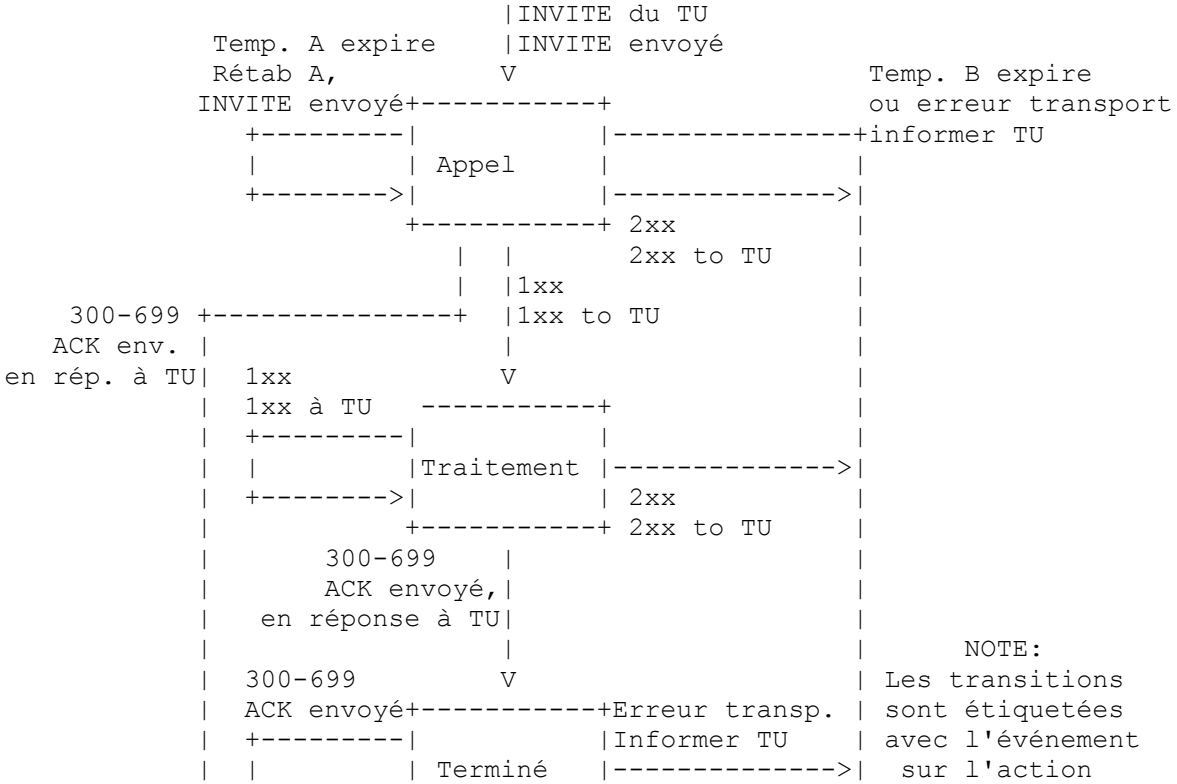
La valeur par défaut pour T1 est 500 ms. T1 est une estimation du RTT entre les transactions client et serveur. Les éléments PEUVENT (bien que cela NE SOIT PAS RECOMMANDÉ) utiliser de plus petites valeurs de T1 au sein de réseaux privés, clos, qui ne permettent pas de connexion Internet générale. T1 PEUT être choisi plus grand, et ceci est RECOMMANDÉ s'il est connu à l'avance (comme sur des liaisons d'accès à forte latence) que le RTT est plus grand. Quelle que soit la valeur de T1, l'attente exponentielle sur les retransmissions décrites dans ce paragraphe DOIT être utilisée.

Si la transaction client est toujours dans l'état "Calling" lorsque le temporisateur B arrive à expiration, la transaction client DEVRAIT informer le TU qu'une fin de temporisation est survenue. La transaction client NE DOIT PAS générer de ACK. La valeur de 64*T1 est égale à la quantité de temps nécessaire pour envoyer sept demandes dans le cas d'un transport non fiable.

Si la transaction client reçoit une réponse provisoire alors qu'elle est dans l'état "Calling", elle passe à l'état "Proceeding". Dans l'état "Proceeding", la transaction client NE DEVRAIT PLUS retransmettre la demande. De plus, la réponse provisoire DOIT être passée au TU. Toutes les réponses provisoires ultérieures DOIVENT être repassées au TU pendant l'état "Proceeding".

Dans les états "Calling" ou "Proceeding", la réception d'une réponse avec un code d'état de 300 à 699 DOIT causer le passage de la transaction client à "Completed". La transaction client DOIT passer la réponse reçue au TU, et elle DOIT générer un ACK de demande, même si le transport est fiable (les lignes directrices pour la construction de l'ACK à partir de la réponse sont données au paragraphe 17.1.1.3) et puis passer le ACK à la couche transport pour transmission. Le ACK DOIT être envoyé à la même adresse, port, et transport que celle à laquelle la demande originelle avait été envoyée. La transaction client DEVRAIT lancer le temporisateur D en entrant dans l'état "Completed", avec une valeur d'au moins 32 secondes pour les transports non fiables, et une valeur de zéro seconde pour les transports fiables. Le temporisateur D reflète la quantité de temps pendant laquelle la transaction de serveur peut rester dans l'état "Completed" lorsque des transports non fiables sont utilisés. Ceci est égal au temporisateur H dans la transaction de serveur INVITE, dont la valeur par défaut est de 64*T1. Cependant, la transaction client ne sait pas la valeur de T1 utilisée par la transaction de serveur, et donc un minimum absolu de 32 s est utilisé au lieu de baser le temporisateur D sur T1.

Toute retransmission de la réponse finale reçue dans l'état "Completed" DOIT amener le ACK à être repassé à la couche transport pour retransmission, mais la réponse nouvellement reçue NE DOIT PAS être repassée au TU. Une retransmission de la réponse est définie comme toute réponse qui pourrait correspondre à la même transaction client sur la base des règles du paragraphe 17.1.3.



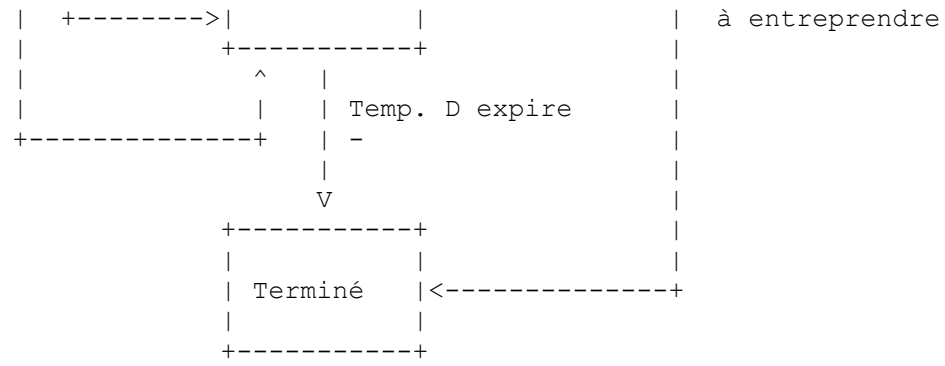


Figure 5 : Transaction client INVITE

Si le temporisateur D arrive à expiration alors que la transaction client est dans l'état "Completed", la transaction client DOIT passer à l'état "Terminated".

Aussi bien dans l'état "Calling" que "Proceeding", la réception d'une 2xx réponse DOIT causer le passage de la transaction client à l'état "Terminated", et la réponse DOIT être repassée au TU. Le traitement de cette réponse dépend de si le TU est un mandataire central ou un UAC central. Un UAC central va traiter la génération de l'ACK pour cette réponse, alors qu'un mandataire central va toujours transmettre la réponse 200 (OK) vers l'amont. Le traitement différent du 200 (OK) entre mandataire et UAC est la raison pour laquelle il ne se fait pas dans la couche transaction.

La transaction client DOIT être détruite à l'instant où elle entre dans l'état "Terminated". Ceci est réellement nécessaire pour garantir un fonctionnement correct. La raison en est que les réponses 2xx à une INVITE sont traitées différemment ; chacune est transmise par un mandataire, et le traitement de l'ACK dans une UAC est différent. Et donc, chaque 2xx a besoin d'être passée à un mandataire central (afin qu'elle puisse être transmise) et à un UAC central (afin qu'il puisse en être accusé réception). Aucun traitement de couche transaction n'a lieu. Chaque fois qu'une réponse est reçue par le transport, si la couche transport ne trouve pas de transaction client correspondante (en utilisant les règles du paragraphe 17.1.3), la réponse est passée directement au central. Comme la transaction client correspondante est détruite par la première 2xx, les 2xx suivantes ne trouveront pas de correspondant et seront donc passées au central.

17.1.1.3 Construction de la demande ACK

Ce paragraphe spécifie la construction des demandes ACK envoyées au sein de la transaction client. Un UAC central qui génère un ACK pour 2xx DOIT suivre les règles décrites à la Section 13.

La demande ACK construite par la transaction client DOIT contenir des valeurs pour Call-ID, From, et URI-de-demande égales aux valeurs de ces champs d'en-tête dans la demande passée au transport par la transaction client (qu'on appellera la "demande originale"). Le champ d'en-tête To dans le ACK DOIT être égal au champ d'en-tête To dans la réponse dont on accuse réception, et donc diffèrera habituellement du champ d'en-tête To dans la demande originelle par l'ajout du paramètre étiquette. Le ACK DOIT contenir un seul champ d'en-tête Via, et celui-ci DOIT être égal au champ d'en-tête Via le plus élevé de la demande originelle. Le champ d'en-tête CSeq dans le ACK DOIT contenir la même valeur pour le numéro de séquence que celle qui était présente dans la demande originelle, mais le paramètre méthode DOIT être égal à "ACK".

Si la demande INVITE dont il est accusé réception de la réponse avait un champ d'en-tête Route, celui-ci DOIT apparaître dans le ACK. Ceci permet de s'assurer que le ACK peut être acheminé de façon appropriée à travers tous mandataires sans état vers l'aval.

Bien que toute demande PUISSE contenir un corps, un corps dans un ACK est particulier car la demande ne peut pas être rejetée si le corps n'est pas compris. Donc, le placement de corps dans ACK pour les réponses non-2xx N'EST PAS RECOMMENDÉ, mais si c'est fait, les types de corps doivent se restreindre à ceux qui apparaissent dans le INVITE, en supposant que la réponse à l'INVITE n'était pas 415. Si cela était, le corps dans le ACK PEUT être de tout type figurant sur la liste dans le champ d'en-tête Accept dans le 415.

Par exemple, considérons la demande suivante :

```
    INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKkjshdyff
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=88sja8x
Max-Forwards: 70
```

Call-ID: 987asjd97y7atg
CSeq: 986759 INVITE

Le ACK de demande pour une réponse finale non-2xx à cette demande ressemblerait à ceci :

ACK sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKkjshdyff
To: Bob <sip:bob@biloxi.com>;tag=99sa0xk
From: Alice <sip:alice@atlanta.com>;tag=88sja8x
Max-Forwards: 70
Call-ID: 987asjd97y7atg
CSeq: 986759 ACK

17.1.2 Transaction Non-INVITE de client

17.1.2.1 Généralités sur la transaction non-INVITE

Les transactions non-INVITE n'utilisent pas de ACK. Ce sont de simples interactions demande-réponse. Pour les transports non fiables, des demandes sont retransmises à un intervalle qui débute à T1 et double jusqu'à ce qu'il atteigne T2. Si une réponse provisoire est reçue, la retransmission continue pour les transports non fiables, mais à un intervalle de T2. La transaction de serveur ne retransmet la dernière réponse envoyée, qui peut être une réponse provisoire ou finale, que lorsqu'une retransmission de la demande est reçue. C'est pourquoi les retransmissions de demandes doivent continuer même après une réponse provisoire ; c'est pour s'assurer de la livraison fiable de la réponse finale.

A la différence d'une transaction INVITE, une transaction non-INVITE n'a pas de traitement particulier pour les réponses 2xx. Il en résulte qu'une seule réponse 2xx à une non-INVITE est livrée à une UAC.

17.1.2.2 Description formelle

L'automate à états pour la transaction client non-INVITE est montré à la Figure 6. Il est très similaire à l'automate à état pour INVITE.

L'état "Trying" est abordé lorsque le TU initialise une nouvelle transaction client avec une demande. En entrant dans cet état, la transaction client DEVRAIT régler le temporisateur F pour arriver à expiration dans 64*T1 secondes. La demande DOIT être passée à la couche transport pour transmission. Si un transport non fiable est utilisé, la transaction client DOIT régler le temporisateur E pour arriver à expiration dans T1 secondes. Si le temporisateur E arrive à expiration alors qu'on est toujours dans cet état, le temporisateur est réinitialisé, mais cette fois avec une valeur de MIN(2*T1, T2). Lorsque le temporisateur arrive une nouvelle fois à expiration, il est réinitialisé à MIN(4*T1, T2). Ce processus continue de sorte que les retransmissions surviennent avec un accroissement exponentiel de l'intervalle qui culmine à T2. La valeur par défaut de T2 est 4 s, et elle représente la quantité de temps qu'une transaction de serveur non-INVITE va prendre pour répondre à une demande, si elle ne répond pas immédiatement. Pour la valeur par défaut de T1 et T2, il en résulte des intervalles de 500 ms, 1 s, 2 s, 4 s, 4 s, 4 s, etc.

Si le temporisateur F arrive à expiration pendant que la transaction client est toujours dans l'état "Trying", la transaction client DEVRAIT informer le TU de la fin de temporisation, et elle DEVRAIT alors entrer dans l'état "Terminated". Si une réponse provisoire est reçue dans l'état "Trying", la réponse DOIT être passée au TU, et la transaction client DEVRAIT ensuite passer à l'état "Proceeding". Si une réponse finale (code d'états 200-699) est reçue dans l'état "Trying", la réponse DOIT être passée au TU, et la transaction client DOIT passer à l'état "Completed".

Si le temporisateur E arrive à expiration pendant l'état "Proceeding", la demande DOIT être passée à la couche transport pour retransmission, et le temporisateur E DOIT être réinitialisé avec une valeur de T2 secondes. Si le temporisateur F arrive à expiration pendant l'état "Proceeding", le TU DOIT être informé d'une fin de temporisation, et la transaction client DOIT passer à l'état "Terminated". Si une réponse finale (code d'états 200-699) est reçue pendant l'état "Proceeding", la réponse DOIT être passée au TU, et la transaction client DOIT passer à l'état "Completed".

Une fois que la transaction client entre dans l'état "Completed", elle DOIT régler le temporisateur K de façon à arriver à expiration dans T4 secondes pour les transports non fiables, et à zéro seconde pour les transports fiables. L'état "Completed" existe pour mettre en mémoire tampon toutes retransmissions de réponses supplémentaires qui peuvent être reçues (c'est pourquoi la transaction client ne reste que pour les transports non fiables). T4 représente la quantité de temps pendant laquelle le réseau va nettoyer les messages entre transactions client et serveur. La valeur par défaut de T4 est 5 s. Une réponse est une retransmission lorsqu'elle correspond à la même transaction, en utilisant les règles spécifiées au paragraphe 17.1.3. Si le temporisateur K arrive à expiration pendant cet état, la transaction client DOIT passer à l'état "Terminated".

Une fois que la transaction est dans l'état "Terminated", elle DOIT être immédiatement détruite.

17.1.3 Correspondance des réponses aux transactions client

Lorsque la couche transport dans le client reçoit une réponse, elle a à déterminer quelle transaction client va traiter la réponse, afin que puisse prendre place le processus des paragraphes 17.1.1 et 17.1.2. Le paramètre de branche dans le champ d'en-tête Via le plus élevé est utilisé à cette fin. Une réponse correspond à une transaction client à deux conditions :

1. Si la réponse a la même valeur de paramètre de branche dans le champ d'en-tête Via le plus élevé que le paramètre de branche dans le champ d'en-tête Via le plus élevé de la demande qui a créé la transaction.
2. Si le paramètre méthode dans le champ d'en-tête CSeq correspond à la méthode de la demande qui a créé la transaction. La méthode est nécessaire car une demande CANCEL constitue une transaction différente, mais partage la même valeur de paramètre de branche.

Si une demande est envoyée via multicast (diffusion groupée), il est possible que cela génère plusieurs réponses de différents serveurs. Ces réponses auront toutes le même paramètre de branche dans le Via le plus élevé, mais varieront par l'étiquette To. La première réponse reçue, sur la base des règles ci-dessus, sera utilisée, et les autres seront vues comme des retransmissions. Ceci n'est pas une erreur ; SIP en diffusion groupée ne fournit qu'un service rudimentaire de "un seul saut à la découverte" qui est limité au traitement d'une seule réponse. Voir les détails au paragraphe 18.1.1.

17.1.4 Traitement des erreurs de transport

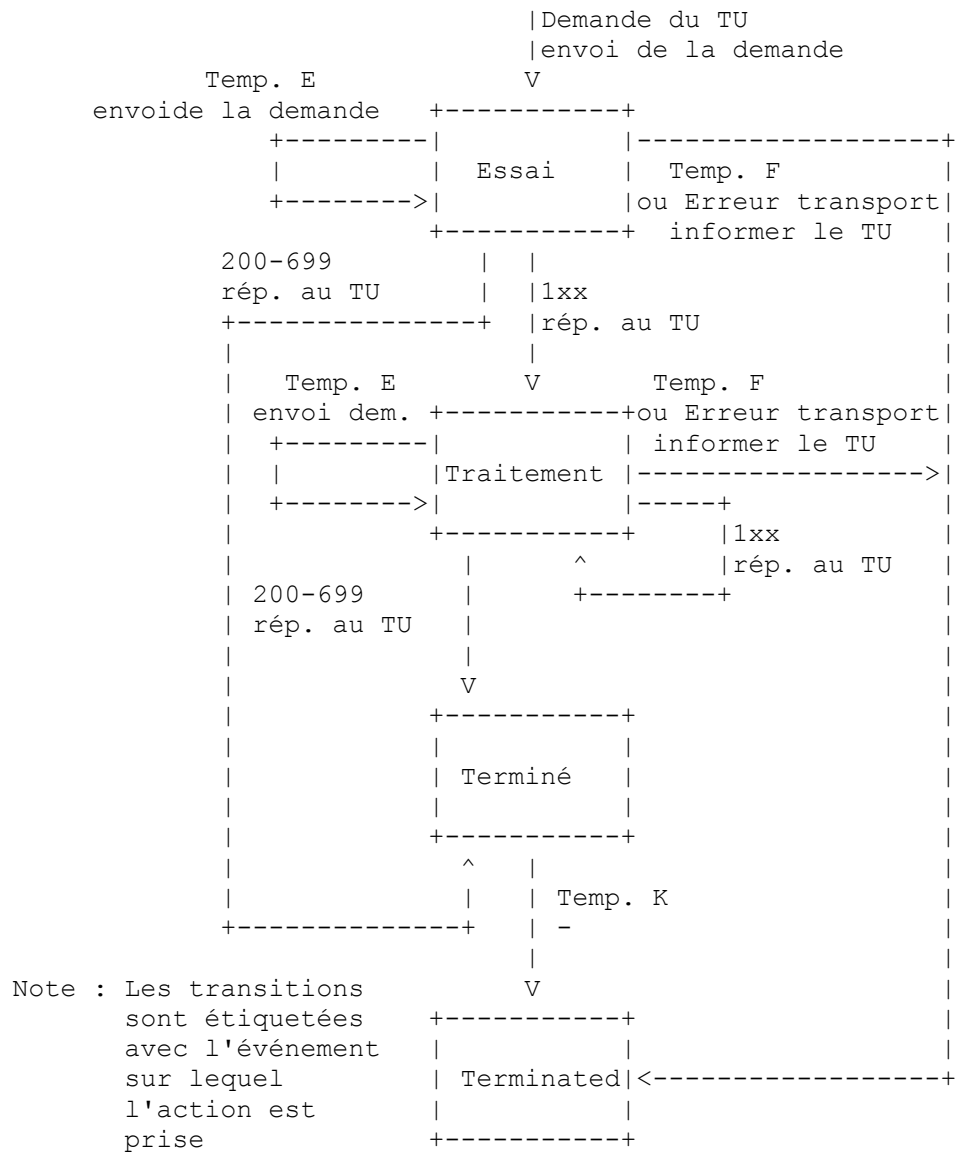


Figure 6 : Transaction client non-INVITE

Lorsque la transaction client envoie une demande à la couche transport pour être envoyée, on suit les procédures suivantes si la couche transport indique une défaillance.

La transaction client DEVRAIT informer le TU qu'une défaillance de transport est survenue, et la transaction client DEVRAIT passer directement à l'état "Terminated". Le TU traitera le mécanisme de récupération de défaillance décrit en [4].

17.2 Transaction de serveur

La transaction serveur est responsable de la livraison des demandes au TU et de la transmission fiable des réponses. Elle le fait par l'intermédiaire d'un automate à états. Les transactions serveur sont créées par le centre lorsqu'une demande est reçue et que le traitement de la transaction est souhaité pour cette demande (ce qui n'est pas toujours le cas).

Comme avec les transactions client, l'automate à états dépend de si la demande reçue est une demande INVITE.

17.2.1 Transaction INVITE du serveur

La Figure 7 montre le diagramme des états pour la transaction de serveur INVITE.
Lorsqu'une transaction de serveur est construite pour une demande, elle entre dans l'état "Proceeding". La transaction de serveur DOIT générer une réponse 100 (En essai) sauf si elle sait que le TU va générer une réponse provisoire ou finale dans les 200 ms, auquel cas elle PEUT générer une réponse 100 (En essai). Cette réponse provisoire est nécessaire pour arrêter rapidement les retransmissions afin d'éviter l'encombrement du réseau. La réponse 100 (En essai) est construite conformément aux procédures du paragraphe 8.2.6, sauf que l'insertion des étiquettes dans le champ d'en-tête To de la réponse (lorsque aucune n'était présente dans la demande) est déclassée de PEUT à NE DEVRAIT PAS. La demande DOIT être passée au TU.

Le TU passe toutes les réponses provisoires à la transaction de serveur. Tant que la transaction de serveur est dans l'état "Proceeding", chacune d'elles DOIT être passée à la couche transport pour transmission. Elles ne sont pas envoyées de façon fiable par la couche transaction (ce n'est pas par elle qu'elles sont retransmises) et ne causent pas un changement dans l'état de la transaction de serveur. Si une retransmission de demande est reçue pendant l'état "Proceeding", la plus récente réponse provisoire reçue du TU DOIT être passée à la couche transport pour retransmission. Une demande est une retransmission si elle correspond à la même transaction de serveur sur la base des règles du paragraphe 17.2.3.
Si, étant dans l'état "Proceeding", le TU passe une réponse 2xx à la transaction de serveur, la transaction de serveur DOIT passer cette réponse à la couche transport pour transmission. Elle n'est pas retransmise par la transaction de serveur ; les retransmissions des réponses 2xx sont traitées par le TU. La transaction de serveur DOIT alors passer à l'état "Terminated".

Dans l'état "Proceeding", si le TU passe une réponse avec code d'état de 300 à 699 à la transaction de serveur, la réponse DOIT être passée à la couche transport pour transmission, et l'automate à états DOIT entrer dans l'état "Completed". Pour les transports non fiables, le temporisateur G est réglé pour arriver à expiration dans T1 secondes, et n'est pas établi pour les transports fiables.

Ceci est un changement par rapport à la RFC 2543, où les réponses étaient toujours retransmises, même sur les transports fiables.

Lorsqu'on est passé à l'état "Completed", le temporisateur H DOIT être réglé pour arriver à expiration en 64*T1 secondes pour tous les transports. Le temporisateur H détermine quand la transaction de serveur abandonne les retransmissions de la réponse. Sa valeur est choisie pour être égale à celle du temporisateur B, la durée pendant laquelle une transaction client va continuer à essayer d'envoyer une demande. Si le temporisateur G arrive à expiration, la réponse est passée à la couche transport une fois de plus pour retransmission, et le temporisateur G est réglé pour arriver à expiration dans MIN(2*T1, T2) secondes. A partir de là, lorsque le temporisateur G arrive à expiration, la réponse est passée à nouveau au transport pour transmission, et le temporisateur G est relancé avec une valeur qui double, jusqu'à ce que cette valeur excède T2, auquel cas il est relancé avec la valeur de T2. Ceci est identique au comportement de retransmission pour des demandes dans l'état "Trying" de la transaction client non-INVITE. De plus, dans l'état "Completed", si une retransmission de demande est reçue, le serveur DEVRAIT passer la réponse au transport pour retransmission.

Si un ACK est reçu lorsque la transaction de serveur est dans l'état "Completed", la transaction de serveur DOIT passer à l'état "Confirmed". Comme le temporisateur G est ignoré dans cet état, toute retransmission de la réponse va cesser.
Si le temporisateur H arrive à expiration pendant l'état "Completed", cela implique que le ACK n'a jamais été reçu. Dans ce cas, la transaction de serveur DOIT passer à l'état "Terminated", et DOIT indiquer au TU qu'un échec de transaction est survenu.

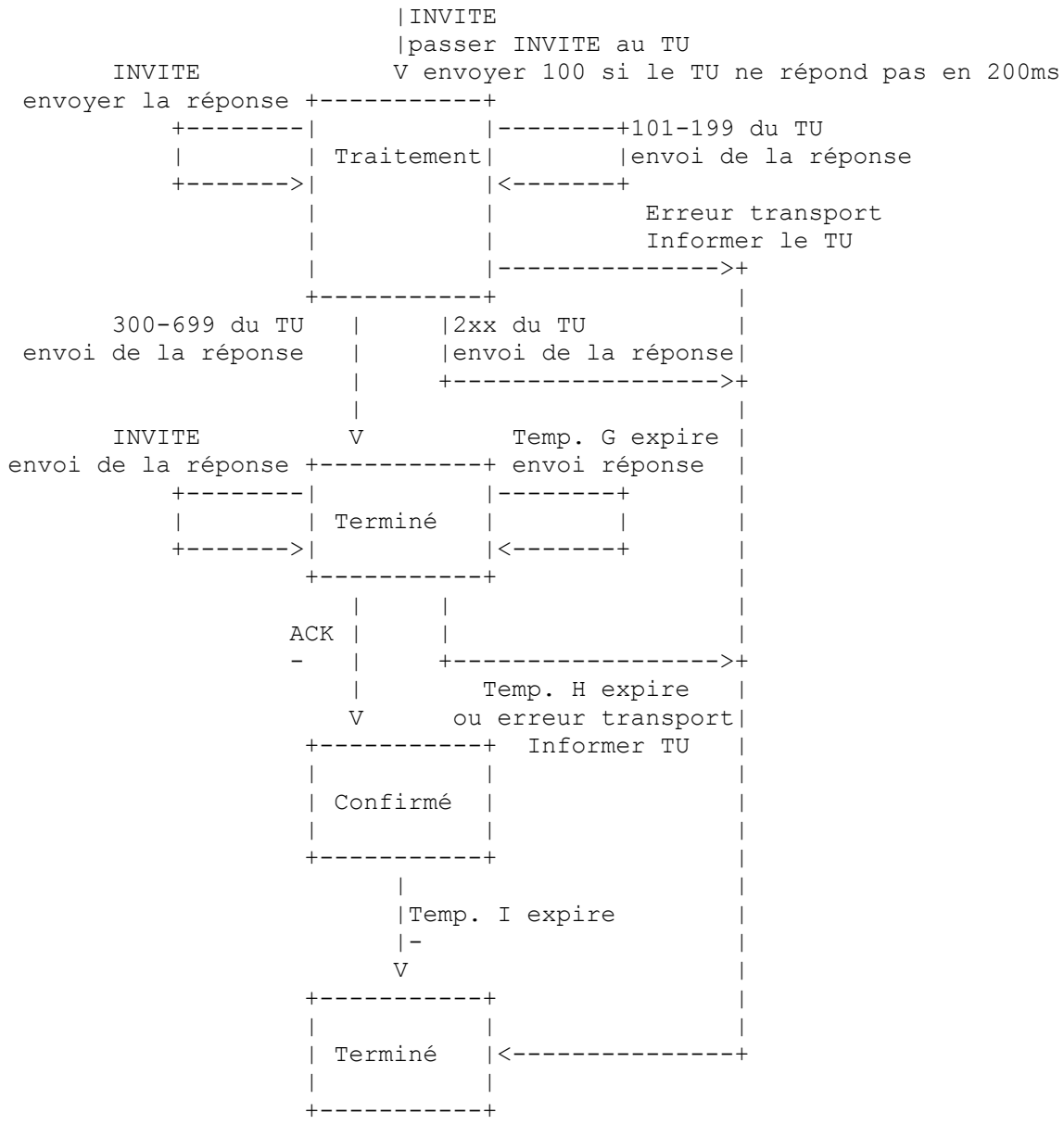


Figure 7 : Transaction serveur INVITE

L'objet de l'état "Confirmed" est d'absorber tout message ACK supplémentaire qui arrive, déclenché à partir de retransmissions de la réponse finale. Lorsqu'on arrive à cet état, le temporisateur I est réglé pour arriver à expiration dans T4 secondes pour les transports non fiables, et zéro seconde pour les transports fiables. Une fois que le temporisateur I arrive à expiration, le serveur DOIT passer à l'état "Terminé".

Une fois que la transaction est dans l'état "Terminé", elle DOIT être immédiatement détruite. Comme avec les transactions client, ceci est nécessaire pour assurer la fiabilité des réponses 2xx à INVITE.

17.2.2 Transaction Non-INVITE du serveur

L'automate à états pour la transaction de serveur non-INVITE est montré à la Figure 8.

L'automate à états est initialisé dans l'état "Trying" et est passé dans une demande autre que INVITE ou ACK à l'initialisation. Cette demande est passée au TU. Une fois dans l'état "Trying", toutes les retransmissions de demandes ultérieures sont éliminées. Une demande est une retransmission si elle correspond à la même transaction de serveur, en utilisant les règles spécifiées au paragraphe 17.2.3.

Dans l'état "Trying", si le TU passe une réponse provisoire à la transaction de serveur, la transaction de serveur DOIT

entrer dans l'état "Proceeding". La réponse DOIT être passée à la couche transport pour transmission. Toutes réponses provisoires ultérieures qui sont reçues du TU dans l'état "Proceeding" DOIVENT être passées à la couche transport pour transmission. Si une retransmission de la demande est reçue dans l'état "Proceeding", la réponse provisoire la plus récemment envoyée DOIT être passée à la couche transport pour retransmission. Si le TU passe une réponse finale (code d'états 200-699) au serveur dans l'état "Proceeding", la transaction DOIT entrer dans l'état "Completed", et la réponse DOIT être passée à la couche transport pour transmission.

Lorsque la transaction de serveur entrer dans l'état "Completed", elle DOIT régler le temporisateur J pour qu'il arrive à expiration dans 64*T1 secondes pour les transports non fiables, et zéro seconde pour les transports fiables. Dans l'état "Completed", la transaction de serveur DOIT passer la réponse finale à la couche transport pour retransmission à chaque fois qu'est reçue une retransmission de la demande. Toute autre réponse finale passée par le TU à la transaction de serveur DOIT être éliminée pendant l'état "Completed". La transaction de serveur reste dans cet état jusqu'à l'arrivée à expiration du temporisateur J, et à ce point, il DOIT passer à l'état "Terminated".

La transaction de serveur DOIT être détruite à l'instant où elle entre dans l'état "Terminated".

17.2.3 Correspondance des demandes avec les transactions du serveur

Lorsqu'une demande est reçu du réseau par le serveur, elle doit être comparée à une transaction existante. Ceci s'accompli de la façon suivante.

Le paramètre de branche dans le champ d'en-tête Via le plus élevé de la demande est examiné. S'il est présent et commence par le cookie magique "z9hG4bK", la demande a été générée par une transaction client conforme à la présente spécification. Donc, le paramètre branche sera unique parmi toutes les transactions envoyées par ce client. La demande correspond à une transaction si :

1. le paramètre de branche dans la demande est égal à celui du champ d'en-tête Via de tête de la demande qui a créé la transaction, et
2. la valeur send-by dans le Via de tête de la demande est égal à celui de la demande qui a créé la transaction, et
3. la méthode de la demande correspond à celle qui a créé la transaction, excepté pour ACK, où la méthode de la demande qui a créé la transaction est INVITE.

Ces règles de correspondance s'appliquent aussi bien aux transactions INVITE et non-INVITE.

La valeur send-by est utilisée au titre du processus de correspondance parce qu'il pourrait y avoir des duplications accidentelles ou malveillantes des paramètres de branche à partir de clients différents.

Si le paramètre de branche dans le champ d'en-tête Via de tête n'est pas présent, ou s'il ne contient pas le cookie magique, les procédures suivantes sont utilisées. Elles existent pour permettre la rétro-compatibilité des mises en œuvre conformes à la RFC 2543.

La demande INVITE correspond à une transaction si l'URI-de-demande, l'étiquette To, l'étiquette From, le Call-ID, le CSeq, et le champ d'en-tête Via de tête correspondent à ceux de la demande INVITE qui a créé la transaction. Dans ce cas, l'INVITE est une retransmission de l'original qui a créé la transaction. La demande ACK correspond à une transaction si l'URI-de-demande, l'étiquette From, le Call-ID, le numéro CSeq (et non la méthode), et le champ d'en-tête Via de tête correspondent à ceux de la demande INVITE qui a créé la transaction, et si l'étiquette To du ACK correspond à l'étiquette To de la réponse envoyée par la transaction de serveur. La correspondance est faite sur la base des règles de correspondance définies pour chacun de ces champs d'en-tête. L'inclusion de l'étiquette dans le champ d'en-tête To dans le processus de correspondance de ACK aide à lever les ambiguïtés de ACK pour les réponses 2xx par rapport au ACK pour d'autres réponses à un mandataire, qui peuvent avoir transmis les deux réponses. (Ceci peut survenir dans des conditions inhabituelles. Précisément, lorsqu'un mandataire a fourché une demande, et ensuite a une défaillance, la réponse peut être livrée à un autre mandataire, ce qui peut se terminer par la transmission de plusieurs réponses vers l'amont). Une demande ACK qui correspond à une transaction INVITE à laquelle correspond un ACK précédent est considéré comme une retransmission de cet ACK précédent.

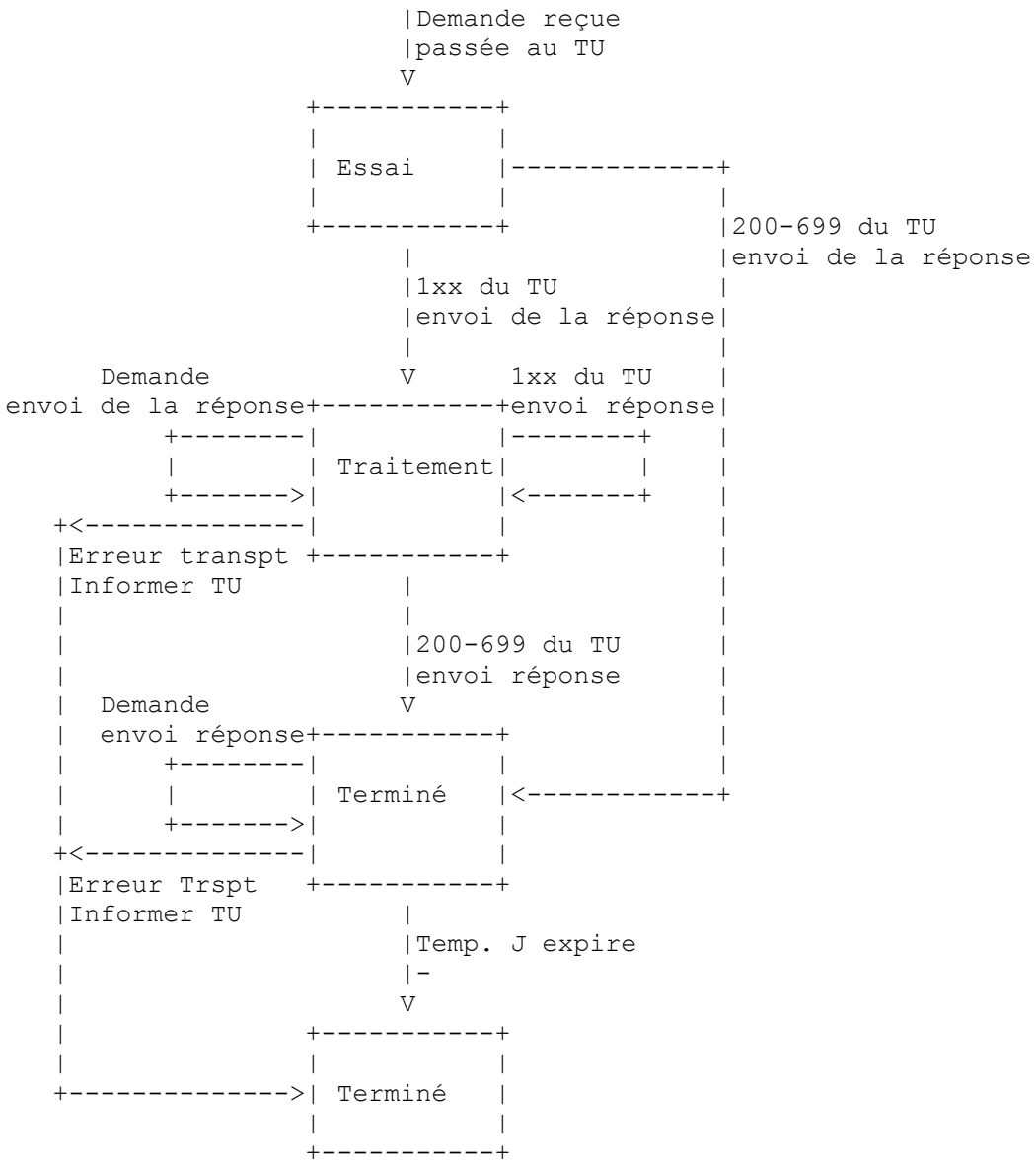


Figure 8 : Transaction serveur non-INVITE

Pour toutes les autres méthodes de demande, une demande correspond à une transaction si l'URI-de-demande, l'étiquette To, l'étiquette From, le Call-ID, le CSeq (y compris la méthode), et le champ d'en-tête Via de tête correspondent à ceux de la demande qui a créé la transaction. La correspondance est établie sur la base des règles de correspondance définies pour chacun de ces champs d'en-tête. Lorsqu'une demande non-INVITE correspond à une transaction existante, elle est une retransmission de la demande qui a créé cette transaction.

