Groupe de travail Réseau
Request for Comments : 2748
Catégorie : En cours de normalisation
janvier 2000
Traduction Claude Brière de L'Isle

S . Herzog, Ed., IPHighway
J. Boyle, Level3
R. Cohen, Cisco
D. Durham, Intel
R. Rajan, AT&T
A. Sastry, Cisco

Protocole COPS (Service commun de politique ouverte)

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

(La présente RFC est mise à jour par la RFC4621)

Conventions utilisées dans ce document
Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" dans le présent document sont à interpréter comme décrit dans la [RFC2119].

Résumé
Le présent document décrit un modèle client/serveur simple pour la prise en charge du contrôle de politique sur les protocoles de signalisation de qualité de service. Le modèle ne fait aucune hypothèse sur les méthodes du serveur de politique, mais se fonde sur le retour de décisions par le serveur aux demandes de politique. Le modèle est conçu pour être extensible afin que d'autres sortes de clients de politique puissent être pris en charge à l'avenir. Cependant le présent document ne prétend pas être la seule approche, ni l'approche préférable, de la mise en application de futurs types de politiques.

Table des matières

1. Introduction

Le présent document décrit un protocole simple de questions/réponses qui peut être utilisé pour échanger des informations de politique entre un serveur de politique (PDP, *Policy Decision Point*) et ses clients (PEP, *Policy Enforcement Point*). Un exemple de client de politique est un routeur RSVP qui doit exercer sur une utilisation de RSVP [RFC2205] un contrôle d'admission fondé sur la politique. On suppose qu'il existe au moins un serveur de politique dans chaque domaine administratif contrôlé. Le modèle de base des interactions entre un serveur de politique et ses clients est compatible avec le document cadre pour le contrôle d'admission fondé sur la politique [RFC2753].

Un objectif clé de ce protocole de contrôle de politique est de commencer par un concept simple mais extensible. Les principales caractéristiques du protocole COPS sont les suivantes :

1. Le protocole emploie un modèle client/serveur dans lequel le PEP envoie des demandes, des mises à jour et des suppressions sur le PDP distant, et le PDP retourne les décisions au PEP.
2. Le protocole utilise TCP comme protocole de transport pour un échange fiable de messages entre les clients de politique et un serveur. Donc, aucun mécanisme supplémentaire n'est nécessaire pour une communication fiable entre un serveur et ses clients.
3. Le protocole est, par sa conception, extensible pour la prise en charge d'objets auto-identifiants et peut prendre en charge diverses informations spécifiques du client sans exiger de modification du protocole COPS lui-même. Le protocole a été créé pour l'administration générale, la configuration, et la mise en application des politiques.
4. COPS fournit la sécurité au niveau du message pour l'authentification, la protection contre la répétition, et l'intégrité du message. COPS peut aussi réutiliser des protocoles de sécurité existants tels que IPSEC [RFC2401] ou TLS pour authentifier et sécuriser le canal entre le PEP et le PDP.
5. Le protocole est à états pleins sous deux aspects principaux : (1) l'état Demande/Décision est partagé entre le client et le serveur et (2) l'état provenant de divers événements (paires Demande/Décision) peut être inter-associé. Par (1) on veut dire que les demandes provenant du client PEP sont installées ou mémorisées par le PDP distant jusqu'à ce qu'elles soient explicitement supprimées par le PEP. En même temps, les décisions provenant du PDP distant peuvent être générées en asynchrone à tout moment pour un état de demande actuellement installé. Par (2) on veut dire que le serveur peut répondre différemment à de nouvelles demandes à cause d'un état Demande/Décision précédemment installé qui s'y rapporte.
6. De plus, le protocole est à états pleins en ce qu'il permet au serveur de pousser les informations de configuration au client, et permet donc au serveur de retirer de tels état chez le client lorsque ils ne sont plus applicables.

1.1 Modèle de base

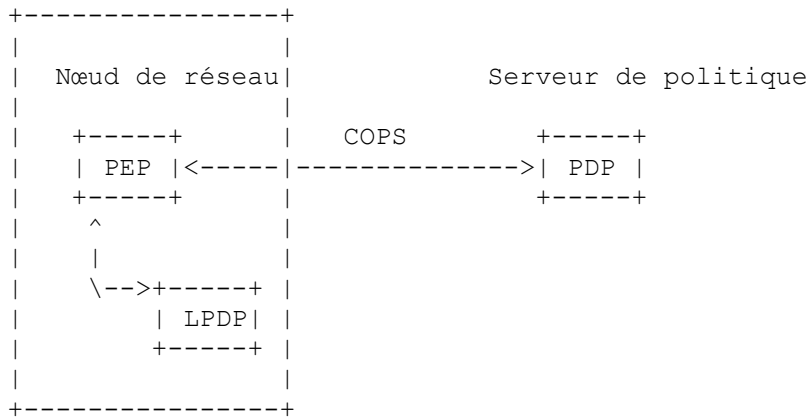


Figure 1 : Illustration de COPS

La Figure 1 illustre la disposition des divers composants de politique dans un exemple COPS normal (tiré de la [RFC2753]). Ici, COPS est utilisé pour communiquer les informations de politique entre un point de mise en application de politique (PEP, *Policy Enforcement Point*) et un point de décision de politique (PDP, *Policy Decision Point*) distant au sein du contexte d'un type de client particulier. Le point de décision de politique local (LPDP, *Local Policy Decision Point*) facultatif peut être utilisé par l'appareil pour prendre des décisions de politique locales en l'absence d'un PDP.

On suppose que chaque client de politique participant est cohérent fonctionnellement avec un PEP [RFC2753]. Le PEP peut communiquer avec un serveur de politique (qu'on appelle ici un PDP distant [RFC2753]) pour obtenir des décisions ou directives de politique.

Le PEP est chargé d'initier une connexion TCP persistente avec un PDP. Le PEP utilise cette connexion TCP pour envoyer des demandes et recevoir des décisions de la part du PDP distant. La communication entre le PEP et le PDP distant est principalement sous la forme d'un échange de demande/décision à état plein, bien que le PDP distant puisse à l'occasion envoyer des décisions non sollicitées au PEP pour forcer des changements dans les états de demandes précédemment approuvés. Le PEP a aussi la capacité de faire rapport au PDP distant de la réussite de la réalisation de la décision locale du PDP, ce qui est utile pour les besoins de comptabilité et de surveillance. Le PEP est chargé de notifier au PDP qu'un état de demande a changé sur le PEP. Enfin, le PEP est chargé de la suppression de tout état qui n'est plus applicable du fait d'événements chez le client ou de décisions produites par le serveur.

Lorsque le PEP envoie une demande de configuration, il s'attend à ce que le PDP envoie en continu des unités désignées de données de configuration au PEP via des messages de décision selon ce qui est applicable pour la demande de configuration. Lorsque une unité de données de configuration désignées est installée avec succès sur le PEP, celui-ci devrait envoyer un message de rapport au PDP pour confirmer l'installation. Le serveur peut alors mettre à jour ou supprimer les informations de configuration désignées via un nouveau message de décision. Lorsque le PDP envoie une décision de suppression de données de configuration désignées chez le PEP, celui-ci va supprimer la configuration spécifiée et envoyer un message de rapport au PDP à titre de confirmation.

Le protocole de politique est conçu pour communiquer des objets auto-identifiants qui contiennent les données nécessaires pour identifier les états de demande, établir le contexte pour une demande, identifier le type de demande, référencer des demandes installées précédemment, relayer des décisions de politique, faire rapport d'erreurs, assurer l'intégrité du message, et transférer des informations spécifiques du client ou de l'espace de noms.

Pour distinguer les différentes sortes de clients, le type de client est identifié dans chaque message. Les différents types de clients peuvent avoir des données spécifiques de client différentes et peuvent exiger des sortes différentes de décisions de politique. Il est prévu que chaque nouveau type de client ait un projet d'utilisation correspondant qui spécifie les particularités de son interaction avec ce protocole de politique.

Le contexte de chaque demande correspond au type d'événement qui l'a déclanché. L'objet COPS Contexte identifie le type de demande et de message (si applicable) qui a déclanché un événement de politique via ses champs de type de message et de type de demande. COPS identifie trois types d'événements d'origine extérieure : (1) l'arrivée d'un message entrant, (2) l'allocation de ressources locales, et (3) la transmission d'un message sortant. Chacun de ces événements peut exiger la prise d'une décision différente. Le contenu d'une demande/décision COPS dépend du contexte. Un quatrième type de demande est utile pour les types de clients qui souhaitent recevoir des informations de configuration de la part du PEP pour produire une demande de configuration pour un appareil nommé spécifique ou un module qui requiert que soient installées des informations de configuration.

Le PEP peut aussi avoir la capacité de prendre une décision de politique locale via son point de décision de politique local (LPDP, *Local Policy Decision Point*) [RFC2753], cependant, le PDP reste tout le temps le point de décision d'autorité. Cela signifie que les informations de décision locale pertinentes doivent être relayées au PDP. C'est-à-dire que l'accès à toutes les informations pertinentes pour prendre la décision finale de politique doit être accordé au PDP. Pour faciliter cette fonctionnalité, le PEP doit envoyer ses informations de décision locale au PDP distant via un objet de décision LPDP. Le PEP doit alors respecter la décision du PDP car elle est absolue.

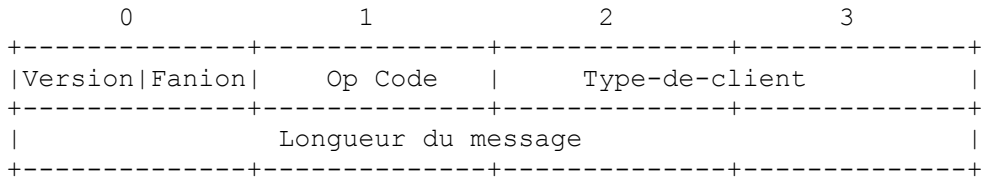
Finalement, la tolérance aux fautes est une capacité exigée pour le présent protocole, en particulier à cause du fait qu'elle est associée à la sécurité et à la gestion de service des appareils répartis dans le réseau. La tolérance aux fautes peut être réalisée en faisant qu'à la fois le PEP et le PDP distant vérifient constamment leur connexion mutuelle via des messages de maintien en vie. Lorsque une défaillance est détectée, le PEP doit essayer de se reconnecter au PDP distant ou tenter de se connecter à un PDP de sauvegarde en remplacement. Lorsque il est déconnecté, le PEP devrait revenir à la prise de décisions locales. Une fois qu'une connexion est rétablie, on attend du PEP qu'il notifie au PDP tout état supprimé ou tout nouvel événement qui a réussi le contrôle local d'admission après la perte de connexion. De plus, le PDP distant peut demander que tout l'état interne du PEP soit resynchronisé (toutes les demandes précédemment installées sont à produire à nouveau). Après défaillance et avant que la nouvelle connexion soit pleinement fonctionnelle, l'interruption de service peut être minimisée si le PEP met en antémémoire les décisions précédemment communiquées et continue de les utiliser pour une durée limitée. Les paragraphes 2.3 et 2.5 détaillent les mécanismes de COPS pour assurer la fiabilité.

2. Le protocole

Cette section décrit les formats de message et les objets échangés entre le PEP et le PDP distant.

2.1 En-tête commun

Chaque message COPS comporte l'en-tête COPS suivi par un certain nombre d'objets typés.



Note globale : //// implique que le champ est réservé, mis à 0.

Les champs de l'en-tête sont :
Version : 4 bits. C'est le numéro de version COPS. La version actuelle est 1.
Fanion : 4 bits. Les valeurs de fanion définies (tous les autres fanions DOIVENT être mis à 0) :
0x1 bit fanion Message sollicité
Ce fanion est mis lorsque le message est sollicité par un autre message COPS. Ce fanion N'EST PAS à établir (valeur = 0) sauf spécification contraire à la section 3.

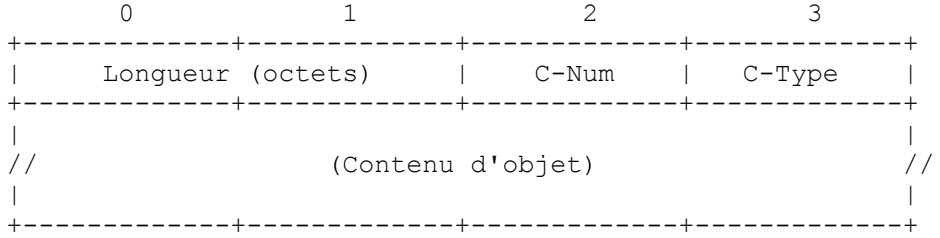
Op Code : 8 bits. C'est le code de fonctionnement de COPS :
1 = Demande (REQ, *Request*)
2 = Décision (DEC)
3 = État de rapport (RPT, *Report State*)
4 = État Demande de suppression (DRQ, *Delete Request State*)
5 = Demande de synchronisation d'état (SSQ, *Synchronize State Request*)
6 = Client-Ouvert (OPN)
7 = Client-Accepté (CAT, *Client-Accept*)
8 = Client-Fermé (CC, *Client-Close*)
9 = Garder-en-vie (KA, *Keep-Alive*)
10 = Synchronisation terminée (SSC, *Synchronize Complete*)

Type-de-client : 16 bits. Le Type-de-client identifie le client de politique. L'interprétation de tous les objets encapsulés se rapporte au type de client. Les types de client qui établissent à 1 le bit de poids fort dans le champ Type-de-client sont spécifiques de l'entreprise (ce sont les Type-de-client de 0x8000 à 0xFFFF). (Voir dans les documents spécifiques du client d'utilisation les identifiants de type de client particulier.) Pour les messages Garder-en-vie, le type de client dans l'en-tête DOIT toujours être à 0 car il sont utilisés pour la vérification de la connexion (et non pour la vérification de la session de client).

Longueur du message : 32 bits. C'est la taille du message en octets, qui inclut l'en-tête COPS standard et tous les objets encapsulés. Les messages DOIVENT être alignés sur les intervalles de 4 octets.

2.2 Formats d'objet spécifique de COPS

Tous les objets suivent le même format d'objet ; chaque objet consiste en un ou plusieurs mots de 32 bits avec un en-tête de quatre octets, en utilisant le format suivant :



La longueur est une valeur de deux octets qui décrit le nombre d'octets (y compris l'en-tête) qui composent l'objet. Si la longueur en octets ne tombe pas sur une limite de mot de 32 bits, le bourrage DOIT être ajouté à la fin de l'objet de telle sorte qu'il soit aligné sur la prochaine frontière de 32 bits avant que l'objet puisse être envoyé sur le réseau. Du côté réception, la frontière d'objet suivante sera trouvée en arrondissant simplement la longueur déclarée de l'objet précédent à la frontière de 32 bits suivante.

Normalement, C-Num identifie la classe d'informations contenues dans l'objet, et le C-Type identifie le sous-type ou la version des informations contenues dans l'objet.

C-num : 8 bits
- 1 = Manette *(Handle)*
- 2 = Contexte
- 3 = Interface entrante
- 4 = Interface sortante
- 5 = Code de cause
- 6 = Décision
- 7 = Décision de LPDP
- 8 = Erreur
- 9 = Informations spécifiques du client
- 10 = Temporisateur de maintien en vie
- 11 = Identification de PEP
- 12 = Type de rapport
- 13 = Adresse du PDP de redirection
- 14 = Adresse du dernier PDP
- 15 = Temporisateur de comptabilité
- 16 = Intégrité du message

C-type : 8 bits. Valeurs définies par C-num.

2.2.1 Objet Manette

L'objet Manette encapsule une valeur unique qui identifie un état installé. Cette identification est utilisée par la plupart des opérations COPS. Un état correspondant à une manette DOIT être explicitement supprimé lorsque il n'est plus applicable. Voir les détails au paragraphe 2.4.

C-Num = 1

C-Type = 1, Manette de client.

Champ de longueur variable, pas d'implication de format autre qu'il est unique parmi les autres manettes de client du même PEP (autrement dit de la connexion TCP COPS) pour un type de client particulier. Il est toujours initialement choisi par le PEP puis supprimé par le PEP lorsque il n'est plus applicable. La manette de client est utilisée pour se référer à un état de demande initié par un PEP particulier et installé au PDP pour un type de client. Un PEP va spécifier une manette de client dans ses messages Demande, dans ses messages Rapport et dans ses messages Supprimer envoyés par le PDP. Dans tous les cas, la manette de client est utilisée pour identifier de façon univoque une demande d'un PEP particulier pour un type de client.

La valeur de la manette de client est réglée par le PEP et est opaque au PDP. Le PDP effectue simplement une comparaison à l'octet près sur la valeur de cet objet par rapport aux valeurs de l'objet Manette des autres demandes actuellement installées.

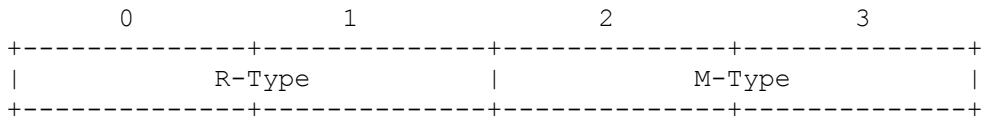
2.2.2 Objet Contexte

Il spécifie le type d'événement qui a déclanché l'interrogation. Il est exigé pour les messages Demande. Le contrôle d'admission, l'allocation de ressources, et la transmission des demandes sont tous sensibles aux types de client qui exportent leur facilité de prise de décision au PDP. Pour les types de client applicables, un PEP peut aussi faire une demande pour recevoir des informations de configuration désignées de la part du PDP. Ces données de configuration désignées peuvent être sous la forme utile pour régler les attributs du système sur un PEP, ou elles peuvent être sous la forme de règles de politique qui sont à vérifier directement par le PEP.

Plusieurs fanions peuvent être établis pour la même demande. Ceci n'est cependant permis que si l'ensemble des informations spécifiques du client dans la demande combinée sont identiques aux informations spécifiques du client qui

seraient spécifiées si des demandes individuelles était faites pour chaque fanion spécifié.

C-num = 2, C-Type = 1



R-Type (Fanion de type de demande)
0x01 = Message entrant / demande de contrôle d'admission
0x02 = Demande d'allocation de ressources
0x04 = Demande de message sortant
0x08 = Demande de configuration

M-Type (Type de message) : Valeurs de types de message de protocole de 16 bits spécifiques du client.

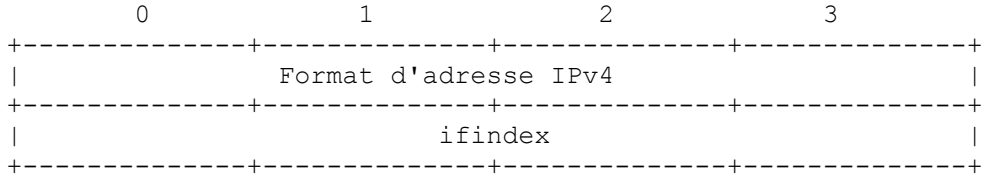
2.2.3 Objet Interface entrante (IN-Int)

L'objet Interface entrante est utilisé pour identifier l'interface d'entrée sur laquelle s'applique une demande particulière et l'adresse d'où est originaire le message reçu. Pour les flux ou messages générés sur l'hôte local du PEP, l'adresse de retour et l'ifindex sont utilisés.