Parce que les règles de correspondance incluent l'URI-de-demande, le serveur ne peut pas faire correspondre une réponse à une transaction. Lorsque le TU passe une réponse à la transaction de serveur, il doit la passer à la transaction de serveur spécifique pour laquelle la réponse est ciblée.

17.2.4 Traitement des erreurs de transport

Lorsque la transaction de serveur envoie une réponse à la couche transport pour être envoyée, les procédures suivantes s'appliquent si la couche transport indique une défaillance.

D'abord, on suit les procédures de [4], qui essayent de livrer la réponse à une sauvegarde. Si cela échoue, sur la base de la définition de l'échec dans [4], la transaction de serveur DEVRAIT informer le TU qu'une défaillance est survenue, et DEVRAIT passer à l'état terminé.

18 Transport

La couche transport est responsable de la transmission réelle des demandes et réponses sur les transports réseau. Cela inclut la détermination de la connexion à utiliser pour une demande ou réponse dans le cas de transports orientés connexion.

La couche transport est responsable de la gestion des connexions persistantes pour les protocoles de transport comme TCP et SCTP, ou TLS parmi eux, y compris ceux qui sont ouverts à la couche transport. Ceci inclut les connexions ouvertes par les transports client ou serveur, de sorte que les connexions soient partagées entre les fonctions de transport client et serveur. Ces connexions sont indexées par le tuplet formé de l'adresse, port, et protocole de transport à l'extrémité distante de la connexion. Lorsque une connexion est ouverte par la couche transport, cet indice est mis à la destination, port et transport IP. Lorsque la connexion est acceptée par la couche transport, cet indice est mis à l'adresse IP de source, au numéro de port, et transport. Noter que, parce que le port de source est souvent éphémère, mais on ne peut savoir si il est éphémère ou choisi par les procédures de [4], les connexions acceptées par la couche transport seront fréquemment non réutilisées. Il en résulte que deux mandataires dans une relation "d'homologue à homologue" utilisant un transport orienté connexion auront fréquemment deux connexions en service, une pour les transactions initialisée dans chaque direction.

Il est RECOMMANDÉ que les connexions soient gardées ouvertes pour des durées définies par les mises en œuvre après l'envoi du dernier message ou sa réception sur cette connexion. Sa durée DEVRAIT au moins être égale à la plus longue durée dont l'élément aura besoin afin d'amener une transaction de l'instanciation à l'état terminé. Ceci est destiné à augmenter la probabilité que les transactions soient terminées sur la même connexion que celle sur laquelle elle ont été initialisées (par exemple, demande, réponse, et dans le cas de INVITE, ACK pour les réponses non-2xx). Cela signifie habituellement au moins 64*T1 (voir au paragraphe 17.1.1.1 pour une définition de T1). Cependant, elle pourrait être plus longue dans un élément qui a un TU en utilisant une grande valeur pour le temporisateur C (point 11 du paragraphe 16.6), par exemple.

Tous les éléments SIP DOIVENT mettre en œuvre UDP et TCP. Les éléments SIP PEUVENT mettre en œuvre d'autres protocoles.

Rendre TCP obligatoire pour l'agent utilisateur est un changement substantiel par rapport à la RFC 2543. C'est venu du besoin de traiter des messages plus longs, qui DOIVENT utiliser TCP, comme expliqué ci-dessous. Et donc, même si un élément n'envoie jamais de gros messages, il peut en recevoir un et a besoin d'être capable de le traiter.

18.1 Clients

18.1.1 Envoi des demandes

Le côté client de la couche transport est responsable de l'envoi de la demande et de la réception des réponses. L'utilisateur de la couche transport passe au client transport la demande, une adresse, port, transport IP, et éventuellement TTL pour les destinations en envoi groupé.

Si une demande est dans les 200 octets du MTU de la voie, ou si elle est supérieure à 1300 octets et que le MTU de la voie est inconnu, la demande DOIT être envoyée en utilisant un protocole de transport à contrôle de l'encombrement de la RFC 2914 [43], tel que TCP. Si cela cause un changement dans le protocole de transport par rapport à celui indiqué dans le Via de tête, la valeur du Via de tête DOIT être changée. Cela empêche la fragmentation des messages sur UDP et donne un contrôle d'encombrement pour les plus grands messages. Cependant, les mises en œuvre DOIVENT être capable de traiter les messages jusqu'à la taille maximale de datagramme. Pour UDP, cette taille est de 65 535 octets, y compris les en-têtes IP et UDP.

Le "tampon" de 200 octets entre la taille de message et le MTU prend en compte le fait que la réponse dans SIP peut être plus grande que la demande. Cela arrive du fait de l'ajout de la valeur de champ d'en-tête Record-Route à la réponse à INVITE, par exemple. Avec la mémoire tampon supplémentaire, la réponse peut être d'environ 170 octets plus grande que la demande, et n'être toujours pas fragmentée sur IPv4 (environ 30 octets sont consommés par IP/UDP, en supposant l'absence d'IPSec). 1300 est choisi lorsque le MTU de voie n'est pas connu, sur la base d'une hypothèse d'un MTU Ethernet de 1500 octets.

Si un élément envoie une demande sur TCP à cause de ces contraintes de taille de message, et que cette demande aurait autrement été envoyée sur UDP, si l'essai d'établissement de la connexion génère un protocole ICMP non accepté, ou

résulte en une réinitialisation de TCP, l'élément DEVRAIT réessayer la demande, en utilisant UDP. Ceci est seulement destiné à fournir la rétrocompatibilité avec les mises en œuvre conformes à la RFC 2543 et ne prenant pas en charge TCP. On suppose que ce comportement sera déconseillé dans une future révision de la présente spécification.

Un client qui envoie une demande à une adresse en diffusion groupée DOIT ajouter le paramètre "maddr" à sa valeur de champ d'en-tête Via qui contient l'adresse de destination en diffusion groupée, et pour IPv4, DEVRAIT ajouter le paramètre "ttl" avec une valeur de 1. L'usage de IPv6 en diffusion groupée n'est pas défini dans la présente spécification, et sera un sujet de normalisation futur quand le besoin s'en fera sentir.

Ces règles résultent en une limitation résolue de la diffusion groupée dans SIP. Sa fonction principale est de fournir un simple service de "un seul saut à la découverte", en délivrant une demande à un groupe de serveurs homogènes, où il est seulement demandé de traiter toute réponse provenant de l'un d'entre eux. Cette fonctionnalité est des plus utiles pour l'enregistrement. En fait, sur la base des règles de traitement des transaction du paragraphe 17.1.3, la transaction client va accepter la première réponse, et verra toutes les autres comme des retransmissions parce qu'elles contiennent toutes le même identifiant de branche Via.

Avant qu'une demande ne soit envoyée, le transport client DOIT insérer une valeur du champ "send-by" dans le champ d'en-tête Via. Ce champ contient une adresse IP ou nom d'hôte, et un port. L'usage d'un FQDN est RECOMMANDÉ. Ce champ est utilisé pour envoyer des réponses dans certaines conditions, décrites ci-dessous. Si le port est absent, la valeur par défaut dépend du transport. Il est 5060 pour UDP, TCP et SCTP, 5061 pour TLS.

Pour les transports fiables, la réponse est normalement envoyée sur la connexion sur laquelle a été reçue la demande. Donc, le transport client DOIT être prêt à recevoir la réponse sur la même connexion qu'utilisée pour envoyer la demande. Dans des conditions d'erreur, le serveur peut essayer d'ouvrir une nouvelle connexion pour envoyer la réponse. Pour traiter ce cas, la couche transport DOIT aussi être prête à recevoir une connexion entrante sur l'adresse IP de source d'où la demande a été envoyée et le numéro de port dans le champ "send-by". Elle DOIT aussi être prête à recevoir des connexions entrantes sur toute adresse et port qui seraient choisis par un serveur sur la base des procédures décrites à la Section 5 de [4].

Pour les transports non fiables en envoi individuel, le transport client DOIT être prêt à recevoir des réponses à l'adresse IP de source d'où la demande est envoyée (car les réponses sont renvoyées à l'adresse de source) et au numéro de port figurant dans le champ "send-by". De plus, comme avec les transports fiables, dans certains cas la réponse sera envoyée ailleurs. Le client DOIT être prêt à recevoir des réponses sur toute adresse et port qui seraient choisis par un serveur sur la base des procédures décrites à la Section 5 de [4].

Pour la diffusion groupée, le transport client DOIT être prêt à recevoir des réponses sur le même groupe et port de diffusion groupée que celui sur lequel la demande est envoyée (c'est-à-dire, il doit être un membre du groupe de diffusion auquel il a envoyé la demande.)

Si une demande est destinée à une adresse, port, et transport IP avec lequel une connexion existante est ouverte, il est RECOMMANDÉ que cette connexion soit utilisée pour envoyer la demande, mais une autre connexion PEUT être ouverte et utilisée.

Si une demande est envoyée en utilisant la diffusion groupée, elle est envoyée à l'adresse, port, et TTL du groupe fournis par l'utilisateur du transport. Si une demande est envoyée en utilisant un transport non fiable en envoi individuel, elle est envoyée à l'adresse et port IP fournis par l'utilisateur du transport.

18.1.2 Réception des réponses

Lorsque une réponse est reçue, le transport client examine la valeur de champ d'en-tête Via de tête. Si la valeur du paramètre "send-by" dans cette valeur de champ d'en-tête ne correspond pas à une valeur que le transport client est configuré pour insérer dans des demandes, la réponse DOIT être éliminée en silence.

S'il existe des transactions client, le transport client utilise les procédures de correspondance du paragraphe 17.1.3 pour essayer de confronter la réponse à une transaction existante. Si il y a correspondance, la réponse DOIT être passée à cette transaction. Autrement, la réponse DOIT être passée au centre (qu'il soit mandataire sans état, mandataire à états pleins, ou agent utilisateur) pour traitement ultérieur. Le traitement de ces réponses "erratiques" dépend du centre (un mandataire les transmettra, alors qu'un agent utilisateur les éliminera, par exemple).

18.2 Serveurs

18.2.1 Réception des demandes

Un serveur DEVRAIT être prêt à recevoir des demandes sur toute combinaison d'adresse IP, port et transport qui peut être le résultat d'une recherche DNS sur un URI SIP ou SIPS [4] menée pour les besoins de la communication avec ce serveur. Dans ce contexte, "mener" inclut de placer un URI dans un champ d'en-tête Contact dans une demande REGISTER ou une réponse redirect, ou dans un champ d'en-tête Record-Route dans une demande ou réponse. Un URI peut aussi être "remis" en le plaçant sur une page web ou une carte de visite. Il est aussi RECOMMANDÉ qu'un serveur écoute les demandes sur les ports SIP par défaut (5060 pour TCP et UDP, 5061 pour TLS sur TCP) sur toutes les interfaces publiques. L'exception typique serait celle des réseaux privés, ou lorsque plusieurs instances de serveur tournent sur le même hôte. Pour tout port et interface qu'écoute un serveur pour UDP, il DOIT écouter sur le même port et interface pour TCP. Ceci parce qu'un message peut avoir besoin d'être envoyé en utilisant TCP, plutôt qu'UDP, si il est trop grand. Il en résulte que l'inverse n'est pas vrai. Un serveur n'a pas besoin d'écouter UDP sur une adresse et port particuliers simplement parce qu'il serait en train d'écouter sur les mêmes adresse et port pour TCP. Il peut, bien sûr, y avoir d'autres raisons qui conduisent un serveur à écouter UDP sur une adresse et port particuliers.

Lorsque le transport serveur reçoit une demande sur tout transport, il DOIT examiner la valeur du paramètre "send-by" dans la valeur de champ d'en-tête Via de tête. Si la portion hôte du paramètre "send-by" contient un nom de domaine, ou s'il contient une adresse IP qui diffère de l'adresse de source du paquet, le serveur DOIT ajouter un paramètre "received" à cette valeur de champ d'en-tête Via. Ce paramètre DOIT contenir l'adresse de source d'où le paquet a été reçu. Ceci est destiné à assister la couche de transport serveur dans l'envoi de la réponse, car elle doit être envoyée à l'adresse IP de source d'où est venue la demande.

Considérons une demande reçue par le transport serveur, qui ressemble en partie à :

```
    INVITE sip:bob@Biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060
```

La demande est reçue avec une adresse IP de source de 192.0.2.4. Avant de passer la demande, le transport ajoute un paramètre "received", de sorte que la demande ressemblerait, en partie à :

```
    INVITE sip:bob@Biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;received=192.0.2.4
```

Ensuite, le transport serveur essaye de confronter la demande à une transaction de serveur. Il le fait en utilisant les règles de correspondance décrites au paragraphe 17.2.3. Si une transaction de serveur correspondante est trouvée, la demande est passée à cette transaction pour traitement. Si aucune correspondance n'est trouvée, la demande est passée au centre, qui peut décider de construire une nouvelle transaction de serveur pour cette demande. Noter que lorsqu'un UAS central envoie une réponse 2xx à INVITE, la transaction de serveur est détruite. Cela signifie que lorsque arrive l'ACK, il n'y aura pas de transaction de serveur correspondante, et sur la base de cette règle, l'ACK est passé à l'UAS central, où il est traité.

18.2.2 Envoi des réponses

Le transport serveur utilise la valeur du champ d'en-tête Via de tête afin de déterminer où envoyer une réponse. Il DOIT suivre le traitement suivant :

- Si le "send-protocol" est un protocole de transport fiable tel que TCP ou SCTP, ou TLS par-dessus eux, la réponse DOIT être envoyée en utilisant la connexion existante à la source de la demande originelle qui a créé la transaction, si cette connexion est toujours ouverte. Ceci exige que le transport serveur maintienne une association entre les transactions serveur et les connexions de transport. Si cette connexion n'est plus ouverte, le serveur DEVRAIT ouvrir une connexion à l'adresse IP dans le paramètre "received", s'il est présent, en utilisant le port dans la valeur "send-by", ou le port par défaut pour ce transport, si aucun port n'est spécifié. Si cette tentative de connexion échoue, le serveur DEVRAIT utiliser les procédures de [4] pour les serveurs afin de déterminer les adresse et port IP pour ouvrir la connexion et y envoyer la réponse.
- Autrement, si la valeur de champ d'en-tête Via contient un paramètre "maddr", la réponse DOIT être transmise aux adresses dont la liste figure là, en utilisant le port indiqué dans "send-by", ou au port 5060 si aucun n'est présent. Si l'adresse est une adresse en diffusion groupée, la réponse DEVRAIT être envoyée en utilisant le TTL indiqué dans le paramètre "ttl", ou avec un TTL de 1 si ce paramètre est absent.
- Autrement (pour les transport non fiables en envoi individuel), si le Via de tête a un paramètre "received", la réponse DOIT être envoyée à l'adresse figurant dans le paramètre "received", en utilisant le port indiqué dans la valeur "send-by", ou en utilisant le port 5060 si aucun n'est spécifié explicitement. Si ceci échoue, par exemple, on obtient une réponse ICMP "port inatteignable", les procédures de la Section 5 de [4] DEVRAIENT être utilisées pour

déterminer où envoyer la réponse.
- o Autrement, si il est étiqueté non récepteur, la réponse DOIT être envoyée à l'adresse indiquée par la valeur "send-by", en utilisant les procédures de la Section 5 de [4].

18.3 Tramage

Dans le cas des transports orientés message(tels que UDP), si le message a un champ d'en-tête Content-Length, le corps de message est supposé contenir cette quantité d'octets. Si il y a des octets additionnels dans le paquet de transport au-delà de la fin du corps de message, ils DOIVENT être supprimés. Si le paquet de transport se termine avant la fin du corps de message, c'est considéré comme une erreur. Si le message est une réponse, il DOIT être éliminé. Si le message est une demande, l'élément DEVRAIT générer une réponse 400 (Mauvaise demande). Si le message n'a pas de champ d'en-tête Content-Length, le corps de message est supposé se terminer à la fin du paquet de transport.

Dans le cas de transports orientés flux, tels que TCP, le champ d'en-tête Content-Length indique la taille du corps. Le champ d'en-tête Content-Length DOIT être utilisé avec les transports orientés flux.

18.4 Traitement d'erreurs

Le traitement d'erreurs est indépendant du fait que le message soit une demande ou une réponse.

Si l'utilisateur du transport demande qu'un message soit envoyé sur un transport non fiable, et que le résultat est une erreur ICMP, le comportement dépend du type d'erreur ICMP. Les erreurs d'hôte, de réseau, de port ou de protocole, non joignables, ou les problèmes d'erreur de paramètre DEVRAIENT être cause que la couche transport informe l'utilisateur du transport d'une défaillance de l'envoi. Les erreurs ICMP d'extinction de source et de dépassement de TTL DEVRAIENT être ignorées.

Si l'utilisateur du transport demande qu'une demande soit envoyée sur un transport fiable, et si le résultat est une défaillance de connexion, la couche transport DEVRAIT informer l'utilisateur du transport d'une défaillance d'envoi.

19 Composants de message communs

Certains composants de messages SIP apparaissent à diverses places au sein des messages SIP (et parfois, en dehors d'eux) et méritent une discussion distincte.

19.1 Indicateurs de ressource uniformes SIP et SIPS

Un URI SIP ou SIPS identifie une ressource de communication. Comme tous les URI, les URI SIP et SIPS peuvent être placés dans des pages web, des messages électroniques, ou de la littérature imprimée. Ils contiennent des informations suffisantes pour initialiser et maintenir une session de communication avec les ressources.

Les exemples de ressources de communication incluent ce qui suit :
- o un utilisateur de service en ligne
- o une apparition sur un téléphone multi-ligne
- o une boîte à lettres sur un système de messagerie
- o un numéro de RTPC à un service de passerelle
- o un groupe (tel que "ventes" ou "bureau d'accueil") dans une organisation

Un URI SIPS spécifie que les ressources sont contactées en confiance. Cela signifie, en particulier, que TLS doit être utilisé entre l'UAC et le domaine qui possède l'URI. A partir de là, les communications sécurisées sont utilisées pour joindre l'utilisateur, et le mécanisme spécifique de sécurité dépend de la politique du domaine. Toute ressource décrite par un URI SIP peut être "mis à niveau" en URI SIPS en changeant simplement le schéma, si on désire communiquer en toute sécurité avec cette ressource.

19.1.1 Composants d'URI SIP et SIPS

Les schémas "sip:" et "sips:" suivent les lignes directrices de la RFC 2396 [5]. Ils utilisent une forme similaire à l'URL mailto, permettant la spécification d'en-têtes de champs de demande SIP et le corps de message SIP. Cela rend possible la spécification du sujet, du type de support, ou de l'urgence des sessions initialisées en utilisant un URI sur une page web ou dans un message électronique. La syntaxe formelle pour un URI SIP ou SIPS est présentée à la Section 25. Sa formé générale, dans le cas d'un URI SIP , est :

sip:user:password@host:port;uri-parameters?headers

Le format pour un URI SIPS est le même, sauf que le schéma est "sips" à la place de sip. Ces jetons, et certains des jetons dans leurs expansions, ont la signification suivante :

user : Identifiant d'une ressource particulière chez l'hôte visé. Le terme "host" dans ce contexte se réfère fréquemment à un domaine. Le "userinfo" d'un URI consiste en son champ d'utilisateur (user), le champ de mot de passe *(password)*, et le signe @ qui les suit. La partie userinfo d'un URI est facultative et PEUT être absente lorsque l'hôte de destination n'a pas de notion des utilisateurs ou lorsque l'hôte lui-même est la ressource identifiée. Si le signe @ est présent dans un URI SIP ou SIPS, le champ d'utilisateur NE DOIT PAS être vide. Si l'hôte visé peut traiter les numéros de téléphone, par exemple, une passerelle téléphonique Internet, un champ d'abonné téléphonique, définis dans la RFC 2806 [9] PEUVENT être utilisés pour remplir le champ d'utilisateur. Il y a des règles d'échappement spéciales pour le codage des champs d'abonné téléphonique dans les URI SIP et SIPS décrits au paragraphe 19.1.2.

password : Mot de passe associé à l'utilisateur. Alors que la syntaxe d'URI SIP et SIPS permet que ce champ soit présent, son utilisation N'EST PAS RECOMMANDÉE, parce que le passage d'informations d'authentification en clair (comme les URI) s'est révélé être une menace pour la sécurité dans presque tous les cas où il a été utilisé. Par exemple, le transport d'un numéro PIN dans ce champ expose le PIN.
Noter que le champ password est juste une extension de la partie d'utilisateur. Les mises en œuvre qui ne souhaitent pas donner de signification particulière à la portion password du champ PEUVENT simplement traiter "user:password" comme une seule chaîne.

host : Hôte fournissant la ressource SIP. La partie hôte contient un nom de domaine pleinement qualifié ou une adresse numérique IPv4 ou IPv6. L'utilisation de la forme de nom de domaine pleinement qualifié est RECOMMANDÉE chaque fois que possible.

port : Numéro de port où la demande est à envoyer.

uri-parameters : Paramètres affectant une demande construite à partir de l'URI.
Les paramètres d'URI sont ajoutés après le composant hostport et sont séparés par un point-virgule.
Les paramètres d'URI prennent la forme :

parameter-name "=" parameter-value

Même si un nombre arbitraire de paramètres d'URI peut être inclus dans un URI, aucun nom de paramètre donné NE DOIT apparaître plus d'une fois.
Ce mécanisme extensible inclut les paramètres transport, maddr, ttl, user, method et lr.
Le paramètre transport détermine le mécanisme de transport à utiliser pour envoyer des messages SIP, comme spécifié dans [4]. SIP peut utiliser tout protocole de transport réseau. Les noms de paramètres sont définis pour UDP (RFC 768 [14]), TCP (RFC 761 [15]), et SCTP (RFC 2960 [16]). Pour un URI SIPS, le paramètre transport DOIT indiquer un transport fiable.
Le paramètre maddr indique l'adresse du serveur à contacter pour cet utilisateur, outrepassant toute adresse déduite du champ hôte. Lorsqu'un paramètre maddr est présent, les composants port et transport de l'URI s'appliquent à l'adresse indiquée dans la valeur du paramètre maddr. La référence [4] décrit la bonne interprétation de transport, maddr, et hostport afin d'obtenir l'adresse de destination, le port, et le transport pour l'envoi d'une demande. Le champ maddr a été utilisé comme une simple forme d'acheminement à source lâche. Il permet à un URI de spécifier un mandataire qui doit être traversé en route vers la destination. Continuer d'utiliser le paramètre maddr de cette façon est fortement déconseillé (les mécanismes qui le permettent sont déconseillés). Les mises en œuvre devraient utiliser à la place le mécanisme Route décrit dans le présent document, qui établit un ensemble des routes préexistant, si nécessaire (voir au paragraphe 8.1.1.1). Ceci fournit un plein URI pour décrire le nœud à traverser.
Le paramètre ttl détermine la valeur de la durée de vie du paquet UDP en diffusion groupée et ne DOIT être utilisé que si maddr est une adresse en diffusion groupée et si le protocole de transport est UDP. Par exemple, pour spécifier un appel à alice@atlanta.com en utilisant la diffusion groupée à 239.255.255.1 avec une ttl de 15, l'URI suivant serait utilisé :

sip:alice@atlanta.com;maddr=239.255.255.1;ttl=15

L'ensemble des chaînes valides d'abonnés téléphoniques est un sous-ensemble des chaînes d'utilisateurs valides. Le paramètre d'URI user existe pour distinguer les numéros de téléphone des noms d'utilisateur qui se trouvent ressembler à des numéros de téléphone. Si la chaîne d'utilisateur contient un numéro de téléphone formaté comme un abonné téléphonique, la valeur de paramètre d'utilisateur "phone" DEVRAIT être présente. Même sans ce paramètre, les receveurs d'URI SIP et SIPS PEUVENT interpréter la partie pré-@ comme un numéro de téléphone si des restrictions locales sur l'espace de nom pour le nom d'utilisateur le permettent.
La méthode de la demande SIP construite à partir de l'URI peut être spécifiée avec le paramètre méthode.
Le paramètre lr, lorsqu'il est présent, indique que l'élément responsable pour cette ressource met en œuvre les mécanismes d'acheminement spécifiés dans le présent document. Ce paramètre sera utilisé dans les URI que les mandataires placent dans les valeurs de champ d'en-tête Record-Route, et peut apparaître dans les URI d'un ensemble de routes préexistant.
Ce paramètre sert à réaliser la rétro-compatibilité avec les systèmes qui mettent en œuvre les mécanismes d'acheminement strict de la RFC 2543 et des projets rfc2543bis à bis-05. Un élément se préparant à envoyer une demande sur la base d'un URI qui ne contient pas ce paramètre peut supposer que l'élément receveur met en œuvre l'acheminement strict et reformate le message pour préserver les informations dans l'URI-de-demande.
Comme le mécanisme uri-parameter est extensible, les éléments SIP DOIVENT ignorer en silence tout uri-parameter qu'ils ne comprennent pas.

headers : Champs d'en-tête à inclure dans une demande construite à partir de l'URI.
Dans la demande SIP, les champs header peuvent être spécifiés avec le mécanisme "?" au sein d'un URI. Les noms et valeurs d'en-tête sont codés en paires hname = hvaleur séparées par l'éperluette (&). Le hname spécial "body" indique que la hvaleur associée est le corps de message de la demande SIP.

Le Tableau 1 résume l'utilisation des composants d'URI SIP et SIPS sur la base du contexte dans lequel apparaît l'URI. La colonne externe décrit les URI qui apparaissent n'importe où à l'extérieur d'un message SIP, par exemple sur une page web ou une carte de visite. Les entrées marquées "o" sont obligatoires, celles marquées "f" sont facultatives, et celles marquées "-" sont non autorisées. Les éléments qui traitent les URI DEVRAIT ignorer tout composant non autorisé présent. La seconde colonne indique la valeur par défaut d'un élément facultatif s'il n'est pas présent. "--" indique que l'élément n'est pas facultatif, ou n'a pas de valeur par défaut.

Les URI dans les champs d'en-tête Contact subissent différentes restrictions selon le contexte dans lequel apparaît le champ d'en-tête. Un ensemble s'applique aux messages qui établissent et maintiennent les dialogues (INVITE et sa réponse 200 (OK)). L'autre s'applique aux messages d'enregistrement et de redirection (REGISTER, sa réponse 200 (OK), et les réponse de classe 3xx à toute méthode).

19.1.2 Exigences d'évitement de caractères

	par défaut	Req.-URI	To	From	Contact enreg./redirect.	dialogueContact /R-R/Route	externe
user	--	f	f	f	f	f	f
password	--	f	f	f	f	f	f
host	--	o	o	o	o	o	o
port	(1)	f	-	-	f	f	f
user-param	ip	f	f	f	f	f	f
method	INVITE	-	-	-	-	-	f
maddr-param	--	f	-	-	f	f	f
ttl-param	1	f	-	-	f	-	f
transp.-param	(2)	f	-	-	f	f	f
lr-param	--	f	-	-	-	f	f
other-param	--	f	f	f	f	f	f
headers	--	-	-	-	f	-	f

(1) La valeur de port par défaut dépend du transport et du schéma. La valeur par défaut est 5060 pour sip: en utilisant UDP, TCP, ou SCTP. Elle est de 5061 pour sip: en utilisant TLS sur TCP et sips: sur TCP.
(2): Le transport par défaut dépend du schéma. Pour sip:, c'est UDP. Pour sips:, c'est TCP.

Tableau 1 : Utilisation et valeurs par défaut des composants d'URI pour valeurs de champ d'en-tête SIP, URI-de-demande et références

SIP suit les exigences et lignes directrices de la RFC 2396 [5] pour la définition de l'ensemble des caractères qui

doivent être esquivés dans un URI SIP, et utilise son mécanisme ""%" HEX HEX" pour l'esquivement. D'après la RFC 2396 [5] :

> L'ensemble des caractères réellement réservés au sein de tout composant d'URI donné est défini par ce composant. En général, un caractère est réservé si la sémantique de l'URI change lorsque le caractère est remplacé par son codage US-ASCII d'esquivement [5]. Les caractères US-ASCII exclus (RFC 2396 [5]), tels qu'espace, caractères de contrôle et caractères utilisés comme délimiteurs d'URI, DOIVENT aussi être esquivés. Les URI NE DOIVENT PAS contenir d'espace et de caractères de contrôle non esquivés.

Pour chaque composant, l'ensemble des expansions BNF valides définit exactement quels caractères peuvent apparaître non esquivés. Tous les autres caractères DOIVENT être esquivés.

Par exemple, "@" n'est pas dans l'ensemble des caractères dans le composant d'utilisateur, de sorte que l'utilisateur "j@s0n" doit avoir au moins le signe @ codé, comme dans "j%40s0n".

L'expansion des jetons de hname et hvaleur à la Section 25 montre que tous les caractères réservés d'URI dans les noms et valeurs de champ d'en-tête DOIVENT être esquivés.

Le sous-ensemble d'abonnés téléphoniques du composant d'utilisateur répond à des considérations d'esquive particulières. L'ensemble des caractères non réservés dans la description des abonnés téléphoniques de la RFC 2806 [9] contient un certain nombre de caractères dans divers éléments de syntaxe qui doivent être esquivés lorsqu'ils sont utilisés dans les URI SIP. Tout caractère survenant dans une description d'abonné téléphonique qui n'apparaît pas dans une expansion du BNF pour la règle d'utilisateur DOIT être esquivée.

Noter que l'esquive de caractère n'est pas autorisée dans le composant host d'un URI SIP ou SIPS (le caractère % n'est pas valide dans son expansion). Ceci va vraisemblablement changer à l'avenir avec la finalisation des exigences pour les noms de domaines internationalisés. Les mises en œuvre en cours NE DOIVENT PAS essayer d'améliorer la robustesse en traitant les caractères esquivés reçus dans le composant host comme littéralement équivalents à leur contrepartie non esquivée. Le comportement requis pour satisfaire aux exigences de IDN peut être significativement différent.

19.1.3 Exemples d'URI SIP et SIPS

sip:alice@atlanta.com
sip:alice:secretword@atlanta.com;transport=tcp
sips:alice@atlanta.com?subject=project%20x&priority=urgent
sip:+1-212-555-1212:1234@gateway.com;user=phone
sips:1212@gateway.com
sip:alice@192.0.2.4
sip:atlanta.com;method=REGISTER?to=alice%40atlanta.com
sip:alice;day=tuesday@atlanta.com

Le dernier échantillon d'URI ci-dessus a une valeur de champ user "alice;day=tuesday". Les règles d'esquive définies ci-dessus permettent que deux points apparaissent non esquivés dans ce champ. Pour les besoins de ce protocole, le champ est opaque. La structure de cette valeur n'est utile qu'à l'élément SIP responsable des ressources.

19.1.4 Comparaison d'URI

Certaines opérations dans la présente spécification requièrent de déterminer si deux URI SIP ou SIPSs sont équivalents. Dans la présente spécification, les registraires ont besoin de comparer les liens dans les URI Contact dans des demandes REGISTER (voir au paragraphe 10.3.). L'égalité des URI SIP et SIPS est comparée selon les règles suivantes :

- Un URI SIP et un URI SIPS ne sont jamais équivalents.
- La comparaison du userinfo des URI SIP et SIPS est sensible à la casse. Ceci inclut les userinfo qui contiennent des mots de passe ou sont formatés comme des numéros d'abonné téléphonique. La comparaison de tous les autres composants de l'URI est insensible à la casse sauf explicitement défini autrement.
- L'ordre des paramètres et champs d'en-tête n'est pas significatif dans la comparaison des URI SIP et SIPS.
- Les caractères autres que ceux qui sont dans l'ensemble "réservé" (voir la RFC 2396 [5]) sont équivalents à leur codage ""%" HEX HEX".
- Une adresse IP qui est le résultat d'une recherche DNS de nom d'hôte ne correspond pas à ce nom d'hôte.

o Pour que deux URI soient égaux, les composants user, password, host, et port doivent correspondre.

Un URI omettant le composant utilisateur ne correspondra pas à un URI qui en inclut un. Un URI omettant le composant mot de passe ne correspondra pas avec un URI qui en inclut un.

Un URI qui omet tout composant avec une valeur par défaut ne correspondra pas à un URI contenant explicitement ce composant avec sa valeur par défaut. Par exemple, un URI qui omet le composant facultatif port ne correspondra pas à un URI qui déclare explicitement le port 5060. Il en est de même pour les composants de paramètre transport, ttl, user, et method.

Définir sip:user@host comme n'étant pas équivalent à sip:user@host:5060 est un changement par rapport à la RFC 2543. Lorsqu'on déduit les adresses des URI, des adresses équivalentes sont attendues d'URI équivalents. L'URI sip:user@host:5060 sera toujours résolu au port 5060. L'URI sip:user@host peut se résoudre à d'autres ports à travers les mécanismes SRV de DNS décrits en détail dans [4].

o Les composants d'URI uri-parameter sont comparés comme suit :
- Tout uri-parameter apparaissant dans les deux URI doit correspondre.
- Un uri-parameter user, ttl, ou method apparaissant dans un seul URI ne correspondra jamais, même si il contient la valeur par défaut.
- Un URI qui inclut un paramètre maddr ne correspondra pas à un URI qui ne contient pas de paramètre maddr.
- Tous les autres uri-parameter apparaissant dans un seul URI sont ignorés dans la comparaison des URI.

o Les composants d'en-tête d'URI ne sont jamais ignorés. Tout composant d'en-tête présent DOIT être présent dans les deux URI et correspondre pour les URI à comparer. Les règles de correspondance sont définies pour chaque champ d'en-tête à la Section 20.

Les URI au sein de chacun des ensembles suivants sont équivalents :
sip:%61lice@atlanta.com;transport=TCP et sip:alice@AtLanTa.CoM;Transport=tcp
sip:carol@chicago.com et sip:carol@chicago.com;newparam=5 et sip:carol@chicago.com;security=on

sip:biloxi.com;transport=tcp;méthode=REGISTER?to=sip:bob%40biloxi.com
sip:biloxi.com;méthode=REGISTER;transport=tcp?to=sip:bob%40biloxi.com

sip:alice@atlanta.com?subject=project%20x&priority=urgent
sip:alice@atlanta.com?priority=urgent&subject=project%20x

Les URI au sein de chacun des ensembles suivants ne sont pas équivalents :
SIP:ALICE@AtLanTa.CoM;Transport=udp
sip:alice@AtLanTa.CoM;Transport=UDP (noms d'utilisateur différents)

sip:bob@biloxi.com (peut se résoudre sur différents ports)
sip:bob@biloxi.com:5060

sip:bob@biloxi.com (peut se résoudre sur différents transports)
sip:bob@biloxi.com;transport=udp

sip:bob@biloxi.com (peut se résoudre sur différents ports et transports)
sip:bob@biloxi.com:6000;transport=tcp

sip:carol@chicago.com
sip:carol@chicago.com?Subject=next%20meeting (composant d'en-tête différent)

sip:bob@phone21.boxesbybob.com
sip:bob@192.0.2.4 (même si c'est ce à quoi se résout phone21.boxesbybob.com)

Noter que l'égalité n'est pas transitive :
o sip:carol@chicago.com et sip:carol@chicago.com;security=on sont équivalents
o sip:carol@chicago.com et sip:carol@chicago.com;security=off sont équivalents
o sip:carol@chicago.com;security=on et sip:carol@chicago.com;security=off ne sont pas équivalents

19.1.5 Formation de demandes à partir d'un URI

Une mise en œuvre doit faire attention lors de la formation de demandes directement à partir d'un URI. Les URI tirés de cartes de visite, de pages web, et même de sources intérieures au protocole telles que des contacts enregistrés peuvent contenir des champs d'en-tête ou parties de corps inappropriées.

Une mise en œuvre DOIT inclure tout paramètre transport, maddr, ttl, ou user fourni dans l'URI-de-demande de la demande formée. Si l'URI contient un paramètre method, sa valeur DOIT être utilisée comme méthode de la demande. Le paramètre method NE DOIT PAS être placé dans l'URI-de-demande. Les paramètres d'URI inconnus DOIVENT être placés dans l'URI-de-demande du message.

Une mise en œuvre DEVRAIT traiter la présence de tout en-tête ou partie de corps dans l'URI comme le désir de les inclure dans le message, et choisir d'honorer la demande composant par composant.

Une mise en œuvre NE DEVRAIT PAS honorer les champs d'en-tête From, Call-ID, CSeq, Via, et Record-Route visiblement dangereux.

Une mise en œuvre NE DEVRAIT honorer AUCUNE valeur de champ d'en-tête Route demandée afin de n'être pas utilisée comme agent involontaire d'attaques malveillantes.

Une mise en œuvre NE DEVRAIT PAS honorer des demandes d'inclusion de champs d'en-tête qui pourraient causer de fausses annonces de sa localisation ou de ses capacités. Ceci inclut : Accept, Accept-Encoding, Accept-Language, Allow, Contact (dans son utilisation de dialogue), Organization, Supported, et User-Agent.

Une mise en œuvre DEVRAIT vérifier la pertinence de tout champ d'en-tête descriptif demandé parmi lesquels : Content-Disposition, Content-Encoding, Content-Language, Content-Length, Content-Type, Date, Mime-Version, et Timestamp.

Si la demande formée de la construction d'un message à partir d'un URI donné n'est pas une demande SIP valide, l'URI est invalide. Une mise en œuvre NE DOIT PAS continuer par la transmission de la demande. Elle devrait à la place poursuivre le cours de l'action due à un URI invalide dans le contexte où il survient.

La demande construite peut être invalide de nombreuses façons, qui incluent, mais sans s'y limiter, l'erreur de syntaxe dans le champ d'en-tête, une combinaison invalide des paramètres d'URI, ou une description incorrecte du corps de message.

L'envoi d'une demande formée à partir d'un URI donné peut exiger des capacités indisponibles pour la mise en œuvre. L'URI peut indiquer l'utilisation d'un transport ou extension non mis en oeuvre, par exemple. Une mise en œuvre DEVRAIT refuser d'envoyer ces demandes plutôt que de les modifier pour correspondre à leurs capacités. Une mise en œuvre NE DOIT PAS envoyer de demande requérant une extension qu'elle ne prend pas en charge.

Par exemple, une telle demande peut être formée grâce à la présence d'un paramètre d'en-tête Require ou un paramètre de méthode d'URI inconnue ou avec une valeur explicitement non prise en charge.

19.1.6 Mise en rapport des URI de SIP et des URL téléphoniques

Lorsqu'un URL tel (RFC 2806 [9]) est converti en URI SIP ou SIPS, la portion abonné téléphonique toute entière d'un tel URL, incluant tous les paramètres, est placée dans la partie useurinfo de l'URI SIP ou SIPS.