Cet objet Interface est aussi utilisé pour identifier l'interface entrante (receveuse) via son ifindex. Le ifindex peut être utilisé pour différencier les sous-interfaces et les interfaces non numérotées (voir le LIH de RSVP pour un exemple). Lorsque SNMP est accepté par le PEP, cet ifindex entier DOIT correspondre à la même valeur d'entier pour l'interface dans le tableau d'index d'interface de la MIB-II SNMP.

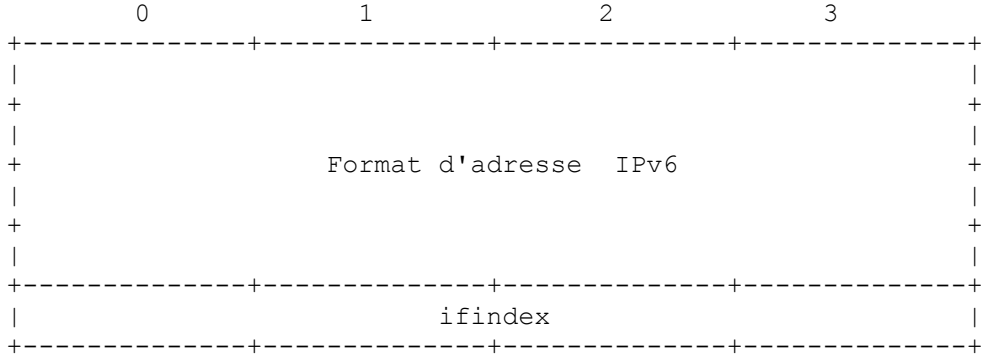
Note : Le ifindex spécifié dans Interface entrante se rapporte normalement au flux des messages de protocole sous-jacents. Le ifindex est l'interface sur laquelle le message de protocole a été reçu.

C-Num = 3
C-Type = 1, Adresse IPv4 + Interface



Pour ce type d'objet d'interface, l'adresse IPv4 spécifie l'adresse IP d'où venait le message entrant.

C-Type = 2, Adresse IPv6 + Interface



Pour ce type d'objet d'interface, l'adresse IPv6 spécifie l'adresse IP d'où est venu le message entrant. Le ifindex est utilisé pour se référer à l'interface locale entrante définie par la MIB-II sur le PEP comme décrit ci-dessus.

2.2.4 Objet Interface sortante (OUT-Int)

Interface sortante est utilisé pour identifier l'interface sortante à laquelle s'applique une demande spécifique et l'adresse à laquelle le message transmis est à envoyer. Pour les flux ou messages destinés à l'hôte local du PEP, l'adresse de retour et le ifindex sont utilisés. Interface sortante a les mêmes formats que l'objet Interface entrante.

Cet objet Interface est aussi utilisé pour identifier l'interface de sortie (de transmission) via son ifindex. Le ifindex peut être utilisé pour différencier les sous-interfaces et les interfaces non numérotées (voir un exemple avec le LIH de RSVP). Lorsque SNMP est pris en charge par le PEP, cet entier ifindex DOIT correspondre à la même valeur d'entier pour l'interface dans le tableau d'index d'interface de la MIB-II SNMP.

Note : Le ifindex spécifié dans Interface sortante se rapporte normalement au flux des messages de protocole sous-jacents. Le ifindex est celui dont un message de protocole est sur le point d'être transmis.

C-Num = 4
C-Type = 1, Adresse IPv4 + Interface

Même format de C-Type que pour l'objet Interface sortante. L'adresse IPv4 spécifie l'adresse IP à laquelle va le message sortant. Le ifindex est utilisé pour se référer à l'interface locale sortante sur le PEP définie par la MIB-II.

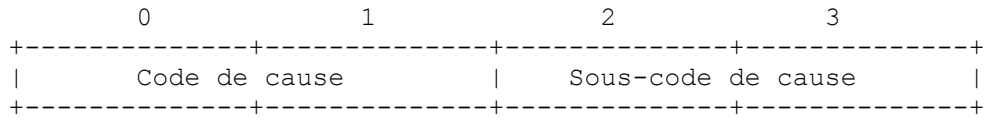
C-Type = 2, Adresse IPv6 + Interface

Même format de C-Type que dans l'objet Interface entrante. Pour ce type de l'objet interface, l'adresse IPv6 spécifie l'adresse IP à laquelle va le message sortant. Le ifindex est utilisé pour se référer à l'interface locale sortante sur le PEP définie par la MIB-II.

2.2.5 Objet Cause

Cet objet spécifie la raison pour laquelle l'état de la demande a été supprimé. Il apparaît dans le message Demande de suppression (DRQ, *delete request*). Le champ Sous-code de cause est réservé pour des codes de causes spécifiques du client plus détaillés définis dans les documents correspondants.

C-Num = 5, C-Type = 1



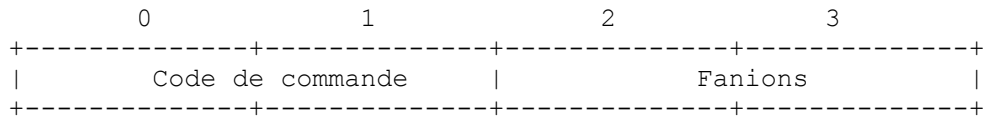
Code de cause :

1 = non spécifié
2 = Gestion
3 = Préempté (un autre état de demande a la préséance)
4 = Supprimer (Utilisé pour communiquer un retrait d'état signalé)
5 = Fin de temporisation (l'état local est arrivé à expiration)
6 = Changement de chemin (le changement invalide l'état de demande)
7 = Ressources insuffisantes (pas de ressources locales disponibles)
8 = Directive du PDP (une décision du PDP a causé la suppression)
9 = Décision non acceptée (la décision du PDP n'est pas acceptée)
10 = Manette de synchronisation inconnue
11 = Manette transitoire (événement sans état)
12 = Décision mal formée (n'a pas pu être récupérée)
13 = Objet COPS inconnu provenant du PDP : le sous-code (octet 2) contient un C-num d'objet inconnu et (octet 3) contient un C-Type d'objet inconnu.

2.2.6 Objet Décision (Decision)

C'est une décision prise par le PDP. Elle apparaît dans les réponses. Les objets de décision non obligatoires spécifiques requis dans une décision sur une demande particulière dépendent du type de client.

C-Num = 6
C-Type = 1, Fanions de décision (obligatoires)



Commandes :

0 = Décision nulle (pas de données de configuration disponibles)
1 = Installer (Admettre la demande/Installer la configuration)
2 = Retirer (Retirer la demande/Retirer la configuration)

Fanions :

0x01 = Déclancher une erreur (Déclanche un message d'erreur s'il est établi)

Note : Déclancher une erreur est applicable aux types de client qui sont capables d'envoyer des notifications d'erreur pour les messages signalés.

Les valeurs de fanion qui ne sont pas applicables au R-Type ou type de client d'un contexte donné DOIVENT être ignorées par le PEP.

C-Type = 2, Données sans état

Ce type d'objet de décision porte des informations supplémentaires sans état qui peuvent être appliquées localement par le PEP. C'est un objet de longueur variable et son format interne DEVRAIT être spécifié dans le document d'extension COPS pertinent pour le type de client particulier. Cet objet est facultatif dans les messages Décision et est interprété par rapport à un contexte donné.

Il est prévu que même les PEP qui "exportent" soient capables de prendre de simples décisions de politique sans état en local dans leur LPDP. Comme ce réglage est bien connu et mis en œuvre partout, les PDP en sont aussi informés (soit de façon universelle, par configuration, soit en utilisant le message Client-Ouvert). Le PDP peut aussi inclure ces informations dans sa décision, et le PEP DOIT les appliquer à l'événement d'allocation de ressource qui a généré la demande.

C-Type = 3, Données de remplacement

Ce type d'objet de décision porte des données de remplacement destinées à remplacer les données existantes dans un message signalé. C'est un objet de longueur variable et son format interne DEVRAIT être spécifié dans le document d'extension COPS pertinent pour le type de client en question. Il est facultatif dans les messages Décision et est interprété par rapport à un contexte donné.

C-Type = 4, Données de décision spécifiques du client

Les types de décision supplémentaires peuvent être introduits en utilisant l'objet Données de décision spécifiques du client. C'est un objet de longueur variable et son format interne DEVRAIT être spécifié dans le document d'extension COPS pertinent pour ce type de client. Il est facultatif dans les messages Décision et est interprété par rapport à un contexte donné.

C-Type = 5, Données de décision désignées

Les informations de configuration désignées sont encapsulées dans cette version de l'objet Décision en réponse aux demandes de configuration. C'est un objet de longueur variable et son format interne DEVRAIT être spécifié dans le document d'extension COPS pertinent pour ce type de client. Il est facultatif dans les messages Décision et est interprété par rapport à un contexte et des fanions Décision donnés.

2.2.7 Objet Décision LPDP (LPDPDecision)

C'est une décision prise par le point de décision de politique locale du PEP (LPDP). Elle peut apparaître dans les demandes. Ces objets correspondent aux objets de décision spécifiques du client définis ci-dessus et ont le même format.

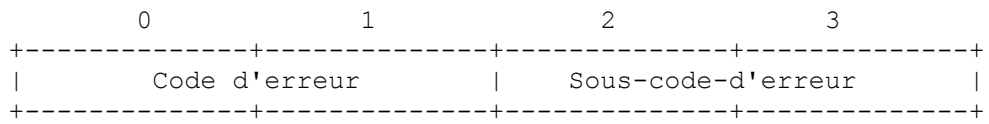
C-Num = 7
C-Type = (même C-Type que pour les objets Décision)

2.2.8 Objet Erreur (Error)

Cet objet est utilisé pour identifier une erreur particulière du protocole COPS. Le champ Sous-code-d'erreur contient des

codes d'erreur supplémentaires détaillés spécifiques du client. Les sous-codes Erreur appropriés pour un type de client particulier DEVRAIENT être spécifiés dans le document d'extensions COPS pertinent.

C-Num = 8, C-Type = 1



Code d'erreur :

1 = Mauvaise manette
2 = Référence de manette invalide
3 = Mauvais format de message (message mal formé)
4 = Incapable de traiter (le serveur abandonne l'interrogation)
5 = Manque d'informations spécifiques du client obligatoires
6 = Type de client non pris en charge
7 = Manque d'un objet COPS obligatoire
8 = Défaillance du client
9 = Défaillance de la communication
10 = Non spécifiée
11 = Fermeture
12 = Redirigé sur le serveur préféré
13 = Objet COPS non identifié : le sous-code (octet 2) contient le C-Num d'un objet inconnu et (octet 3) contient le C-Type d'un objet inconnu.
14 = Échec d'authentification
15 = Authentification exigée

2.2.9 Objet Informations spécifiques du client (ClientSI)

Les divers types de cet objet sont exigés pour les demandes, et utilisés dans les rapports lorsque nécessaire. Il contient des informations spécifiques du type de client.

C-Num = 9,
C-Type = 1, ClientSI signalé.

Champ de longueur variable. Tous les objets/attributs spécifiques du protocole de signalisation ou de l'état interne d'un client sont encapsulés dans un ou plusieurs objets Informations spécifiques du client signalés. Le format des données encapsulées dans l'objet ClientSI est déterminé par le type de client.

C-Type = 2, ClientSI désigné

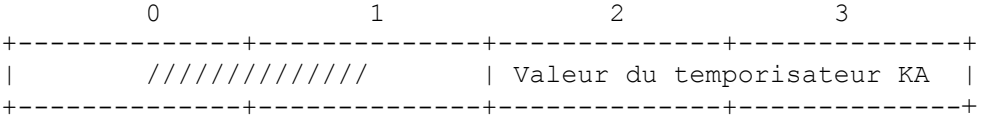
Champ de longueur variable. Il contient les informations de configuration désignées utiles pour relayer des informations spécifiques sur le PEP, une demande, ou un état configuré, au serveur PDP.

2.2.10 Objet Temporisateur de garde en vie (KATimer)

Les durées sont codées par des valeurs d'entier de 2 octets et sont en unités de secondes. La valeur du temporisateur est traitée comme une différence (delta).

C-Num = 10,
C-Type = 1, Valeur du temporisateur de garde en vie

Objet Temporisateur utilisé pour spécifier l'intervalle maximum sur lequel un message COPS DOIT être envoyé ou reçu. La gamme des temporisations finies est de 1 à 65535 secondes représentée par un entier non signé de deux octets. La valeur zéro implique l'infini.



2.2.11 Objet Identification de PEP (PEPID)

L'objet Identification de PEP est utilisé pour identifier le client PEP auprès du PDP distant. Il est exigé pour les messages Client-Ouvert.

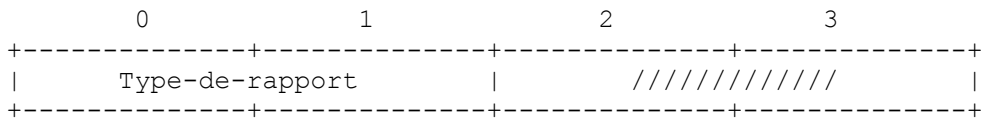
C-Num = 11, C-Type = 1

Champ de longueur variable. C'est une chaîne ASCII terminée par NUL qui est aussi bourrée de zéros jusqu'à une frontière de mot de 32 bits (afin que la longueur de l'objet soit un multiple de 4 octets). Le PEPID DOIT contenir une chaîne ASCII qui identifie de façon univoque le PEP au sein du domaine de politique d'une manière persistante à travers les réinitialisations du PEP. Par exemple, ce peut être l'adresse IP allouée en statique au PEP ou un nom du DNS. Cet identifiant peut être utilisé en toute sécurité par un PDP comme manette pour identifier le PEP dans ses règles de politique.

2.2.12 Objet Type-de-rapport (Report-Type)

C'est le type de rapport sur l'état de la demande associé à une manette :

C-Num = 12, C-Type = 1

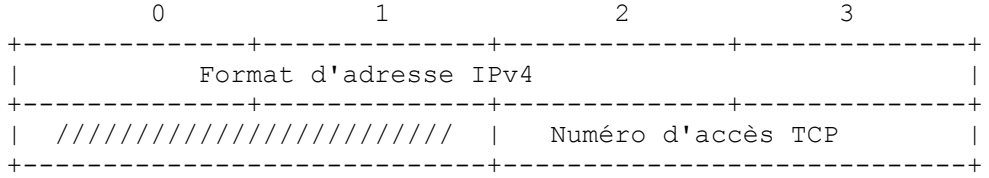


Type-de-rapport:
1 = Succès : la décision a réussi au PEP
2 = Échec : la décision n'a pas pu être menée à son terme par le PEP
3 = Comptabilité : mise à jour de comptabilité pour un état installé

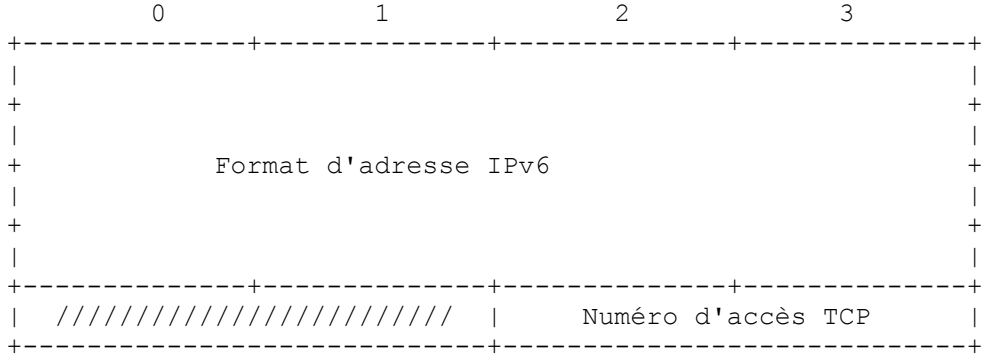
2.2.13 Adresse de redirection de PDP (PDPRedirAddr)

Lorsque un PDP clôt une session de PEP pour un type de client particulier, il peut facultativement utiliser cet objet pour rediriger le PEP sur l'adresse et le numéro d'accès TCP du serveur PDP spécifié :

C-Num = 13,
C-Type = 1, Adresse IPv4 + accès TCP



C-Type = 2, Adresse IPv6 + accès TCP



2.2.14 Dernière adresse de PDP (LastPDPAddr)

Lorsque un PEP envoie un message Client-Ouvert pour un type de client particulier, le PEP DEVRAIT spécifier le dernier PDP qu'il a réussi à ouvrir (ce qui signifie qu'il a reçu un Client-Accepté) depuis le dernier réamorçage du PEP. Si aucun PDP n'a été utilisé depuis le dernier réamorçage, le PEP ne va simplement pas inclure cet objet dans le message Client-

Ouvert.

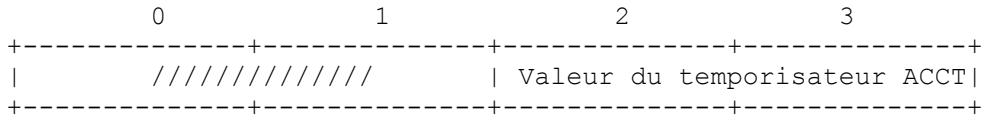
C-Num = 14,
C-Type = 1, adresse IPv4 (même format que PDPRedirAddr)
C-Type = 2, adresse IPv6 (même format que PDPRedirAddr)

2.2.15 Objet Temporisateur de comptabilité (AcctTimer)

Les durées sont codées comme des valeurs d'entier de 2 octets et sont en unités de secondes. La valeur du temporisateur est traitée comme une différence.

C-Num = 15,
C-Type = 1, valeur de temporisateur de comptabilité

C'est la valeur du temporisateur facultatif utilisé pour déterminer l'intervalle minimum entre les rapports périodiques de type de comptabilité. Elle est utilisée par le PDP pour décrire au PEP un intervalle acceptable entre les mises à jour non sollicitées de comptabilité via les messages Rapport lorsque ils sont applicables. Elle donne une méthode pour que le PDP contrôle la quantité de trafic comptable vu par le réseau. La gamme des valeurs finies de temps est de 1 à 65535 secondes représentées par un entier non signé de deux octets. Une valeur de zéro signifie qu'il DEVRAIT n'y avoir pas de mise à jour comptable non sollicitée.