Et donc, tel:+358-555-1234567;postd=pp22 devient
sip:+358-555-1234567;postd=pp22@foo.com;user=phone
ou
sips:+358-555-1234567;postd=pp22@foo.com;user=phone
et non
sip:+358-555-1234567@foo.com;postd=pp22;user=phone
ou
sips:+358-555-1234567@foo.com;postd=pp22;user=phone

En général, les URL équivalents à "tel" convertis en URI SIP ou SIPS de cette façon peuvent ne pas produire les URI

SIP ou SIPS équivalents. Le userinfo des URI SIP et SIPS sont comparés comme une chaîne sensible à la casse. Les variations dans les portions insensibles à la casse des URL tel et la remise en ordre des paramètres d'URL tel n'affectent pas l'équivalence des URL tel, mais affecte l'équivalence des URI SIP formés à partir d'eux.

Par exemple,
tel:+358-555-1234567;postd=pp22
tel:+358-555-1234567;POSTD=PP22
sont équivalents, alors que
sip:+358-555-1234567;postd=pp22@foo.com;user=phone
sip:+358-555-1234567;POSTD=PP22@foo.com;user=phone
ne le sont pas.

De même,
tel:+358-555-1234567;postd=pp22;isub=1411
tel:+358-555-1234567;isub=1411;postd=pp22
sont équivalents, alors que
sip:+358-555-1234567;postd=pp22;isub=1411@foo.com;user=phone
sip:+358-555-1234567;isub=1411;postd=pp22@foo.com;user=phone
ne le sont pas.

Pour atténuer ce problème, les éléments de construction des champs d'abonné téléphonique à placer dans la partie userinfo d'un URI SIP ou SIPS DEVRAIENT ramener toute portion insensible à la casse d'abonné téléphonique en minuscules, et ordonner les paramètres d'abonné téléphonique de façon lexicologique par nom de paramètre, excepté les sous adresses RNIS et les numéros de poste, qui viennent d'abord et dans cet ordre. (Tous les composants d'un URL tel, excepté les paramètres pour extension future, sont définis pour être comparés indépendamment de la casse.)

Suivant cette suggestion,
tel:+358-555-1234567;postd=pp22
tel:+358-555-1234567;POSTD=PP22
deviennent tous deux
sip:+358-555-1234567;postd=pp22@foo.com;user=phone
et
tel:+358-555-1234567;tsp=a.b;phone-context=5
tel:+358-555-1234567;phone-context=5;tsp=a.b
deviennent tous deux
sip:+358-555-1234567;phone-context=5;tsp=a.b@foo.com;user=phone

19.2 Étiquettes d'option

Les étiquettes d'option sont des identifiants univoques utilisés pour désigner de nouvelles options (extensions) dans SIP. Ces étiquettes sont utilisées dans les champs d'en-tête Require (paragraphe 20.32), Proxy-Require (paragraphe 20.29), Supported (paragraphe 20.37) et Unsupported (paragraphe 20.40). Noter que ces options apparaissent comme paramètres dans ces champs d'en-tête sous une forme option-étiquette = jeton (voir à la Section 25 la définition de jeton).

Les étiquettes d'option sont définies dans les RFC établissant des normes. Ceci est un changement par rapport aux pratiques du passé, et est institué pour assurer la continuité de l'interopérabilité entre fabricants (voir la discussion aux paragraphes 20.32 et 20.37). Un registre IANA d'étiquettes d'option est utilisé pour assurer un référencement facile.

19.3 Étiquettes

Le paramètre "tag" *(étiquette)* est utilisé dans les champ d'en-tête To et From des messages SIP. Il sert de mécanisme général d'identification de dialogue, et c'est une combinaison du Call-ID avec deux étiquettes, une de chaque participant au dialogue. Lorsqu'un agent utilisateur envoie une demande en-dehors d'un dialogue, il contient seulement une étiquette From, fournissant "la moitié" de l'ID de dialogue. Le dialogue est complété à partir de la ou des réponses, chacune d'elle contribuant à la seconde moitié dans le champ d'en-tête To. Le fourchement des demandes SIP signifie que plusieurs dialogues peuvent être établis à partir d'une seule demande. Cela aussi explique le besoin d'un identifiant de dialogue à deux faces ; sans une contribution des receveurs, le générateur ne pourrait lever les ambiguïtés sur les multiples dialogues établis à partir d'une seule demande.

Lorsque une étiquette est générée par un agent utilisateur pour insertion dans une demande ou réponse, il DOIT être mondialement unique et cryptographiquement aléatoire avec au moins 32 bits à caractère aléatoire. Une propriété de cette exigence de sélection est qu'un agent utilisateur placera une étiquette différente dans l'en-tête From d'un INVITE de celle qu'il placerait dans l'en-tête To de la réponse au même INVITE. Ceci est nécessaire pour qu'un agent utilisateur s'invite lui-même à une session, un cas courant pour "épingler" les appels dans les passerelles RTPC. De même, deux INVITE pour des appels différents auront des étiquettes From différentes, et deux réponses pour des appels différents auront des étiquettes To différentes.

A côté des exigences d'unicité mondiale, l'algorithme de génération d'une étiquette est spécifique de la mise en œuvre. Les étiquettes sont utiles dans les systèmes tolérants, où un dialogue est destiné à être récupéré sur un serveur de remplacement après une défaillance. Un UAS peut choisir l'étiquette de telle façon qu'une sauvegarde puisse reconnaître qu'une demande fait part d'un dialogue sur le serveur défaillant, et donc déterminer qu'il devrait essayer de récupérer le dialogue et tout autre état qui lui est associé.

20 Champs d'en-tête

La syntaxe générale pour les champs d'en-tête est traitée au paragraphe 7.3. La présente section fait la liste de l'ensemble complet des champs d'en-tête avec des notes sur la syntaxe, la signification, et l'utilisation. Tout au long de la présente section, on utilise [HX.Y] pour se référer au paragraphe X.Y de la spécification HTTP/1.1 actuelle (RFC 2616 [8]). On donne des exemples de chaque champ d'en-tête.

Les informations sur les champs d'en-tête en rapport avec les méthodes et le traitement par le mandataire sont résumées dans les Tableaux 2 et 3.

La colonne "où" décrit les types de demande et réponse dans lesquels le champ d'en-tête peut être utilisé. Les valeurs dans cette colonne sont :
R : le champ d'en-tête peut seulement apparaître dans des demandes ;
r : le champ d'en-tête peut seulement apparaître dans des réponses ;
2xx, 4xx, etc. : une valeur ou gamme numérique indique les codes de réponse avec lesquels le champ d'en-tête peut être utilisé ;
c : le champ d'en-tête est copié de la demande à la réponse. Une entrée vide dans la colonne "où" indique que le champ d'en-tête peut être présent dans toutes les demandes et réponses.

La colonne "mandataire" décrit les opérations qu'un mandataire peut effectuer sur un champ d'en-tête :
a : un mandataire peut ajouter ou enchaîner le champ d'en-tête s'il n'est pas présent.
m : un mandataire peut modifier une valeur de champ d'en-tête existante.
d : un mandataire peut supprimer une valeur de champ d'en-tête.
r : un mandataire doit être capable de lire le champ d'en-tête, et donc ce champ d'en-tête ne peut être chiffré.

Les six colonnes suivantes se rapportent à la présence d'un champ d'en-tête dans une méthode :
c : conditionnel ; les exigences sur le champ d'en-tête dépendent du contexte du message.
m : le champ d'en-tête est obligatoire *(mandatory)*.
m* : le champ d'en-tête DEVRAIT être envoyé, mais les clients/serveurs doivent être prêts à recevoir les messages sans ce champ d'en-tête.
o : le champ d'en-tête est facultatif *(optional)*.
t : le champ d'en-tête DEVRAIT être envoyé, mais les clients/serveurs doivent être prêts à recevoir les messages sans ce champ d'en-tête. Si un protocole fondé sur le flux (tel que TCP) est utilisé comme transport, le champ d'en-tête DOIT être envoyé.
* : le champ d'en-tête est exigé si le corps de message n'est pas vide. Voir les détails aux paragraphes 20.14, 20.15 et 7.4.
- : le champs d'en-tête n'est pas applicable.

"facultatif" signifie qu'un élément PEUT inclure le champ d'en-tête dans une demande ou réponse, et un agent utilisateur PEUT ignorer le champ d'en-tête s'il est présent dans la demande ou réponse (l'exception à cette règle est le champ d'en-tête Require exposé au paragraphe 20.32). Un champ d'en-tête "mandatory" DOIT être présent dans une demande, et DOIT être compris par l'UAS qui reçoit la demande. Un champ d'en-tête obligatoire de réponse DOIT être présent dans la réponse, et le champ d'en-tête DOIT être compris par l'UAC qui traite la réponse. "Non applicable" signifie que le champ d'en-tête NE DOIT PAS être présent dans une demande. Si il en est placé un par erreur dans une demande, il DOIT être ignoré par l'UAS qui reçoit la demande. De même, un champ d'en-tête qualifié de "non applicable" pour une réponse signifie que l'UAS NE DOIT PAS placer le champ d'en-tête dans la réponse, et l'UAC

DOIT ignorer le champ d'en-tête dans la réponse.

Un agent utilisateur DEVRAIT ignorer les paramètres d'en-tête d'extension qu'il ne comprend pas.

Une forme compacte de certains noms de champ d'en-tête courants est aussi définie pour l'utiliser lorsque la taille totale du message pose problème.

Les champs d'en-tête Contact, From, et To contiennent un URI. Si l'URI contient une virgule, un point d'interrogation ou deux ponts, l'URI DOIT être contenu dans des crochets angulaires (< et >). Tous les paramètres d'URI sont contenus dans ces crochets. Si l'URI n'est pas inclus dans des crochets angulaires, tout paramètre délimité par deux points est un paramètre d'en-tête, et non un paramètre d'URI.

20.1 Accept

Le champ d'en-tête Accept suit la syntaxe définie dans [H14.1]. La sémantique sera aussi identique, avec l'exception que si aucun champ d'en-tête Accept n'est présent, le serveur DEVRAIT supposer une valeur par défaut de application/sdp.

Un champ d'en-tête Accept vide signifie qu'aucun format n'est acceptable.

Exemple : Accept: application/sdp;level=1, application/x-private, text/html

Champs d'en-tête	**Où**	**mandataire**	**ACK**	**BYE**	**CAN**	**INV**	**OPT**	**REG**
Accept	R		-	o	-	o	m*	o
Accept	2xx		-	-	-	o	m*	o
Accept	415		-	c	-	c	C	c
Accept-Encoding	R		-	o	-	o	O	o
Accept-Encoding	2xx		-	-	-	o	m*	o
Accept-Encoding	415		-	c	-	c	C	c
Accept-Language	R		-	o	-	o	O	o
Accept-Language	2xx		-	-	-	o	m*	o
Accept-Language	415		-	c	-	c	C	c
Alert-Info	R	ar	-	-	-	o	-	-
Alert-Info	180	ar	-	-	-	o	-	-
Allow	R		-	o	-	o	O	o
Allow	2xx		-	o	-	m*	m*	o
Allow	r		-	o	-	o	O	o
Allow	405		-	m	-	m	M	m
Authentication-Info	2xx		-	o	-	o	O	o
Authorization	R		o	o	o	o	O	o
Call-ID	c	r	m	m	m	m	M	m
Call-Info		ar	-	-	-	o	O	o
Contact	R		o	-	-	m	O	o
Contact	1xx		-	-	-	o	-	-
Contact	2xx		-	-	-	m	O	o
Contact	3xx	d	-	o	-	o	O	o
Contact	485		-	o	-	o	O	o
Content-Disposition			o	o	-	o	O	o
Content-Encoding			o	o	-	o	O	o
Content-Language			o	o	-	o	O	o
Content-Length		ar	t	t	t	t	T	t
Content-Type			*	*	-	*	*	*
Cseq	c	r	m	m	m	m	M	m
Date		a	o	o	o	o	O	o
Error-Info	300-699	a	-	o	o	o	O	o
Expires			-	-	-	o	-	o
From	c	r	m	m	m	m	M	m
In-Reply-To	R		-	-	-	o	-	-
Max-Forwards	R	amr	m	m	m	m	M	m
Min-Expires	423	-	-	-	-	-	M	
MIME-Version			o	o	-	o	O	o
Organization		ar	-	-	-	o	O	o
Priority	R	ar	-	-	-	o	-	-
Proxy-Authenticate	407	ar	-	m	-	m	M	m
Proxy-Authenticate	401	ar	-	o	o	o	O	o

Proxy-Authorization	R	dr	o	o	-	o	O	o
Proxy-Require	R	ar	-	o	-	o	O	o
Record-Route	R	ar	o	o	o	o	O	-
Record-Route	2xx,18x	mr	-	o	o	o	O	-
Reply-To			-	-	-	o	-	-
Require		ar	-	c	-	c	C	c
Retry-After	404,413,480,486		-	o	o	o	O	o
	500,503		-	o	o	o	O	o
	600,603		-	o	o	o	O	o
Route	R	adr	c	c	c	c	C	c
Server	r		-	o	o	o	O	o
Subject	R		-	-	-	o	-	-
Supported	R		-	o	o	m*	O	o
Supported	2xx		-	o	o	m*	m*	o
Timestamp			o	o	o	o	O	o
To	c(1)	r	m	m	m	m	M	m
Unsupported	420		-	m	-	m	M	m
User-Agent			o	o	o	o	O	o
Via	R	amr	m	m	m	m	M	m
Via	rc	dr	m	m	m	m	M	m
Warning	r		-	o	o	o	O	o
WWW-Authenticate	401	ar	-	m	-	m	M	m
WWW-Authenticate	407	ar	-	o	-	o	O	o

(1) : copiés avec l'ajout possible d'étiquette

Tableau 3 : Résumé des champs d'en-tête;

20.2 Accept-Encoding

Le champ d'en-tête Accept-Encoding *(Codage-accepté)* est similaire à Accept, mais restreint les codages de contenu [H3.5] qui sont acceptables dans la réponse. Voir [H14.3]. La sémantique dans SIP est identique à celle définie dans [H14.3].

Un champ d'en-tête Accept-Encoding vide est permis. Il est équivalent à Accept-Encoding: identity, c'est-à-dire, seul le codage de l'identité, signifiant pas de codage, est permis.

Si aucun champ d'en-tête Accept-Encoding n'est présent, le serveur DEVRAIT supposer une valeur par défaut de identity.

Ceci diffère légèrement de la définition HTTP, qui indique que lorsqu'il est absent, tout codage peut être utilisé, mais le codage de l'identité est préféré.

Exemple : Accept-Encoding: gzip

20.3 Accept-Language

Le champ d'en-tête Accept-Language *(Langage-accepté)* est utilisé dans des demandes pour indiquer les langages préférés pour les phrases de cause, de description de sessions, ou les réponses d'état portées comme corps de message dans la réponse. Si aucun champ d'en-tête Accept-Language n'est présent, le serveur DEVRAIT supposer que tous les langages sont acceptables pour le client.

Le champ d'en-tête Accept-Language suit la syntaxe définie dans [H14.4]. Les règles pour ordonner les langages sur la base du paramètre "q" s'appliquent aussi dans SIP.

Exemple : Accept-Language: da, en-gb;q=0.8, en;q=0.7

20.4 Alert-Info

Lorsqu'il est présent dans une demande INVITE, le champ d'en-tête Alert-Info *(informations d'alerte)* spécifie une tonalité de sonnerie de remplacement à l'UAS. Lorsqu'il est présent dans une réponse 180 (Sonnerie), le champ d'en-tête Alert-Info spécifie une tonalité de retour d'appel de remplacement à l'UAC. Une utilisation typique est pour un

mandataire d'insérer ce champ d'en-tête pour fournir une caractéristique de sonnerie distincte.

Le champ d'en-tête Alert-Info peut introduire des risques pour la sécurité. Ces risques et la façon de les traiter sont exposés au paragraphe 20.9, qui traite du champ d'en-tête Call-Info qui présente des risques identiques.

De plus, un utilisateur DEVRAIT être capable de désactiver cette caractéristique à volonté.

Ceci aide à prévenir les perturbations qui pourraient résulter de l'utilisation de ce champ d'en-tête par des éléments douteux

Exemple : Alert-Info: <http://www.example.com/sounds/moo.wav>

20.5 Allow

Le champ d'en-tête Allow fait la liste des ensembles de méthodes prises en charge par l'agent utilisateur qui génère le message.

Toutes les méthodes, y compris ACK et CANCEL, comprises par l'agent utilisateur DOIVENT être incluses dans la liste des méthodes dans le champ d'en-tête Allow, lorsqu'il est présent. L'absence d'un champ d'en-tête Allow NE DOIT PAS être interprétée comme signifiant que l'agent utilisateur qui envoie le message n'accepte aucune méthode. Elle implique plutôt que l'agent utilisateur ne fournit aucune information sur les méthodes qu'il accepte.

Fournir un champ d'en-tête Allow dans les réponses aux méthodes autres que OPTIONS réduit le nombre de messages nécessaires.

Exemple : Allow: INVITE, ACK, OPTIONS, CANCEL, BYE

20.6 Authentication-Info

Le champ d'en-tête Authentication-Info *(Informations d'authentification)* sert à l'authentification mutuelle avec HTTP Digest. Un UAS PEUT inclure ce champ d'en-tête dans une réponse 2xx à une demande dont l'authentification a été réussie en utilisant digest sur la base du champ d'en-tête Authorization.

La syntaxe et la sémantique suivent celle spécifiée dans la RFC 2617 [17].

Exemple : Authentication-Info: nextnonce="47364c23432d2e131a5fb210812c"

20.7 Authorization

Le champs d'en-tête Authorization contient les accréditifs d'authentification d'un agent utilisateur. Le paragraphe 22.2 évoque l'utilisation du champ d'en-tête Authorization , et le paragraphe 22.4 décrit sa syntaxe et sa sémantique lorsqu'il est utilisé avec l'authentification HTTP.

Ce champ d'en-tête, ainsi que Proxy-Authorization, rompt avec les règles générales sur les valeurs multiples de champ d'en-tête. Bien que ce ne soit pas une liste séparée par des virgules, ce nom de champ d'en-tête peut être présent plusieurs fois, et NE DOIT PAS être combiné en une seule ligne d'en-tête en utilisant les règles habituelles décrites au paragraphe 7.3.

Dans l'exemple ci-dessous, il n'y a pas de guillemets autour du paramètre Digest :

Authorization: Digest username="Alice", realm="atlanta.com",
nonce="84a4cc6f3082121f32b42a2187831a9e",
réponse="7587245234b3434cc3412213e5f113a5432"

20.8 Call-ID

Le champ d'en-tête Call-ID *(identifiant d'appel)* identifie de façon univoque une invitation particulière ou tous les enregistrements d'un client particulier. Une seule conférence multimédia peut donner naissance à plusieurs appels avec différents Call-ID, par exemple, si un utilisateur invite un seul individu plusieurs fois à la même conférence (à long terme). Les Call-ID sont sensibles à la casse et sont simplement comparés octet par octet.
La forme compacte du champ d'en-tête Call-ID est i.

Exemples : Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@biloxi.com
i:f81d4fae-7dec-11d0-a765-00a0c91e6bf6@192.0.2.4

20.9 Call-Info

Le champ d'en-tête Call-Info *(Informations d'appel)* donne des informations supplémentaires sur l'appelant ou sur l'appelé, selon qu'il se trouve dans une demande ou une réponse. L'objet de l'URI est décrit par le paramètre "purpose" *(objet)*. Le paramètre "icon" désigne une image convenable comme icône de représentation de l'appelant ou de l'appelé. Le paramètre "info" décrit l'appelant ou l'appelé en général, par exemple, à travers une page web. Le paramètre "card" fournit une carte de visite, par exemple, en format vCard [36] ou LDIF [37]. Des jetons supplémentaires peuvent être enregistrés en utilisant IANA et les procédures de la Section 27.

L'utilisation du champ d'en-tête Call-Info peut présenter des risques pour la sécurité. Si un appelé va chercher les URI fournis par un appelant malveillant, il peut subir le risque d'afficher un contenu inapproprié ou offensant, dangereux ou illégal, et ainsi de suite. Donc, il est RECOMMANDÉ qu'un agent utilisateur n'éclaircisse les informations du champ d'en-tête Call-Info que s'il peut vérifier l'authenticité de l'élément d'origine du champ d'en-tête et faire confiance à cet élément. Il n'est pas nécessaire que ce soit l'agent utilisateur homologue ; un mandataire peut insérer ce champ d'en-tête dans des demandes.

Exemple : Call-Info: <http://wwww.example.com/alice/photo.jpg> ;purpose=icon,
<http://www.example.com/alice/> ;purpose=info

20.10 Contact

Une valeur de champ d'en-tête Contact fournit un URI dont la signification dépend du type de demande ou réponse qui y figure.

Une valeur de champ d'en-tête Contact peut contenir un nom d'affichage, un URI avec les paramètres d'URI, et des paramètres d'en-tête.

Le présent document définit les paramètres Contact "q" et "expires". Ces paramètres ne sont utilisé que lorsque Contact est présent dans une demande ou réponse REGISTER, ou dans une réponse 3xx. Des paramètres supplémentaires peuvent être définis dans d'autres spécifications.

Lorsque la valeur de champ d'en-tête contient un nom d'affichage, l'URI incluant tous les paramètres d'URI est inscrit dans "<" et ">". Si aucun <"" et ">" n'est présent, tous les paramètres après l'URI sont des paramètres d'en-tête, et non les paramètres d'URI. Le nom d'affichage peut être des jetons, ou une chaîne entre guillemets, si un jeu de caractères plus étendu est désiré.

Même si le "display-name" est vide, la forme "name-addr" DOIT être utilisée si le "addr-spec" contient une virgule, un point-virgule, ou un point d'interrogation. Il peut y avoir ou n'y avoir pas d'espace insécable entre le display-name et le "<".

Ces règles pour l'analyse grammaticale du nom d'affichage, de l'URI et des paramètres d'URI, et les paramètres d'en-tête s'appliquent aussi pour les champs d'en-tête To et From.

Le champ d'en-tête Contact a un rôle similaire à celui du champ d'en-tête Location dans HTTP. Cependant, les champs d'en-tête HTTP n'admettent qu'une seule adresse, qui n'est pas entre guillemets. Comme les URI peuvent contenir des virgules et des point-virgules comme caractères réservés, ils peuvent être pris, respectivement, pour des délimiteur d'en-tête ou de paramètre.

La forme compacte du champ d'en-tête Contact est m (pour "moved").

Exemples : Contact: "Mr. Watson" <sip:watson@worcester.bell-telephone.com> ;q=0.7; expires=3600,
"Mr. Watson" <mailto:watson@bell-telephone.com> ;q=0.1
m: <sips:bob@192.0.2.4>;expires=60

20.11 Content-Disposition

Le champ d'en-tête Content-Disposition *(Disposition-du-contenu)* décrit comment le corps de message ou, pour les messages multiparties, une partie de corps de message doit être interprétée par l'UAC ou l'UAS. Ce champ d'en-tête SIP étend le Content-Type de MIME (RFC 2183 [18]).

Plusieurs nouveaux "disposition-types" de l'en-tête Content-Disposition sont définis par SIP. La valeur "session" indique que la partie corps décrit une session, pour les appels ou des supports précoces (pré appel). La valeur "render" indique que la partie corps devrait être affichée ou rendue autrement à l'utilisateur. Noter que la valeur "render" est utilisée plutôt que "inline" pour éviter la connotation que le corps MIME est affiché comme une partie du rendu du message entier (dans la mesure où les corps MIME des messages SIP sont souvent non affichés chez l'utilisateur). Pour la rétrocompatibilité, si le champ d'en-tête Content-Disposition manque, le serveur DEVRAIT supposer que les corps d'application/sdp Content-Type ont la disposition "session", alors que d'autres types de contenu ont "render".

Le type de disposition "icon" indique que la partie corps contient une image convenable comme représentation imagée de l'appelant ou de l'appelé qui pourra être rendue de façon informationnelle par un agent utilisateur lorsqu'un message a été reçu, ou de façon persistante lorsqu'un dialogue s'instaure. La valeur "alert" indique que la partie corps contient des informations, telles qu'un clip audio, qui devrait être rendu par l'agent utilisateur dans une tentative d'alerter l'utilisateur à la réception d'une demande, généralement une demande qui initialise un dialogue ; ce corps d'alerte pourrait, par exemple, être rendu par une sonnerie pour un appel téléphonique après qu'une réponse provisoire 180 Sonnerie a été envoyée.

Tout corps MIME avec un "disposition-type" qui restitue le contenu à l'utilisateur devrait n'être traité que lorsqu'un message a été correctement authentifié.

Le paramètre de traitement, handling-param, décrit comment l'UAS devrait réagir si il reçoit un corps de message dont le type de contenu ou de disposition n'est pas compris. Le paramètre a des valeurs définies de "optional" et "required". Si le paramètre de traitement est manquant, la valeur "required" DEVRAIT être supposée. Le paramètre de traitement est décrit dans la RFC 3204 [19].

Si ce champs d'en-tête est manquant, le type MIME détermine la disposition de contenu par défaut. Si il n'y en a aucune, "render" est supposé.

Exemple : Content-Disposition: session

20.12 Content-Encoding

Le champ d'en-tête Content-Encoding *(Codage-du-contenu)* est utilisé pour modifier le "media-type". Lorsqu'il est présent, sa valeur indique quels codages additionnels de contenu ont été appliqués au corps d'entité, et donc quel mécanisme de décodage DOIT être appliqué afin d'obtenir le media-type référencé par le champ d'en-tête Content-Type. Content-Encoding est principalement utilisé pour permettre de compresser un corps de message sans perdre l'identité de son type de support sous-jacent.

Si plusieurs codages ont été appliqués à un corps d'entité, la liste des codages de contenu DOIT figurer dans l'ordre dans lequel ils ont été appliqués.

Toutes les valeurs de content-coding sont insensibles à la casse. L'IANA agit comme teneur de registre pour les jetons de valeur de content-coding. Voir [H3.5] pour une définition de la syntaxe pour content-coding.

Les clients PEUVENT appliquer les codages de contenu au corps dans les demandes. Un serveur PEUT appliquer les codages de contenu aux corps dans les réponses. Le serveur DOIT seulement utiliser les codages listés dans le champ d'en-tête Accept-Encoding dans la demande.
La forme compacte du champ d'en-tête Content-Encoding est e.

Exemples : Content-Encoding: gzip
e: tar

20.13 Langage-du-contenu

Voir [H14.12]. Exemple : Content-Language: fr

20.14 Content-Length

Le champ d'en-tête Content-Length *(Longueur-du-contenu)* indique la taille du corps du message, en nombre décimal d'octets, envoyé au receveur. Les applications DEVRAIENT utiliser ce champ pour indiquer la taille du corps du message à transférer, indépendamment du type de support de l'entité. Si un protocole fondé sur le flux (tel que TCP) est utilisé comme transport, le champ d'en-tête DOIT être utilisé.

La taille du corps de message n'inclut pas le CRLF qui sépare les champs d'en-tête et le corps. Tout Content-Length supérieur ou égal à zéro est une valeur valide. Si aucun corps n'est présent dans un message, la valeur du champ d'en-tête Content-Length DOIT être mise à zéro.

La capacité d'omettre Content-Length simplifie la création de scripts de style cgit qui génèrent des réponses de façon dynamique.
La forme compacte du champs d'en-tête est l.

Exemples : Content-Length: 349
l: 173

20.15 Content-Type

Le champ d'en-tête Content-Type *(Type-de-contenu)* indique le type de support du corps de message envoyé au receveur. L'élément "media-type" est défini dans [H3.7]. Le champ d'en-tête Content-Type DOIT être présent si le corps n'est pas vide. Si le corps est vide, et si un champ d'en-tête Content-Type est présent, il indique que le corps de ce type spécifique a une longueur de zéro (par exemple, un fichier audio vide).
La forme compacte du champ d'en-tête est c.

Exemples : Content-Type: application/sdp
c: text/html; charset=ISO-8859-4

20.16 CSeq

Un champ d'en-tête CSeq dans une demande contient un seul numéro de séquence décimal et la méthode de la demande. Le numéro de séquence DOIT être exprimable comme un entier de 32 bits non signé. La partie méthode de CSeq est sensible à la casse. Le champ d'en-tête CSeq sert à ordonner les transactions au sein d'un dialogue, à fournir un moyen d'identifier les transactions de façon univoque, et à différentier entre les nouvelles demandes et les retransmissions. Deux champs d'en-tête CSeq sont considérés égaux si le numéro de séquence et la méthode de demande sont identiques.
Exemple : CSeq: 4711 INVITE

20.17 Date

Le champ d'en-tête Date contient la date et l'heure. A la différence de HTTP/1.1, SIP ne prend en charge que le format le plus récent de la RFC 1123 [20] pour les dates. Comme dans [H3.3], SIP restreint la zone d'heure dans la date SIP à "GMT", alors que la RFC 1123 permet toute zone d'heure. Une date de la RFC 1123 est sensible à la casse. Le champ d'en-tête Date reflète l'heure à laquelle la demande ou la réponse a été envoyée pour la première fois.

Le champ d'en-tête Date peut être utilisé par de simples systèmes de terminaison sans horloge assistée par batterie pour

acquérir une notion de l'heure en cours. Cependant, dans sa forme GMT, elle requiert des clients qu'ils connaissent leur décalage par rapport au GMT.

Exemple : Date: Sat, 13 Nov 2010 23:29:00 GMT

20.18 Error-Info

Le champ d'en-tête Error-Info fournit un pointeur sur des informations supplémentaires sur la réponse d'état d'erreur.

Les UAC SIP ont des capacités d'interface d'utilisateur qui vont de la fenêtre pop-up et des clients logiciels audio sur micro-ordinateur à des téléphones seulement audio ou des points d'extrémité connectés via des passerelles. Plutôt que de forcer un serveur qui génère une erreur à choisir entre l'envoi d'un code d'état d'erreur avec une phrase de cause détaillée et jouer un enregistrement audio, le champ d'en-tête Error-Info permet d'envoyer les deux. L'UAC a alors le choix de l'indicateur d'erreur à donner à l'appelant.

Un UAC PEUT traiter un URI SIP ou SIPS dans un champ d'en-tête Error-Info comme si c'était un Contact dans une demande redirect et générer un nouvel INVITE, résultant en l'établissement d'une session d'annonces enregistrées. Un URI non-SIP PEUT être rendu à l'utilisateur.

Exemples : SIP/2.0 404 Le numéro demandé n'est pas en service
Error-Info: <sip:not-dans-service-recording@atlanta.com>

20.19 Expires

Le champ d'en-tête Expires donne l'heure relative après laquelle le message (ou le contenu) expire.

La signification précise dépend de la méthode.

L'heure d'expiration dans un INVITE n'affecte pas la durée de la session réelle qui peut résulter de l'invitation. Les protocoles de description de session peuvent cependant offrir la capacité d'exprimer des limites à la durée de session.

La valeur de ce champ est un nombre entier de secondes (en décimales) entre 0 et (2**32)-1, mesuré depuis la réception de la demande.

Exemple : Expires: 5

20.20 From

Le champ d'en-tête From indique l'initiateur de la demande. Il peut être différent de l'initiateur du dialogue. Les demandes envoyées par l'appelé à l'appelant utilisent l'adresse de l'appelé dans le champ d'en-tête From.

Le "display-name" facultatif est destiné à être rendu par une interface d'utilisateur humain. Un système DEVRAIT utiliser le nom d'affichage "Anonymous" si l'identité du client doit rester cachée. Même si le "display-name" est vide, la forme "name-addr" DOIT être utilisée si le "addr-spec" contient une virgule, un point d'interrogation, ou un point-virgule. Les questions de syntaxe sont exposées au paragraphe 7.3.1.

Deux champs d'en-tête From sont équivalents si leurs URI correspondent, et si leurs paramètres correspondent. Les paramètres d'extension dans un champ d'en-tête, non présents dans l'autre sont ignorés pour les besoins de la comparaison. Cela signifie que le nom d'affichage et la présence ou l'absence de crochets angulaires n'affectent pas la comparaison.

Voir au paragraphe 20.10 les règles d'analyse d'un nom d'affichage, de l'URI et des paramètres d'URI, et des paramètres de champ d'en-tête.
La forme compacte du champ d'en-tête From est f.
Exemples : From: "A. G. Bell" <sip:agb@bell-telephone.com> ;tag=a48s
From: sip:+12125551212@server.phone2net.com;tag=887s
f: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8

20.21 In-Reply-To

Le champ d'en-tête In-Reply-To énumère les Call-ID auquel cet appel se réfère ou qu'il retourne. Ces Call-ID peuvent être mémorisés par le client puis inclus dans ce champ d'en-tête dans un appel en retour.

Ceci permet aux systèmes automatiques de distribution d'appel d'acheminer les appels en retour à celui qui est à l'origine du premier appel. Cela permet aussi aux appelés de filtrer les appels, de sorte que seuls les appels en retour pour les appels dont ils sont à l'origine seront acceptés. Ce champ n'est pas un substitut pour l'authentification de la demande.
Exemple : In-Reply-To: 70710@saturn.bell-tel.com, 17320@saturn.bell-tel.com

20.22 Max-Forwards

Le champ d'en-tête Max-Forwards doit être utilisé avec toute méthode SIP pour limiter le nombre de mandataires ou passerelles qui peuvent transmettre la demande au prochain serveur vers l'aval. Cela peut aussi être utile lorsque le client essaye de retracer une chaîne de demande qui paraît être défaillante ou avoir un bouclage à mi-chaîne.

La valeur de Max-Forwards est un entier dans la gamme 0-255 qui indique le nombre restant de fois que ce message de demande est autorisé à être retransmis. Ce compte est décrémenté par chaque serveur qui transmet la demande. La valeur initiale recommandée est 70.

Ce champ d'en-tête devrait être inséré par les éléments qui ne peuvent pas autrement garantir la détection de boucle. Par exemple, une B2BUA devrai insérer un champ d'en-tête Max-Forwards.

Exemple : Max-Forwards: 6

20.23 Min-Expires

Le champ d'en-tête Min-Expires porte l'intervalle minimum de rafraîchissement pris en charge pour les éléments à état conditionnel gérés par ce serveur. Cela inclut les champs d'en-tête Contact qui sont mémorisés par un registraire. Le champ d'en-tête contient un nombre décimal entier de secondes de 0 à (2**32)-1. L'utilisation du champ d'en-tête dans une réponse 423 (Intervalle trop bref) est décrit aux paragraphes 10.2.8, 10.3, et 21.4.17.

Exemple : Min-Expires: 60

20.24 MIME-Version

Voir [H19.4.1].
Exemple : MIME-Version: 1.0

20.25 Organization

Le champ d'en-tête Organization porte le nom de l'organisation à laquelle appartient l'élément SIP qui produit la demande ou la réponse.
Le champ PEUT être utilisé le logiciel client pour filtrer les appels.

Exemple : Organization: Boxes by Bob

20.26 Priority

Le champ d'en-tête Priority indique l'urgence de la demande telle que perçue par le client. Le champ d'en-tête Priority décrit la priorité que devrait avoir la demande SIP pour le récepteur humain ou son agent. Par exemple, il peut être

intégré dans des décisions sur l'acheminement et l'acceptabilité des appels. Pour ces décisions, un message ne contenant pas de champ d'en-tête Priority DEVRAIT être traité comme s'il spécifiait une priorité "normale". Le champ d'en-tête Priority n'influence pas l'utilisation de ressources de communication telles que la priorité de transmission des paquets dans les routeurs ou l'accès aux circuits dans les passerelles RTPC. Le champ d'en-tête Priority peut avoir les valeurs "non-urgent", "normal", "pressant", et "urgence", mais des valeurs supplémentaires peuvent être définies ailleurs. Il est RECOMMANDÉ que la valeur "urgence" ne soit utilisée que lorsque la vie humaine, l'intégrité physique, ou les biens sont en danger imminent. Autrement, il n'y a pas de sémantique définie pour ce champ d'en-tête.

Ces valeurs sont celles de la RFC 2076 [38], avec l'ajout de "urgence".

Exemples : Subject: Une tornade vient sur vous !
Priority: urgence
ou
Subject: Plans pour la fin de semaine
Priority: non-urgent

20.27 Proxy-Authenticate

Une valeur de champ d'en-tête Proxy-Authenticate contient une requête d'authentification.

L'utilisation de ce champ d'en-tête est définie dans [H14.33]. Voir au paragraphe 22.3 pour des détails complémentaires sur son utilisation.

Exemple : Proxy-Authenticate: Digest realm="atlanta.com",domain="sip:ss1.carrier.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359",opaque="", stale=FALSE, algorithm=MD5

20.28 Proxy-Authorization

Le champ d'en-tête Proxy-Authorization permet au client de s'identifier (ou d'identifier son utilisateur) à un mandataire qui exige l'authentification. Une valeur de champ Proxy-Authorization consiste en accréditations qui contiennent les informations d'authentification de l'agent utilisateur pour le mandataire et/ou le domaine des ressources demandées.

Voir au paragraphe 22.3 la définition de l'utilisation de ce champ d'en-tête.

Ce champ d'en-tête, ainsi que Authorization, enfreint les règles générales sur les noms multiples de champ d'en-tête. Bien que ce ne soit pas une liste séparée par des virgules, ce nom de champ d'en-tête peut être présent plusieurs fois, et NE DOIT PAS être combiné en un seule ligne d'en-tête en utilisant les règles usuelles décrites au paragraphe 7.3.1.

Exemple : Proxy-Authorization: Digest username="Alice", realm="atlanta.com",
nonce="c60f3082ee1212b402a21831ae",
réponse="245f23415f11432b3434341c022"

20.29 Proxy-Require

Le champ d'en-tête Proxy-Require est utilisé pour indiquer des caractéristiques particulières du mandataire qui doivent être prises en charge par le mandataire. Voir au paragraphe 20.32 les détails complémentaires sur les mécanismes de ce message et un exemple d'utilisation.

Exemple : Proxy-Require: foo

20.30 Record-Route

Le champ d'en-tête Record-Route est inséré par les mandataires dans une demande pour forcer des demandes futures dans le dialogue à être acheminées par l'intermédiaire du mandataire.
Des exemples de son utilisation avec le champ d'en-tête Route sont décrits au paragraphe 16.12.1.

Exemple : Record-Route: <sip:server10.biloxi.com;lr>,<sip:bigbox3.site3.atlanta.com;lr>

20.31 Reply-To

Le champ d'en-tête Reply-To contient un URI de retour logique qui peut être différent du champ d'en-tête From. Par exemple, l'URI PEUT être utilisé pour retourner des appels manqués ou une session non établie. Si l'utilisateur souhaite rester anonyme, le champ d'en-tête DEVRAIT être omis de la demande ou rempli d'une façon telle qu'elle ne révèle aucune information privée.

Même si le "display-name" est vide, la forme "name-addr" DOIT être utilisée si "addr-spec" contient une virgule, point d'interrogation, ou un point-virgule. Les questions de syntaxe sont exposées au paragraphe 7.3.1.

Exemple : Reply-To: Bob <sip:bob@biloxi.com>

20.32 Require

Le champ d'en-tête Require est utilisé par les UAC pour dire aux UAS les options que l'UAC attend que l'UAS prenne en charge afin de traiter la demande. Bien qu'il soit un champ d'en-tête facultatif, Require NE DOIT PAS être ignoré si il est présent.

Le champ d'en-tête Require contient une liste des étiquettes d'option, décrite au paragraphe 19.2. Chaque étiquette d'option définit une extension SIP qui DOIT être comprise pour traiter la demande. Fréquemment, ceci est utilisé pour indiquer qu'un ensemble spécifique de champ d'en-tête d'extension doit être compris. Un UAC conforme à la présente spécification DOIT seulement inclure les étiquettes d'option correspondantes aux RFC de normalisation *(standard track)*.
Exemple : Require: 100rel

20.33 Retry-After

Le champ d'en-tête Retry-After (*Réessayer plus tard*) peut être utilisé avec une réponse 500 (Erreur interne au serveur) ou 503 (Service indisponible) pour indiquer la durée attendue de l'indisponibilité du service pour le client demandeur, et avec une réponse 404 (Non trouvé), 413 (Entité de demande trop grande), 480 (Temporairement indisponible), 486 (Occupé ici), 600 (Occupé), ou 603 pour indiquer quand l'appelé prévoit d'être à nouveau disponible. La valeur de ce champ est un nombre entier positif de secondes (en décimal) après l'heure de la réponse.

Un commentaire facultatif peut être utilisé pour indiquer des informations supplémentaires sur l'heure de rappel. Un paramètre facultatif "durée" indique pendant combien de temps l'appelé sera joignable à partir de l'heure initiale de disponibilité. Si aucun paramètre de durée n'est donné, le service est supposé être disponible indéfiniment.

Exemples : Retry-After: 18000;durée=3600
Retry-After: 120 (je suis en réunion)

20.34 Route

Le champ d'en-tête est utilisé pour forcer l'acheminement d'une demande à travers la liste de l'ensemble des mandataires. Des exemples de l'utilisation du champ d'en-tête Route figurent au paragraphe 16.12.1.

Exemple : Route: <sip:bigbox3.site3.atlanta.com;lr>,
<sip:server10.biloxi.com;lr>

20.35 Server

Le champ d'en-tête Server contient des informations sur le logiciel utilisé par l'UAS pour traiter la demande.

Révéler la version spécifique de logiciel du serveur peut rendre le serveur plus vulnérable aux attaques contre un

logiciel qui est connu comme offrant de failles à la sécurité. Les mises en œuvre DEVRAIENT faire du champ d'en-tête Server une option configurable.

Exemple : Server: HomeServer v2

20.36 Subject

Le champ d'en-tête Subject donne un résumé ou indique la nature de l'appel, permettant un filtrage des appels sans avoir à analyser la description de session. La description de session n'a pas à utiliser les mêmes indications de sujet que l'invitation.
La forme compacte du champ d'en-tête Subject est s.

Exemple : Subject: Besoin de plus de parkings
s: Soutien technique

20.37 Supported

Le champ d'en-tête Supported *(Accepté)* énumère toutes les extensions acceptées par l'UAC ou l'UAS.

Le champ d'en-tête Supported contient une liste des étiquettes d'option, décrites au paragraphe 19.2, qui sont comprises par l'UAC ou l'UAS. Un agent utilisateur conforme à la présente spécification DOIT seulement inclure les étiquettes d'option correspondant aux RFC de normalisation. S'il est vide, il signifie qu'aucune extension n'est acceptée.
La forme compacte du champ d'en-tête Supported est k.

Exemple : Supported: 100rel

20.38 Timestamp

Le champ d'en-tête Timestamp *(Horodatage)* décrit le moment où l'UAC a envoyé la demande à l'UAS.

Voir au paragraphe 8.2.6 les détails sur la façon de générer une réponse à une demande qui contient ce champ d'en-tête. Bien qu'il n'y ait pas de comportement normalisé défini ici qui fasse usage de cet en-tête, il permet à des extensions ou applications SIP d'obtenir des estimations de délai d'aller-retour.

Exemple : Timestamp: 54

20.39 To

Le champ d'en-tête To spécifie le récepteur logique de la demande.

Le "display-name" facultatif est destiné à être rendu par une interface d'utilisateur humain. Le paramètre "tag" sert de mécanisme général pour l'identification de dialogue.

Voir au paragraphe 19.3 les détails du paramètre "tag".

La comparaison d'égalité des champs d'en-tête To est identique à la comparaison de champ d'en-tête From. Voir au paragraphe 20.10 les règles d'analyse d'un nom d'affichage, d'URI et des paramètres d'URI, et des paramètres de champ d'en-tête.
La forme compacte du champ d'en-tête To est t.
Voici des exemples valides du champ d'en-tête To :

To: The Operator <sip:operator@cs.columbia.edu>;tag=287447
t: sip:+12125551212@serveur.phone2net.com

20.40 Unsupported

Le champ d'en-tête Unsupported *(Non accepté)* fait la liste des caractéristiques non acceptées par l'UAS. Voir les motivations au paragraphe 20.32.
Exemple : Unsupported: foo

20.41 User-Agent

Le champ d'en-tête User-Agent *(Agent d'utilisation)* contient des informations sur l'UAC d'origine de la demande. La sémantique de ce champ d'en-tête est définie en [H14.43].

Révéler la version spécifique de logiciel de l'agent utilisateur peut rendre l'agent utilisateur plus vulnérable aux attaques contre un logiciel connu pour offrir des failles à la sécurité. Les mises en œuvre DEVRAIENT faire du champ d'en-tête User-Agent une option configurable.
Exemple : User-Agent: Téléphone logiciel Beta1.5

20.42 Via

Le champ d'en-tête Via indique le chemin pris par la demande jusqu'à présent et indique le chemin qui devrait être suivi dans l'acheminement des réponses. Le paramètre ID de branche dans les valeurs de champ d'en-tête Via sert d'identifiant de transaction, et est utilisé par les mandataires pour détecter les boucles.

Une valeur de champ d'en-tête Via contient le protocole de transport utilisé pour envoyer le message, le nom d'hôte du client ou l'adresse réseau, et si possible le numéro de port auquel il souhaite recevoir les réponses. Une valeur de champ d'en-tête Via peut aussi contenir des paramètres tels que "maddr", "ttl", "received", et "branch", dont la signification et l'utilisation sont décrites dans d'autres sections. Pour les mises en œuvre conformes à la présente spécification, la valeur du paramètre branch DOIT débuter par le cookie magique "z9hG4bK", comme exposé au paragraphe 8.1.1.7.

Les protocoles de transport définis ici sont "UDP", "TCP", "TLS", et "SCTP". "TLS" signifie TLS sur TCP. Lorsque une demande est envoyée à un URI SIPS, le protocole indique toujours "SIP", et le protocole de transport est TLS.
La forme compacte du champ d'en-tête Via est v.

Exemple : Via: SIP/2.0/UDP erlang.bell-telephone.com:5060;branch=z9hG4bK87asdks7
Via: SIP/2.0/UDP 192.0.2.1:5060 ;received=192.0.2.207 ;branch=z9hG4bK77asjd

Dans cet exemple, le message est originaire d'un hôte multi domiciliations avec deux adresses, 192.0.2.1 et 192.0.2.207. L'expéditeur a mal jugé de l'interface réseau qui devait être utilisée. Erlang.bell-telephone.com a remarqué la confusion et a ajouté un paramètre à la précédente valeur de champ d'en-tête Via du saut, contenant l'adresse dont le paquet est réellement venu.

L'hôte ou d'adresse réseau et le numéro de port ne sont pas exigés pour suivre la syntaxe d'URI SIP. Précisément, l'espace insécable (LWS) d'un côté ou de l'autre du ":" ou "/" est admis, comme montré ici :

Via: SIP / 2.0 / UDP first.example.com: 4000;ttl=16;maddr=224.2.0.1;branch=z9hG4bKa7c6a8dlze.1

Même si la présente spécification exige que le paramètre branch soit présent dans toute les demandes, le BNF pour le champ d'en-tête indique qu'il est facultatif. Ceci permet l'interopération avec les éléments de la RFC 2543, qui n'avaient pas à insérer le paramètre branch.

Deux champ d'en-tête Via sont égaux si leurs champs send-protocol et send-by sont égaux, si tous deux ont le même ensemble de paramètres, et si les valeurs de tous les paramètres sont égales.

20.43 Warning

Le champ d'en-tête Warning *(Avertissement)* est utilisé pour porter des informations supplémentaires sur l'état d'une réponse. Les valeurs de champ d'en-tête Warning sont envoyées avec les réponses et contiennent un code d'avertissement, nom d'hôte, et texte d'avertissement à trois chiffres

Le "warn-text" devrait être dans un langage naturel qui ait des chances d'être intelligible à l'utilisateur humain qui reçoit la réponse. Cette décision peut se fonder sur toute information disponible, comme la localisation de l'utilisateur, le champ Accept-Language dans une demande, ou le champ Content-Language dans une réponse. Le langage par défaut est i-default [21].

La liste des "warn-code" actuellement définis figure ci-dessous, avec un texte d'avertissement recommandé et une description de leur signification. Ces avertissement décrivent des défaillances induites par la description de session. Le premier chiffre des codes d'avertissement commençant par "3" indique des avertissements spécifiques de SIP. Les avertissements 300 à 329 sont réservés pour indiquer des problèmes de mots clé dans la description de session, 330 à 339 se rapportent aux services réseau de base demandés dans la description de session, 370 à 379 se rapportent aux paramètres quantitatifs de qualité de service demandés dans la description de session, et 390 à 399 sont les avertissements divers qui ne rentrent pas dans les catégories précédentes.

300 Protocole réseau incompatible : Un ou plusieurs protocoles réseau contenus dans la description de session ne sont pas disponibles.
301 Formats d'adresse réseau incompatibles : Un ou plusieurs formats d'adresse réseau contenus dans la description de session ne sont pas disponibles.
302 Protocole de transport incompatible : Un ou plusieurs protocoles de transport décrits dans la description de session ne sont pas disponibles.
303 Unités de bande passante incompatibles : Une ou plusieurs unités de mesure de bande passante contenues dans la description de session ne sont pas comprises.
304 Type de support non disponible : Un ou plusieurs types de support contenus dans la description de session ne sont pas disponibles.
305 Format de support incompatible : Un ou plusieurs formats de supports contenus dans la description de session ne sont pas disponibles.
306 Attribut non compris : Un ou plusieurs des attributs de supports dans la description de session ne sont pas acceptés.
307 Paramètre de description de session non compris : Un paramètre autre que ceux listés ci-dessus n'est pas compris.
330 Diffusion groupée non disponible : Le site où est localisé l'utilisateur n'accepte pas la diffusion groupée.
331 Envoi individuel non disponible : Le site où est localisé l'utilisateur n'accepte pas les communications en envoi individuel (habituellement à cause de la présence d'un pare-feu).
370 Bande passante insuffisante : La bande passante spécifiée dans la description de session ou définie par les supports excède celle connue comme disponible.
399 Avertissements divers : Le texte d'avertissement peut inclure des informations arbitraires à présenter à un utilisateur humain ou à enregistrer. Un système recevant cet avertissement NE DOIT PAS en tirer d'action automatique.

1xx et 2xx ont été pris par HTTP/1.1.

Des "warn-code" supplémentaires peuvent être définis à travers l'IANA, comme défini au paragraphe 27.2.

Exemples : Warning: 307 isi.edu "Paramètre de Session 'foo' non compris"
Warning: 301 isi.edu "Type d'adresse réseau 'E.164' incompatible"

20.44 WWW-Authenticate

Une valeur de champ d'en-tête WWW-Authenticate contient une requête d'authentification. Voir au paragraphe 22.2 les détails complémentaires sur son utilisation.

Exemple : WWW-Authenticate: Digest realm="atlanta.com",
domain="sip:boxesbybob.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359",
opaque="", stale=FALSE, algorithm=MD5

21 Codes de réponse

Les codes de réponse sont cohérents avec les codes de réponse HTTP/1.1, et les étendent. Tous les codes de réponse HTTP/1.1 ne sont pas appropriés, et seuls ceux qui le sont sont donnés ici. D'autres codes de réponse HTTP/1.1 NE

DEVRAIENT PAS être utilisés. SIP définit une autre nouvelle classe, 6xx.

21.1 1xx Provisoire

Les réponses provisoires, aussi connues comme réponses informatives, indiquent que le serveur contacté est en train d'effectuer certaines actions et n'a pas encore de réponse définitive. Un serveur envoie une réponse 1xx si il s'attend à ce que l'obtention d'une réponse finale prenne plus de 200 ms. Noter que la transmission des réponses 1xx n'est pas fiable. Elle n'oblige jamais le client à envoyer un ACK. Les réponses provisoires (1xx) PEUVENT contenir un corps de message, y compris de description de session.

21.1.1 100 Essai

Cette réponse indique que la demande a été reçue par le serveur du saut suivant et que certaine action non spécifiée est en train d'être effectuée au titre de cet appel (par exemple, une base de données est consultée). Cette réponse, comme toutes les autres réponses provisoires, arrête les retransmissions d'un INVITE par un UAC. La réponse 100 (En cours d'essai) est différente des autres réponses provisoires, en ce qu'elle n'est jamais transmise vers l'amont par un mandataire à états pleins.

21.1.2 180 Sonnerie en cours

L'agent utilisateur qui reçoit l'INVITE est en train d'essayer d'alerter l'utilisateur. Cette réponse PEUT être utilisée pour initialiser la sonnerie de retour d'appel locale.

21.1.3 181 L'appel est en cours de transmission

Un serveur PEUT utiliser ce code d'état pour indiquer que l'appel est en train d'être retransmis à un ensemble de destinations différent.

21.1.4 182 En file d'attente

Le demandé est temporairement indisponible, mais le serveur a décidé de mettre l'appel en file d'attente plutôt que de le rejeter. Lorsque l'appelé devient disponible, il retournera la réponse d'état finale appropriée. La phrase de cause PEUT donner plus de précisions sur l'état de l'appel, par exemple, "5 appels en file d'attente ; le temps d'attente prévu est de 15 minutes". Le serveur PEUT produire plusieurs réponses 182 (En file d'attente) pour tenir l'appelant au courant de l'état de l'appel en file d'attente.

21.1.5 183 Avancement de la session

La réponse 183 (Avancement de la session) est utilisée pour porter des informations sur la progression de l'appel qui n'est pas autrement classifiée. La phrase de cause, le champ d'en-tête, ou le corps de message PEUVENT être utilisés pour porter plus de précisions sur la progression de l'appel

21.2 2xx Réussite

La demande a réussi.

21.2.1 200 OK

La demande a réussi. Les informations retournées avec la réponse dépendent de la méthode utilisée dans la demande.

21.3 3xx Réorientation

Les réponses 3xx donnent des informations sur la nouvelle localisation de l'utilisateur, ou sur des services de remplacement qui pourraient être capables de satisfaire à l'appel.

21.3.1 300 Choix multiples

L'adresse dans la demande peut se résoudre en plusieurs choix, chacun avec sa propre localisation spécifique, et l'utilisateur (ou agent utilisateur) peut choisir un point de terminaison de communication préféré et rediriger sa demande sur cette localisation.
La réponse PEUT inclure un corps de message contenant une liste de caractéristiques et localisation(s) de ressources à partir desquelles l'utilisateur ou agent utilisateur peut choisir la plus appropriée, si c'est permis par le champ d'en-tête Accept de la demande. Cependant, aucun type MIME n'a été défini pour ce corps de message.
Les choix DEVRAIENT aussi être énumérés comme champs Contact (paragraphe 20.10). A la différence de HTTP, la réponse SIP PEUT contenir plusieurs champs Contact ou une liste d'adresses dans un champ Contact. Les agents d'utilisateur PEUVENT utiliser la valeur du champ d'en-tête Contact pour une redirection automatique ou PEUVENT demander à l'utilisateur de confirmer un choix. Cependant, la présente spécification ne définit aucune norme pour une telle sélection automatique.
Cette réponse d'état est appropriée si l'appelé peut être joint à plusieurs localisations différentes et le serveur ne peut pas ou préfère ne pas mandater la demande.

21.3.2 301 Déplacement définitif

L'utilisateur ne peut plus être joint à l'adresse figurant dans l'URI-de-demande, et le client demandeur DEVRAIT réessayer à la nouvelle adresse donnée par le champ d'en-tête Contact (paragraphe 20.10). Le demandeur DEVRAIT mettre à jour tout répertoire local, carnet d'adresses, et mémoires caches de localisation d'utilisateur avec cette nouvelle valeur et rediriger les demandes futures sur la ou les adresses listées.

21.3.3 302 Temporairement déplacé

Le client demandeur DEVRAIT réessayer la demande à la ou aux nouvelles adresses données par le champ d'en-tête Contact (paragraphe 20.10). L'URI-de-demande de la nouvelle demande utilise la valeur du champ d'en-tête Contact dans la réponse.

La durée de validité de l'URI Contact peut être indiquée par un champ d'en-tête Expires (paragraphe 20.19) ou un paramètre expires dans le champ d'en-tête Contact. Les mandataires et les agents d'utilisateur PEUVENT tous deux mettre cet URI en mémoire cache pour la durée du délai d'expiration. Si il n'y a pas de délai d'expiration explicite, l'adresse n'est valide pour recommencer qu'une seule fois, et NE DOIT PAS être cachée pour des transactions ultérieures.

Si l'URI caché tiré du champ d'en-tête Contact échoue, l'URI-de-demande de la demande redirigée PEUT être essayé encore une seule fois.

L'URI temporaire peut être périmé plus tôt que le délai d'expiration, et un nouvel URI temporaire peut être disponible.

21.3.4 305 Utiliser un mandataire

On DOIT accéder à la ressource demandée par l'intermédiaire du mandataire donné par le champ Contact. Le champ Contact donne l'URI du mandataire. Le receveur est censé répéter cette seule demande via le mandataire. Les réponses 305 (Utiliser un mandataire) DOIVENT être générées par les seuls UAS.

21.3.5 380 Service de remplacement

L'appel n'a pas réussi, mais des services de remplacement sont possibles.

Les services de remplacement sont décrits dans le corps de message de la réponse. Les formats pour de tels corps ne sont pas définis ici, et pourront être le sujet d'une normalisation future.

21.4 4xx Défaillance de la demande

Les réponses 4xx sont des réponses d'échec bien déterminées provenant d'un serveur particulier. Le client NE DEVRAIT PAS réessayer la même demande sans modification (par exemple, en ajoutant l'autorisation appropriée). Cependant, la même demande sur un serveur différent pourrait réussir.

21.4.1 400 Mauvaise demande

La demande n'a pas pu être comprise à cause d'une syntaxe mal formée. La phrase de cause DEVRAIT identifier le problème de syntaxe plus en détail, par exemple, "Champ d'en-tête Call-ID manquant".

21.4.2 401 Non autorisé

La demande exige une authentification de l'utilisateur. Cette réponse est produite par les UAS et registraires, alors que la réponse 407 (Authentification du mandataire exigée) est utilisée par les serveurs mandataires.

21.4.3 402 Payement exigé

Réservé pour utilisation future.

21.4.4 403 Interdit

Le serveur comprend la demande, mais refuse d'y satisfaire. L'autorisation n'y fera rien, et la demande NE DEVRAIT PAS être répétée.

21.4.5 404 Pas trouvé

Le serveur a des informations certaines que l'utilisateur n'existe pas au domaine spécifié dans l'URI-de-demande. Cet état est aussi retourné si le domaine dans l'URI-de-demande ne correspond à aucun des domaines traités par le receveur de la demande.

21.4.6 405 Méthode non autorisée

La méthode spécifiée dans la ligne de demande est comprise, mais non autorisée pour l'adresse identifiée par l'URI-de-demande.

La réponse DOIT inclure un champ d'en-tête Allow contenant une liste de méthodes valides pour l'adresse indiquée.

21.4.7 406 Non acceptable

La ressource identifiée par la demande est seulement capable de générer des entités de réponse qui ont des caractéristiques de contenu non acceptables en fonction du champ d'en-tête Accept envoyé dans la demande.

21.4.8 407 Authentification du mandataire requise

Ce code est similaire à 401 (Non autorisé), mais indique que le client DOIT d'abord s'authentifier avec le mandataire. L'authentification d'accès SIP est expliquée à la Section 26 et au paragraphe 22.3.

Ce code d'état peut être utilisé pour des applications où, plutôt que le demandé, l'accès au canal de communication (par exemple, une passerelle de téléphonie) exige l'authentification.

21.4.9 408 Expiration du délai de demande

Le serveur n'a pas pu produire une réponse dans le délai requis, par exemple, s'il n'a pas pu déterminer la localisation de l'utilisateur dans les temps. Le client PEUT répéter la demande sans modifications à tout moment ultérieurement.

21.4.10 410 Parti

La ressource demandée n'est plus disponible au serveur et aucune adresse de retransmission n'est connue. Cette condition est supposée être permanente. Si le serveur ne connaît pas, ou n' aucune facilite pour déterminer si la condition est permanente ou non, c'est le code d'état 404 (Non trouvé) qui DEVRAIT être utilisé à la place.

21.4.11 413 Entité de demande trop longue

Le serveur refuse de traiter une demande parce que l'entité corps de la demande est supérieure à ce que le serveur est désireux ou capable de traiter. Le serveur PEUT fermer la connexion pour empêcher le client de continuer la demande. Si la condition est temporaire, le serveur DEVRAIT inclure un champ d'en-tête Retry-After pour indiquer qu'elle est temporaire et après quel délai le client PEUT réessayer.

21.4.12 414 URI de demande trop long

Le serveur refuse de servir la demande car l'URI-de-demande est plus long que ce que le serveur désire interpréter.

21.4.13 415 Type de support non accepté

Le serveur refuse de servir la demande parce que le corps de message de la demande est dans un format qui n'est pas accepté par le serveur pour la méthode demandée. Le serveur DOIT retourner une liste de formats acceptables en utilisant les champs d'en-tête Accept, Accept-Encoding, ou Accept-Language, selon le problème spécifique du contenu. Le traitement de cette réponse par l'UAC est décrit au paragraphe 8.1.3.5.

21.4.14 416 Schéma d'URI non accepté

Le serveur ne peut pas traiter la demande parce que le schéma de l'URI dans l'URI-de-demande est inconnu du serveur. Le traitement de cette réponse par le client est décrit au paragraphe 8.1.3.5.

21.4.15 420 Mauvaise extension

Le serveur n'a pas compris l'extension de protocole spécifiée dans un champ d'en-tête Proxy-Require (paragraphe 20.29) ou Require (paragraphe 20.32). Le serveur DOIT inclure une liste des extensions non prises en charge dans un champ d'en-tête Unsupported dans la réponse. Le traitement de cette réponse par l'UAC est décrit au paragraphe 8.1.3.5.

21.4.16 421 Extension requise

L'UAS a besoin d'une extension particulière pour traiter la demande, mais cette extension ne figure pas dans la liste du champ d'en-tête Supported dans la demande. Les réponses avec ce code d'état DOIVENT contenir un champ d'en-tête

Require faisant la liste des extensions requises.

Un UAS NE DEVRAIT utiliser cette réponse que s'il ne peut vraiment fournir aucun service utile au client. Au lieu de cela, si une extension souhaitable ne figure pas sur la liste du champ d'en-tête Supported, les serveurs DEVRAIENT traiter la demande en utilisant les capacités SIP de base et toute extension acceptée par le client.

21.4.17 423 Intervalle trop bref

Le serveur rejette la demande parce que le délai d'expiration des ressources rafraîchi par la demande est trop court. Cette réponse peut être utilisée par un registraire pour rejeter un enregistrement dont le délai d'expiration du champ d'en-tête Contact est trop bref. L'utilisation de cette réponse et le champ d'en-tête Min-Expires qui s'y rapporte sont décrits aux paragraphes 10.2.8, 10.3, et 20.23.

21.4.18 480 Temporairement indisponible

Le système de terminaison du demandé a été contacté avec succès mais le demandé est actuellement indisponible (par exemple, il n'est pas enregistré, ou bien il est enregistré mais se trouve dans un état qui empêche les communications avec l'appelé, ou il a activé le dispositif "ne pas déranger"). La réponse PEUT indiquer un meilleur moment pour appeler dans le champ d'en-tête Retry-After. L'utilisateur peut aussi être disponible ailleurs (à l'insu de ce serveur). La phrase de cause DEVRAIT indiquer une raison plus précise comme pourquoi l'appelé est indisponible. Cette valeur DEVRAIT être réglable par l'agent utilisateur. L'état 486 (Occupé ici) PEUT être utilisé pour indiquer plus précisément une raison particulière pour l'échec de l'appel.

Cet état est aussi retourné par une redirection ou un serveur mandataire qui reconnaît l'utilisateur identifié par l'URI-de-demande, mais n'a pas actuellement de localisation de retransmission valide pour cet utilisateur.

21.4.19 481 L'appel/transaction n'existe pas

Cet état indique que l'UAS a reçu une demande qui ne correspond à aucun dialogue ou transaction existant.

21.4.20 482 Boucle détectée

Le serveur a détecté une boucle (paragraphe 16.3 Item 4).

21.4.21 483 Trop de bonds

Le serveur a reçu une demande qui contient un champ d'en-tête Max-Forwards (paragraphe 20.22) avec la valeur zéro.

21.4.22 484 Adresse incomplète

Le serveur a reçu une demande avec un URI-de-demande incomplet. Des informations supplémentaires DEVRAIENT être fournies dans la phrase de cause.

Ce code d'état permet la numérotation en recouvrement. Avec la numérotation en recouvrement, le client ne connaît pas la longueur de la chaîne de numérotation. Il envoie des chaînes de longueur croissante, pressant l'utilisateur d'envoyer des entrées supplémentaires, jusqu'à ce qu'il ne reçoive plus de réponse d'état 484 (Adresse incomplète).

21.4.23 485 Ambiguïté

L'URI-de-demande est ambigu. La réponse PEUT contenir une liste d'adresses possibles non ambiguës dans le champ d'en-tête Contact. Révéler les solutions de remplacement peut porter une atteinte à la vie privée de l'utilisateur ou de l'organisation. Il DOIT être possible de configurer un serveur pour qu'il réponde par l'état 404 (Pas trouvé) ou de

supprimer la liste des choix possibles pour les URI-de-demande ambigus.

Exemple de réponse à une demande avec l'URI-de-demande sip:lee@example.com :
SIP/2.0 485 Ambigu
Contact: Carol Lee <sip:carol.lee@example.com>
Contact: Ping Lee <sip:p.lee@example.com>
Contact: Lee M. Foote <sips:lee.foote@example.com>

Certains systèmes de messagerie électronique et de messagerie vocale fournissent cette fonction. Un code d'état distinct de 3xx est utilisé dans la mesure où les sémantiques sont différentes : pour 300, il est supposé que la même personne ou service sera joint par les choix fournis. Alors qu'un choix automatisé ou une recherche séquentielle a un sens pour une réponse 3xx, une intervention de l'utilisateur est requise pour une réponse 485 (Ambigu).

21.4.24 486 Occupé ici

Le système de terminaison de l'appelé a été contacté avec succès, mais l'appelé ne peut ou ne veut pas actuellement prendre un appel supplémentaire sur ce système de terminaison. La réponse PEUT indiquer un meilleur moment pour appeler dans le champ d'en-tête Retry-After. L'utilisateur pourrait aussi être disponible ailleurs, comme sur un service de messagerie vocale. L'état 600 (Occupé partout) DEVRAIT être utilisé si le client sait qu'aucun autre système de terminaison ne sera capable d'accepter cet appel.

21.4.25 487 Demande terminée

La demande a été terminée par un BYE ou une demande CANCEL. Cette réponse n'est jamais retournée pour une demande CANCEL elle-même.

21.4.26 488 Non acceptable ici

La réponse a la même signification que 606 (Non acceptable), mais s'applique seulement aux ressources spécifiques adressées par l'URI-de-demande et la demande peut réussir ailleurs.

Un corps de message qui contient une description de capacités support PEUT être présente dans la réponse, qui est formatée conformément au champ d'en-tête Accept dans l'INVITE (ou l'application/sdp si il n'est pas présent), comme un corps de message dans une réponse 200 (OK) à une demande OPTIONS.

21.4.27 491 Demande en cours

La demande a été reçue par un UAS qui avait une demande en cours dans le même dialogue. Le paragraphe 14.2 décrit comment résoudre de telles situations.

21.4.28 493 Indéchiffrable

La demande a été reçue par un UAS qui contenait un corps MIME chiffré pour lequel le receveur ne possède pas ou ne fournira pas de clé de déchiffrement appropriée. Cette réponse PEUT avoir un seul corps contenant une clé publique appropriée qui devrait être utilisée pour chiffrer les corps MIME envoyés à cet agent utilisateur. Des précisions sur l'utilisation de ce code de réponse se trouvent au paragraphe 23.2.

21.5 5xx Défaillance de serveur

Les réponses 5xx sont des réponses d'échec qui indique que le serveur lui-même s'est trompé.

21.5.1 500 Erreur interne du serveur

Le serveur a rencontré une condition inattendue qui l'a empêché de satisfaire la demande. Le client PEUT afficher la condition d'erreur spécifique et PEUT réessayer la demande après plusieurs secondes.

Si la condition est temporaire, Le serveur PEUT indiquer quand le client peut réessayer la demande en utilisant le champ d'en-tête Retry-After.

21.5.2 501 Non mis en oeuvre

Le serveur ne prend pas en charge la fonctionnalité requise pour satisfaire la demande. C'est la réponse appropriée lorsqu'un UAS ne reconnaît pas la méthode de demande et n'est pas capable de la prendre en charge pour les utilisateurs. (Les mandataires transmettent toutes les demandes sans considération de la méthode.)

Noter qu'une réponse 405 (Méthode non admise) est envoyée lorsque le serveur reconnaît la méthode de demande, mais que cette méthode est non autorisée ou non prise en charge.

21.5.3 502 Mauvaise passerelle

Le serveur, alors qu'il agit comme une passerelle ou un mandataire, a reçu une réponse invalide de la part du serveur aval auquel il a accédé en essayant de satisfaire la demande.

21.5.4 503 Service indisponible

Le serveur est temporairement incapable de traiter la demande du fait d'une surcharge temporaire ou de la maintenance du serveur. Le serveur PEUT indiquer quand le client devrait réessayer la demande dans un champ d'en-tête Retry-After. Si aucun Retry-After n'est donné, le client DOIT agir comme s'il avait reçu une réponse 500 (Erreur interne du serveur).

Un client (mandataire ou UAC) recevant une 503 (Service indisponible) DEVRAIT essayer de transmettre la demande sur un serveur de remplacement. Il NE DEVRAIT PAS transmettre d'autres demandes à ce serveur pour la durée spécifiée dans le champ d'en-tête Retry-After, s'il est présent.

Les serveurs PEUVENT refuser la connexion ou abandonner la demande au lieu de répondre avec 503 (Service indisponible).

21.5.5 504 Expiration du délai de serveur

Le serveur n'a pas reçu de réponse dans les temps d'un serveur externe auquel il a accédé en essayant de traiter la demande. La réponse 408 (Expiration du délai de réponse) devrait être utilisée à la place s'il n'y a pas de réponse dans la période spécifiée dans le champ d'en-tête Expires de la part du serveur amont.

21.5.6 505 Version non acceptée

Le serveur ne prend pas en charge, ou refuse de prendre en charge, la version de protocole SIP qui a été utilisée dans la demande. Le serveur indique qu'il n'est pas en mesure ou ne veut pas achever la demande en utilisant la même version majeure que le client, autrement qu'avec ce message d'erreur.

21.5.7 513 Message trop long

Le serveur n'a pas été capable de traiter la demande car la longueur du message excède ses capacités.

21.6 6xx Echecs totaux

Les réponses 6xx indiquent qu'un serveur a des informations certaines sur un utilisateur particulier, et non pas seulement sur l'instance particulière indiquée dans l'URI-de-demande.

21.6.1 600 Occupé partout

Le système de terminaison de l'appelé a été contacté avec succès mais l'appelé est occupé et ne souhaite pas prendre l'appel en ce moment. La réponse PEUT indiquer un meilleur moment pour appeler dans le champ d'en-tête Retry-After. Si l'appelé ne souhaite pas révéler la raison pour laquelle il décline l'appel, il utilise à la place le code d'état 603 (Refus). Cette réponse d'état n'est retournée que si le client sait qu'aucun autre point d'extrémité (tel qu'une messagerie vocale) ne répondra à la demande. Autrement, 486 (Occupé ici) devrait être retournée.

21.6.2 603 Refus

La machine de l'appelé a été contactée avec succès mais l'utilisateur souhaite explicitement ne pas participer , ou ne le peut pas. La réponse PEUT indiquer un meilleur moment pour appeler dans le champ d'en-tête Retry-After. Cette réponse d'état n'est retournée que si le client sait qu'aucun autre point d'extrémité ne répondra à la demande.

21.6.3 604 N'existe nulle part

Le serveur a des informations certaines que l'utilisateur indiqué dans l'URI-de-demande n'existe nulle part.

21.6.4 606 Non acceptable

L'agent d'utilisateur a été contacté avec succès mais certains aspects de la description de session, tels que le support demandé, la bande passante, ou le style d'adressage ne sont pas acceptables.

Une réponse 606 (Non acceptable) signifie que l'utilisateur souhaite communiquer, mais ne peut pas prendre en charge de façon adéquate la session décrite. La réponse 606 (Non acceptable) PEUT contenir une liste des raisons dans un champ d'en-tête Warning décrivant pourquoi la session décrite ne peut être prise en charge. Les codes de cause de Warning figurent au paragraphe 20.43.

Un corps de message contenant une description des capacités support PEUT être présente dans la réponse, et elle est formatée conformément au champ d'en-tête Accept dans l'INVITE (ou l'application/sdp si elle n'est pas présente), identique au corps de message dans une réponse 200 (OK) à une demande OPTIONS.

Il est souhaité qu'il ne soit pas trop fréquemment nécessaire de recourir à la négociation, et quand un nouvel utilisateur est invité à se joindre à une conférence déjà existante, la négociation peut n'être pas possible. Il appartient à l'initiateur de l'invitation de décider d'agir ou non par une réponse 606 (Non acceptable).

Cette réponse d'état n'est retournée que si le client sait qu'aucun autre point d'extrémité ne répondra à la demande.

22 Usage de l'authentification HTTP

SIP fournit un mécanisme sans état, fondé sur la confrontation pour l'authentification qui s'appuie sur celle de HTTP. A chaque fois qu'un serveur mandataire ou agent utilisateur reçoit une demande (avec les exceptions données au paragraphe 22.1), il PEUT mettre au défi l'initiateur de la demande de fournir l'assurance de son identité. Une fois que l'origine a été identifiée, le receveur de la demande DEVRAIT s'assurer que cet utilisateur est ou non autorisé à faire la demande en question. Aucun système d'autorisation n'est recommandé ou discuté dans le présent document.

Le mécanisme d'authentification "Digest" décrit dans cette section fournit seulement l'authentification du message et la protection contre la répétition, sans l'intégrité ou la confidentialité du message. Les mesures de protection en-deça et au-delà de celles fournies par Digest doivent être prises pour empêcher les attaquants actives de modifier les demandes et réponses SIP.

Noter que du fait de sa faible sécurité, l'utilisation de l'authentification "de base" a été déconseillée. Les serveurs NE

DOIVENT PAS accepter d'accréditifs en utilisant le schéma d'autorisation "Basic", et les serveurs NE DOIVENT PAS se confronter à "Basic". Ceci est un changement par rapport à la RFC 2543.

22.1 Cadre de travail

Le cadre de travail de l'authentification SIP est étroitement parallèle à celui de HTTP (RFC 2617 [17]). En particulier, le BNF pour auth-scheme, auth-param, challenge, realm, realm-valeur, et les accréditifs est identique (bien que l'usage de "Basic" comme schéma ne soit pas permis). Dans SIP, un UAS utilise la réponse 401 (Non autorisé) pour contester l'identité d'un UAC. De plus, les registraires et les serveurs redirecteurs PEUVENT faire usage des réponses 401 (Non autorisé) pour l'authentification, mais les mandataires NE DOIVENT PAS, et à la place PEUVENT utiliser la réponse 407 (Authentification du mandataire exigée). Les exigences pour l'inclusion de Proxy-Authenticate, Proxy-Authorization, WWW-Authenticate, et Authorization dans les divers messages sont identiques à celles décrites dans la RFC 2617 [17].

Comme SIP n'a pas le concept d'un URL racine canonique, la notion d'espaces de protection est interprétée différemment dans SIP. Seule la chaîne de domaine définit le domaine de protection. Ceci est un changement par rapport à la RFC 2543, dans laquelle l'URI-de-demande et le domaine définissent ensemble le domaine de protection.

Cette définition précédente du domaine de protection a causé une certaine confusion car l'URI-de-demande envoyé par l'UAC et l'URI-de-demande reçu par le serveur questionneur pourraient être différents, et bien sûr, la forme finale de l'URI-de-demande pourrait n'être pas connue de l'UAC. Aussi, la définition précédente dépendait de la présence d'un URI SIP dans l'URI-de-demande et semblait écarte les autres schémas d'URI (par exemple, l'URL tel).

Les opérateurs des agent utilisateurs ou serveurs mandataires qui veulent authentifier les demandes reçues DOIVENT adhérer aux lignes directrices suivantes pour la création d'une chaîne de domaine pour leur serveur :

- Les chaînes de domaine DOIVENT être uniques au monde. Il est RECOMMANDÉ qu'une chaîne de domaine contienne un nom d'hôte ou un nom de domaine, suivant la recommandation du paragraphe 3.2.1 de la RFC 2617 [17].
- Les chaînes de domaine DEVRAIENT présenter un identifiant lisible par l'homme qui puisse être rendu à un utilisateur.

Par exemple :

```
INVITE sip:bob@biloxi.com SIP/2.0
Authorization: Digest realm="biloxi.com", <...>
```

Généralement, l'authentification SI¨P est significative pour un domaine spécifique, un domaine de protection. Et donc, pour l'authentification par Digest, chaque domaine de protection a son propre ensemble de noms d'utilisateurs et ses mots de passe. Si un serveur n'exige pas l'authentification pour une demande particulière, il PEUT accepter un nom d'utilisateur par défaut, "anonyme", qui n'a pas de mot de passe (password de ""). De même, les UAC représentant de nombreux utilisateurs, tels que les passerelles RTPC, PEUVENT avoir leur propre nom d'utilisateur et mot de passe spécifique d'un appareil, plutôt que des comptes pour les utilisateurs particuliers, pour leur domaine.