2.2.16 Objet Intégrité du message (Integrity)

L'objet Intégrité comporte un numéro de séquence et un résumé de message utile pour l'authentification et la validation d'un message COPS. Lorsque il est utilisé, l'intégrité est protégée à la fin d'un message COPS comme le dernier objet COPS. Le résumé est alors calculé sur la totalité d'un message COPS particulier jusqu'à la valeur du résumé elle-même sans l'inclure. L'envoyeur d'un message COPS va calculer et remplir la portion résumé de l'objet Intégrité. Le receveur d'un message COPS va alors calculer un résumé sur le message reçu et vérifier qu'il correspond au résumé contenu dans l'objet Intégrité reçu.

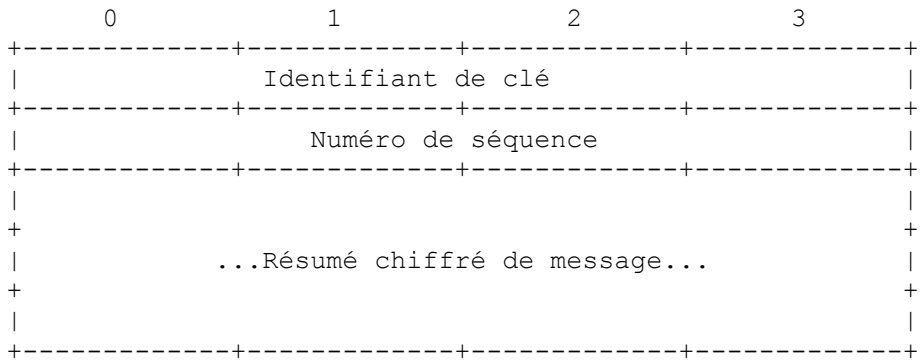
C-Num = 16,
C-Type = 1, résumé HMAC

L'objet Intégrité HMAC utilise HMAC (Hachage à clé pour l'authentification de message) [RFC2104] pour calculer le résumé de message sur la base d'une clé partagée entre le PEP et son PDP.

Cet objet Intégrité spécifie un identifiant de clé de 32 bits utilisé pour identifier une clé spécifique partagée par un PEP particulier et son PDP ainsi que l'algorithme cryptographique à utiliser. L'identifiant de clé permet que plusieurs clés simultanées existent sur le PEP avec les clés correspondantes sur le PDP pour le PEPID donné. La clé identifiée par l'identifiant de clé a été utilisée pour calculer le résumé de message dans l'objet Intégrité. Toutes les mises en œuvre DOIVENT, au minimum, accepter HMAC-MD5-96, qui est HMAC avec l'algorithme MD5 de résumé de message de la [RFC1321] tronqué à 96 bits pour calculer le résumé de message.

Cet objet comporte aussi un numéro de séquence qui est un entier non signé de 32 bits utilisé pour éviter les attaques en répétition. Le numéro de séquence est initié durant un échange initial de message Client-Ouvert/Client-Accepté et est alors incrémenté de un chaque fois qu'un nouveau message est envoyé sur la connexion TCP dans la même direction. Si le numéro de séquence atteint la valeur de 0xFFFFFFFF, l'incrément suivant va simplement revenir à une valeur de zéro.

Le résumé de longueur variable est calculé sur un message COPS en commençant par l'en-tête COPS jusqu'à l'objet Intégrité (qui DOIT être le dernier objet dans un message COPS) INCLUANT l'en-tête de l'objet Intégrité, l'identifiant de clé, et le numéro de séquence. Le champ Résumé chiffré de message n'est pas inclus au titre du calcul du résumé. Dans le cas de HMAC-MD5-96, HMAC-MD5 va produire un résumé de 128 bits qui est alors tronqué à 96 bits avant d'être mémorisé ou confronté au champ Résumé chiffré de message comme spécifié dans la [RFC2104]. Le résumé chiffré de message DOIT faire 96 bits lorsque HMAC-MD5-96 est utilisé.



2.3 Communication

Le protocole COPS utilise une seule connexion TCP persistante entre le PEP et un PDP distant. Une mise en œuvre de PDP par serveur DOIT écouter sur un numéro d'accès TCP bien connu (COPS = 3288 [IANA]). Le PEP est chargé d'initier la connexion TCP avec un PDP. La localisation du PDP distant peut être configurée, ou obtenue via un mécanisme de localisation de service [RFC2608]. La découverte de service sort cependant du domaine d'application du présent protocole.

Si un seul PEP peut prendre en charge plusieurs types de client, il peut envoyer plusieurs messages Client-Ouvert, chacun spécifiant un type de client particulier à un PDP sur une ou plusieurs connexions TCP. De même, un PDP qui réside à une adresse et numéro d'accès donnés peut prendre en charge un ou plusieurs types de client. Suivant les types de client qu'il prend en charge, un PDP a la capacité d'accepter ou rejeter chaque type de client de façon indépendante. Si un type de client est rejeté, le PDP peut rediriger le PEP sur une adresse de PEP et accès TCP de remplacement pour un certain type de client via COPS. Différents numéros d'accès TCP peuvent être utilisés pour rediriger le PEP sur une autre mise en œuvre de PDP fonctionnant sur le même serveur. Des dispositions supplémentaires pour la prise en charge de plusieurs types de client (peut-être provenant de fabricants de PDP indépendants) sur un seul serveur de PDP distant ne sont pas fournies par le protocole COPS, mais sont plutôt laissées à l'architecture logicielle de cette plateforme de serveur.

Il est possible qu'un seul PEP ait des connexions ouvertes avec plusieurs PDP. C'est le cas lorsque il y a des PDP physiquement différents qui prennent en charge différents types de client, comme indiqué à la figure 2.

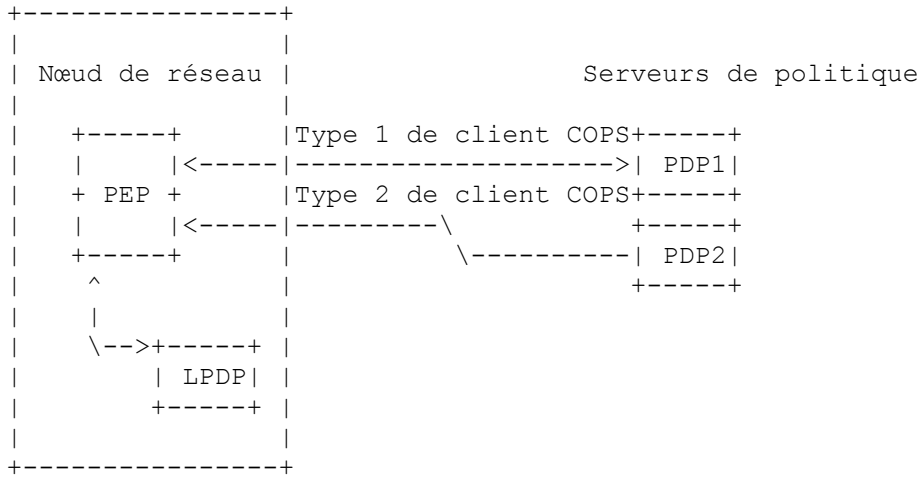


Figure 2 : illustration de multiples PDP

Lorsque une connexion TCP est supprimée ou perdue, on s'attend à ce que le PDP finisse par nettoyer tous les états de demandes en cours qui se rapportent aux échanges de demande/décision avec le PEP. Lorsque le PEP détecte une perte de connexion due à une condition de fin de temporisation, il DEVRAIT envoyer explicitement un message Client-Fermé pour chaque type de client ouvert contenant un objet <Erreur> qui indique le code d'erreur "Échec de communication". De plus, le PEP DEVRAIT tenter continuellement de contacter le PDP principal ou, si il n'y réussit pas, tout PDP de sauvegarde connu. Précisément, le PEP DEVRAIT continuer d'essayer tous les PDP pertinents avec lesquels il a été configuré jusqu'à ce qu'il puisse établir une connexion. Si un PEP est en communication avec un PDP de sauvegarde, le PDP principal devient disponible, le PDP de sauvegarde est chargé de rediriger le PEP sur le PDP principal (via un message <Client-Fermé> contenant un objet <PDPRedirAddr> qui identifie le PDP principal à utiliser pour chaque type de client affecté). Le paragraphe 2.5 donne des détails sur le comportement de synchronisation entre les PEP et les PDP.

2.4 *Utilisation de la manette de c*lient

La manette client est utilisée pour identifier un état de demande unique pour un seul PEP par type de client. Les manettes de client sont choisies par le PEP et sont opaques au PDP. Le PDP utilise simplement la manette de demande pour identifier de façon univoque l'état de la demande pour un type de client particulier sur une connexion TCP particulière et lier de façon générique ses décisions à la demande correspondante. Les manettes de client sont initiées dans les messages de demande et sont ensuite utilisées par les messages de demande, de décision, et de rapport suivants pour faire référence au même état de demande. Lorsque le PEP est prêt à retirer un état de demande local, il va produire un message Supprimer au PDP pour la manette de client correspondante. Une manette DOIT être supprimée explicitement par le PEP avant qu'elle puisse être utilisée par le PEP pour identifier un nouvel état de demande. Les manettes qui se réfèrent à des états de demande différents DOIVENT être uniques au sein du contexte d'une connexion TCP et d'un type de client particuliers.

2.5 Comportement de synchronisation

Lorsque il est déconnecté d'un PDP, le PEP DEVRAIT revenir à des prises de décision locales. Une fois qu'une connexion est rétablie, on attend du PEP qu'il notifie au PDP tous les événements qui ont satisfait au contrôle d'admission local. De plus, le PDP distant peut demander que tous les états internes du PEP soient resynchronisés (toutes les demandes installées précédemment sont à produire à nouveau) en envoyant un message État de synchronisation.