Alors qu'un serveur peut légitimement questionner la plupart des demandes SIP, il y a deux demandes définies par le présent document qui exigent un traitement particulier pour l'authentification: ACK et CANCEL.

Dans un schéma d'authentification qui utilise les réponses pour porter les valeurs utilisées pour calculer les noms de circonstance (comme avec Digest), certains problèmes apparaissent pour les demandes qui n'appellent pas de réponse, y compris ACK. Pour cette raison, tout accréditif dans l'INVITE qui ont été acceptés par un serveur DOIVENT être acceptés par ce serveur pour le ACK. Les UAC qui créent un message ACK dupliqueront toutes les valeurs de champs d'en-tête Authorization et Proxy-Authorization qui apparaissent dans l'INVITE auquel l'ACK correspond. Les serveurs NE DOIVENT PAS essayer de questionner un ACK.

Bien que la méthode CANCEL prenne une réponse (une 2xx), les serveurs NE DOIVENT PAS essayer de mettre en question les demandes CANCEL car ces demandes ne peuvent pas être resoumises. Généralement, une demande CANCEL DEVRAIT être acceptée par un serveur si elle vient du même saut que celui qui a envoyé la demande à annuler (pourvu que soit en place un moyen de tri du transport ou une association de sécurité de la couche réseau, comme indiqué au paragraphe 26.2.1).

Lorsqu'un UAC reçoit une mise en question, il DEVRAIT rapporte à l'utilisateur le contenu du paramètre "domaine" dans la mise en question (qui apparaît dans un champ d'en-tête WWW-Authenticate ou Proxy-Authenticate) si l'UAC ne connaît pas déjà un accréditif pour le domaine en question. Un fournisseur de service qui pré-configure les agents

d'utilisateur avec des accréditifs pour son domaine devrait savoir que les utilisateurs n'auront pas l'opportunité de présenter leur propres accréditifs pour ce domaine lorsqu'ils sont mis en question par un appareil pré-configuré.

Finalement, noter que même si un UAC peut louer des accréditifs qui soient associés au domaine approprié, il existe une possibilité que ces accréditifs ne soient plus valides ou que le serveur questionneur n'accepte pas ces accréditifs pour quelque raison que ce soit (particulièrement lorsque "anonyme" sans mot de passe est soumis). Dans ce cas, un serveur peut répéter sa mise en question, ou il peut répondre par une 403 Interdit. Un UAC NE DOIT PAS réessayer des demandes avec les accréditifs qui viennent d'être rejetés (bien que la demande puisse être réessayée si le nom occasionnel était un faux).

22.2 Authentification d'utilisateur à utilisateur

Lorsqu'un UAS reçoit une demande d'un UAC, l'UAS PEUT authentifier l'origine avant que la demande ne soit traitée. Si aucun accréditif (dans le champ d'en-tête Authorization) n'est fourni dans la demande, l'UAS peut mettre en demeure l'origine de fournir un accréditif en rejetant la demande avec un code d'état 401 (Non autorisé).

Le champs d'en-tête de réponse WWW-Authenticate DOIT être inclus dans les messages de réponse 401 (Non autorisé). La valeur du champ consiste en au moins une mise en question qui indique le ou les schéma d'authentification et les paramètres applicables au domaine.

Exemple du champs d'en-tête WWW-Authenticate dans une mise en question 401 :

```
        WWW-Authenticate: Digest
realm="biloxi.com",
qop="auth,auth-int",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

Lorsque l'UAC d'origine reçoit la 401 (Non autorisé), il DEVRAIT, s'il en est capable, régénérer la demande avec l'accréditif approprié. L'UAC peut exiger une entrée de l'utilisateur d'origine avant de continuer. Une fois que l'accréditif d'authentification a été fourni (directement par l'utilisateur, ou découvert dans une interrogation interne), les agents d'utilisateur DEVRAIENT mettre en mémoire l'accréditif pour une valeur donnée de champ d'en-tête To et de "domaine" et essayer de réutiliser ces valeurs sur la prochaine demande pour cette destination. Les agents d'utilisateur PEUT mettre en mémoire l'accréditif de la façon qu'ils veulent.

Si aucun accréditif ne peut être situé pour un domaine, les UAC PEUVENT réessayer la demande avec le nom d'utilisateur "anonyme" et pas de mot de passe (un mot de passe de "").

Une fois que l'accréditif a été situé, tout agent utilisateur qui souhaite s'authentifier auprès d'un UAS ou registraire – habituellement, mais pas nécessairement, après la réception d'une réponse 401 (Non autorisé) -- PEUT le faire en incluant un champ d'en-tête Authorization avec la demande. La valeur du champ Authorization consiste en un accréditif contenant les informations d'authentification de l'agent utilisateur pour le domaine des ressources demandées ainsi que les paramètres requis pour la prise en charge de l'authentification et la protection contre la répétition.

Exemple de champ d'en-tête Authorization :

```
Authorization: Digest username="bob",
realm="biloxi.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
uri="sip:bob@biloxi.com",
qop=auth,
nc=00000001,
cnonce="0a4f113b",
réponse="6629fae49393a05397450978507c4ef1",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

Lorsqu'un UAC resoumet une demande avec son accréditif après avoir reçu une réponse 401 (Non autorisé) ou 407 (Authentification du mandataire exigée), il DOIT incrémenter la valeur du champ d'en-tête CSeq comme il le ferait normalement lors de l'envoi d'une demande mise à jour.

22.3 Authentification de mandataire à utilisateur

De la même façon, lorsqu'un UAC envoie une demande à un serveur mandataire, le serveur mandataire PEUT authentifier l'origine avant que la demande ne soit traitée. Si aucun accréditif (dans le champ d'en-tête Proxy-Authorization) n'est fourni dans la demande, le mandataire peut mettre au défi l'origine de fournir un accréditif en rejetant la demande avec un code d'état 407 (Authentification du mandataire exigée). Le mandataire DOIT remplir le message 407 (Authentification du mandataire exigée) avec une valeur de champ d'en-tête Proxy-Authenticate applicable au mandataire pour la ressource demandée.

L'utilisation de Proxy-Authenticate et Proxy-Authorization est parallèle à celle décrite dans [17], avec une différence. Les mandataires NE DOIVENT PAS ajouter des valeurs au champ d'en-tête Proxy-Authorization. Toutes les réponses 407 (Authentification du mandataire exigée) réponses DOIT être transmises vers l'amont à l'UAC en suivant les procédures pour toutes les autres réponses. Il est de la responsabilité de l'UAC d'ajouter la valeur de champ d'en-tête Proxy-Authorization contenant l'accréditif pour le domaine du mandataire qui a réclamé l'authentification.

Si un mandataire devait resoumettre une demande ajoutant une valeur de champ d'en-tête Proxy-Authorization, il aurait besoin d'incrémenter le CSeq dans la nouvelle demande. Cependant, cela causerait l'élimination de la réponse de l'UAS par l'UAC qui a soumis la demande originelle, car la valeur du CSeq serait différente.

Lorsque l'UAC d'origine reçoit la réponse 407 (Authentification du mandataire exigée) il DEVRAIT, s'il en est capable, régénérer la demande avec l'accréditif approprié. Il devrait suivre la même procédure pour l'affichage du paramètre "domaine" qu'indiqué plus haut pour répondre à 401.

Si aucun accréditif ne peut être trouvé pour un domaine, les UAC PEUVENT réessayer la demande avec un nom d'utilisateur "anonyme" et pas de mot de passe (un mot de passe de "").

L'UAC DEVRAIT aussi mettre en mémoire l'accréditif utilisé dans la demande régénérée.

La règle suivante est RECOMMANDÉE pour la mise en mémoire cache d'accréditif par le mandataire :
Si un agent utilisateur reçoit une valeur de champs d'en-tête Proxy-Authenticate dans une réponse 401/407 à une demande avec un Call-ID particulier, il devrait incorporer l'accréditif pour ce domaine dans toutes les demandes ultérieures qui contiennent le même Call-ID. Ces accréditif NE DOIVENT PAS être mis en mémoire cache à travers les dialogues ; cependant, si un agent utilisateur est configuré avec le domaine de son mandataire local extérieur, lorsqu'il en existe un, cet agent utilisateur PEUT cacher l'accréditif pour ce domaine à travers les dialogues. Noter que cela signifie qu'une demande future dans un dialogue pourrait contenir des accréditifs qui ne sont demandés par aucun mandataire le long du chemin d'en-tête Route.

Tout agent utilisateur qui souhaite s'authentifier auprès d'un serveur mandataire -- habituellement, mais pas nécessairement, après réception d'une réponse 407 (Authentification du mandataire exigée) -- PEUT le faire en incluant une valeur de champ d'en-tête Proxy-Authorization avec la demande. Le champs d'en-tête de demande Proxy-Authorization permet au client de s'identifier (lui ou son utilisateur) auprès d'un mandataire qui exige l'authentification. La valeur de champ d'en-tête Proxy-Authorization consiste en un accréditif contenant les informations d'authentification de l'agent utilisateur pour le mandataire et/ou le domaine des ressources demandées.

Une valeur de champ d'en-tête Proxy-Authorization ne s'applique qu'au mandataire dont le domaine est identifié dans le paramètre "realm" (ce mandataire peut avoir demandé précédemment l'authentification en utilisant le champ Proxy-Authenticate). Lorsque plusieurs mandataires sont utilisés dans une chaîne, une valeur de champ d'en-tête Proxy-Authorization NE DOIT PAS être utilisée par un mandataire dont le domaine ne correspond pas au paramètre "realm" spécifié dans cette valeur.

Noter que si un schéma d'authentification qui ne prend pas en charge les domaines est utilisé dans le champ d'en-tête Proxy-Authorization, un serveur mandataire DOIT essayer d'analyser toutes les valeurs de champ d'en-tête Proxy-Authorization pour déterminer si un d'entre eux a ce que le serveur mandataire considère comme un accréditif valide. Comme ceci peut consommer une grande quantité de temps dans les grands réseaux, les serveurs mandataires DEVRAIENT utiliser un schéma d'authentification qui prenne en charge les domaines dans le champ d'en-tête Proxy-Authorization.

Si une demande est fourchée (comme indiqué au paragraphe 16.7), divers serveurs mandataires et/ou agents d'utilisateur peuvent souhaiter mettre en question l'UAC. Dans ce cas, le serveur mandataire fourcheur est responsable de l'agrégation de ces mises en question en une seule réponse. Chaque valeur de WWW-Authenticate et Proxy-Authenticate reçue dans les réponses à la demande fourchue DOIVENT être placées dans la réponse unique qui est

envoyée par le mandataire fourcheur à l'agent utilisateur ; l'ordre de ces valeurs de champs d'en-tête n'est pas significatif.

Lorsqu'un serveur mandataire produit une mise en question dans la réponse à une demande, il ne traitera pas la demande jusqu'à ce que l'UAC ait réessayé la demande avec un accréditif valide. Un mandataire fourcheur peut transmettre simultanément une demande à plusieurs serveurs mandataires qui exigent l'authentification, chacun d'eux à son tour ne transmettra pas la demande jusqu'à ce que l'UAC d'origine se soit authentifié dans leur domaine respectif. Si l'UAC ne fournit pas d'accréditif pour chaque mise en question, les serveurs mandataires qui ont effectué les mises en question ne transmettront pas les demandes à l'agent utilisateur où l'utilisateur de destination pourrait être localisé, et donc, les avantages du fourchement sont largement perdus.

Lorsqu'il resoumet sa demande dans la réponse à une 401 (Non autorisé) ou 407 (Authentification du mandataire exigée) qui contient plusieurs mises en question, un UAC PEUT inclure une valeur Authorization pour chaque valeur WWW-Authenticate et une valeur Proxy-Authorization pour chaque valeur Proxy-Authenticate pour laquelle l'UAC souhaite fournir un accréditif. Comme noté ci-dessus, plusieurs accréditifs dans une demande DEVRAIENT être différenciés par le paramètre "realm".

Il est possible pour des mises en question multiples associées au même domaine d'apparaître dans le même 401 (Non autorisé) ou 407 (Authentification du mandataire exigée). Cela peut arriver, par exemple, lorsque plusieurs mandataires au sein du même domaine administratif, qui utilisent un domaine commun, sont atteint par une demande fourchue. Lorsqu'il réessaye une demande, un UAC PEUT donc fournir plusieurs accréditifs dans le champ d'en-tête Authorization ou Proxy-Authorization avec la même valeur de paramètre "realm". Le même accréditif DEVRAIT être utilisé pour le même domaine.

22.4 Schéma d'authentification Digest

La présente section décrit les modifications et éclaircissements nécessaires pour appliquer le schéma d'authentification HTTP Digest à SIP. L'utilisation du schéma SIP est presque entièrement identique à celui pour HTTP [17].

Comme la RFC 2543 est fondé sur HTTP Digest comme défini dans la RFC 2069 [39], les serveur SIP qui prennent en charge la RFC 2617 DOIVENT s'assurer qu'ils sont rétro-compatibles avec la RFC 2069. Les procédures pour cette rétro-compatibilité sont spécifiées dans la RFC 2617. Noter cependant, que les serveur SIP NE DOIVENT PAS accepter ou demander l'authentification de base.

Les règles pour l'authentification Digest suivent celles définies dans [17], avec "HTTP/1.1" remplacé par "SIP/2.0" en plus des différences suivantes :

1. L'URI inclus dans la mise en question a le BNF suivant :
   URI = SIP-URI / SIPS-URI
2. Le BNF dans la RFC 2617 a une erreur dans ce que le paramètre 'uri' du champ d'en-tête Authorization pour l'authentification HTTP Digest n'est pas comprise entre des guillemets simples. (L'exemple du paragraphe 3.5 de la RFC 2617 est correct.) Pour SIP, 'uri' DOIT être compris entre des guillemets simples.
3. Le BNF pour digest-uri-value est:
   digest-uri-value = URI-de-demande ; comme définie dans la Section 25
4. L'exemple de procédure pour choisir un nom occasionnel *(nonce)* sur la base de Etag ne marche pas pour SIP.
5. Le texte de la RFC 2617 [17] concernant le fonctionnement de la mémoire cache ne s'applique pas à SIP.
6. La RFC 2617 [17] exige qu'un serveur vérifie que l'URI dans la ligne de demande et l'URI inclus dans le champ d'en-tête Authorization pointent sur la même ressource. Dans un contexte SIP, ces deux URI peuvent se référer à des utilisateurs différents, du fait de la retransmission à certains mandataires. Donc, dans SIP, un serveur PEUT vérifier que l'URI-de-demande dans la valeur de champ d'en-tête Authorization correspond à un utilisateur pour lequel le serveur veut accepter des demandes retransmises ou directes, mais ce n'est pas nécessairement une défaillance si les deux champs ne sont pas équivalents.
7. A titre de précision pour le calcul de la valeur A2 pour l'assurance de l'intégrité du message dans le schéma d'authentification Digest, les mises en œuvre devraient supposer, lorsque le corps d'entité est vide (c'est-à-dire, lorsque les messages SIP n'ont pas de corps) que le hachage du corps d'entité se résout en hachage MD5 d'une chaîne vide, ou :
   H(entity-body) = MD5("") = "d41d8cd98f00b204e9800998ecf8427e"
8. La RFC 2617 note qu'une valeur de cnonce NE DOIT PAS être envoyée dans un champ d'en-tête Authorization (et par extension Proxy-Authorization) si aucune directive qop n'a été envoyée. Donc, tout algorithme qui dépend du cnonce (y compris "MD5-Sess") exige que la directive qop soit envoyée. L'utilisation du paramètre "qop" est facultative dans la RFC 2617 pour les besoins de la rétro-compatibilité avec la RFC 2069 ; comme la RFC 2543 était fondée sur la RFC 2069, la réception du paramètre "qop" doit malheureusement rester facultative pour les clients et

les serveurs. Cependant, les serveurs DOIT toujours envoyer un paramètre "qop" dans les valeurs de champ d'en-tête WWW-Authenticate et Proxy-Authenticate. Si un client reçoit un paramètre "qop" dans un champ d'en-tête de mise en question, il DOIT envoyer le paramètre "qop" dans tout champ d'en-tête d'autorisation en résultant.

La RFC 2543 ne permettait pas l'usage du champ d'en-tête Authentication-Info (qui l'était effectivement dans la RFC 2069). Cependant, nous permettons maintenant l'usage de ce champ d'en-tête, car il donne des vérifications d'intégrité sur les corps de message et une authentification mutuelle. La RFC 2617 [17] définit le mécanisme de la rétro-compatibilité en utilisant l'attribut qop dans la demande. Ce mécanisme DOIT être utilisé par un serveur pour déterminer si le client accepte le nouveau mécanisme de la RFC 2617 qui n'était pas spécifié dans la RFC 2069.

23 S/MIME

Les messages SIP portent des corps MIME et la norme MIME inclut des mécanismes pour sécuriser les contenus MIME pour assurer à la fois l'intégrité et la confidentialité (qui comportent les types MIME 'multipart/signed' et 'application/pkcs7-mime', voir les RFC 1847 [22], RFC 2630 [23] et RFC 2633 [24]). Les développeurs devraient noter, cependant, qu'il peut y avoir certains intermédiaires rares de réseau (qui ne sont pas les serveurs mandataires typiques) qui s'appuient seulement sur la vision ou la modification des corps de messages SIP (particulièrement SDP), et que le MIME sécurisé peut empêcher ces types d'intermédiaires de fonctionner.

Ceci s'applique particulièrement à certains types de pare-feu.

Le mécanisme PGP de chiffrement des champs d'en-tête et corps de message SIP décrits dans la RFC 2543 a été déconseillé.

23.1 Certificats S/MIME

Les certificats qui sont utilisés pour identifier un utilisateur terminal pour les besoins de S/MIME diffèrent de ceux utilisés par les serveurs sur un aspect important – plutôt que de certifier que l'identité du détenteur correspond à un nom d'hôte particulier, ces certificats garantissent que le détenteur est identifié par une adresse d'utilisateur terminal. Cette adresse est composée de l'enchaînement des portions "userinfo" "@" et "domainname" d'un URI SIP ou SIPS (en d'autres termes, une adresse électronique de la forme "bob@biloxi.com"), correspondant plus communément à une adresse d'enregistrement d'utilisateur.

Ces certificats sont aussi associés à des clés qui sont utilisées pour signer ou chiffrer les corps de messages SIP. Les corps sont signés avec la clé privée de l'envoyeur (qui peut inclure leur clé publique avec le message approprié), mais les corps sont chiffrés avec la clé publique du receveur prévu. Visiblement, les envoyeurs doivent avoir une pré connaissance de la clé publique des receveurs afin de chiffrer les corps de message. Les clés publiques peuvent être mémorisées chez un agent utilisateur sur un porte-clés virtuel.

Chaque agent utilisateur qui prend en charge S/MIME DOIT contenir un porte-clés spécifique pour les certificats d'utilisateurs finals. Ce porte-clés devrait faire la transposition entre les adresses d'enregistrement et les certificats correspondants. Avec le temps, les utilisateurs DEVRAIENT utiliser le même certificat quand ils remplissent l'URI d'origine de signalisation (le champ d'en-tête From) avec la même adresse-d'enregistrement.

Tout mécanismes dépendant de l'existence des certificats d'utilisateur final est sérieusement limité en ce qu'il n'y a aujourd'hui virtuellement aucune autorité consolidée qui fournissent des certificats pour les applications d'utilisateur final. Cependant, les usagers DEVRAIENT acquérir des certificats d'autorités de certification connues du public. A titre de remplacement, les usagers PEUVENT créer des certificats auto-signés. Les implications de certificats auto-signés sont examinées plus à fond au paragraphe 26.4.2. Les mises en oeuvre peuvent aussi utiliser des certificats pré-configurés dans des développements dans lesquels existe une relation de confiance préalable entre toutes les entités SIP.

Au-delà du problème de l'acquisition d'un certificat d'utilisateur final, il y a quelques répertoires centralisés bien connus qui distribuent des certificats d'utilisateur final. Cependant, le détenteur d'un certificat DEVRAIT publier son certificat dans tout répertoire public approprié. De même, les UAC DEVRAIENT prendre en charge un mécanisme pour importer (manuellement ou automatiquement) les certificats découverts dans les répertoires publics qui correspondent aux URI cibles des demandes SIP.

23.2 Échange de clé S/MIME

SIP lui-même peut aussi être utilisé comme moyen de distribuer des clés publiques de la manière suivante.

Chaque fois que le message CMS SignedData est utilisé dans S/MIME pour SIP, il DOIT contenir le certificat portant la clé publique nécessaire pour vérifier la signature.

Lorsque un UAC envoie une demande contenant un corps S/MIME qui initialise un dialogue, ou envoie une demande non-INVITE en-dehors du contexte d'un dialogue, l'UAC DEVRAIT structurer le corps comme un corps S/MIME 'multipart/signed' CMS SignedData . Si le service CMS désiré est EnvelopedData (et que la clé publique de l'usager cible est connu), l'UAC DEVRAIT envoyer le message EnvelopedData encapsulé dans un message SignedData.

Lorsqu'un UAS reçoit une demande contenant un corps CMS S/MIME qui inclut un certificat, l'UAS DEVRAIT d'abord valider le certificat, si possible, avec tout certificat racine disponible pour les autorités de certification. L'UAS DEVRAIT aussi déterminer le sujet du certificat (pour S/MIME, le SubjectAltName contiendra l'identité appropriée) et comparer cette valeur au champ d'en-tête From de la demande. Si le certificat ne peut être vérifié, parce qu'il est auto-signé, ou signé par une autorité inconnue, ou si il est vérifiable mais son sujet ne correspond pas au champ d'en-tête From de la demande, l'UAS DOIT notifier à son usager l'état du certificat (y compris le sujet du certificat, son signataire, et toute information clé d'identification) et demander une permission explicite avant de continuer. Si le certificat a été vérifié avec succès et que le sujet du certificat correspond au champ d'en-tête From de la demande SIP, ou si l'utilisateur (après notification) autorise explicitement l'utilisation du certificat, l'UAS DEVRAIT ajouter ce certificat à un porte-clés local, indexé par adresse-d'enregistrement de détenteur du certificat.

Lorsqu'un UAS envoie une réponse contenant un corps S/MIME qui répond à la première demande dans un dialogue, ou à une réponse à une demande non-INVITE en-dehors du contexte d'un dialogue, l'UAS DEVRAIT structurer le corps comme un corps S/MIME 'multipart/signed' CMS SignedData. Si le service CMS désiré est EnvelopedData, l'UAS DEVRAIT envoyer le message EnvelopedData encapsulé dans un message SignedData.

Lorsqu'un UAC reçoit une réponse contenant un corps CMS S/MIME qui comporte un certificat, l'UAC DEVRAIT d'abord valider le certificat, si possible, avec tout certificat racine approprié. L'UAC DEVRAIT aussi déterminer le sujet du certificat et comparer cette valeur au champ To de la réponse ; bien que les voies puissent très bien être différentes, et cela n'est pas nécessairement une indication d'une atteinte à la sécurité. Si le certificat ne peut pas être vérifié parce qu'il est auto-signé, ou signé par une autorité inconnue, l'UAC DOIT notifier à son usager le statut du certificat (y compris le sujet du certificat, son signataire, et toute information d'identification clé) et demander une permission explicite avant de continuer. Si le certificat a été vérifié avec succès, et si le sujet du certificat correspond au champ d'en-tête To de la réponse, ou si l'utilisateur (après notification) autorise explicitement l'utilisation du certificat, l'UAC DEVRAIT ajouter ce certificat à un porte-clés local, indexé par l'adresse-d'enregistrement du détenteur du certificat. Si l'UAC n'a pas transmis son propre certificat à l'UAS dans une transaction précédente, il DEVRAIT utiliser un corps CMS SignedData pour sa prochaine demande ou réponse.

Dans les occasions ultérieures, lorsque l'agent utilisateur reçoit des demandes ou réponses qui contiennent un champ d'en-tête From correspondant à une valeur figurant dans son porte-clés, l'agent utilisateur DEVRAIT comparer le certificat offert dans ces messages avec le certificat existant dans son porte-clés. S'il y a une discordance, l'agent utilisateur DOIT notifier à son usager un changement du certificat (de préférence dans des termes qui indiquent que c'est une atteinte potentielle à la sécurité) et obtenir la permission de l'utilisateur avant de continuer à traiter la signalisation. Si l'utilisateur autorise ce certificat, il DEVRAIT être ajouté au porte-clés avec toutes les autres valeurs précédentes pour cette adresse-d'enregistrement.

Bien noter cependant, que ce mécanisme d'échange de clés ne garantit pas un échange de clés sécurisé lorsque sont utilisés les certificats auto-signés, ou les certificats signés par une obscure autorité - il est vulnérable aux attaques bien connues. Dans l'opinion des auteurs, la sécurité qu'il fournit est cependant notablement meilleure que rien ; il est en fait comparable à l'application SSH largement utilisée. Ces limitations sont examinée plus en détail au paragraphe 26.4.2.

Si un agent utilisateur reçoit un corps S/MIME qui a été chiffré avec une clé publique inconnue du receveur, il DOIT rejeter la demande avec une réponse 493 (Indéchiffrable). Cette réponse DEVRAIT contenir un certificat valide pour le répondant (correspondant, si possible, à toute adresse d'enregistrement donnée dans le champ d'en-tête To de la demande rejetée) au sein d'un corps MIME avec un paramètre 'certs-only' "smime-type".

Un 493 (Indéchiffrable) envoyé sans aucun certificat indique que le répondent ne peut ou ne veut pas utiliser les messages chiffrés S/MIME, bien qu'ils puissent encore prendre en charge les signatures S/MIME.

Noter qu'un agent utilisateur qui reçoit une demande contenant un corps S/MIME qui n'est pas facultatif (avec un paramètre "handling" d'en-tête Content-Disposition de "required") DOIT rejeter la demande avec une réponse 415 Type de support non pris en charge si le type MIME n'est pas compris. Un agent utilisateur qui reçoit une telle réponse lorsque S/MIME est envoyé DEVRAIT notifier à son usager que l'appareil distant ne prend pas S/MIME en charge, et il PEUT ultérieurement renvoyer la demande sans S/MIME, si c'est approprié ; cependant, cette réponse 415 peut constituer une attaque de dépréciation.

Si un agent utilisateur envoie un corps S/MIME dans une demande, mais reçoit une réponse qui contient un corps MIME qui n'est pas sécurisé, l'UAC DEVRAIT notifier à son usager que la session pourrait n'être pas sécurisée. Cependant, si un agent utilisateur qui prend en charge S/MIME reçoit une demande avec un corps non sécurisé, il NE DEVRAIT PAS répondre avec un corps sécurisé, mais si il attend S/MIME de la part de l'envoyeur (par exemple, parce que la valeur du champ d'en-tête From correspond à une identité sur sa chaîne de clés), l'UAS DEVRAIT notifier à son usager que la session pourrait n'être pas sécurisée.

Un certain nombre de conditions soulevées dans le texte précédent invitent à notifier à l'utilisateur la survenance d'un événement de gestion/certificat anormal. Les Usagers peuvent bien se demander ce qu'ils devraient faire dans ces circonstances. Tout d'abord, un changement inattendu dans un certificat, ou une absence de sécurité alors que la sécurité est prévue, sont des raisons de faire attention mais n'indiquent pas nécessairement qu'une attaque est en cours. L'utilisateur peut abandonner toute tentative de connexion ou refuser une demande de connexion reçue ; dans le langage de la téléphonie, il peut raccrocher et rappeler. L'usager peut souhaiter trouver un moyen de remplacement pour contacter l'autre partie et confirmer que leur clé a légitimement été changée. Noter que parfois les usagers sont obligés de changer leurs certificats, par exemple, lorsqu'ils soupçonnent que la confidentialité de leur clé privée a été compromise. Lorsque leur clé privée n'est plus privée, les usagers doivent légitimement générer une nouvelle clé et rétablir la confiance avec tout usager détenteur de leur vieille clé.

Finalement, si durant le cours d'un dialogue, un agent utilisateur reçoit un certificat dans un message CMS SignedData qui ne correspond pas avec les certificats échangé précédemment durant un dialogue, l'agent utilisateur DOIT notifier son changement à l'usager, de préférence dans des termes qui indiquent qu'il s'agit d'une atteinte possible à la sécurité.

23.3 Sécurisation des corps MIME

Il y a deux types de corps MIME sécurisés qui présentent un intérêt pour SIP : l'utilisation de ces corps devrait suivre la spécification S/MIME [24] avec quelques variations.

- "multipart/signed" ne DOIT être utilisé qu'avec des signatures CMS détachées. Cela permet la rétro-compatibilité avec les receveurs non-conformes à S/MIME.
- les corps S/MIME DEVRAIENT avoir un champ d'en-tête Content-Disposition, et la valeur du paramètre "handling" DEVRAIT être "required."
- Si un UAC n'a pas sur son porte-clés de certificat associé à l'adresse-d'enregistrement à laquelle il veut envoyer une demande, il ne peut pas envoyer un message MIME "application/pkcs7-mime" chiffré. Les UAC PEUVENT envoyer une demande initiale telle qu'un message OPTIONS avec une signature CMS détachée afin de solliciter le certificat du côté distant (la signature DEVRAIT être sur un corps "message/sip" du type décrit au paragraphe 23.4). Noter que les futurs travaux de normalisation sur S/MIME pourraient définir des clés non fondées sur des certificats.
- L'envoyeur de corps S/MIME DEVRAIT utiliser l'attribut "SMIMECapabilities" (voir au paragraphe 2.5.2 de [24]) pour exprimer ses capacités et préférences pour les communications ultérieures. Noter particulièrement que l'envoyeur PEUT utiliser la capacité "preferSignedData" pour encourager le receveur à répondre avec des messages CMS SignedData (par exemple, lors de l'envoi d'une demande OPTIONS comme décrit ci-dessus).
- Les mises en oeuvre S/MIME DOIVENT au minimum prendre en charge SHA1 comme algorithmes de signature numérique, et 3DES comme algorithme de chiffrement. Tous les autres algorithmes de signature et de chiffrement PEUVENT être acceptés. Les mises en oeuvre peuvent négocier la prise en charge de ces algorithmes avec l'attribut "SMIMECapabilities".
- Chaque corps S/MIME dans un message SIP DEVRAIT être signé avec un seul certificat. Si un agent utilisateur reçoit un message avec plusieurs signatures, la signature la plus externe devrait être traitée comme le seul certificat pour ce corps. Les signatures parallèles NE DEVRAIENT PAS être utilisées.

Ci-après figure un exemple de corps S/MIME SDP chiffré au sein d'un message SIP :

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;étiquette=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
```

Max-Forwards: 70
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
name=smime.p7m
Content-Disposition: attachment; filename=smime.p7m
handling=required

Content-Type: application/sdp

v=0

o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com

s=-
t=0 0
c=IN IP4 pc33.atlanta.com
m=audio 3456 RTP/AVP 0 1 3 99

a=rtpmap:0 PCMU/8000

23.4 Confidentialité et intégrité des en-têtes SIP en utilisant S/MIME : Tunnelage de SIP

Comme moyen de fourniture d'un certain degré d'authentification, intégrité ou confidentialité de bout en bout pour les champs d'en-tête SIP, S/MIME peut encapsuler des messages SIP entiers dans des corps MIME du type "message/sip" et leur appliquer alors la sécurité MIME de la même manière qu'à des corps SIP normaux. Ces demandes et réponses SIP encapsulées ne constituent pas un dialogue ou une transaction séparé, elles sont une copie du message "extérieur" qui est utilisé pour vérifier l'intégrité ou pour fournir des informations supplémentaires.

Si un UAS reçoit une demande qui contient un corps S/MIME "message/sip" tunnelé , il DEVRAIT inclure un corps "message/sip" tunnelé dans la réponse avec le même smime-type.

Tous les corps MIME traditionnels (tels que SDP) DEVRAIENT être rattachés au message "interne" de sorte qu'ils puissent aussi bénéficier de la sécurité S/MIME. Noter que ce corps "message/sip" peut être envoyé comme partie d'un corps MIME "multipart/mixed" si des types MIME non sécurisés devaient aussi être transmis dans une demande.

23.4.1 Propriétés d'intégrité et de confidentialité des en-têtes SIP

Lorsque les mécanismes S/MIME d'intégrité ou de confidentialité sont utilisés, il peut y avoir des discordances entre les valeurs dans le message "interne" et les valeurs dans le message "externe". Les règles de traitement de telles différences pour tous les champs d'en-tête décrits dans le présent document sont données dans ce paragraphe.

Noter que pour les besoins d'un horodatage lâche, tous les messages SIP qui tunnellent "message/sip" DEVRAIENT contenir un en-tête Date à la fois dans les en-têtes "interne" et "externe".

23.4.1.1 Intégrité

Chaque fois que des vérifications d'intégrité sont effectuées, l'intégrité d'un champ d'en-tête devrait être déterminée en comparant la valeur des champs d'en-tête dans le corps signé et celui des messages "externes" en utilisant les règles de comparaison de SIP comme décrit à la Section 20.

Le champs d'en-tête qui peut être légitimement modifié par les serveurs mandataires sont : URI-de-demande, Via, Record-Route, Route, Max-Forwards, et Proxy- Authorization. Si ces champs d'en-tête ne sont pas intacts de bout en bout, les mises en oeuvre NE DEVRAIENT PAS considérer cela comme une atteinte à la sécurité. Les changements affectant tout autre champ d'en-tête défini dans le présent document constitue une violation d'intégrité ; les usagers DOIVENT recevoir notification d'une discordance.

23.4.1.2 Confidentialité

Lorsque les messages sont chiffrés, des champs d'en-tête peuvent être inclus dans le corps chiffré alors qu'ils ne sont pas présents dans le message "externe".

Certains champs d'en-tête doivent toujours avoir une version en clair parce que ce sont des champs d'en-tête exigés dans les demandes et réponses - cela inclut : To, From, Call-ID, CSeq, Contact.
Alors qu'il n'est probablement pas utile de fournir une version de remplacement chiffrée pour Call-ID, CSeq, ou Contact, la fourniture d'une alternative aux informations dans le message "externe" To ou From est permise. Noter que les valeurs dans un corps chiffré ne sont pas utilisées pour les besoins de l'identification des transactions ou dialogues – elles sont purement informatives. Si dans un corps chiffré le champ d'en-tête From diffère de la valeur dans le message "externe", la valeur dans le corps chiffré DEVRAIT être affichée à l'utilisateur, mais NE DOIT PAS être utilisée dans le champ d'en-tête "externe" de tout futur message.

Au départ, un agent utilisateur va vouloir chiffrer les champs d'en-tête qui ont une sémantique de bout en bout, ce qui inclut : Subject, Reply-To, Organization, Accept, Accept-Encoding, Accept-Language, Alert-Info, Error-Info, Authentication-Info, Expires, In-Reply-To, Require, Supported, Unsupported, Retry-After, User-Agent, Server, et Warning. Si un de ces champs d'en-tête est présent dans un corps chiffré, il devrait être utilisé à la place de tout champ d'en-tête "externe", que cela entraîne l'affichage des valeurs de champ d'en-tête aux usagers ou l'établissement d'états internes dans l'agent utilisateur. Ils NE DEVRAIENT PAS cependant être utilisés dans les en-têtes "externes" de futurs messages.

S'il est présent, le champ d'en-tête Date DOIT toujours être le même dans les en-têtes "internes" et "externes".

Comme les corps MIME sont attachés au message "interne", les mises en oeuvre vont normalement chiffrer les champs d'en-tête spécifiques de MIME, y compris : MIME-Version, Content-Type, Content-Length, Content-Language, Content-Encoding et Content-Disposition. Le message "externe" aura les bons champs d'en-tête MIME pour les corps S/MIME. Ces champs d'en-tête (et tout corps MIME qu'ils préfacent) devraient être traités comme des champs d'en-tête et corps MIME normaux reçus dans un message SIP.

Il n'est pas particulièrement utile de chiffrer les champs d'en-tête suivants : Min-Expires, Timestamp, Authorization, Priority, et WWW-Authenticate. Cette catégorie inclut aussi ces champs d'en-tête qui peuvent être changés par les serveurs mandataires (décrits dans la section précédente). Les agents d'utilisateur NE DEVRAIT jamais inclure ceux-ci dans un message "interne" si il ne sont pas inclus dans le message "externe". Les agents d'utilisateur qui reçoivent un de ces champs d'en-tête dans un corps chiffré DEVRAIENT ignorer les valeurs chiffrées.

Noter que les extensions à SIP peuvent définir des champs d'en-tête supplémentaires ; les auteurs de ces extensions devraient décrire les propriétés d'intégrité et de confidentialité de tels champs d'en-tête. Si un agent utilisateur SIP rencontre un champ d'en-tête inconnu avec une violation d'intégrité, il DOIT ignorer le champ d'en-tête.

23.4.2 Intégrité et authentification de tunnelage

Le tunnelage des messages SIP au sein des corps S/MIME peut fournir l'intégrité pour les champs d'en-tête SIP si les champs d'en-tête que l'envoyeur souhaite sécuriser sont dupliqués dans un corps MIME "message/sip" signé avec une signature CMS détachée.

Pourvu que le corps "message/sip" contienne au moins les identifiants de dialogue fondamentaux (To, From, Call-ID, CSeq), un corps MIME signé peut fournir une authentification limitée. Au minimum, si le certificat utilisé pour signer le corps est inconnu du receveur et ne peut être vérifié, la signature peut être utilisée pour garantir qu'une demande ultérieure dans un dialogue a été transmise par le même détenteur de certificat qui a initialisé le dialogue. Si le receveur du corps MIME signé a une incitation forte à faire confiance au certificat (il a pu être validé, il a été acquis auprès d'un dépositaire de confiance, ou il a été fréquemment utilisé) la signature peut alors être prise comme une forte assertion de l'identité du sujet du certificat.

Afin d'éliminer les confusions possibles sur l'ajout ou le retrait de champs d'en-tête entiers, les envoyeurs DEVRAIENT dupliquer tous les champs d'en-tête de la demande au corps signé. Tout corps de message qui requiert une protection d'intégrité DOIT être attaché au message "interne".

Si un en-tête Date est présent dans un message avec un corps signé, le receveur DEVRAIT comparer la valeur du champ d'en-tête avec sa propre horloge interne, le cas échéant. Si une discordance horaire significative est détectée (de l'ordre d'une heure ou plus), l'agent utilisateur DEVRAIT alerter l'utilisateur de l'anomalie, et noter que c'est une

atteinte potentielle) la sécurité.

Si une violation d'intégrité est détectée dans un message par son receveur, le message PEUT être rejeté avec une réponse 403 (Interdit) si c'est une demande, ou tout dialogue existant PEUT être terminé. Les agents d'utilisateur DEVRAIENT notifier ces circonstances aux usagers et demander des directives explicites sur la façon de continuer.

Ce qui suit est un exemple de l'utilisation d'un corps "message/sip" tunnelé

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: multipart/signed;
protocol="application/pkcs7-signature";
micalg=sha1; boundary=boundary42
Content-Length: 568
--boundary42
Content-Type: message/sip
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <bob@biloxi.com>
From: Alice <alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 147

        v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 pc33.atlanta.com
        t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

--boundary42
        Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;
handling=required
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756
  --boundary42-
```

23.4.3 Chiffrement de tunnelage

Il peut aussi être souhaitable d'utiliser ce mécanisme pour chiffrer un corps de "message/sip" au sein d'un corps S/MIME de message CMS EnvelopedData, mais en pratique, la plupart des champs d'en-tête son au moins utiles au réseau ; l'utilisation générale du chiffrement avec S/MIME sécurise les corps de message comme SDP plutôt que les en-têtes de message. Certains champs d'en-tête informatifs, comme Subject ou Organization pourraient peut-être garantir la sécurité de bout en bout. Les en-têtes définis par les futures applications SIP pourraient aussi exiger un obscurcissement.