Après une défaillance et avant qu'une nouvelle connexion soit pleinement fonctionnelle, l'interruption de service peut être minimisée si le PEP met en antémémoire les décisions communiquées précédemment et continue de les utiliser pour une durée appropriée. Les règles spécifiques d'un tel comportement seront définies dans les spécifications COPS d'extension de type de client appropriées.

Un PEP qui met en antémémoire l'état d'un échange précédent avec un PDP déconnecté DOIT communiquer ce fait à tout PDP avec lequel il est capable de se reconnecter ultérieurement. Cela se fait en incluant l'adresse et l'accès TCP du dernier PDP pour lequel le PEP met encore l'état en antémémoire dans le message Client-Ouvert. L'objet <LastPDPAddr> ne sera inclus que pour le dernier PDP avec lequel le PEP était en synchronisation complète. Si l'interruption de service était temporaire et si le PDP contient encore l'état complet pour le PEP, le PDP peut choisir de ne pas synchroniser tous les états. Il est encore de la responsabilité du PEP de mettre à jour le PDP de tous les changements d'état qui sont survenus durant l'interruption de service, y compris des états communiqués au précédent PDP qui ont été supprimés après la perte de la connexion. Ceux-ci DOIVENT être explicitement supprimés après qu'une connexion est rétablie. Si le PDP produit une demande de synchronisation, le PEP DOIT passer tous les états en cours au PDP, suivis par un message État de synchronisation achevé (terminant ainsi le processus de synchronisation). Si le PEP a une défaillance et perd tous les états en antémémoire pour un type de client, il va simplement ne pas inclure un <LastPDPAddr> dans son message Client-Ouvert.

3. Contenu du message

La présente section décrit les messages de base échangés entre un PEP et un PDP distant ainsi que leur contenu. Par convention, l'ordre des objets est supposé être celui indiqué dans le BNF *(format Backus-Naur)* pour chaque message COPS sauf notation contraire. L'objet Intégrité, s'il est inclus, DOIT toujours être le dernier objet du message. Si la sécurité est exigée et si un message est reçu sans un objet Intégrité valide, le receveur DOIT envoyer un message Client-Fermé pour le Client-Type=0 spécifiant le code d'erreur approprié.

3.1 Demande (REQ) PEP -> PDP

Le PEP établit une manette de client d'état de demande pour laquelle le PDP distant peut conserver l'état. Le PDP distant utilise alors cette manette pour se référer aux informations échangées et aux décisions communiquées sur la connexion TCP vers un PEP particulier pour un type de client donné.

Une fois qu'une manette à états pleins est établie pour une nouvelle demande, toutes les modifications ultérieures de la demande peuvent être faites en utilisant le message REQ qui spécifie la manette installée précédemment. Le PEP est chargé de notifier au PDP chaque fois que son état local change, afin que l'état du PDP soit capable de refléter précisément l'état du PEP.

Le format du message Demande est le suivant :

```
<Message Demande> ::=  <En-tête commun>
                       <Manette de client>
                       <Contexte>
                       [<IN-Int>]
                       [<OUT-Int>]
                       [<ClientSI(s)>]
                       [<LPDPDecision(s)>]
                       [<Intégrité>]

<ClientSI(s)> ::= <ClientSI> | <ClientSI(s)> <ClientSI>

<LPDPDecision(s)> ::= <LPDPDecision> |
                      <LPDPDecision(s)> <LPDPDecision>

<LPDPDecision> ::= [<Contexte>]
                   <LPDPDecision: Fanions>
                   [<LPDPDecision: Données sans état>]
                   [<LPDPDecision: Données de remplacement>]
                   [<LPDPDecision: Données de ClientSI>]
                   [<LPDPDecision: Données désignées>]
```

L'objet Contexte est utilisé pour déterminer le contexte dans lequel tous les autres objets sont à interpréter. Il est aussi utilisé pour déterminer le genre de décision que le serveur de politique va retourner. Cette décision peut se rapporter au contrôle d'admission, à l'allocation de ressource, à la transmission et substitution d'objet, ou à la configuration.

Les objets Interface sont utilisés pour déterminer l'interface correspondante sur laquelle un message de protocole de signalisation a été reçu ou est sur le point d'être envoyé. Ils sont normalement utilisés si le client est participant sur le chemin d'un protocole de signalisation ou si le client demande des données de configuration pour une interface particulière.

ClientSI, l'objet d'information spécifique du client contient les données spécifiques du type de client pour lequel une décision de politique doit être prise. Dans le cas de configuration, le ClientSI désigné peut inclure des information désignées sur le module, interface, ou fonctionnalité à configurer. L'ordre de plusieurs ClientSI n'est pas important.

Finalement, l'objet LPDPDecision contient les informations qui concernent la décision locale prise par le LPDP.

Les messages de demande mal formés DOIVENT résulter en un message Décision spécifié par le PDP avec le code d'erreur approprié.

3.2 Décision (DEC) PDP -> PEP

Le PDP répond à la REQ par un message DEC qui inclut la manette de client associée et un ou plusieurs objets de décision groupés par rapport à une paire de type d'objet Contexte et d'objet Fanion de décision. Si il y a une erreur de protocole, un objet Erreur est retourné à la place.

Il est exigé que le premier message de décision pour une demande nouvelle/mise à jour ait le fanion de message sollicité établi (valeur = 1) dans l'en-tête COPS. Cela évite le problème de garder la trace de à quelle mise à jour de demande (c'est-à-dire, à quelle demande produite à nouveau pour la même manette) correspond une décision particulière. Il est important que, pour une manette donnée, il y ait au plus une décision sollicitée en cours par demande. Cela signifie essentiellement que le PEP NE DEVRAIT PAS produire plus d'une REQ (pour une manette donnée) avant qu'il reçoive une DEC correspondante avec le fanion de message sollicité établi. Le PDP DOIT toujours produire les décisions pour les demandes sur une manette particulière dans l'ordre dans lequel elles arrivent et toutes les demandes DOIVENT avoir une décision correspondante.

Pour éviter les impasses, le PEP peut toujours mettre une fin de temporisation après avoir produit une demande qui ne reçoit pas de décision. Il DOIT alors supprimer la manette périmée, et peut réessayer en utilisant une nouvelle manette.

Le format du message Décision est le suivant :

```
<Message Décision> ::= <En-tête commun>
                       <Manette de client>
                       <Décision(s)> | <Erreur>
```

[<Intégrité>]

<Décision(s)> ::= <Décision> | <Décision(s)> <Décision>

<Décision> ::= <Contexte>
<Décision: Fanions>
[<Décision: Données sans état>]
[<Décision: Données de remplacement>]
[<Décision: Données de ClientSI>]
[<Décision: Données désignées>]

Le message Décision peut comporter un objet Erreur ou un ou plusieurs contextes plus les objets de décision associés. Les problèmes du protocole COPS sont rapportés dans l'objet Erreur (par exemple une erreur de format de la demande originale y compris des messages de demande mal formés, des objets COPS inconnus dans la Demande, etc.). Le ou les objets Décision applicables dépendent du contexte et du type de client. La seule exigence de rangement pour les objets Décision est que le type d'objet Fanion de décision exigé DOIT précéder les autres types d'objet Décision par lien de contexte.

3.3 État de rapport (RPT) PEP -> PDP

Le message RPT est utilisé par le PEP pour communiquer au PDP la réussite ou l'échec de l'exécution de la décision du PDP, ou de faire rapport d'un changement d'état en rapport avec la comptabilité. Le Type-de-rapport spécifie le genre de rapport et le ClientSI facultatif peut porter des informations supplémentaires par type de client.

Pour chaque message DEC contenant un contexte de configuration qui est reçu par un PEP, le PEP DOIT générer un message État-de-rapport correspondant avec le fanion Message-sollicité établi pour décrire la réussite ou l'échec à appliquer la décision de configuration. De plus, les décisions d'exportation à partir du PDP PEUVENT résulter en un État-de-rapport sollicité correspondant de la part du PEP selon le contexte et le type de client. Les messages RPT sollicités par des décisions pour une certaine manette de client DOIVENT établir le fanion Message-sollicité et DOIVENT être envoyés dans le même ordre que celui de la réception de leur message Décision correspondant. Il NE DOIT jamais y avoir plus d'un message État-de-rapport généré avec le fanion Message-sollicité établi par décision.

L'état de rapport peut aussi être utilisé pour faire des mises à jour périodiques d'informations spécifiques de client pour des besoins de comptabilité et de surveillance d'état, selon le type du client. Dans de tels cas, le type de rapport comptable devrait être spécifié en utilisant l'objet Informations spécifiques de client approprié.

<État-de-rapport> ::== <En-tête commun>
<Manette de client>
<Type-de-rapport>
[<ClientSI>]
[<Intégrité>]

3.4 Supprimer l'état de demande (DRQ) PEP -> PDP

Lorsque il est envoyé du PEP, ce message indique au PDP distant que l'état identifié par la manette de client n'est plus disponible/pertinent. Cette information sera alors utilisée par le PDP distant pour initier les actions d'entretien appropriées. L'objet Code de cause est interprété par rapport au type de client et signifie la raison de la suppression.

Le format du message Supprimer l'état de demande est le suivant :

<Demande de suppression> ::= <En-tête commun>
<Manette de client>
<Raison>
[<Intégrité>]

Étant donnée la nature à états pleins de COPS, il est important que lorsque un état de demande est finalement supprimé du PEP, un message DRQ pour cet état de demande soit envoyé au PDP afin que l'état correspondant puisse être aussi retiré sur le PDP. Les états de demande qui ne sont pas explicitement supprimés par le PEP vont être conservés par le PDP jusqu'à ce que la session de client soit fermée ou que la connexion se termine.