Une autre application possible du chiffrement des champs d'en-tête est l'anonymat sélectif. Une demande pourrait être construite avec un champ d'en-tête From ne contenant pas d'informations personnelles (par exemple, sip:anonymous@anonymizer.invalid). Cependant, un second champ d'en-tête From contenant la véritable adresse-d'enregistrement du générateur pourrait être chiffré au sein d'un corps de "message/sip" MIME où il ne serait visible qu'aux points d'extrémité d'un dialogue.

Noter que si ce mécanisme est utilisé pour l'anonymat, le champ d'en-tête From ne sera plus utilisable par le receveur d'un message comme indice de leur chaîne de clé de certificat pour restituer la bonne clé S/MIME associée à l'envoyeur. Le message doit d'abord être décrypté, et le champ d'en-tête From "interne" DOIT être utilisé comme indice.

Pour fournir l'intégrité de bout en bout, les corps MIME "message/sip" chiffrés DEVRAIENT être signés par l'envoyeur. Cela crée un corps MIME "multipart/signed" qui contient un corps chiffré et une signature, tous deux de type "application/pkcs7-mime".

Dans l'exemple suivant de message chiffré et signé, le textes encadré est chiffré :

```
        INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.com>
From: Anonymous <sip:anonymous@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:pc33.atlanta.com>
Content-Type: multipart/signed;
protocol="application/pkcs7-signature";
micalg=sha1; boundary=boundary42
Content-Length: 568

--boundary42
        Content-Type: application/pkcs7-mime; smime-type=enveloped-data; name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m
handling=required
Content-Length: 231

Content-Type: message/sip

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
To: Bob <bob@biloxi.com>
From: Alice <alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE*
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.com>

Content-Type: application/sdp

v=0
o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com
s=Session SDP
t=0 0
c=IN IP4 pc33.atlanta.com
m=audio 3456 RTP/AVP 0 1 3 99
a=rtpmap:0 PCMU/8000

--boundary42
        Content-Type: application/pkcs7-signature; name=smime.p7s
```

Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;
handling=required
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756
--boundary42-

24 Exemples

Dans les exemples suivants, on a souvent omis le corps de message et les champs d'en-tête Content-Length et Content-Type correspondants pour abréger.

24.1 Enregistrement

Bob s'enregistre au départ. Le flux de messages est montré à la Figure 9. Noter que l'authentification habituellement exigée pour l'enregistrement n'est pas montrée pour simplifier.

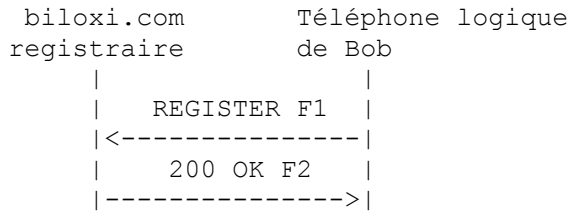


Figure 9 : Exemple d'enregistrement SIP

F1 REGISTER de Bob --> Registraire
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:bob@192.0.2.4>
Expires: 7200
Content-Length: 0

L'enregistrement expire après deux heures. Le registraire répond par un 200 OK :

F2 200 OK du registraire --> Bob
SIP/2.0 200 OK
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7;received=192.0.2.4
To: Bob <sip:bob@biloxi.com>;tag=2493k59kd
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:bob@192.0.2.4>
Expires: 7200
Content-Length: 0

24.2 Établissement de session

Cet exemple contient tous les détails de l'établissement de session exemple de la Section 4. Le flux de message est montré dans la Figure 1. Noter que ces flux montrent l'ensemble minimum requis de champs d'en-tête – certains autres

champs d'en-tête comme Allow et Supported devraient normalement être présents.

F1 INVITE d'Alice --> mandataire atlanta.com
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(On ne montre pas le SDP d'Alice)

F2 100 Essai du mandataire atlanta.com --> Alice
SIP/2.0 100 Trying
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Content-Length: 0

F3 INVITE du mandataire atlanta.com --> mandataire biloxi.com
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
Max-Forwards: 69
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(On ne montre pas le SDP d'Alice)

F4 100 Essai du mandataire biloxi.com --> mandataire atlanta.com
SIP/2.0 100 Trying
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1 ;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Content-Length: 0

F5 INVITE du mandataire biloxi.com --> Bob
INVITE sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8 ;received=192.0.2.1
Max-Forwards: 68
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(On ne montre pas le SDP d'Alice)

F6 180 Sonnerie de Bob --> mandataire biloxi.com
SIP/2.0 180 Sonnerie
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0

F7 180 Sonnerie de biloxi.com --> mandataire atlanta.com
SIP/2.0 180 Sonnerie
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0

F8 180 Sonnerie du mandataire atlanta.com --> Alice
SIP/2.0 180 Sonnerie
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0

F9 200 OK de Bob --> mandataire biloxi.com
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131

(On ne montre pas le SDP de Bob)

F10 200 OK du mandatairebiloxi.com --> mandataire atlanta.com
SIP/2.0 200 OK
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp

Content-Length: 131

(On ne montre pas le SDP de Bob)

F11 200 OK du mandataire atlanta.com --> Alice
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131

(On ne montre pas le SDP de Bob)

F12 ACK d'Alice --> Bob
ACK sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds9
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 ACK
Content-Length: 0

La session entre Alice et Bob est maintenant établie.

Bob raccroche le premier. Noter que le téléphone SIP de Bob maintient son propre espace de numérotation CSeq, qui, dans cet exemple, commence par 231. Comme Bob fait la demande, les URI To et From et les étiquettes ont été sautés.

F13 BYE de Bob --> Alice
BYE sip:alice@pc33.atlanta.com SIP/2.0
Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>;tag=a6c85cf
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0

F14 200 OK d'Alice --> Bob
SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
From: Bob <sip:bob@biloxi.com>;tag=a6c85cf
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0

Le document des flux d'appel SIP [40] contient d'autres exemples de messages SIP.

25 BNF augmenté pour le protocole SIP

Tous les mécanismes spécifiés dans ce document sont décrits à la fois en prose et en forme Backus-Naur augmentée (BNF) définie dans la RFC 2234 [10]. Le paragraphe 6.1 de la RFC 2234 définit un ensemble de règles centrales qui sont utilisées par cette spécification, et ne sont pas répétées ici. Les mises en œuvre ont besoin d'être familiarisées avec la notation et le contenu de la RFC 2234 afin de comprendre la présente spécification. Certaines règles de base sont en majuscules , telles que SP, LWS, HTAB, CRLF, DIGIT, ALPHA, etc. Des crochets angulaires sont utilisés dans les définitions pour clarifier l'utilisation des noms de règles.

L'utilisation des crochets carrés est syntaxiquement redondante. Elle est utilisée comme conseil sémantique pour indiquer que l'utilisation du paramètre spécifique est facultative.

25.1 Règles de base

Les règles suivantes sont utilisées tout au long de la présente spécification pour décrire la construction de l'analyse grammaticale de base. L'ensemble des caractères codés de l'US-ASCII est défini par ANSI X3.4-1986.

alphanum = ALPHA / DIGIT

Plusieurs règles sont tirées de la RFC 2396 [5] mais sont mises à jour pour les rendre conformes à la RFC 2234 [10]. Ceci inclut :

reserved = ";" / "/" / "?" / ":" / "@" / "&" / "=" / "+" / "$" / ","
unreserved = alphanum / mark
mark = "-" / "_" / "." / "!" / "~" / "*" / "'" / "(" / ")"
escaped = "%" HEXDIG HEXDIG

Les valeurs de champs d'en-tête SIP peuvent être développées sur plusieurs lignes si la ligne de continuation commence par un espace ou une tabulation horizontale. Tous les espaces blancs linéaires, y compris folding, ont la même sémantique que SP. Un receveur PEUT remplacer tout espace blanc linéaire par un seul SP avant d'interpréter la valeur du champ ou de transmettre le message vers l'aval. Ceci est destiné à se comporter exactement comme HTTP/1.1, comme décrit dans la RFC 2616 [8]. La construction SWS est utilisée lorsqu l'espace blanc linéaire est facultatif, habituellement entre jetons et séparateurs.

LWS = [*WSP CRLF] 1*WSP ; espace blanc linéaire
SWS = [LWS] ; espace blanc séparé

Pour séparer le nom d'en-tête du reste de la valeur, on utilise les deux points, ce qui, d'après la règle ci-dessus, permet un espace blanc devant, mais pas de rupture de ligne, et l'espace après, incluant une rupture de ligne. Le HCOLON définit cette construction.

HCOLON = *(SP / HTAB) ":" SWS

La règle TEXT-UTF8 n'est utilisée que pour les contenus de champs descriptifs et les valeurs qui ne sont pas destinées à être interprétées par l'analyseur de message. Les mots de *TEXT-UTF8 contiennent des caractères provenant de l'ensemble de caractères *(charset)* UTF-8 (RFC 2279 [7]). La règle TEXT-UTF8-TRIM est utilisée pour les contenus de champs descriptifs qui ne sont pas des chaînes entre guillemets, où le LWS de tête et de queue n'est pas significatif. A cet égard, SIP diffère de HTTP, qui utilise l'ensemble de caractères ISO 8859-1.

TEXT-UTF8-TRIM = 1*TEXT-UTF8char *(*LWS TEXT-UTF8char)
TEXT-UTF8char = %x21-7E / UTF8-NONASCII
UTF8-NONASCII = %xC0-DF 1UTF8-CONT
/ %xE0-EF 2UTF8-CONT
/ %xF0-F7 3UTF8-CONT
/ %xF8-Fb 4UTF8-CONT
/ %xFC-FD 5UTF8-CONT
UTF8-CONT = %x80-BF

Un CRLF n'est admis dans la définition de TEXT-UTF8-TRIM qu'en tant que partie d'une continuation de champ d'en-tête. Il est prévu que le folding LWS sera remplacé par un seul SP avant l'interprétation de la valeur TEXT-UTF8-TRIM.

Les caractères numériques hexadécimaux sont utilisés dans plusieurs éléments des protocoles. Certains éléments (authentification) forcent les caractères alphabétiques de l'hexadécimal à être en minuscules.

LHEX = DIGIT / %x61-66 ;lowercase a-f

De nombreuses valeurs de champs d'en-tête SIP consistent en mots séparés par des LWS ou des caractères spéciaux.

Sauf mention contraire les jetons sont insensibles à la casse. Ces caractères spéciaux DOIVENT être dans une chaîne entre guillemets pour être utilisés au sein d'une valeur de paramètre. Le mot construit est utilisé dans le Call-ID pour permettre d'utiliser la plupart des séparateurs.

```
jeton = 1*(alphanum / "-" / "." / "!" / "%" / "*" / "_" / "+" / "`" / "'" / "~" )
séparateurs =  "(" / ")" / "<" / ">" / "@" / "," / ";" / ":" / "\" / DQUOTE / "/" / "[" / "]" / "?" / "=" / "{" / "}" / SP / HTAB
mot = 1*(alphanum / "-" / "." / "!" / "%" / "*" / "(" / ")" / "<" / ">" / "/" / "[" / "]" / "?" / "{" / "}" )
```

Lorsque les jetons sont utilisés ou que les séparateurs sont utilisés entre les éléments, l'espace blancs est souvent permis avant ou après ces caractères :

```
STAR   =  SWS "*" SWS ; astérisque
SLASH  =  SWS "/" SWS ; barre oblique
EQUAL  =  SWS "=" SWS ; égal
LPAREN =  SWS "(" SWS ; parenthèse gauche
RPAREN =  SWS ")" SWS ; parenthèse droite
RAQUOT =  ">" SWS ; crochet angulaire droit
LAQUOT =  SWS "<"; crochet angulaire gauche
COMMA  =  SWS "," SWS ; virgule
SEMI   =  SWS ";" SWS ; point virgule
COLON  =  SWS ":" SWS ; deux points
LDQUOT =  SWS DQUOTE ; guillemet double ouvert
RDQUOT =  DQUOTE SWS ; guillemet double fermé
```

Les commentaires peuvent être inclus dans certains champs d'en-tête SIP en mettant le texte du commentaire entre parenthèses. Les commentaires ne sont admis que dans les champs qui contiennent "comment" au titre de leur définition de valeur de cham. Dans tous les autres champs, les parenthèses sont considérées comme faisant partie de la valeur du champ.

```
comment = LPAREN *(ctext / quoted-pair / comment) RPAREN
ctext =  %x21-27 / %x2A-5B / %x5D-7E / UTF8-NONASCII / LWS
```

ctext inclut tous les caractères excepté les parenthèses gauche et droite et la barre oblique inversée. Une chaîne de texte est analysée comme un seul mot si elle est entre des doubles guillemets. Dans les chaînes entre guillemets, les guillemets doubles (") et les barres obliques inversées (\) n'ont pas besoin d'être codées.

```
quoted-string  =  SWS DQUOTE *(qdtext / quoted-pair ) DQUOTE
qdtext         =  LWS / %x21 / %x23-5B / %x5D-7E / UTF8-NONASCII
```

Le caractère barre oblique inversée ("\") ne PEUT être utilisé comme mécanisme de mise entre guillemets à un seul caractère que dans une chaîne entre guillemets et dans la construction de commentaire. A la différence de HTTP/1.1, les caractères CR et LF ne peuvent pas être formatés par ce mécanisme pour éviter les conflits avec le retour à la ligne et la séparation d'en-tête.

```
quoted-pair        =  "\" (%x00-09 / %x0B-0C / %x0E-7F)
SIP-URI=  "sip:" [ userinfo ] hostport uri-parameters [ headers ]
SIPS-URI           =  "sips:" [ userinfo ] hostport uri-parameters [ headers ]
userinfo =  ( user / telephone-subscriber ) [ ":" password ] "@"
user       =  1*( unreserved / escaped / user-unreserved )
user-unreserved  =  "&" / "=" / "+" / "$" / "," / ";" / "?" / "/"
password           =  *( unreserved / escaped / "&" / "=" / "+" / "$" / "," )
hostport =  host [ ":" port ]
host       =  hostname / IPv4address / IPv6reference
hostname           =  *( domainlabel "." ) toplabel [ "." ]
domainlabel        =  alphanum / alphanum *( alphanum / "-" ) alphanum
toplabel =  ALPHA / ALPHA *( alphanum / "-" ) alphanum
IPv4address        =  1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT
IPv6reference      =  "[" IPv6address "]"
IPv6address        =  hexpart [ ":" IPv4address ]
hexpart  =  hexseq / hexseq "::" [ hexseq ] / "::" [ hexseq ]
hexseq   =  hex4 *( ":" hex4)
hex4       =  1*4HEXDIG
port       =  1*DIGIT
```

Le BNF pour le numéro d'abonné téléphonique peut être trouvé dans la RFC 2806 [9]. Noter, cependant, que tout caractère admis ici qui n'est pas autorisé dans la partie utilisateur de l'URI SIP DOIT être formaté.

```
uri-parameters    = *( ";" uri-parameter)
uri-parameter     = transport-param / user-param / method-param / ttl-param / maddr-param / lr-param / other-param
transport-param   = "transport=" ( "udp" / "tcp" / "sctp" / "tls" / other-transport)
other-transport   = token
user-param        = "user=" ( "phone" / "ip" / other-user)
other-user        = token
method-param      = "method=" Method
ttl-param         = "ttl=" ttl
maddr-param       = "maddr=" host
lr-param=  "lr"
other-param       = pname [ "=" pvalue ]
pname   = 1*paramchar
pvalue  = 1*paramchar
paramchar         = param-unreserved / unreserved / escaped
param-unreserved  = "[" / "]" / "/" / ":" / "&" / "+" / "$"
headers           = "?" header *( "&" header )
header  = hname "=" hvalue
hname   = 1*( hnv-unreserved / unreserved / escaped )
hvalue  = *( hnv-unreserved / unreserved / escaped )
hnv-unreserved  = "[" / "]" / "/" / "?" / ":" / "+" / "$"
SIP-message       = Request / Response
Request =  Request-Line
           *( message-header )
           CRLF
             [ message-body ]
Request-Line      = Method SP URI-de-demande SP SIP-Version CRLF
URI-de-demande=  SIP-URI / SIPS-URI / absoluteURI
absoluteURI       = scheme ":" ( hier-part / opaque-part )
hier-part=  ( net-path / abs-path ) [ "?" query ]
net-path =  "//" authority [ abs-path ]
abs-path=  "/" path-segments
opaque-part       = uric-no-slash *uric
uric      = reserved / unreserved / escaped
uric-no-slash     = unreserved / escaped / ";" / "?" / ":" / "@" / "&" / "=" / "+" / "$" / ","
path-segments     = segment *( "/" segment )
segment=  *pchar *( ";" param )
param   = *pchar
pchar   = unreserved / escaped / ":" / "@" / "&" / "=" / "+" / "$" / ","
scheme  = ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )
authority           =  srvr / reg-name
srvr      = [ [ userinfo "@" ] hostport ]
reg-name          = 1*( unreserved / escaped / "$" / "," / ";" / ":" / "@" / "&" / "=" / "+" )
query   = *uric
SIP-Version       = "SIP" "/" 1*DIGIT "." 1*DIGIT

message-header  = (Accept
                / Accept-Encoding
                / Accept-Language
                / Alert-Info
                / Allow
                / Authentication-Info
                / Authorization
                / Call-ID
                / Call-Info
                / Contact
               / Content-Disposition
                / Content-Encoding
                / Content-Language
                / Content-Length
```

```
                / Content-Type
                / CSeq
                / Date
                / Error-Info
                / Expires
                / From
                / In-Reply-To
                / Max-Forwards
                / MIME-Version
                / Min-Expires
                / Organization
                / Priority
                / Proxy-Authenticate
                / Proxy-Authorization
                / Proxy-Require
                / Record-Route
                / Reply-To
                / Require
                / Retry-After
                / Route
                / Server
                / Subject
                / Supported
                / Timestamp
                / To
                / Unsupported
                / User-Agent
                / Via
                / Warning
                / WWW-Authenticate
                / extension-header) CRLF

INVITEm         = %x49.4E.56.49.54.45 ; INVITE dans caps
ACKm  = %x41.43.4B ; ACK dans caps
OPTIONSm        = %x4F.50.54.49.4F.4E.53 ; OPTIONS dans caps
BYEm  = %x42.59.45 ; BYE dans caps
CANCELm         = %x43.41.4E.43.45.4C ; CANCEL dans caps
REGISTERm       = %x52.45.47.49.53.54.45.52 ; REGISTER dans caps
Method = INVITEm / ACKm / OPTIONSm / BYEm / CANCELm / REGISTERm / extension-method
extension-method  = token
Response        = Ligne-d'état *( en-tête-de-message ) CRLF [ corps-de-message ]
Ligne-d'état    = Version-SIP SP Code-d'état SP Phrase-de-cause CRLF
Code-d'état     = Informatif
        / Redirection
        / Succès
        / Erreur-Client
        / Erreur-Serveur
        / Echec-Global
        / code-d'extension
code-d'extension = 3DIGIT
Phrase-de-cause  = *(réservé / non-réservé / formaté / UTF8-NONASCII / UTF8-CONT / SP / HTAB)

Informatif      = "100" ; Essai
        / "180" ; Sonnerie
        / "181" ; Appel en cours de transmission
        / "182" ; En file d'attente
        / "183" ; Session en cours

Réussite        = "200" ; OK

Redirection     = "300" ; Choix Multiple
        / "301" ; Déplacement Permanent
        / "302" ; Déplacement Temporaire
```

```
        /  "305"  ;  Utiliser un mandataire
        /  "380"  ;  Service de remplacement

Erreur-Client     =  "400"  ;  Mauvaise Demande
        /  "401"  ;  Non autorisé
        /  "402"  ;  Payement exigé
        /  "403"  ;  Interdit
        /  "404"  ;  Pas Trouvé
        /  "405"  ;  Méthode Non Admise
        /  "406"  ;  Non Acceptable
        /  "407"  ;  Authentification du mandataire exigée
        /  "408"  ;  Expiration du délai de réponse
        /  "410"  ;  Parti
        /  "413"  ;  Entité de Demande Trop Longue
        /  "414"  ;  URI-de-demande Trop Long
        /  "415"  ;  Type de Support Non Accepté
        /  "416"  ;  Schéma d'URI Non Accepté
        /  "420"  ;  Mauvaise Extension
        /  "421"  ;  Extension Exigée
        /  "423"  ;  Intervalle Trop Bref
        /  "480"  ;  Temporairement non disponible
        /  "481"  ;  Chemin d'appel/Transaction Inexistant
        /  "482"  ;  Boucle Détectée
        /  "483"  ;  Trop de Sauts
        /  "484"  ;  Adresse Incomplète
        /  "485"  ;  Ambigu
        /  "486"  ;  Occupé Ici
        /  "487"  ;  Demande Terminée
        /  "488"  ;  Non Acceptable ici
        /  "491"  ;  Demande en cours
        /  "493"  ;  Indéchiffrable

Erreur-du-Serveur =  "500"  ;  Erreur Interne du Serveur
        /  "501"  ;  Non Mis en oeuvre
        /  "502"  ;  Mauvaise passerelle
        /  "503"  ;  Service Indisponible
        /  "504"  ;  Expiration du délai du serveur
        /  "505"  ;  Version SIP non acceptée
        /  "513"  ;  Message trop long

Echec-Global      =  "600"  ;  Occupé partout
        /  "603"  ;  Refus
        /  "604"  ;  N'existe nulle part
        /  "606"  ;  Non acceptable

Accept            =  "Accept" HCOLON [ accept-range *(COMMA accept-range) ]
accept-range      =  media-range *(SEMI accept-param)
media-range       =  ( "*/*" / ( m-type SLASH "*" ) / ( m-type SLASH m-subtype ) ) *( SEMI m-parameter )
accept-param      =  ("q" EQUAL qvaleur) / generic-param
qvalue    =  ( "0" [ "." 0*3DIGIT ] ) / ( "1" [ "." 0*3("0") ] )
generic-param     =  token [ EQUAL gen-value ]
gen-value         =  token / host / quoted-string

Accept-Encoding=  "Accept-Encoding" HCOLON [ encoding *(COMMA encoding) ]
encoding          =  codings *(SEMI accept-param)
codings =  content-coding / "*"
content-coding    =  token

Accept-Language   =  "Accept-Language" HCOLON [ language *(COMMA language) ]
language          = language-range *(SEMI accept-param)
language-range    = ( ( 1*8ALPHA *( "-" 1*8ALPHA ) ) / "*" )
Alert-Info        = "Alert-Info" HCOLON alert-param *(COMMA alert-param)
alert-param       = LAQUOT absoluteURI RAQUOT *( SEMI generic-param )
```

```
Allow    = "Allow" HCOLON [Method *(COMMA Method)]
Authorization     = "Authorization" HCOLON accréditif
accréditif        = ("Digest" LWS digest-response) / autre-réponse
digest-response   = dig-resp *(COMMA dig-resp)
dig-resp          = username / realm / nonce / digest-uri / dresponse / algorithm / cnonce / opaque / message-qop /
                    nonce-count / auth-param
username          = "username" EQUAL username-value
username-value    = quoted-string
digest-uri        = "uri" EQUAL LDQUOT digest-uri-value RDQUOT
digest-uri-value  = rquest-uri ; Equal to request-uri as specified by HTTP/1.1
message-qop       = "qop" EQUAL qop-value
cnonce   = "cnonce" EQUAL cnonce-value
cnonce-value      = nonce-value
nonce-count       = "nc" EQUAL nc-value
nc-value= 8LHEX
dresponse         =  "response" EQUAL request-digest
request-digest    =  LDQUOT 32LHEX RDQUOT
auth-param        =  auth-param-name EQUAL ( token / quoted-string )
auth-param-name=  token
other-response    =  auth-scheme LWS auth-param  *(COMMA auth-param)
auth-scheme       =  token

Authentication-Info  =  "Authentication-Info" HCOLON ainfo  *(COMMA ainfo)
Ainfo    =  nextnonce / message-qop / response-auth / cnonce / nonce-count
Nextnonce         =  "nextnonce" EQUAL nonce-value
response-auth     =  "rspauth" EQUAL response-digest
response-digest   =  LDQUOT *LHEX RDQUOT

Call-ID           =  ( "Call-ID" / "i" ) HCOLON callid
callid    =  word [ "@" word ]

Call-Info         =  "Call-Info" HCOLON info *(COMMA info)
info      =  LAQUOT absoluteURI RAQUOT *( SEMI info-param)
info-param        =  ( "purpose" EQUAL ( "icon" / "info" / "card" / token ) ) / generic-param

Contac            =  ("Contact" / "m" ) HCOLON ( STAR / (contact-param *(COMMA contact-param)))
contact-param     =  (name-addr / addr-spec) *(SEMI contact-params)
name-addr         =  [ display-name ] LAQUOT addr-spec RAQUOT
addr-spec         =  SIP-URI / SIPS-URI / absoluteURI
display-name      =  *(token LWS)/ quoted-string

contact-params    =  c-p-q / c-p-expires / contact-extension
c-p-q     =  "q" EQUAL qvaleur
c-p-expires       =  "expires" EQUAL delta-seconds
contact-extension=  generic-param
delta-seconds     =  1*DIGIT

Content-Disposition =  "Content-Disposition" HCOLON disp-type *( SEMI disp-param )
disp-type         =  "render" / "session" / "icon" / "alert" / disp-extension-token
disp-param        =  handling-param / generic-param
handling-param    =  "handling" EQUAL ( "optional" / "required" / other-handling )
other-handling    =  token
disp-extension-token        =  token

Content-Encoding    =  ( "Content-Encoding" / "e" ) HCOLON content-coding *(COMMA content-coding)

Content-Language =  "Content-Language" HCOLON language-tag *(COMMA language-tag)
language-tag      =  primary-tag *( "-" subtag )
primary-tag       =  1*8ALPHA
subtag    =  1*8ALPHA

Content-Length   =  ( "Content-Length" / "l" ) HCOLON 1*DIGIT
Content-Type     =  ( "Content-Type" / "c" ) HCOLON media-type
media-type        =  m-type SLASH m-subtype *(SEMI m-parameter)
```

```
m-type  =  discrete-type / composite-type
discrete-type      =  "text" / "image" / "audio" / "video" / "application" / extension-token
composite-type   =  "message" / "multipart" / extension-token
extension-token  =  ietf-token / x-token
ietf-token          =  token
x-token  =  "x-" token
m-subtype          =  extension-token / iana-token
iana-token          =  token
m-paramètre       =  m-attribute EQUAL m-value
m-attribute         =  token
m-valeur            =  token / quoted-string

Cseq                =  "CSeq" HCOLON 1*DIGIT LWS Method

Date                =  "Date" HCOLON SIP-date
SIP-date=  rfc1123-date
rfc1123-date       =  wkday "," SP date1 SP time SP "GMT"
date1     =  2DIGIT SP month SP 4DIGIT ; day month year (e.g., 02 Jun 1982)
time      =  2DIGIT ":" 2DIGIT ":" 2DIGIT ; 00:00:00 - 23:59:59
wkday   =  "Mon" / "Tue" / "Wed" / "Thu" / "Fri" / "Sat" / "Sun"
month   =  "Jan" / "Feb" / "Mar" / "Apr" / "May" / "Jun" / "Jul" / "Aug" / "Sep" / "Oct" / "Nov" / "Dec"

Error-Info          =  "Error-Info" HCOLON error-uri *(COMMA error-uri)
error-uri=  LAQUOT absoluteURI RAQUOT *( SEMI generic-param )

Expires             =  "Expires" HCOLON delta-seconds
From     =  ( "From" / "f" ) HCOLON from-spec
from-spec           =  ( name-addr / addr-spec ) *( SEMI from-param )
from-param          =  tag-param / generic-param
tag-param           =  "tag" EQUAL token

In-Reply-To         =  "In-Reply-To" HCOLON callid *(COMMA callid)

Max-Forwards    =  "Max-Forwards" HCOLON 1*DIGIT

MIME-Version    =  "MIME-Version" HCOLON 1*DIGIT "." 1*DIGIT

Min-Expires         =  "Min-Expires" HCOLON delta-seconds

Organization        =  "Organization" HCOLON [TEXT-UTF8-TRIM]

Priority            =  "Priority" HCOLON priority-value
priority-value      =  "emergency" / "urgent" / "normal" / "non-urgent" / other-priority
other-priority      =  token

Proxy-Authenticate  =  "Proxy-Authenticate" HCOLON challenge
challenge           =  ("Digest" LWS digest-cln *(COMMA digest-cln)) / other-challenge
other-challenge   =  auth-scheme LWS auth-param      *(COMMA auth-param)
digest-cln          =  realm / domain / nonce / opaque / stale / algorithm / qop-options / auth-param
realm     =  "realm" EQUAL realm-value
realm-value         =  quoted-string
domain  =  "domain" EQUAL LDQUOT URI *( 1*SP URI ) RDQUOT
URI       =  absoluteURI / abs-path
nonce     =  "nonce" EQUAL nonce-value
nonce-value         =  quoted-string
opaque  =  "opaque" EQUAL quoted-string
stale     =  "stale" EQUAL ( "true" / "false" )
algorithm           =  "algorithm" EQUAL ( "MD5" / "MD5-sess" / token )
qop-options         =  "qop" EQUAL LDQUOT qop-valeur    *("," qop-valeur) RDQUOT
qop-value           =  "auth" / "auth-int" / token

Proxy-Authorization =  "Proxy-Authorization" HCOLON credential
Proxy-Require    =  "Proxy-Require" HCOLON option-tag          *(COMMA option-tag)
```

option-tag = token

Record-Route = "Record-Route" HCOLON rec-route *(COMMA rec-route)
rec-route = name-addr *(SEMI rr-param)
rr-param= generic-param

Reply-To = "Reply-To" HCOLON rplyto-spec
rplyto-spec = (name-addr / addr-spec) *(SEMI rplyto-param)
rplyto-param = generic-param
Require = "Require" HCOLON option-étiquette *(COMMA option-étiquette)

Retry-Afte = "Retry-After" HCOLON delta-seconds [comment] *(SEMI retry-param)

retry-param = ("duration" EQUAL delta-seconds) / generic-param

Route = "Route" HCOLON route-param *(COMMA route-param)
route-param = name-addr *(SEMI rr-param)

Server = "Server" HCOLON serveur-val *(LWS serveur-val)
serveur-val = product / comment
product = token [SLASH product-version]
product-version = token

Subject = ("Subject" / "s") HCOLON [TEXT-UTF8-TRIM]

Supported = ("Supported" / "k") HCOLON [option-tag *(COMMA option-tag)]

Timestamp = "Timestamp" HCOLON 1*(DIGIT) ["." *(DIGIT)] [LWS delay]
Delay = *(DIGIT) ["." *(DIGIT)]

To = ("To" / "t") HCOLON (name-addr / addr-spec) *(SEMI to-param)
to-param = tag-param / generic-param

Unsupported = "Unsupported" HCOLON option-étiquette *(COMMA option-étiquette)
User-Agent = "User-Agent" HCOLON serveur-val *(LWS serveur-val)

Via = ("Via" / "v") HCOLON via-parm *(COMMA via-parm)
via-parm = envoyé-protocol LWS envoyé-by *(SEMI via-params)
via-params = via-ttl / via-maddr / via-received / via-branch / via-extension
via-ttl = "ttl" EQUAL ttl
via-maddr = "maddr" EQUAL host
via-received = "received" EQUAL (IPv4address / IPv6address)
via-branch = "branch" EQUAL token
via-extension = generic-param
send-protocol = protocol-name SLASH protocol-version SLASH transport
protocol-name = "SIP" / token
protocol-version = token
transport = "UDP" / "TCP" / "TLS" / "SCTP" / other-transport
send-by = host [COLON port]
ttl = 1*3DIGIT ; 0 to 255

Warning = "Warning" HCOLON warning-valeur *(COMMA warning-value)
warning-value = warn-code SP warn-agent SP warn-text
warn-code = 3DIGIT
warn-agent = hostport / pseudonym ; nom ou pseudonyme du serveur qui ajoute l'en-tête d'avertissement à utiliser pour le déboggage
warn-text = quoted-string
pseudonym = token

WWW-Authenticate = "WWW-Authenticate" HCOLON challenge

extension-header = header-name HCOLON header-value

```
header-name     = token
header-value    = *(TEXT-UTF8char / UTF8-CONT / LWS)
message-body    = *OCTET
```

26 Considérations sur la sécurité : Modèle de menace et recommandations d'utilisation de la sécurité

SIP est un protocole qu'il n'est pas facile de sécuriser. Son utilisation d'intermédiaires, ses relations de confiance multi-faces, son usage prévu entre éléments sans confiance du tout, et son fonctionnement d'usager à usager font que le problème de la sécurité est loin d'être trivial. Des solutions de sécurité pour aujourd'hui sont nécessaires, sans coordination excessive, dans une grande variété d'environnements et d'utilisations. Afin de satisfaire à ces besoins divers, plusieurs mécanismes distincts applicables aux différents aspects et usages de SIP sont nécessaires.

Noter que la sécurité de la signalisation SIP elle-même n'a pas d'impact sur la sécurité des protocoles utilisés de concert avec SIP comme RTP, ou sur les implications de sécurité que pourraient porter les corps spécifiques de SIP (bien que la sécurité de MIME joue un rôle substantiel dans la sécurisation de SIP). Tout support associé à une session peut être chiffré de bout en bout indépendamment de toute signalisation SIP associée. Le chiffrage de support est en-dehors du domaine d'application du présent document.

Les considérations qui suivent examinent d'abord un ensemble de modèles de menaces classiques qui s'identifient en gros aux besoins de sécurité de SIP. L'ensemble des services de sécurité pour contrer ces menaces est ensuite détaillé, suivi par une explication de plusieurs mécanismes de sécurité qui peuvent être utilisés pour fournir ces services. Ensuite sont énumérées les exigences pour les mises en œuvre de SIP, ainsi que des exemples de développement dans lesquels ces mécanismes de sécurité pourraient être utilisés pour améliorer la sécurité de SIP. Quelques notes sur la confidentialité concluent cette section.

26.1 Modèles d'attaques et de menaces

Ce paragraphe précise certaines menaces qui devraient être communes à la plupart des développements de SIP. Ces menaces ont été choisies spécifiquement pour illustrer chacun des services de sécurité exigés par SIP.

Les exemples suivants ne fournissent en aucune façon une liste exhaustive des menaces contre SIP ; elles sont plutôt des menaces "classiques" qui démontrent le besoin de services de sécurité particuliers qui puissent potentiellement empêcher toutes les catégories de menaces.

Ces attaques supposent un environnement dans lequel les attaquants ont la possibilité de lire tout paquet sur le réseau – on prévoit que SIP sera fréquemment utilisé sur l'Internet public. Les attaquants sur le réseau peuvent être capables de modifier les paquets (peut-être à quelque intermédiaire compromis). Les attaquants peuvent souhaiter voler des services, espionner les communications, ou interrompre les sessions.

26.1.1 Capture d'enregistrement

Le mécanisme d'enregistrement de SIP permet à un agent utilisateur de s'identifier auprès d'un registraire comme un appareil auquel est localisé un usager (désigné par une adresse d'enregistrement). Un registraire atteste de l'identité affirmée dans le champ d'en-tête From d'un message REGISTER pour déterminer si cette demande peut modifier les adresses de contact associées à l'adresse-d'enregistrement dans le champ d'en-tête To. Alors que ces deux champs sont fréquemment les mêmes, il y a de nombreux développements valides dans lesquels une tierce partie peut enregistrer des contacts au nom d'un utilisateur.

Le champ d'en-tête From d'une demande SIP, peut cependant être modifié arbitrairement par le possesseur d'un agent utilisateur, et cela ouvre la porte à des enregistrements malveillants. Un attaquant qui réussit à s'emparer de l'identité d'une partie autorisée à changer les contacts associés à une address-of- record pourrait, par exemple, désenregistrer tous les contacts existants pour un URI et enregistrer ensuite son propre appareil comme l'adresse de contact appropriée, dirigeant par ce moyen toutes les demandes de l'usager affecté sur l'appareil de l'attaquant.

Cette menace appartient à une famille de menaces qui s'appuient sur l'absence d'assurance cryptographique du générateur de la demande. Tout UAS SIP qui représente un service de valeur (par exemple, une passerelle qui fait

interagir les demandes SIP avec les appels téléphoniques traditionnels) pourrait vouloir contrôler l'accès à ses ressources en authentifiant les demandes qu'il reçoit. Même les agents d'utilisateur d'usager terminal, par exemple les téléphones SIP, ont intérêt à s'assurer de l'identité de celui qui est à l'origine des demandes.

Cette menace démontre le besoin de services de sécurité permettant aux entités SIP d'authentifier l'origine des demandes.

26.1.2 Usurpation du nom d'un serveur

Le domaine auquel une demande est destinée est généralement spécifié dans l'URI-de-demande. Les agents d'utilisateur contactent normalement directement un serveur dans ce domaine afin de livrer une demande. Cependant, il est toujours possible qu'un attaquant puisse s'emparer de l'identité du serveur distant, et que la demande de l'agent utilisateur soit interceptée par quelque autre partie.