Les messages Décision mal formés DOIVENT déclancher une DRQ spécifiant le code de cause d'erreur approprié

(Mauvais format de message) et tout état associé sur le PEP DEVRAIT soit être supprimé, soit être redemandé. Si une Décision contient un objet Décision COPS inconnu, le PEP DOIT supprimer sa demande en spécifiant le code de cause Objet COPS inconnu parce que le PEP ne sera pas capable de se conformer aux informations contenues dans l'objet inconnu. Dans tous les cas, après avoir produit une DRQ, le PEP peut réessayer la demande correspondante.

3.5 Demande de synchronisation d'état (SSQ) PDP -> PEP

Le format du message Demande de synchronisation d'état est le suivant :

```
<Synchronisation d'état> ::= <En-tête commun>
                [<Manette de client>]
                [<Intégrité>]
```

Ce message indique que le PDP distant souhaite que le client (qui apparaît dans l'en-tête commun) renvoie son état. Si la Manette de client facultative est présente, seul l'état associé à la manette est synchronisé. Si le PEP ne reconnaît pas la manette demandée, il DOIT immédiatement envoyer un message DRQ au PDP pou avoir la manette qui était spécifiée dans le message SSQ. Si aucune manette n'est spécifiée dans le message SSQ, tous les états de client actifs DOIVENT être synchronisés avec le PDP.

Le client effectue une synchronisation d'état en produisant à nouveau des interrogations de demande du type de client spécifié pour l'état existant dans le PEP. Lorsque la synchronisation est terminée, le PEP DOIT produire un message État de synchronisation terminé au PDP.

3.6 Client-Ouvert (OPN) PEP -> PDP

Le message Client-Ouvert peut être utilisé par le PEP pour spécifier au PDP les types de client que le PEP peut prendre en charge, le dernier PDP auquel le PEP s'est connecté pour le type de client donné, et/ou la négociation de caractéristiques spécifiques du client. Un message Client-Ouvert peut être envoyé au PDP à tout moment et plusieurs messages Client-Ouvert pour le même type de client sont permis (au cas de changements d'état global).

```
<Client-Ouvert>  ::= <En-tête commun>
            <PEPID>
            [<ClientSI>]
            [<LastPDPAddr>]
            [<Intégrité>]
```

Le PEPID est un nom symbolique, de longueur variable, qui identifie de façon univoque le client spécifique au PDP (voir au paragraphe 2.2.11).

Un objet ClientSI désigné peut être inclus pour relayer des information globales supplémentaires sur le PEP au PDP lorsque nécessaire (comme spécifié dans le document d'extensions approprié pour le type de client).

Le PEP peut aussi fournir un objet Dernière adresse du PDP dans son message Client-Ouvert spécifiant le dernier PDP (pour le type de client donné) pour lequel il conserve encore des décisions en antémémoire depuis son dernier réamorçage. Un PDP peut utiliser ces informations pour déterminer le comportement de synchronisation approprié (voir au paragraphe 2.5).

Si le PDP reçoit un message Client-Ouvert mal formé, il DOIT générer un message Client-Fermé spécifiant le code d'erreur approprié.

3.7 Client-Accepté (CAT) PDP -> PEP

Le message Client-Accepté est utilisé pour répondre positivement au message Client-Ouvert. Ce message va retourner au PEP un objet Temporisateur qui indique l'intervalle de temps maximum entre les messages Garder-en-vie. Facultativement, un temporisateur spécifiant l'intervalle minimum admis entre les messages de rapport de comptabilité peut être inclus lorsque applicable.

```
<Client-Accepté>  ::= <En-tête commun>
                <Temporisateur KA>
```

[<Temporisateur ACCT>]
[<Intégrité>]

Si le PDP refuse le client, il va à la place produire un message Client-Fermé.

Le temporisateur KA correspond au délai intermédiaire maximum acceptable entre la génération de messages par le PDP et le PEP. La valeur du temporisateur est déterminée par le PDP et est spécifiée en secondes. Une valeur de temporisateur de 0 implique qu'aucune vérification de connexion secondaire n'est nécessaire.

Le temporisateur ACCT facultatif permet au PDP d'indiquer au PEP que des rapports comptables périodiques NE DEVRAIENT PAS excéder l'intervalle spécifié par le temporisateur par manette de client. Cela permet au PDP de contrôler le taux auquel les rapports comptables sont envoyés au PEP (lorsque applicable).

En général, les messages Rapport de type comptable sont envoyés au PDP à un moment déterminé comme approprié par le PEP. Le temporisateur de comptabilité est simplement utilisé par le PDP pour surveiller le taux d'envoi de ces mises à jour (c'est-à-dire, pour empêcher le PEP d'inonder le PDP de rapports comptables). Ne pas inclure cet objet implique qu'il n'y a pas de restriction de la part du PDP sur le taux de génération des mises à jour comptables.

Si le PEP reçoit un message Client-Accepté mal formé, il DOIT générer un message Client-Fermé qui spécifie le code d'erreur approprié.

3.8 Client-Fermé (CC) PEP -> PDP, PDP -> PEP

Le message Client-Fermé peut être produit par le PDP ou par le PEP pour notifier à l'autre qu'un type particulier de client n'est plus pris en charge.

<Client-Fermé> ::= <En-tête commun>
<Erreur>
[<PDPRedirAddr>]
[<Intégrité>]

L'objet Erreur est inclus pour décrire la raison de la fermeture (par exemple, le type de client demandé n'est pas pris en charge par le PDP distant, ou la défaillance du client).

Un PDP PEUT facultativement inclure un objet Adresse de redirection de PDP afin d'informer le PEP d'un PDP de remplacement qu'il DEVRAIT utiliser pour le type de client spécifié dans l'en-tête commun.

3.9 Garder-en-vie (KA) PEP -> PDP, PDP -> PEP

Le message Garder-en-vie DOIT être transmis par le PEP au sein de la période définie par le minimum de toutes les valeurs de temporisateur KA spécifiées dans tous les messages CAT reçus pour la connexion. Un message KA DOIT être généré de façon aléatoire entre 1/4 et 3/4 de cet intervalle minimum de temporisateur KA. Lorsque le PDP reçoit un message Garder-en-vie du PEP, il DOIT faire en écho un Garder-en-vie au PEP. Ce message fait une validation pour chaque côté que la connexion est toujours en fonctionnement même lorsque il n'y a pas d'autre échange de message.

Note : Le type de client dans l'en-tête DOIT toujours être réglé à 0 car le KA est utilisé pour la vérification de la convexion (et non pour la vérification de la session client).

<Garder-en-vie> ::= <En-tête commun>
[<Intégrité>]

Le client et le serveur PEUVENT tous deux supposer que la connexion TCP est insuffisante pour le type de client avec la valeur de temps minimum (spécifiée dans le message CAT) si aucune activité de communication n'est détectée pendant une durée excédant la période de temporisation. Pour le PEP, une telle détection implique que le PDP distant, ou la connexion, est défaillant et le PEP DEVRAIT maintenant tenter d'utiliser un PDP de remplacement/de sauvegarde.

3.10 Synchronisation d'état achevée (SSC) PEP -> PDP

Le message Synchronisation d'état achevée est envoyé par le PEP au PDP après que le PDP a envoyé une Demande de synchronisation d'état au PEP et que le PEP a terminé la synchronisation. Il est utile afin que le PDP sache quand tous les vieux états de client ont été redemandés avec succès, et donc, que le PEP et le PDP sont complètement synchronisés. L'objet Manette de client n'a besoin d'être inclus que si le message Synchronisation d'état correspondant faisait à l'origine référence à une manette spécifique.

<Synchronisation d'état achevée> ::= <En-tête commun>
[<Manette de client>]
[<Intégrité>]

4. Fonctionnement normal

La présente section décrit les échanges normaux entre les serveurs de PDP distant et les clients de PEP.

4.1 Négociation de sécurité et de numéro de séquence

Le message de sécurité COPS est négocié une fois par connexion et couvre toutes les communications sur une connexion particulière. Si le niveau de sécurité COPS est exigé, il DOIT être négocié durant l'échange initial de message Client-Ouvert/Client-Accepté en spécifiant un Type de client de zéro (qui est réservé à la négociation de niveau de sécurité de connexion et à la vérification de la connexion).

Si un PEP n'est pas configuré pour utiliser la sécurité COPS avec un PDP, il va simplement envoyer les messages Client-Ouvert de PDP pour le type de client pris en charge comme spécifié au paragraphe 4.3 et il ne va inclure l'objet Intégrité dans aucun des messages COPS.

Autrement, la sécurité peut être initiée par le PEP si il envoie au PDP un message Client-Ouvert avec un Type de client = 0 avant d'ouvrir aucun autre Type de client. Si le PDP reçoit un Client-Ouvert avec un Type de client = 0 après qu'un autre Type de client a déjà été ouvert avec succès, il DOIT retourner un message Client-Fermé (pour Type de client = 0) à ce PEP. Ce premier message Client-Ouvert DOIT spécifier un Type de client de zéro et DOIT fournir le PEPID et un objet Intégrité COPS. Cet objet Intégrité va contenir le numéro de séquence initial dont le PEP exige que le PDP l'incrémente durant la communication suivante après l'échange initial Client-Ouvert/Client-Accepté et l'identifiant de clé qui identifie l'algorithme et la clé utilisés pour calculer le résumé.