Par exemple, considérons un cas dans lequel un serveur redirecteur d'un domaine, chicago.com, s'approprie l'identité d'un serveur redirecteur d'un autre domaine, biloxi.com. Un agent utilisateur envoie une demande à biloxi.com, mais le serveur redirecteur de chicago.com répond avec une réponse falsifiée appropriée aux champs d'en-tête SIP pour une réponse de biloxi.com. Les adresses de contact falsifiées dans la réponse de redirection pourraient diriger l'agent utilisateur d'origine sur des ressources inappropriées ou non sûres, ou simplement empêcher des demandes pour biloxi.com de réussir.

Cette famille de menaces a une vaste descendance, dont beaucoup sont critiques. Au titre de la menace de capture d'enregistrement, considérons le cas dans lequel un enregistrement envoyé à biloxi.com est intercepté par chicago.com, qui répond à l'enregistrement intercepté par une réponse 301 (Déplacement Permanent) falsifiée. Cette réponse qui peut sembler venir de biloxi.com désigne chicago.com comme le registraire approprié. Toutes les demandes REGISTER futures de l'agent utilisateur d'origine iront désormais à chicago.com.

La prévention de cette menace exige des moyens pour que les agents d'utilisateur puissent authentifier les serveurs à qui ils envoient les demandes.

26.1.3 Altération des corps de message

Il est de fait que les agents d'utilisateur SIP acheminent des demandes à travers des serveurs mandataires de confiance. Indépendamment de la façon dont cette confiance est établie (l'authentification des mandataires est exposée dans un autre paragraphe), un agent utilisateur peut faire confiance à un serveur mandataire pour acheminer une demande, mais pas pour inspecter ou éventuellement modifier les corps contenus dans cette demande.

Considérons un agent utilisateur qui utilise des corps de message SIP pour communiquer des clé de chiffrement de session pour une session de média. Bien qu'il fasse confiance au serveur mandataire du domaine qu'il contacte pour délivrer correctement la signalisation, il peut ne pas vouloir que les administrateurs de ce domaine soient capables de décrypter la session de média qui suit. C'est pire si le serveur mandataire étant activement malveillant, il peut modifier la clé de session, en agissant comme intermédiaire, ou peut-être en changeant les caractéristiques de sécurité demandées par l'agent utilisateur d'origine.

Cette famille de menaces ne s'applique pas seulement aux clés de session, mais à la plupart des formes concevables de contenu portées de bout en bout dans SIP. Cela peut inclure des corps MIME qui devraient être rendus à l'utilisateur, SDP, ou des signaux de téléphonie encapsulés, entre autres. Les attaquants peuvent tenter de modifier les corps SDP, par exemple, afin de pointer les flux de support RTP sur un dispositif enregistreur afin d'espionner des communications vocales ultérieures.

Noter aussi que certains champs d'en-tête dans SIP sont signifiants de bout en bout, par exemple, Subject. Les agents d'utilisateur veulent protéger ces champs d'en-tête aussi bien sur les corps (un intermédiaire malveillant qui change le champ d'en-tête Subject peut faire apparaître une requête importante comme un pourriel, par exemple). Cependant, comme de nombreux champs d'en-tête sont légitimement inspectés ou altérés par les serveurs mandataires lorsqu'une demande est acheminée, tous les champs d'en-tête ne doivent pas être sécurisés de bout en bout.

Pour ces raisons, l'agent utilisateur peut vouloir sécuriser les corps de message SIP, et dans certains cas limités de champs d'en-tête, de bout en bout. Les services de sécurité nécessaires pour les corps de message incluent la confidentialité, l'intégrité, et l'authentification. Ces services de bout en bout devraient être indépendants des moyens

utilisés pour sécuriser les interactions avec les intermédiaires comme les serveurs mandataires.

26.1.4 Suppression de sessions

Une fois qu'un dialogue a été établi par l'échange de messages initial, des demandes ultérieures peuvent être envoyées pour modifier l'état du dialogue et/ou de la session. Il est critique que les éléments principaux dans une session puissent être certains que de telles demandes ne sont pas falsifiées par des attaquants.

Considérons un cas dans lequel un attaquant tiers capture quelques messages initiaux dans un dialogue partagé par deux parties afin d'en apprendre les paramètres de la session (l'étiquette To, l'étiquette From, et ainsi de suite) puis insère une demande BYE dans la session. L'attaquant peut choisir de falsifier la demande de façon qu'elle semble venir de l'un des participants. Une fois que le BYE est reçu par sa cible, la session va être interrompue prématurément.

Des menaces similaires de mi-session incluent la transmission de fausses ré-INVITE qui altèrent la session (éventuellement en réduisant la sécurité de la session ou en redirigeant des flux de support au titre d'une attaque par enregistrement).

La contre mesure la plus efficace à cette menace est l'authentification de l'envoyeur par le BYE. Dans ce cas, le receveur a seulement besoin de savoir que le BYE est venu de la partie avec laquelle a été établi le dialogue correspondant (par opposition à la certification de l'identité absolue de l'envoyeur). Aussi, si l'attaquant est incapable d'apprendre les paramètres de session du fait de la confidentialité, il ne sera pas possible de falsifier le BYE. Cependant, certains intermédiaires (comme les serveurs mandataires) auront besoin d'inspecter ces paramètres lors de l'établissement de la session.

26.1.5 Déni de service et amplification

Les attaques de déni de service visent à rendre indisponible un élément de réseau particulier, habituellement en dirigeant un montant excessif de trafic réseau sur ses interfaces. Une attaque distribuée de déni de service permet à un seul usager d'être la cause que plusieurs réseaux hôtes submergent un hôte cible avec une grande quantité de trafic réseau.

Dans de nombreuses architectures, les serveurs mandataires SIP font face à l'Internet public pour accepter des demandes venant de points de terminaison IP du monde entier. SIP crée un certain nombre d'opportunités potentielles pour des attaques distribuées de déni de service qui doivent être reconnues et contrées par les mises en œuvre et opérateurs de systèmes SIP.

Les attaquants peuvent créer des bogues dans des demandes, qui contienne une adresse IP de source falsifiée et un champ d'en-tête Via correspondant qui identifie un hôte ciblé comme l'origine de la demande et envoyer ensuite cette demande à un grand nombre d'éléments de réseau SIP, en utilisant de cette façon les pauvres agents d'utilisateur ou mandataires SIP pour générer du trafic de déni de service visant la cible.

De la même façon, les attaquants peuvent utiliser des valeurs de champ d'en-tête Route falsifiées dans une demande qui identifie l'hôte cible puis envoie de tels messages à des mandataires fourcheurs qui vont amplifier l'échange de messages envoyé à la cible.

Record-Route pourrait être utilisé pour un effet similaire lorsque l'attaquant est certain que le dialogue SIP initié par la demande débouchera dans de nombreuses transactions prenant leur origine dans la direction opposée.

Un certain nombre d'attaques de déni de service se manifestent si des demandes REGISTER ne sont pas correctement authentifiées et autorisées par les registraires. Les attaquants peuvent désenregistrer certains ou tous les usagers dans un domaine administratif, les empêchant par là d'être invités à de nouvelles sessions. Un attaquant pourrait aussi enregistrer un grand nombre de contacts désignant le même hôte pour une adresse-d'enregistrement donnée afin d'utiliser le registraire et tout serveur mandataire associé comme amplificateurs dans une attaque de déni de service. Les attaquants peuvent aussi essayer d'épuiser la mémoire disponible et les ressources de disque d'un registraire en enregistrant un nombre énorme de liens.

L'utilisation de la diffusion groupée pour transmettre les demandes SIP peut augmenter considérablement le potentiel d'attaques de déni de service.

Ces problèmes démontrent un besoin général de définition d'architectures qui minimisent le risque de déni de service, et la nécessité d'être soigneux dans les recommandations pour les mécanismes de sécurité de cette classe d'attaques.

26.2 Mécanismes de sécurité

D'après les menaces décrites ci-dessus, les services de sécurité fondamentaux nécessaires pour le protocole SIP sont : préserver la confidentialité et l'intégrité de l'échange de messages, empêcher les attaques en répétition ou le détournement de message, fournir les moyens d'authentification et de préservation de la confidentialité des participants à une session, et empêcher les attaques de déni de service. Au sein des messages SIP, les corps exigent de leur côté les services de sécurité de confidentialité, d'intégrité, et d'authentification.

Plutôt que de définir de nouveaux mécanismes de sécurité spécifiques de SIP, SIP réutilise chaque fois que possible les modèles de sécurité existants dérivés de l'espace HTTP et SMTP.

Le chiffrement complet des messages fournit le meilleur moyen de préserver la confidentialité de la signalisation – elle peut aussi garantir que les messages ne sont pas modifiés par des intermédiaires malveillants. Cependant, les demandes et réponses SIP ne peuvent pas être simplement chiffrées de bout en bout dans leur totalité parce que des champs de message comme URI-de-demande, Route, et Via doivent être visibles par les mandataires dans la plupart des architectures de réseau de sorte que les demandes SIP soient acheminées correctement. Noter que les serveurs mandataires ont aussi besoin de modifier certaines caractéristiques des messages (comme d'ajouter les valeurs de champ d'en-tête Via) afin que SIP fonctionne. Les serveurs mandataires doivent donc être de confiance, dans une certaine mesure, pour les agents d'utilisateur SIP. A cette fin, on recommande pour SIP les mécanismes de sécurité de couche inférieure qui chiffrent la totalité des demandes ou réponses SIP sur le fil saut par saut, et qui permettent aux points d'extrémité de vérifier l'identité des serveurs mandataires à qui ils envoient les demandes.

Les entités SIP ont aussi besoin de s'identifier les uns les autres d'une façon sécurisée. Lorsqu'un point d'extrémité SIP certifie l'identité de son utilisateur à un agent utilisateur homologue ou à un serveur mandataire, cette identité devrait être vérifiable d'une certaine façon. Un mécanisme d'authentification cryptographique est fourni dans SIP pour répondre à cette exigence.

Un mécanisme de sécurité indépendant pour les corps de message SIP apporte un moyen de remplacement pour l'authentification mutuelle de bout en bout, ainsi qu'une limite sur le degré de confiance que les agents utilisateurs doivent prêter aux intermédiaires.

26.2.1 Sécurité des couches Transport et Réseau

La sécurité de couche Transport ou Réseau chiffre le trafic de signalisation, garantissant la confidentialité et l'intégrité du message.

Souvent, les certificats sont utilisés pour l'établissement de la sécurité des couches inférieures, et ces certificats peuvent aussi être utilisés pour fournir un moyen d'authentification dans de nombreuses architectures.

Deux solutions courantes pour fournir la sécurité au couches Transport et Réseau sont, respectivement, TLS [25] et IPSec [26].

IPSec est un ensemble d'outils de protocole de couche Réseau qui peut être utilisé collectivement comme un moyen de remplacement sécurisé pour l'IP traditionnel (Protocole Internet). IPSec est plus communément utilisé dans les architectures dans lesquelles un ensemble d'hôtes ou domaines administratifs ont une relation de confiance existante avec une autre. IPSec est habituellement mis en œuvre au niveau du système d'exploitation dans un hôte, ou sur une passerelle de sécurité qui fournit la confidentialité et l'intégrité pour tous trafics reçus d'une interface particulière (comme dans une architecture de VPN). IPSec peut aussi être utilisé saut par saut.

Dans de nombreuses architectures, IPSec n'exige pas l'intégration avec des applications SIP ; IPSec est peut-être mieux adapté aux développements dans lesquels l'ajout de la sécurité directement aux hôtes SIP serait ardu. Les agents d'utilisateur qui ont une relation de clé pré-partagée avec leur serveur mandataire de premier saut sont aussi de bons candidats à l'utilisation d'IPSec. Tout développement d'IPSec pour SIP nécessite un profil IPSec qui décrive les outils de protocole qui seront requis pour la sécurisation de SIP. Un tel profil n'est pas donné dans le présent document.

TLS fournit la sécurité de couche transport sur les protocoles orientés connexion (pour les besoins du présent document, TCP); "tls" (signifiant TLS sur TCP) peut être spécifié comme le protocole de transport désiré au sein d'une valeur de champ d'en-tête Via ou d'un URI SIP. TLS est bien adapté aux architectures dans lesquelles la sécurité saut par saut est exigée entre hôtes sans association de confiance pré-existante. Par exemple, Alice a confiance dans son mandataire de serveur local qui, après un échange de certificats, décide de faire confiance au serveur mandataire local de Bob, auquel Bob fait confiance, et donc Bob et Alice peuvent communiquer en toute sécurité.

TLS doit être étroitement couplé avec une application SIP. Noter que les mécanismes de transport sont spécifiés saut par saut dans SIP, et donc un agent utilisateur qui envoie des demandes TLS à un serveur mandataire n'a pas l'assurance que TLS va être utilisé de bout en bout.

La suite de chiffrement TLS_RSA_WITH_AES_128_CBC_SHA [6] DOIT être prise en charge au minimum par les mises en œuvre lorsque TLS est utilisé dans une application SIP. Pour les besoins de la rétro-compatibilité, les serveurs mandataires, les serveurs redirecteurs, et les registraires DEVRAIENT prendre en charge TLS_RSA_WITH_3DES_EDE_CBC_SHA. Les mises en œuvre PEUVENT aussi accepter d'autres suites de chiffrement.

26.2.2 Schéma d'URI SIPS

Le schéma d'URI SIPS adhère à la syntaxe de l'URI SIP (décrite en 19), bien que la chaîne du schéma soit "sips" plutôt que "sip". La sémantique de SIPS est cependant très différente de celle de l'URI SIP. SIPS permet aux ressources de spécifier qu'elle devraient être jointes en toute sécurité.

Un URI SIPS peut être utilisé comme adresse-d'enregistrement pour un utilisateur particulier - l'URI par lequel l'utilisateur est connu canoniquement (sur sa carte de visite, dans le champ d'en-tête From de ses demandes, dans le champ d'en-tête To des demandes REGISTER). Lorsqu'il est utilisé comme l'URI-de-demande d'une demande, le schéma SIPS signifie que chaque saut sur lequel la demande est transmise, jusqu'à ce que la demande atteigne l'entité SIP responsable pour la portion de domaine de l'URI-de-demande, doit être sécurisé avec TLS ; une fois qu'il atteint le domaine en question, il est traité conformément à la politique locale de sécurité et d'acheminement, éventuellement en utilisant TLS pour le dernier saut vers un UAS. Lorsqu'il est utilisé par l'origine d'une demande (comme ce serait le cas si elle utilisait un URI SIPS comme adresse-d'enregistrement de la cible), SIPS impose que tout le chemin de la demande vers le domaine cible soit sécurisé.

Le schéma SIPS est applicable à de nombreuses autres occasions dans lesquelles les URI SIP sont utilisés aujourd'hui dans SIP en plus de l'URI-de-demande, y compris dans les adresses-d'enregistrement, les adresses de contact (le contenu des en-têtes Contact, y compris celui des méthodes REGISTER), et les en-têtes Route. Dans chaque instance, le schéma d'URI SIPS permet à ces champs existants de désigner des ressources sécurisées. La façon dont un URI SIPS est déréférencé dans l'un de ces contextes a ses propres propriétés de sécurité qui sont précisées dans [4].

L'utilisation de SIPS entraîne en particulier que l'authentification mutuelle TLS DEVRAIT être employée, comme DEVRAIT l'être la suite de chiffrement TLS_RSA_WITH_AES_128_CBC_SHA. Les certificats reçus dans le processus d'authentification DEVRAIENT être validés avec des certificats racine détenus par le client ; l'échec de validation d'un certificat DEVRAIT entraîner l'échec de la demande.

Noter que dans le schéma d'URI SIPS, le transport est indépendant de TLS, et donc "sips:alice@atlanta.com;transport=tcp" et "sips:alice@atlanta.com;transport=sctp" sont tous deux valides (noter cependant que UDP est n'est pas un transport valide pour SIPS). L'utilisation de "transport=tls" a par conséquent été déconseillée, en partie parce qu'il est spécifique d'une demande à un seul saut. Ceci est un changement par rapport à la RFC 2543.

Les usagers qui distribuent un URI SIPS comme adresse-d'enregistrement peuvent choisir de faire fonctionner des appareils qui refusent des demandes sur des transports non sécurisés.

26.2.3 Authentification HTTP

SIP fournit une capacité de confrontation, sur la base de l'authentification HTTP, qui s'appuie sur les codes de réponse 401 et 407 ainsi que sur les champs d'en-tête pour transporter la confrontation et l'accréditif. Sans modification majeure, la réutilisation du schéma d'authentification HTTP Digest dans SIP permet une protection contre la répétition et l'authentification unilatérale.

L'utilisation de l'authentification par Digest dans SIP est précisée à la Section 22.

26.2.4 S/MIME

Comme exposé ci-dessus, le chiffrement de tout le message SIP de bout en bout pour les besoins de la confidentialité n'est pas approprié à cause des intermédiaires du réseau (comme les serveurs mandataires) qui ont besoin de voir certains champs d'en-tête afin d'acheminer correctement les messages, et si ces intermédiaires sont exclus des associations de sécurité, les messages SIP ne pourront pas être acheminables.

Cependant, S/MIME permet aux agents d'utilisateur SIP de chiffrer les corps MIME au sein de SIP, en sécurisant ces corps de bout en bout sans affecter les en-têtes de message. S/MIME peut fournir une confidentialité et intégrité de bout en bout pour les corps de message, ainsi que l'authentification mutuelle. Il est aussi possible d'utiliser S/MIME pour fournir une forme d'intégrité et de confidentialité pour les champs d'en-tête SIP à travers le tunnelage de message SIP.

L'usage de S/MIME dans SIP est détaillé à la Section 23.

26.3 Mise en œuvre des mécanismes de sécurité

26.3.1 Prescriptions pour les mises en œuvre de SIP

Les serveurs mandataires, les serveurs redirecteurs, et les registraires DOIVENT mettre en œuvre TLS, et DOIVENT prendre en charge l'authentification à la fois mutuelle et unilatérale. Il est fortement RECOMMANDÉ que les agents d'utilisateur soient capables d'initier TLS ; les agents d'utilisateur PEUVENT aussi être capables d'agir comme un serveur TLS. Les serveurs mandataires, les serveurs redirecteurs, et les registraires DEVRAIENT posséder un certificat de site dont le sujet corresponde à leur nom d'hôte canonique. Les agents d'utilisateur PEUVENT avoir des certificats en propre pour l'authentification mutuelle avec TLS, mais aucune disposition n'est prise dans le présent document pour leur utilisation. Tous les éléments SIP qui prennent en charge TLS DOIVENT avoir un mécanisme pour valider les certificats reçus durant les négociations TLS ; cela entraîne la possession d'un ou plusieurs certificats racine produits par l'autorité de certification (de préférence des distributeurs bien connus de certificats de site comparables à ceux qui délivrent des certificats racine pour les navigateurs de la toile).

Tous les éléments SIP qui prennent en charge TLS DOIVENT aussi accepter le schéma d'URI SIPS.

Les serveurs mandataires, les serveurs redirecteurs, les registraires, et les agents d'utilisateur PEUVENT aussi mettre en œuvre IPSec ou d'autres protocoles de sécurité de couche inférieure.

Lorsqu'un agent utilisateur tente de contacter un serveur mandataire, un serveur redirecteur, ou un registraire, l'UAC DEVRAIT initier une connexion TLS sur laquelle il enverra les messages SIP. Dans certaines architectures, les UAS PEUVENT aussi recevoir des demandes sur de telles connexions TLS.

Les serveurs mandataires, les serveurs redirecteurs, les registraires, et les agents d'utilisateur DOIVENT mettre en œuvre Digest Authorization, avec tous les aspects exigés à la Section 22. Les serveurs mandataires, les serveurs redirecteurs et les registraires DEVRAIENT être configurés avec au moins un domaine Digest, et au moins une chaîne "realm" acceptée par un serveur donné DEVRAIT correspondre au nom d'hôte ou nom de domaine du serveur.

Les agents d'utilisateur PEUVENT prendre en charge la signature et le chiffrement des corps MIME, et le transfert des accréditif avec S/MIME comme indiqué à la Section 23. Si un agent utilisateur détient un ou plusieurs certificats racine d'autorités de certification afin de valider les certificats pour TLS ou IPSec, il DEVRAIT être capable de les réutiliser pour vérifier les certificats S/MIME, en tant que de besoin. Un agent utilisateur PEUT détenir des certificats racine spécialement pour valider les certificats S/MIME.

Noter qu'il est prévu que les futures extensions de sécurité pourront mettre à niveau la force normative associée aux mises en œuvre S/MIME quand S/MIME apparaîtra et que la dimension du problème sera mieux comprise.

26.3.2 Solutions pour la sécurité

Le fonctionnement concerté de ces mécanismes de sécurité peut suivre les modèles existant de la sécurité de la toile et de la messagerie électronique dans une certaine mesure. A haut niveau, les agents d'utilisateur s'authentifient eux-mêmes aux serveurs (serveurs mandataires, serveurs redirecteurs, et registraires) avec un nom d'utilisateur et mot de passe Digest ; les serveurs s'authentifient eux-mêmes aux agents d'utilisateur un saut plus loin, ou à un autre serveur un

saut plus loin (et vice versa), avec un certificat de site délivré par TLS.

Au niveau d'homologue à homologue, les agents d'utilisateur font ordinairement confiance au réseau pour s'authentifier les uns les autres ; cependant, S/MIME peut aussi être utilisé pour fournir l'authentification directe lorsque le réseau ne le fait pas, ou si le réseau lui-même n'est pas de confiance.

Ce qui suit est une illustration exemplaire dans laquelle ces mécanismes de sécurité sont utilisés par divers agents d'utilisateur et serveurs pour prévenir les types de menaces décrites au paragraphe 26.1. Alors que les mises en œuvre et les administrateurs de réseau PEUVENT suivre les lignes directrices normatives données dans le reste de la présente section, ceci n'est fourni qu'à titre d'exemple de mise en œuvre.

26.3.2.1 Enregistrement

Lorsqu'un agent utilisateur vient en ligne et s'enregistre avec son domaine administratif local, il DEVRAIT établir une connexion TLS avec son registraire (La Section 10 décrit comment l'agent utilisateur joint son registraire). Le registraire DEVRAIT offrir un certificat à l'agent utilisateur, et le site identifié par le certificat DOIT correspondre au domaine dans lequel l'agent utilisateur à l'intention de s'enregistrer ; par exemple, si l'agent utilisateur à l'intention de s'enregistrer à l'adresse-d'enregistrement 'alice@atlanta.com', le certificat de site doit identifier un hôte au sein du domaine atlanta.com (comme sip.atlanta.com). Lorsqu'il reçoit le message du certificat TLS, l'agent utilisateur DEVRAIT vérifier le certificat et inspecter le site identifié par le certificat. Si le certificat est invalide, révoqué, ou si il n'identifie pas la partie appropriée, l'agent utilisateur NE DOIT PAS envoyer le message REGISTER et poursuivre l'enregistrement.

Lorsqu'un certificat valide a été fourni par le registraire, l'agent utilisateur sait que le registraire n'est pas un agresseur qui peut rediriger l'agent utilisateur, voler son mot de passe, ou tenter aucune attaque de ce genre.

L'agent utilisateur crée alors une demande REGISTER qui DEVRAIT être adressée à un URI-de-demande correspondant au certificat de site reçu du registraire. Lorsque l'agent utilisateur envoie la demande REGISTER sur la connexion TLS existante, le registraire DEVRAIT mettre en question la demande avec une réponse 401 (Authentification du mandataire exigée). Le paramètre "realm" au sein du champ d'en-tête Proxy-Authenticate de la réponse DEVRAIT correspondre au domaine précédemment donné par le certificat de site. Lorsque l'UAC reçoit la mise en question, il DEVRAIT demander un accréditif à l'utilisateur ou prendre un accréditif approprié dans un porte clés correspondant au paramètre "realm" de la mise en question. Le nom d'utilisateur de cet accréditif DEVRAIT correspondre avec la portion "userinfo" de l'URI dans le champ d'en-tête To de la demande REGISTER. Une fois que l'accréditif Digest a été inséré dans un champ d'en-tête Proxy-Authorization approprié, le REGISTER devrait être resoumis au registraire.

Comme le registraire exige de l'agent utilisateur qu'il s'authentifie lui-même, il serait difficile à un agresseur de falsifier des demandes REGISTER pour l'adresse-d'enregistrement de l'utilisateur. Noter aussi que comme le REGISTER est envoyé sur une connexion TLS confidentielle, l'attaquant ne sera pas capable d'intercepter le REGISTER pour enregistrer l'accréditif en vue d'une possible attaque en répétition.

Une fois que l'enregistrement a été accepté par le registraire, l'agent utilisateur DEVRAIT quitter cette connexion TLS ouverte pourvu que le registraire agisse aussi comme serveur mandataire auquel les demandes sont envoyées pour les usagers dans ce domaine administratif. La connexion TLS existante sera réutilisée pour livrer les demandes entrantes à l'agent utilisateur qui vient de terminer l'enregistrement.

Parce que l'agent utilisateur s'est déjà authentifié au serveur de l'autre côté de la connexion TLS, toutes les demandes qui viennent sur cette connexion sont connues pour être passées par le serveur mandataire – les agresseurs ne peuvent pas créer des demandes falsifiées qui apparaîtraient comme étant envoyées par ce serveur mandataire.

26.3.2.2 Demandes interdomaine

Disons maintenant que l'agent utilisateur d'Alice voudrait initialiser une session avec un usager dans un domaine administratif distant, à savoir "bob@biloxi.com". Nous dirons aussi que le domaine administratif local (atlanta.com) a un mandataire local externe.

Le serveur mandataire qui traite les demandes externes pour un domaine administratif PEUT aussi agir comme un mandataire local externe ; pour simplifier, nous supposerons que c'est le cas pour atlanta.com (autrement, l'agent utilisateur initialiserait une nouvelle connexion TLS avec un serveur distinct à ce point). En supposant que le client a achevé le processus d'enregistrement décrit au paragraphe précédant, il DEVRAIT réutiliser la connexion TLS avec le

serveur mandataire local lorsqu'il envoie une demande INVITE à un autre usager. L'agent utilisateur DEVRAIT réutiliser l'accréditif mis en mémoire dans l'INVITE pour éviter de solliciter inutilement l'utilisateur.

Lorsque le serveur mandataire local externe a validé l'accréditif présenté par l'agent utilisateur dans l'INVITE, il DEVRAIT inspecter l'URI-de-demande pour déterminer comment le message devrait être acheminé (voir en [4]). Si la portion "domainname" de l'URI-de-demande correspond au domaine local (atlanta.com) plutôt qu'à biloxi.com, le serveur mandataire consultera son service de localisation pour déterminer comment le mieux joindre l'usager demandé.

Si "alice@atlanta.com" avait tenté de contacter, disons, "alex@atlanta.com", le mandataire local aurait passé la demande sur la connexion TLS qu'Alex a établie avec le registraire lors de son inscription. Comme Alex recevrait cette demande sur son canal authentifié, il serait sûr que la demande d'Alice a été autorisée par le serveur mandataire du domaine administratif local.

Cependant, dans cette instance l'URI-de-demande désigne un domaine distant. Le serveur mandataire local externe à atlanta.com DEVRAIT donc établir une connexion TLS avec le serveur mandataire distant à biloxi.com. Comme les deux participants à cette connexion TLS sont des serveurs qui possèdent des certificats de site, c'est l'authentification TLS mutuelle qui DEVRAIT survenir. Chaque côté de la connexion DEVRAIT vérifier et inspecter le certificat de l'autre, en notant le nom de domaine qui apparaît dans le certificat pour le comparer avec les champs d'en-tête des messages SIP. Le serveur mandataire atlanta.com, par exemple, DEVRAIT vérifier à ce stade que le certificat reçu du côté distant correspond au domaine biloxi.com. Une fois que c'est fait, et que la négociation TLS est achevée, il en résulte un canal sécurisé entre les deux mandataires, et le mandataire atlanta.com peut transmettre la demande INVITE à biloxi.com.

Le serveur mandataire à biloxi.com DEVRAIT inspecter le certificat du serveur mandataire à atlanta.com à son tour et comparer le domaine allégué par le certificat avec la portion "domainname" du champ d'en-tête From dans la demande INVITE. Le mandataire biloxi PEUT avoir une politique de sécurité stricte qui exige qu'il rejette les demandes qui ne correspondent pas au domaine administratif d'où elles ont été mandatées.

De telles politiques de sécurité pourraient être instituées pour empêcher l'équivalent SIP de 'open relays' de SMTP qui sont fréquemment exploités pour générer des pourriels.

Cette politique, cependant, ne garantit que le fait que la demande est venue du domaine qui est écrit ; elle ne permet pas à biloxi.com de vérifier comment atlanta.com a authentifié Alice. C'est seulement si biloxi.com a d'autres moyens de connaître la politique d'authentification de atlanta.com qu'il aura la possibilité de s'assurer qu'Alice a prouvé son identité. biloxi.com peut alors instituer une politique encore plus stricte qui interdit des demandes venant de domaines qui ne sont pas connus administrativement comme partageant une politique commune d'authentification avec biloxi.com.

Une fois que l'INVITE a été approuvé par le mandataire biloxi, le serveur mandataire DEVRAIT identifier le canal TLS existant, s'il en est un, associé à l'utilisateur ciblé par cette demande (dans ce cas "bob@biloxi.com"). L'INVITE devrait être mandaté à Bob à travers ce canal. Comme la demande est reçue sur une connexion TLS qui avait été authentifiée précédemment comme le mandataire biloxi, Bob sait que le champ d'en-tête From n'a pas été touché et que atlanta.com a validé Alice, bien qu'il n'ait pas de certitude sur l'identité d'Alice.

Avant de transmettre la demande, les deux serveurs mandataires DEVRAIENT ajouter un champ d'en-tête Record-Route à la demande de façon que toutes les demandes futures de ce dialogue passent à travers les serveurs mandataires. Les serveurs mandataires peuvent ainsi continuer de fournir des services de sécurité pour la durée de vie de ce dialogue. Si les serveurs mandataires ne s'ajoutent pas eux-mêmes à Record-Route, les futurs messages passeront directement de bout en bout entre Alice et Bob sans aucun service de sécurité (à moins que les deux parties s'accordent sur une sécurité indépendante de bout en bout comme S/MIME). A cet égard ; le modèle trapézoïde de SIP peut fournir une bonne structure où les conventions d'accord entre sites mandataires établissent un canal raisonnablement sécurisé entre Alice et Bob.

Une attaque contre cette architecture serait, par exemple, incapable de falsifier une demande BYE et de l'insérer dans le flux de signalisation entre Bob et Alice parce que l'attaquant n'a aucun moyen d'attester les paramètres de la session et aussi parce que le mécanisme d'intégrité protège transitivement le trafic entre Alice et Bob.

26.3.2.3 Demandes d'homologue à homologue

Considérons maintenant un agent utilisateur attestant l'identité "carol@chicago.com" qui n'a pas de mandataire local externe. Lorsque Carol souhaite envoyer un INVITE à "bob@biloxi.com", son agent utilisateur DEVRAIT initier

directement une connexion TLS avec le mandataire biloxi (en utilisant le mécanisme décrit dans [4] pour déterminer comment le mieux joindre l'URI-de-demande donné). Lorsque son agent utilisateur reçoit un certificat du mandataire biloxi, il DEVRAIT être vérifié normalement avant qu'elle ne passe son INVITE à travers la connexion TLS. Cependant, Carol n'a pas de moyens de prouver son identité au mandataire biloxi, mais elle a une signature CMS-detached sur un corps "message/sip" dans l'INVITE. Il est invraisemblable que dans cette instance, Carol ait un accréditif dans le domaine biloxi.com, car elle n'a pas d'association formelle avec biloxi.com. Le mandataire biloxi PEUT aussi avoir une politique stricte qui l'empêche même de se soucier de mettre en question des demandes qui n'ont pas biloxi.com dans la portion "domainname" du champ d'en-tête From - il traite ces usagers comme non authentifiés.

Le mandataire biloxi a une politique pour Bob qui veut que toutes les demandes non authentifiées devraient être redirigées sur l'adresse de contact appropriée enregistrée pour 'bob@biloxi.com', à savoir <sip:bob@192.0.2.4>. Carol reçoit la réponse de redirection sur la connexion TLS qu'elle a établie avec le mandataire biloxi, de sorte qu'elle a confiance en la véracité de l'adresse de contact.

Carol DEVRAIT alors établir une connexion TCP avec l'adresse désignée et envoyer un nouvel INVITE avec un URI-de-demande contenant l'adresse de contact reçue (recalculant la signature dans le corps lorsque la demande est prête). Bob reçoit cet INVITE sur une interface non sécurisée, mais son agent utilisateur inspecte et, dans ce cas, reconnaît le champ d'en-tête From de la demande et ensuite voit qu'un certificat mémorisé localement correspond à celui présenté dans la signature du corps de l'INVITE. Il répond dans un mode similaire, s'authentifiant auprès de Carol, et un dialogue sécurisé commence.

Parfois des pare-feu ou des traducteurs NAT peuvent, dans un domaine administratif, empêcher l'établissement d'une connexion TCP directe avec un agent utilisateur. Dans ces cas, les serveurs mandataires ont aussi la possibilité de relayer des demandes aux agents d'utilisateur d'une façon qui n'a pas d'implication sur la confiance (par exemple, en se passant de la connexion TLS existante et en transmettant la demande en TCP clair) comme le dicte la politique locale.

26.3.2.4 Protection contre le déni de service

Afin de minimiser le risque d'une attaque de déni de service contre les architectures en utilisant ces solutions de sécurité, les mises en œuvre devraient prendre note des lignes directrices suivantes.

Lorsque l'hôte sur lequel fonctionne un serveur mandataire SIP est routable depuis l'Internet public, il DEVRAIT être déployé dans un domaine administratif avec des politiques opérationnelles défensives (blocage du trafic acheminé depuis la source, préférence au filtrage du trafic ping). TLS et IPSec peuvent aussi faire tous deux usage d'hôtes fortifiés au bords des domaines administratifs qui participent aux associations de sécurité pour agréger des tunnels et prises sécurisés. Ces hôtes fortifiés peuvent aussi subir le choc d'attaques de déni de service, en s'assurant que les hôtes SIP au sein du domaine administratif ne sont pas encombrés par des échanges de messages superflus.

Quelles que soient les solutions de sécurité déployées, les flots de messages dirigés sur les serveurs mandataires peuvent verrouiller les ressources du serveur mandataire et empêcher le trafic désirable d'atteindre sa destination. Il y a une dépense de calcul associée au traitement d'une transaction SIP à un serveur mandataire, et cette dépense est plus grande pour les serveurs mandataires à états pleins que pour les serveurs mandataires sans état. Donc, les mandataires à états pleins sont plus susceptibles d'être inondés que les serveurs mandataires sans état.

Les agents d'utilisateur et serveurs mandataires DEVRAIENT mettre en question les demandes avec une seule 401 (Non autorisé) ou 407 (Authentification du mandataire exigée), laissant de côté l'algorithme normal de retransmission de réponse, et donc se comportant sans état à l'égard des demandes non authentifiées.

La retransmission de la réponse d'état 401 (Non autorisé) ou 407 (Authentification du mandataire exigée) amplifie le problème si un attaquant utilise une valeur de champ d'en-tête falsifiée (comme Via) pour diriger du trafic sur un tiers.

En résumé, l'authentification mutuelle des serveurs mandataires par un mécanisme tel que TLS réduit de façon significative le potentiel que des intermédiaires illégaux introduisent des demandes ou réponses falsifiées qui puissent empêcher le service. Cela rend la tâche commensurablement plus difficile pour les attaquants de faire d'innocents nœuds SIP des agents d'amplification.

26.4 Limitations

Bien que ces mécanismes de sécurité, lorsqu'ils sont appliqués d'une façon judicieuse, puissent déjouer bien des menaces, il y a des limites à la portée des mécanismes qui peuvent être compris par les mises en œuvre et les opérateurs de réseaux.

26.4.1 Résumé HTTP

Une des principales limitations de l'utilisation de HTTP Digest dans SIP est que les mécanismes d'intégrité dans Digest ne fonctionnent pas très bien pour SIP. Précisément, ils offrent une protection de l'URI-de-demande et de la méthode d'un message, mais pas pour tous les champs d'en-tête que les agents d'utilisateur voudrait le plus vraisemblablement voir sécuriser.

Le mécanisme existant de protection contre la répétition décrit dans la RFC 2617 a aussi certaines limitations pour SIP. Le mécanisme next-nonce *(prochain nom occasionnel)*, par exemple, ne prend pas en charge les demandes tunnelées. Le mécanisme nonce-count *(compte de noms occasionnels)* devrait être utilisé pour la protection la répétition.

Une autre limitation de HTTP Digest est la portée des domaines. Digest a une valeur lorsqu'un usager veut s'authentifier auprès d'une ressource avec laquelle il a une association préexistante, comme un fournisseur de service dont l'utilisateur est un abonné (ce qui est un scénario assez courant et donc Digest fournit une fonction extrêmement utile). Par contraste, la portée de TLS est interdomaine ou multidomaine, car les certificats sont souvent vérifiables globalement, de sorte que l'agent utilisateur peut authentifier le serveur sans association préexistante.

26.4.2 S/MIME

Le plus gros défaut du mécanisme S/MIME est l'absence d'une infrastructure prévalente de clés publiques pour les utilisateurs finals. Si les certificats auto-signés (ou les certificats qui ne peuvent être vérifiés par un des participants à un dialogue) sont utilisés, le mécanisme d'échange de clé fondé sur SIP décrit au paragraphe 23.2 est susceptible d'une attaque d'intrusion avec laquelle un attaquant peut inspecter et modifier les corps S/MIME. L'attaquant intercepte le premier échange de clés entre les deux parties d'un dialogue, retire les signatures CMS-detached existantes de la demande et de la réponse, et insère une signature CMS-detached différente contenant un certificat fourni par l'attaquant (mais qui semble être un certificat pour la bonne adresse-d'enregistrement). Chaque partie pensera avoir échangé les clés avec l'autre, alors qu'en fait chacun a la clé publique de l'attaquant.

Il est important de noter que l'attaquant peut seulement faire levier sur cette vulnérabilité lors du premier échange de clés entre deux parties – lors des occasions suivantes, l'altération de la clé serait remarquée par les agents d'utilisateur. Il serait aussi difficile pour l'attaquant de rester dans le chemin de tous les dialogues futurs entre les deux parties (au fil des jours, des semaines ou des années).

SSH est susceptible de la même attaque par intrusion sur le premier échange de clés ; cependant, il est largement connu qu'alors que SSH n'est pas parfait, il améliore la sécurité des connexions. L'utilisation de clés d'identification pourrait aider dans SIP, exactement comme pour SSH. Par exemple, si deux parties utilisent SIP pour établir une session de communications vocale, chacun peut lire l'identification de la clé qu'il a reçue de l'autre, qui peut être comparée à l'original. Il serait certainement plus difficile pour l'intrus d'émuler les voix des participants que leur signalisation (une pratique qui était utilisée avec le téléphone Clipper sécurisé sur la base d'une puce).

Le mécanisme S/MIME permet aux agents d'utilisateur d'envoyer des demandes chiffrées sans préambule si elles possèdent un certificat pour l'adresse-d'enregistrement de destination sur leur porte-clés. Cependant, il est possible que tout appareil particulier enregistré pour une adresse-d'enregistrement ne détienne pas de certificat précédemment employé par l'utilisateur actuel de l'appareil, et il sera donc incapable de traiter correctement une demande chiffrée, ce qui peut conduire à des erreurs de signalisation évitables. Ceci est particulièrement vraisemblable lorsqu'une demande chiffré est fourchue.

Les clés associées à S/MIME sont très utiles associées à un utilisateur particulier (une adresse-d'enregistrement) plutôt qu'à un appareil (un agent utilisateur). Lorsque des usagers se déplacent d'un appareil à l'autre, il peut être difficile de transporter des clés privées en toute sécurité entre les agents d'utilisateur ; la façon dont de telles clés peuvent être acquises par un appareil est en dehors du domaine d'application du présent document.

Une autre difficulté plus prosaïque du mécanisme S/MIME est qu'il peut avoir pour résultat de très longs messages, particulièrement lorsque le mécanisme de tunnelage SIP, décrit au paragraphe 23.4, est utilisé. Pour cette raison, il est RECOMMANDÉ que TCP soit utilisé comme un protocole de transport lorsque le tunnelage S/MIME est employé.

26.4.3 TLS

Le souci le plus fréquemment exprimé à propos de TLS est qu'il ne peut pas fonctionner sur UDP ; TLS exige un protocole de transport sous-jacent orienté connexion, ce qui pour les besoins de ce document signifie TCP.

Il peut aussi être ardu pour un serveur mandataire local externe et/ou registraire de maintenir de nombreuses connexions TLS de longue durée simultanées avec de nombreux agents d'utilisateur. Cela introduit de vrais soucis d'échelle, particulièrement pour les suites de chiffrement intensives. Maintenir la redondance de connexions TLS de longue durée, particulièrement lorsqu'un agent utilisateur est seul responsable de leur établissement, pourrait aussi être lourd.

TLS permet seulement aux entités SIP d'authentifier les serveurs dont elles sont adjacentes ; TLS offre une sécurité strictement saut par saut. Ni TLS, ni aucun autre mécanisme spécifié dans la présent document, ne permet aux clients d'authentifier les serveurs mandataires avec lesquels ils ne peuvent former une connexion TCP directe.

26.4.4 Les URI de SIPS

Utiliser réellement TLS sur chaque segment d'un chemin de demande implique que l'UAS de terminaison doit être joignable sur TLS (peut-être en s'enregistrant avec un URI SIPS comme adresse de contact). C'est l'utilisation préférée de SIPS. De nombreuses architectures valides utilisent cependant TLS pour sécuriser une partie du chemin de demande, mais s'appuient sur quelque autre mécanisme pour le saut final vers un UAS, par exemple. Et donc SIPS ne peut pas garantir que l'utilisation de TLS sera vraiment de bout en bout. Noter que comme de nombreux agents d'utilisateur n'accepteront pas les connexions TLS entrantes, même ceux des agents d'utilisateur qui prennent en charge TLS peuvent être obligés de maintenir des connexions TLS persistantes, comme décrit dans le paragraphe sur les limites de TLS ci-dessus, afin de recevoir des demandes sur TLS comme un UAS.

Les services de localisation ne sont pas obligés de fournir un lien SIPS pour un URI-de-demande SIPS. Bien que les services de localisation soient normalement remplis par les enregistrements d'usagers (comme indiqué au paragraphe 10.2.1), divers autres protocoles et interfaces pourraient fournir les adresses de contact pour un AOR, et ces outils ont la liberté de transposer les URI SIPS en URI SIP en tant que de besoin. Lorsqu'il est interrogé sur des liens, un service de localisation retourne ses adresses de contact sans considération pour le fait qu'il ait ou non reçu une demande avec un URI-de-demande SIPS. Si un serveur redirecteur accède au service de localisation, il appartient à l'entité qui traite le champ d'en-tête Contact d'une redirection de déterminer les propriétés des adresses de contact.

S'assurer que TLS sera utilisé pour tous les segments de la demande jusqu'au domaine cible est assez complexe. Il est possible que les serveurs mandataires authentifiés cryptographiquement le long du chemin qui ne sont pas conformes ou sont compromis puissent choisir de ne pas suivre les règles de transmission associées à SIPS (et les règles générales de transmission du paragraphe 16.6). De tels intermédiaires malveillants pourraient, par exemple, recibler une demande venant d'un URI SIPS sur un URI SIP dans une tentative de dégradation de la sécurité.

Autrement, un intermédiaire pourrait légitimement recibler une demande d'un URI SIP sur un URI SIPS. Les receveurs d'une demande dont l'URI-de-demande utilise le schéma d'URI SIPS ne peuvent donc supposer sur la seule base de l'URI-de-demande que SIPS était utilisé pour la totalité du chemin de la demande (à partir du client).

Pour répondre à ces préoccupations, il est RECOMMANDÉ aux receveurs d'une demande dont l'URI-de-demande contient un URI SIP ou SIPS d'inspecter la valeur du champ d'en-tête To pour voir si il contient un URI SIPS (noter cependant que le fait que cet URI ait le même schéma mais qu'il ne soit pas équivalent à l'URI dans le champ d'en-tête To ne constitue pas une atteinte à la sécurité). Bien que les clients puissent choisir de remplir différemment l'URI-de-demande et le champ d'en-tête To d'une demande, lorsque SIPS est utilisé, cette disparité pourrait être interprétée comme une possible violation de la sécurité, et la demande pourrait par conséquent être rejetée par son receveur. Les receveurs PEUVENT aussi inspecter la chaîne d'en-tête Via afin de vérifier une seconde fois si TLS a été utilisé ou non pour la totalité du chemin de demande jusqu'à atteindre le domaine administratif local. S/MIME peut aussi être utilisé par l'UAC d'origine pour aider à s'assurer que la forme originale du champ d'en-tête To est portée de bout en bout.

Si l'UAS a des raisons de croire que le schéma de l'URI-de-demande a été modifié à tort dans le transit, l'agent utilisateur DEVRAIT notifier à son usager une potentielle atteinte à la sécurité.

Comme mesure supplémentaire pour empêcher les attaques en dégradation, les entités qui n'acceptent que des demandes SIPS PEUVENT aussi refuser les connexions sur les accès non sécurisés.

L'utilisateur final va indubitablement discerner la différence entre les URI SIPS et SIP, et il peut les éditer manuellement dans une réponse à des stimuli. Ceci peut être bénéfique à la sécurité ou la dégrader. Par exemple, si un attaquant corrompt une mémoire cache DNS en insérant une ensemble d'enregistrements falsifiés qui efface effectivement tous les enregistrements SIPS pour un serveur mandataire, toute demande SIPS qui traverse alors ce serveur mandataire peut échouer. Lorsque cependant un utilisateur voit que des appels répétés à un AOR SIPS échouent, il peut sur certains appareils convertir manuellement le schéma de SIPS en SIP et réessayer. Bien sûr, il y a quelques sauvegardes contre cela (si l'agent utilisateur de destination est vraiment paranoïaque, il peut refuser toutes les

demandes non-SIPS), mais c'est une limitation qui vaut mieux que rien du tout. D'un autre côté, les usagers pourraient aussi deviner que 'SIPS' serait valide même lorsqu'il est seulement présenté avec un URI SIP.

26.5 Confidentialité

Les messages SIP contiennent fréquemment des information sensibles sur leurs expéditeurs – pas seulement ce qu'ils ont à dire , mais avec qui ils communiquent, quand ils communiquent et pour combien de temps, et d'où ils participent aux sessions. De nombreuses applications et leurs utilisateurs exigent que ces sortes d'informations privées soient cachées à toute partie qui n'a pas besoin de les connaître.

Noter qu'il y a aussi moins de façons directes dans lesquelles des informations privées peuvent être divulguées. Si un usager ou service choisit d'être joignable à une adresse qu'on peut deviner d'après le nom de la personne et l'affiliation organisationnelle (que décrivent la plupart des adresses-d'enregistrement), la méthode traditionnelle pour assurer la confidentialité en ayant un numéro de téléphone en "liste rouge" est compromise. Un service de localisation d'usagers peut porter atteinte à la confidentialité du receveur d'une invitation à une session en divulguant ses tenants et aboutissants spécifiques à l'appelant ; une mise en œuvre DEVRAIT par conséquent être capable de restreindre, utilisateur par utilisateur, quelle sorte d'informations de localisation et de disponibilité est délivrée à certaines classes d'appelants. C'est toute une classe de problèmes qu'on prévoit d'étudier plus en détails dans les travaux en cours sur SIP.

Dans certains cas, les usagers peuvent vouloir dissimuler les informations personnelles dans les champs d'en-tête qui portent l'identité. Ceci peut non seulement s'appliquer aux en-têtes From et ceux qui s'y rapportent pour représenter l'origine de la demande, mais aussi l'en-tête To – il peut n'être pas approprié de convoyer à la destination finale un pseudonyme de numérotation rapide, ou identifiant non développé pour un groupe de cibles, dont chacune serait retirée de l'URI-de-demande lorsque la demande est acheminée, mais ne serait pas changée dans le champ d'en-tête To si les deux étaient initialement identiques. Et donc, il PEUT être souhaitable pour des raisons de confidentialité de créer un champ d'en-tête To qui diffère de l'URI-de-demande.

27 Considérations relatives à l'IANA

Tous les noms de méthode, les noms de champ d'en-tête, les codes d'état, et les étiquettes d'option utilisées dans les applications SIP sont enregistrées auprès de l'IANA à travers des instructions dans une section Considérations relatives à l'IANA dans une RFC.

La présente spécification donne pour instruction à l'IANA de créer quatre nouveaux sous-registres sous http://www.iana.org/assignments/sip-parametershttp://www.iana.org/assignments/sip-parameters

Option Tags, Warning Codes (warn-codes), Methods et Response Codes, s'ajoutant au sous-registre de Header fields qui y est déjà présent.

27.1 Étiquettes d'option

La présente spécification établit le sous-registre Option Tags sous http://www.iana.org/assignments/sip-parameters .

Les étiquettes d'option sont utilisées dans des champs d'en-tête tels que Require, Supported, Proxy-Require, et Unsupported dans la prise en charge des mécanismes de compatibilité de SIP pour les extensions (paragraphe 19.2). L'étiquette d'option elle-même est une chaîne qui est associée à une option SIP particulière (c'est-à-dire, une extension). Elle identifie l'option pour les points d'extrémité SIP.

Les étiquettes d'option sont enregistrées par l'IANA lorsqu'elles sont publiées dans les RFC de normalisation. La section Considérations relatives à l'IANA de la RFC doit inclure les informations suivantes, qui apparaissent dans le registre de l'IANA avec le numéro de publication de la RFC.

- Nom de l'étiquette d'option. Le nom PEUT être de longueur quelconque, mais DEVRAIT être inférieur ou égal à vingt caractères. Le nom DOIT consister uniquement en caractères alphanumériques (Section 25).
- Texte descriptif de l'extension.

27.2 Codes d'avertissement

La présente spécification établit le sous-registre Warn-codes sous http://www.iana.org/assignments/sip-parameters et initie son remplissage avec les codes d'avertissement dont la liste figure au paragraphe 20.43. Des codes d'avertissement sont publiés avec la RFC concernée.

Le texte descriptif pour le tableau des warn-codes est :

Les codes d'avertissement fournissent des informations supplémentaires à celles du code d'état dans les messages de réponse SIP lorsque l'échec de la transaction résulte d'un problème de protocole de description de session (SDP) (RFC 2327 [1]).

Le "warn-code" consiste en trois chiffres. Un premier chiffre de "3" indique des avertissements spécifiques de SIP. Jusqu'à ce qu'une spécification future décrive l'utilisation de warn-codes autres que 3xx, seuls les warn-codes 3xx peuvent être enregistrés.

Les avertissements 300 à 329 sont réservés pour indiquer des problèmes avec les mots-clé dans la description de session, 330 à 339 sont relatifs aux services réseau de base nécessaires dans la description de session, 370 à 379 se rapportent aux paramètres quantitatifs de qualité de service nécessaires à la description de session, et 390 à 399 sont divers avertissements qui ne rentrent pas dans une des catégories précédentes.

27.3 Noms de champs d'en-tête

Ceci rend obsolètes les instructions de l'IANA sur le sous-registre header sous http://www.iana.org/assignments/sip-parameters.

Les informations suivantes doivent être fournies dans la publication d'une RFC afin d'enregistrer un nouveau nom de champ d'en-tête :
- o le numéro de la RFC dans laquelle l'en-tête est enregistré ;
- o le nom du champ d'en-tête enregistré ;
- o une version de forme compacte pour ce champ d'en-tête, s'il en est définie une.

Certains champs d'en-tête courants et largement utilisés PEUVENT se voir attribuer une forme compact d'une seule lettre (paragraphe 7.3.3). Les formes compactes ne peuvent être allouées qu'après révision par le groupe de travail SIP , suivie par une publication de RFC.

27.4 Codes de méthode et de réponse

La présente spécification établit le sous-registre Method et Response-Code sous http://www.iana.org/assignments/sip-parameters et initie son remplissage comme suit. Le tableau initial de Methods est :

INVITE	[RFC3261]
ACK	[RFC3261]
BYE	[RFC3261]
CANCEL	[RFC3261]
REGISTER	[RFC3261]
OPTIONS	[RFC3261]
INFO	[RFC2976]

Le tableau de codes de réponse est initialement rempli d'après la Section 21, les portions marquées Informational, Success, Redirection, Client-Error, Server-Error, et Global-Failure. Le tableau a le format suivant :

Type (par exemple, Informational)

Numéro	Phrase de cause par défaut	[RFC3261]

Les informations suivantes doivent être fournies dans une publication de RFC afin d'enregistrer un nouveau code de réponse ou de méthode :
- o le numéro de la RFC dans laquelle le code de méthode ou de réponse est enregistré ;
- o le numéro du code de réponse ou le nom de la méthode enregistré ;
- o la phrase de cause par défaut pour ce code de réponse, si applicable;

27.5 Type MIME "message/sip"

Le présent document enregistre le type de support MIME "message/sip" afin de permettre aux messages SIP d'être tunnelés comme corps au sein de SIP, principalement pour les besoins de la sécurité de bout en bout. Ce type de support est défini par les informations suivantes :

- o nom de type de support : message
- o nom de sous-type de support : sip
- o paramètres exigés : aucun
- o paramètres facultatifs : version
- o version : numéro de version SIP du message inclus (par exemple, "2.0"). S'il est absent, le numéro de version par défaut est "2.0".
- o schéma de codage : les messages SIP consistent en un en-tête de 8 bits facultativement suivi d'un objet de données MIME binaires. Comme tels, les messages SIP doivent être traités en binaire. Dans des circonstances normales les messages SIP sont transportés par des moyens de transport à capacités binaires, aucun codage spécial n'est nécessaire.
- o considérations sur la sécurité : voir ci-dessous.

Les motifs et exemples de cette utilisation comme mécanisme de sécurité de concert avec S/MIME sont donnés au paragraphe 23.4.

27.6 Enregistrement de paramètres de nouveau contenu-disposition

Le présent document enregistre aussi quatre nouveaux en-têtes de contenu-disposition "disposition-types": alert, icon, session et render. Les auteurs demandent que ces valeurs soient inscrites dans le registre IANA pour Content-Dispositions.

Les descriptions de ces "disposition-types", y compris les motifs et exemples, sont données au paragraphe 20.11.
Les descriptions brèves pour le registre IANA sont :

alert	le corps est une tonalité de sonnerie personnalisée pour alerter l'utilisateur
icon	le corps est affiché comme icône à l'utilisateur
render	le corps devrait être affiché à l'utilisateur
session	le corps décrit une session de communications, par exemple, comme corps SDP de la RFC 2327.

28 Changements par rapport à la RFC 2543

La présente RFC révise la RFC 2543. Elle est pour l'essentiel rétro-compatible avec la RFC 2543. Les changements décrits ici corrigent de nombreuses erreurs découvertes dans la RFC 2543 et fournissent des information sur des scénarios qui ne sont pas détaillés dans la RFC 2543. Le protocole a été présenté ici dans un modèle plus proprement structuré.

On marque les différences de comportement fonctionnel qui sont des changements substantiels par rapport à la RFC 2543, qui ont un impact sur l'interopérabilité ou corrigent le fonctionnement dans certains cas, et les comportements fonctionnels qui sont différents de ceux de la RFC 2543 mais ne sont pas une source potentielle de problèmes d'interopérabilité. Il y a eu aussi d'innombrables précisions qui ne sont pas documentées ici.

28.1 Changements fonctionnels majeurs

- o Lorsqu'un UAC souhaite terminer un appel avant qu'il y soit répondu, il envoie CANCEL. Si l'INVITE d'origine retourne toujours une 2xx, l'UAC envoie alors BYE. BYE ne peut être envoyé que sur une branche d'appel existante (qu'on appelle maintenant un dialogue dans cette RFC), tandis qu'il pouvait être envoyé à tout moment dans la RFC 2543.
- o Le BNF SIP a été converti pour être conforme à la RFC 2234.
- o Le BNF d'URL SIP a été rendu plus général, permettant un plus grand ensemble de caractères dans la partie utilisateur. De plus, les règles de comparaison ont été simplifiées pour être principalement insensibles à la casse, et le traitement détaillé de la comparaison de la présence de paramètres a été décrit. Le changement le plus substantiel est qu'un URI avec un paramètre à la valeur par défaut ne correspond pas à un URI qui n'a pas ce paramètre.
- o Retrait du masquage de Via. Cela pose de sérieux problèmes de confiance, car il s'appuie sur le prochain saut pour

effectuer le processus d'obfuscation. A la place de cela, le masquage de Via peut être fait par un choix de la mise en œuvre locale dans les mandataires à états pleins, et il n'est donc plus documenté.

- Dans la RFC 2543, les transactions CANCEL et INVITE été entremêlées. Elles sont maintenant séparées. Lorsque un utilisateur envoie une INVITE et ensuite une CANCEL, la transaction INVITE se termine toujours normalement. Un UAS doit répondre à la demande INVITE originale par une réponse 487.
- De la même façon les transactions CANCEL et BYE étaient entremêlées ; la RFC 2543 permettait que l'UAS n'envoie pas de réponse à INVITE à réception d'un BYE. Ceci n'est plus permis. L'INVITE d'origine a besoin d'une réponse.
- Dans la RFC 2543, les agents d'utilisateur avaient seulement besoin de prendre en charge UDP. Dans la présente RFC, les agents d'utilisateur doivent prendre en charge UDP et TCP.
- Dans la RFC 2543, un mandataire fourchu passe une seule mise en question provenant des éléments aval dans le cas de multiples mises en question. Dans cette RFC, les mandataires sont supposés collecter toutes les mises en question et les placer dans la réponse transmise.
- Dans l'accréditif Digest, l'URI doit être entre guillemets ; ceci n'est pas clair dans les RFC 2617 et RFC 2069 qui sont toutes deux peu cohérentes sur ce point.
- Le traitement SDP a été mis dans une spécification distincte [13], et plus complètement spécifiée comme un processus formel d'échange offre/réponse qui est effectivement tunnelé à travers SIP. SDP est permis dans INVITE/200 ou 200/ACK pour les mises en œuvre de base SIP ; la RFC 2543 faisait allusion à la capacité de l'utiliser dans INVITE, 200, et ACK dans une seule transaction, mais ce n'était pas bien spécifié. Des utilisations de SDP plus complexes sont permises dans les extensions.
- Ajout de la pleine prise en charge de IPv6 dans les URI et dans le champ d'en-tête Via. La prise en charge d'IPv6 dans Via a exigé que ses paramètres de champs d'en-tête permettent les caractères crochets et deux points. Ces caractères n'étaient pas admis précédemment. En théorie, cela pourrait causer des problèmes d'interopérabilité avec les mises en œuvre plus anciennes. Cependant, nous avons observé que la plupart des mises en œuvre acceptent tout caractère ASCII autre que de contrôle dans ces paramètres.
- La procédure SRV de DNS est maintenant documentée dans une spécification distincte [4]. Cette procédure utilise les enregistrements de ressource à la fois SRV et NAPTR et ne combine plus les données à travers les enregistrements SRV comme décrit dans la RFC 2543.
- La détection de boucle a été rendue facultative, supplantée par une utilisation obligatoire de Max-Forwards. La procédure de détection de boucle dans la RFC 2543 avait une sérieuse bogue qui pouvait faire rapport de "spirale" comme d'une condition d'erreur alors qu'elle ne l'est pas. La procédure de détection de boucle facultative est plus complètement et correctement spécifiée ici.
- L'usage des étiquettes est maintenant obligatoire (il était facultatif dans la RFC 2543), car elles sont maintenant les éléments fondamentaux de l'identification du dialogue.
- Ajout du champ d'en-tête Supported, permettant au client d'indiquer quelles extensions sont prises en charge à un serveur, d'appliquer ces extensions à la réponse, et d'indiquer leur usage avec un Require dans la réponse.
- Les paramètres d'extension manquaient dans le BNF pour plusieurs champs d'en-tête, et ils ont été ajoutés.
- La spécification du traitement de la construction de Route et Record-Route était très insuffisante dans la RFC 2543, et aussi n'avait pas la bonne approche. Elle a été substantiellement retravaillée dans la présente spécification (et largement simplifiée), et ceci est sans doute le plus gros changement. La rétro-compatibilité est toujours fournie pour les développements qui n'utilisent pas "pre-loaded routes" *(acheminement pré-chargé)*, lorsque la demande initiale avait un ensemble de valeurs de champ d'en-tête Route obtenu d'une certaine façon en-dehors de Record-Route. Dans ces situations, le nouveau mécanisme n'est pas interopérable.
- Dans la RFC 2543, les lignes dans un message pouvaient être terminées avec CR, LF, ou CRLF. La présente spécification ne permet plus que le CRLF.
- L'utilisation de Route dans CANCEL et ACK n'était pas bien définie dans la RFC 2543. Elle est maintenant bien spécifiée ; si une demande avait un champ d'en-tête Route, son CANCEL ou ACK pour une réponse non-2xx à la demande devait porter la même valeur de champ d'en-tête Route. Les ACK pour les réponses 2xx utilisent les valeurs Route apprises du Record-Route des réponses 2xx.
- La RFC 2543 permettait plusieurs demandes dans un seul paquet UDP. Cet usage a été retiré.
- L'usage du temps absolu dans le champ d'en-tête et paramètre Expire a été retiré. Il causait des problèmes d'interopérabilité dans des éléments qui n'étaient pas synchronisés. Des temps relatifs sont utilisés à la place.
- L'utilisation du paramètre branch de la valeur du champ d'en-tête Via est maintenant obligatoire pour tous les éléments. Il joue maintenant le rôle d'identifiant unique de transaction. Cela évite les règles complexes d'identification de transaction de la RFC 2543 qui étaient porteuses d'erreurs. Un cookie magique est utilisé dans la valeur de paramètre pour déterminer si le saut précédent avait rendu le paramètre unique au monde, et la comparaison revient aux vieilles règles lorsqu'il n'est pas présent. Et donc, l'interopérabilité est assurée.
- Dans la RFC 2543, la fermeture d'une connexion TCP était équivalente à un CANCEL. C'était presque impossible à mettre en œuvre (et faux) pour les connexions TCP entre mandataires. Cela a été éliminé, de sorte qu'il n'y a plus de couplage entre l'état de la connexion TCP et le traitement SIP.
- La RFC 2543 restait silencieuse sur le point de savoir si un agent utilisateur pouvait initier une nouvelle transaction avec un homologue alors qu'une autre était en cours. Ce point est maintenant spécifié. C'est admis pour des demandes non-INVITE, et non admis pour INVITE.
- PGP a été retiré. Ce n'était pas suffisamment spécifié, et non compatible avec le PGP MIME plus complet. Il a été

remplacé par S/MIME.

- Ajout du schéma d'URI "sips" pour TLS de bout en bout. Ce schéma n'est pas rétro-compatible avec la RFC 2543. Les éléments existants qui reçoivent une demande avec une schéma d'URI SIPS dans l'URI-de-demande vont vraisemblablement rejeter la demande. C'est une vraie caractéristique ; elle garantit qu'un appel à un URI SIPS n'est délivré que si tous les sauts du chemin peuvent être sécurisés.
- D'autres caractéristiques de sécurité ont été ajoutées avec TLS, et elle sont décrites dans une section sur les considérations de sécurité beaucoup plus développée.
- Dans la RFC 2543, un mandataire n'était pas obligé de transmettre les réponses provisoires de 101 à 199 vers l'amont. Cela a été rendu obligatoire, ce qui est important, car de nombreuses caractéristiques ultérieures dépendent de la livraison de toutes les réponses provisoires de 101 à 199.
- Il n'était pas dit grand-chose du code de réponse 503 dans la RFC 2543. Il a depuis trouvé une utilisation conséquente pour l'indication des conditions d'échec ou de surcharge chez les mandataires. Cela exige un traitement un peu particulier. La réception d'une réponse 503 devrait déclancher une tentative de contact du prochain élément dans le résultat d'une recherche SRV de DNS. La réponse 503 n'est transmise vers l'amont par un mandataire que sous certaines conditions.
- La RFC 2543 définit, mais ne spécifie pas suffisamment, un mécanisme pour l'authentification d'un serveur par un agent utilisateur. Cela a été supprimé. Les procédures d'authentification mutuelle de la RFC 2617 sont admises à la place.
- Un agent utilisateur ne peut pas envoyer un BYE pour un appel jusqu'à ce qu'il ait reçu un ACK pour l'INVITE initial. C'était admis dans la RFC 2543 mais conduit à une compétition potentielle.
- Un agent utilisateur ou mandataire ne peut pas envoyer CANCEL pour une transaction tant qu'il n'a pas obtenu une réponse provisoire pour la demande. C'était admis dans la RFC 2543 mais conduit à une compétition potentielle.
- Le paramètre action a été déconseillé dans les enregistrements. Il était insuffisant pour tous services utiles, et causait des conflits lors de l'application du traitement aux mandataires.
- La RFC 2543 avait un certain nombre de cas particuliers pour la diffusion groupée. Par exemple, certaines réponses étaient supprimées, les temporisateurs étaient ajustés, et ainsi de suite. La diffusion groupée joue maintenant un rôle plus limité, et le fonctionnement du protocole n'est plus affecté par l'usage de la diffusion groupée par opposition à l'envoi individuel. Les limitations qui en résultent sont précisées.
- L'authentification de base a été retirée entièrement et son usage interdit.
- Les mandataires ne transmettent plus une réponse 6xx immédiatement à sa réception. A la place, ils émettent CANCEL immédiatement sur les branches en cours. Cela évite une condition potentielle de concurrence qui aurait pour résultat qu'un UAC obtienne une réponse 6xx suivie d'une 2xx. Dans tous les cas excepté cette condition de concurrence, le résultat sera le même – la réponse 6xx est transmise vers l'amont.
- La RFC 2543 ne traitait pas le problème de la fusion de demandes. Cela survient lorsqu'une demande est fourchée chez un mandataire et se rejoint plus tard à un élément. Le traitement de la fusion n'est fait que par un agent utilisateur, et des procédures sont définies pour rejeter toutes les demandes sauf la première.

28.2 Changements fonctionnels mineurs

- Ajout des champs d'en-tête Alert-Info, Error-Info, et Call-Info pour la présentation facultative du contenu à l'usager.
- Ajout des champs d'en-tête Content-Language, Content-Disposition et MIME-Version.
- Ajout du mécanisme "glare handling" pour régler le cas où deux parties s'envoient simultanément l'une l'autre un re-INVITE. Il utilise le nouveau code d'erreur 491 (Demande en cours).
- Ajout des champs d'en-tête In-Reply-To et Reply-To pour prendre en charge le retour des appels ou messages manqués ultérieurement.
- Ajout de TLS et SCTP comme transports SIP valides.
- Il y avait divers mécanismes décrits pour le traitement des échecs à tout moment durant un appel ; ils sont maintenant unifiés. BYE est envoyé pour terminer.
- La RFC 2543 rendait obligatoire la retransmission des réponses INVITE sur TCP, mais notait qu'il n'était réellement nécessaire que pour 2xx. C'était un artifice pour pallier une structuration de protocole insuffisante. Avec la définition d'une couche de transaction plus cohérente, ce n'est plus nécessaire. Seules les réponses 2xx aux INVITE sont retransmises sur TCP.
- Les machines de serveur client et transaction sont maintenant pilotées sur la base de temporisations plutôt que de compteurs de retransmissions. Cela permet aux machines d'état d'être correctement spécifiées pour TCP et UDP.
- Le champ d'en-tête Date est utilisé dans les réponses REGISTER pour fournir un moyen simple pour l'auto-configuration des dates dans les agents utilisateurs.
- Permettre à un registraire de rejeter les enregistrements avec des délais d'expirations de trop courte durée. Définition du code de réponse 423 et de Min-Expires à cette fin.

29 Références normatives

[1] Handley, M. et V. Jacobson, "SDP: Session Description Protocol", RFC 2327, avril 1998.
[2] Bradner, S., "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigences", BCP 14, RFC 2119, mars 1997.
[3] Resnick, P., "Internet Message Format", RFC 2822, avril 2001.
[4] Rosenberg, J. et H. Schulzrinne, "SIP: Locating Serveur SIPs", RFC 3263, juin 2002.
[5] Berners-Lee, T., Fielding, R. et L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, août 1998.
[6] Chown, P., "Advanced Encryption Standard (AES) Ciphersuites pour Transport Layer Security (TLS)", RFC 3268, juin 2002.
[7] Yergeau, F., "UTF-8, un format de transformation d'ISO 10646", RFC 2279, janvier 1998.
[8] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. et T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, juin 1999.
[9] Vaha-Sipila, A., "URLs pour Telephone Calls", RFC 2806, avril 2000.
[10] Crocker, D. et P. Overell, "Augmented BNF pour Syntax Specifications: ABNF", RFC 2234, novembre 1997.
[11] Freed, F. et N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, novembre 1996.
[12] Eastlake, D., Crocker, S. et J. Schiller, "Randomness Recommendations for Security", RFC 1750, décembre 1994.
[13] Rosenberg, J. et H. Schulzrinne, "An Offer/Answer Model avec SDP", RFC 3264, juin 2002.
[14] Postel, J., "User Datagram Protocol", STD 6, RFC 768, août 1980.
[15] Postel, J., "DoD Standard Transmission Control Protocol", RFC 761, janvier 1980.
[16] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, Ritina, I., Kalla, M., Zhang, L. et V. Paxson, "Stream Control Transmission Protocol", RFC 2960, octobre 2000.
[17] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. et L. Stewart, "HTTP authentification: Basic et Digest Access Authentication", RFC 2617, juin 1999.
[18] Troost, R., Dorner, S. et K. Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition header field", RFC 2183, août 1997.
[19] Zimmerer, E., Peterson, J., Vemuri, A., Ong, L., Audet, F., Watson, M. et M. Zonoun, "MIME media types for ISUP and QSIG Objects", RFC 3204, décembre 2001.
[20] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, octobre 1989.
[21] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, janvier 1998.
[22] Galvin, J., Murphy, S., Crocker, S. et N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, octobre 1995.
[23] Housley, R., "Cryptographic Message Syntax", RFC 2630, juin 1999.
[24] Ramsdell B., "S/MIME Version 3 Message Specification", RFC 2633, juin 1999.
[25] Dierks, T. et C. Allen, "The TLS Protocol Version 1.0", RFC 2246, janvier 1999.
[26] Kent, S. et R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, novembre 1998.

30 Références informatives

[27] R. Pandya, "Emerging mobile and personal communication systems," IEEE Communications Magazine, Vol. 33, pp. 44--52, juin 1995.
[28] Schulzrinne, H., Casner, S., Frederick, R. et V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, janvier 1996.
[29] Schulzrinne, H., Rao, R. et R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, avril 1998.
[30] Cuervo, F., Greene, N., Rayhan, A., Huitema, C., Rosen, B. et J. Segers, "Megaco Protocol Version 1.0", RFC 3015, novembre 2000.
[31] Handley, M., Schulzrinne, H., Schooler, E. et J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, mars 1999.
[32] Hoffman, P., Masinter, L. et J. Zawinski, "The mailto URL scheme", RFC 2368, juillet 1998.
[33] E. M. Schooler, "A multicast user directory service for synchronous rendezvous," Master's Thesis CS-TR-96-18, Department of Computer Science, California Institute of Technology, Pasadena, California, août 1996.
[34] Donovan, S., "The SIP INFO Method", RFC 2976, octobre 2000.
[35] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, avril 1992.
[36] Dawson, F. et T. Howes, "vCard MIME Directory Profile", RFC 2426, septembre 1998.
[37] Good, G., "The LDAP Data Interchange Format (LDIF) - Technical Specification", RFC 2849, juin 2000.
[38] Palme, J., "Common Internet Message Headers", RFC 2076, février 1997.
[39] Franks, J., Hallam-Baker, P., Hostetler, J., Leach, P., Luotonen, A., Sink, E. et L. Stewart, "An Extension to HTTP: Digest Access Authentication", RFC 2069, janvier 1997.

[40] Johnston, A., Donovan, S., Sparks, R., Cunningham, C., Willis, D., Rosenberg, J., Summers, K. et H. Schulzrinne, "SIP Call Flow Examples", Travail en cours.
[41] E. M. Schooler, "Case study: multimedia conference control in a packet-switched teleconferencing system," Journal of Internetworking: Research et Experience, Vol. 4, pp. 99--120, juin 1993. ISI reprint series ISI/RS-93-359.
[42] H. Schulzrinne, "Personal mobility for multimedia services in the Internet," dans European Workshop on Interactive Distributed Multimedia Systems and Services (IDMS), (Berlin, Germany), mars 1996.
[43] Floyd, S., "Congestion Control Principles", RFC 2914, septembre 2000.

Tableau des valeurs des temporisateurs

Le Table 4 résume les significations et valeurs par défaut des divers temporisateurs utilisés par la présente spécification.

Temporisateur	**Valeur**	**Paragraphe**	**Signification**
T1	500 ms par défaut	17.1.1.1	Estimation du délai d'aller retour
T2	4 s	17.1.2.2	Intervalle maximum de retransmission pour des demandes non INVITE et réponses INVITE
T4	5 s	17.1.2.2	Durée maximum qu'un message restera dans le réseau
Temporisateur A	Initiallement T1	17.1.1.2	Pour UDP seulement, intervalle de retransmission de demande INVITE
Temporisateurr B	64*T1	17.1.1.2	Temporisateur de délai de transaction INVITE
Temporisateur C	> 3 min	16.6, point 11	Délai de transaction de mandataire INVITE
Temporisateur D	> 32 s pour UDP 0 s pour TCP/SCTP	17.1.1.2	Délai d'attente pour retransmission de réponse
Temporisateur E	Initiallement T1	17.1.2.2	Intervalle de retransmission de demande non-INVITE, pour UDP seulement
Temporisateur F	64*T1	17.1.2.2	Temporisateur de délai de transaction non-INVITE
Temporisateur G	Initiallement T1	17.2.1	Intervalle de retransmission de réponse INVITE
Temporisateur H	64*T1	17.2.1	Délai d'attente pour la réception de ACK
Temporisateur I	T4 pour UDP 0 s pour TCP/SCTP	17.2.1	Délai d'attente pour la retransmission de ACK
Temporisateur J	64*T1 pour UDP 0 s pour TCP/SCTP	17.2.2	Délai d'attente pour la retransmission de demande non-INVITE
Temporisateur	T4 pour UDP 0 s pour TCP/SCTP	17.1.2.2	Délai d'attente pour la retransmission de réponse

Tableau 4 : Récapitulation des temporisateurs

Remerciements

Nous souhaitons remercier les membres des groupes de travail MMUSIC et SIP de l'IETF pour leurs commentaires et suggestions. Des commentaires détaillés ont été fournis par Ofir Arkin, Brian Bidulock, Jim Buller, Neil Deason, Dave Devanathan, Keith Drage, Bill Fenner, Cedric Fluckiger, Yaron Goland, John Hearty, Bernie Hoeneisen, Jo Hornsby, Phil Hoffer, Christian Huitema, Hisham Khartabil, Jean Jervis, Gadi Karmi, Peter Kjellerstedt, Anders Kristensen, Jonathan Lennox, Gethin Liddell, Allison Mankin, William Marshall, Rohan Mahy, Keith Moore, Vern Paxson, Bob Penfield, Moshe J. Sambol, Chip Sharp, Igor Slepchin, Eric Tremblay, et Rick Workman.
Brian Rosen a founi le BNF compilé. Jean Mahoney a fourni l'assistance technique à la rédaction. Ce travail est fondé, entre autres, sur [41,42].

Adresse des auteurs

Les adresses des auteurs sont dans l'ordre alphabétique pour les éditeurs, les rédacteurs, puis viennent les auteurs originaux de la RFC 2543. Tous les auteurs de la liste ont contribué activement à de grandes portions du texte de ce document.

Jonathan Rosenberg
Dynamicsoft

72 Eagle Rock Ave
East Hanover, NJ 07936
USA
mél : jdrosen@dynamicsoft.com

Henning Schulzrinne
Dept. of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
USA
mél : schulzrinne@cs.columbia.edu

Gonzalo Camarillo
Ericsson

Advanced Signalling Research Lab.
FIN-02420 Jorvas

Email : Gonzalo.Camarillo@ericsson.com

Alan Johnston
WorldCom
100 South 4th Street
St. Louis, MO 63102
USA
mél : alan.johnston@wcom.com

Jon Peterson
NeuStar, Inc
1800 Sutter Street, Suite 570
Concord, CA 94520
USA
mél : jon.peterson@neustar.com

Robert Sparks
dynamicsoft, Inc.
5100 Tennyson Parkway, Suite 1200
Plano, Texas 75024
USA
mél : rsparks@dynamicsoft.com

Mark Handley
International Computer Science Institute
1947 Center St, Suite 600
Berkeley, CA 94704
USA
mél : mjh@icir.org

Eve Schooler
AT&T Labs-Research
75 Willow Road
Menlo Park, CA 94025
USA
mél : schooler@research.att.com

Propriété intellectuelle

Déclaration de copyright

Remerciement

Le financement de la fonction d'édition des RFC est actuellement fourni par l'Internet Society.