De même, si le PDP accepte la clé et l'algorithme de sécurité du PEP en validant le résumé de message qui utilise la clé identifiée, le PDP DOIT envoyer un message Client-Accepté avec un Type de client de zéro au PEP et portant un objet Intégrité. Cet objet Intégrité va contenir le numéro de séquence initial dont le PDP exige que le PEP l'incrémente durant toute communication ultérieure avec le PDP et l'ID de clé qui identifie la clé et l'algorithme utilisés pour calculer le résumé.

Si le PEP, du point de vue d'un PDP qui exige la sécurité, échoue à, ou n'effectue jamais, la négociation de sécurité en n'envoyant pas un message initial Client-Ouvert avec un Type de client = 0 incluant un objet Intégrité valide, le PDP DOIT envoyer au PEP un message Client-Fermé avec un Type de client = 0 spécifiant le code d'erreur approprié. De même, si le PDP, du point de vue d'un PEP qui exige la sécurité, échoue à la négociation de sécurité en ne renvoyant pas un message Client-Accepté avec un Type de client = 0 incluant une objet Intégrité valide, le PEP DOIT envoyer au PDP un message Client-Fermé avec un Type de client = 0 spécifiant le code d'erreur approprié. Un tel message Client-Fermé n'a pas besoin de porter un objet Intégrité (car la négociation de sécurité n'est pas encore achevée).

L'initialisation de la sécurité peut échouer pour une des raisons suivantes : 1. Le côté qui reçoit le message exige la sécurité de niveau COPS mais un objet Intégrité n'a pas été fourni (Code d'erreur Authentification exigée). 2. Un objet COPS Intégrité a été fourni mais avec un C-Type inconnu/inacceptable (code d'erreur Objet COPS inconnu spécifiant le C-Num et C-Type non acceptés). 3. Le résumé de message ou l'ID de clé dans l'objet Intégrité fourni était incorrect et donc le message n'a pas pu être authentifié en utilisant la clé identifiée (code d'erreur Échec d'authentification).

Une fois que la négociation initiale de sécurité s'est achevée, le PEP va savoir quels numéros de séquence attend le PDP et le PDP va savoir quels numéros de séquence attend le PEP. TOUS les messages COPS doivent alors comporter l'objet Intégrité négocié en spécifiant le numéro de séquence correct avec le résumé de message approprié (incluant les messages Client-Ouvert/Client-Accepté pour les Types de client spécifiés). TOUS les messages suivants du PDP au PEP DOIVENT résulter en l'incrémentation du numéro de séquence fourni par le PEP dans l'objet Intégrité du message Client-Ouvert initial. De même, TOUS les messages suivants du PEP au PDP DOIVENT résulter en l'incrémentation du numéro de

séquence fourni par le PDP dans l'objet Intégrité du message Client-Accepté initial. Les numéros de séquence sont incrémentés de un en commençant par le numéro de séquence initial correspondant. Par exemple, si le numéro de séquence spécifié au PEP par le PDP dans le Client-Accepté initial était 10, le message suivant qu'envoie le PEP au PDP va fournir un objet Intégrité avec un numéro de séquence de 11. Puis le prochain message que le PEP enverra au PDP aura un numéro de séquence de 12 et ainsi de suite. Si un message reçu ultérieurement contient un mauvais numéro de séquence, un ID de clé inconnu, un résumé de message invalide, ou où il manque l'objet Intégrité après que l'intégrité a été négociée, un message Client-Fermé DOIT alors être généré pour le Type de client zéro, contenant un objet Intégrité valide et spécifiant le code d'erreur approprié. La connexion devrait alors être fermée.

4.2 Maintenance des clés

La maintenance des clés sort du domaine d'application du présent document, mais les mises en œuvre de COPS DOIVENT au moins fournir la capacité de configurer manuellement en local les clés et leurs paramètres. La clé utilisée pour produire le résumé de message de l'objet Intégrité est identifiée par le champ ID de clé. Donc, un paramètre ID de clé est utilisé pour identifier une des potentiellement multiples clés simultanées partagées par le PEP et le PDP. Un ID de clé se rapporte à un PEPID particulier sur le PDP ou à un PDP particulier sur le PEP. Chaque clé doit aussi être configurée avec des paramètres de durée de vie pour la période pendant laquelle elle est valide ainsi qu'un paramètre d'algorithme cryptographique associé spécifiant l'algorithme à utiliser avec la clé. Au minimum, toutes les mises en œuvre COPS DOIVENT accepter l'algorithme de chiffrement HMAC-MD5-96 [RFC2104], [RFC1321] pour calculer un résumé de message à inclure dans le Résumé de message chiffré de l'objet Intégrité qui est ajouté au message.

Il est de bonne pratique de changer régulièrement les clés. Les clés DOIVENT être configurables de telle façon que leur durée de vie se chevauche afin de permettre des transitions en douceur entre les clés. Au milieu de la durée du chevauchement des durées de vie entre deux clés, les envoyeurs devraient passer de l'utilisation de la clé actuelle à la prochaine, à la durée de vie plus longue. Pendant ce temps, les receveurs acceptent simplement toute clé identifiée qu'ils reçoivent pendant sa durée de vie configurée et rejettent celles qui ne le sont pas.

4.3 Initialisation de PEP

Un peu après l'établissement d'une connexion entre le PEP et un PDP distant et après la négociation de la sécurité (si nécessaire) le PEP va envoyer un ou plusieurs messages Client-Ouvert au PDP distant, un pour chaque type de client pris en charge par le PEP. Le messager Client-Ouvert DOIT contenir l'adresse du dernier PDP avec lequel le PEP a encore en antémémoire un jeu complet de décisions. Si aucune décision n'est en antémémoire du précédent PDP, l'objet LastPDPAddr NE DOIT PAS être inclus dans le message Client-Ouvert (voir au paragraphe 2.5). Chaque message Client-Ouvert DOIT au moins contenir l'en-tête commun qui note un type client pris en charge par le PEP. Le PDP distant va alors répondre avec des messages Client-Accepté distincts pour chacun des types de client demandés par le PEP et que le PDP peut aussi accepter.

Si un type de client spécifique n'est pas accepté par le PDP, le PDP va plutôt répondre avec un Type de client qui spécifie le type de client qui n'est pas accepté et il va éventuellement suggérer une adresse et accès de PDP de remplacement. Autrement, le PDP va envoyer un Client-Accepté qui spécifie l'intervalle de temps entre les messages Garder-en-vie et le PEP peut alors commencer à produire de requêtes au PDP.

4.4 Opérations d'exportation

Dans le scénario d'exportation, lorsque le PEP reçoit un événement qui exige une nouvelle décision de politique, il envoie un message de demande au PDP distant. Ce qui qualifie spécifiquement un événement pour un type de client particulier DEVRAIT être spécifié dans le document particulier pour ce type de client. Le PDP distant prend alors une décision et renvoie un message de décision au PEP. Comme la demande est à états pleins, la demande sera mémorisée, ou installée, sur le PDP distant. La manette unique (unique par connexion TCP et type de client) spécifiée à la fois dans la demande et dans sa décision correspondante, identifie cet état de demande. Le PEP est chargé de supprimer cet état de demande une fois que la demande n'est plus applicable.

Le PEP peut mettre à jour un état de demande installé antérieurement en produisant à nouveau une demande pour la manette installée précédemment. Le PDP distant est alors supposé prendre une nouvelle décision et renvoyer un message de décision au PEP. De même, le serveur PEUT changer une décision prise antérieurement sur tout état de demande actuellement installé à tout moment en produisant un message de décision non sollicité. À tout moment, le module PEP est supposé être lié par les décisions du PDP et notifier au PDP tout changement d'état.

4.5 Opérations de configuration

Dans le scénario de configuration, comme dans le scénario d'exportation, le PEP va faire une demande de configuration au PDP pour une certaine interface, module, ou fonctionnalité qui peut être spécifié dans l'objet Informations spécifiques du client désigné. Le PDP va alors envoyer éventuellement plusieurs décisions contenant les unités désignées de données de configuration au PEP. Le PEP est supposé installer et utiliser la configuration en local. Une configuration désignée particulière peut être mise à jour en envoyant simplement des messages de décision supplémentaires pour la même configuration désignée. Lorsque le PDP ne souhaite plus que le PEP utilise une partie des informations de configuration, il envoie un message de décision qui spécifie la configuration désignée et un objet Fanions de décision avec la commande de configuration retirée. Le PEP DEVRAIT alors procéder au retrait de la configuration correspondante et envoyer un message de rapport au PDP qui spécifie qu'il a été supprimé.

Dans tous les cas, le PEP PEUT notifier au PDP distant le statut local d'un état installé en utilisant le message de rapport lorsque c'est approprié. Le message de rapport est à utiliser pour signifier quand la facturation peut commencer, quelles actions ont été prises, ou de produire des mises à jour périodiques pour des besoins de surveillance et de comptabilité selon le client. Ce message peut porter des informations spécifiques du client quand nécessaire.

4.6 Opérations de maintien en vie

Le message Garder-en-vie est utilisé pour valider que la connexion entre le client et le serveur fonctionne bien même lorsque il n'y a pas d'autre échange de message du PEP au PDP. Le PEP DOIT générer un message COPS KA de façon aléatoire entre le quart et les trois quarts de l'intervalle minimum de temporisateur KA spécifié par le PDP dans le message Client-Accepté. À réception d'un message Garder-en-vie du PEP, le PDP DOIT alors répondre au message Garder-en-vie en faisant écho au message Garder-en-vie au PEP. Si l'un ou l'autre côté ne reçoit pas un message Garder-en-vie ou un autre message COPS durant l'intervalle minimum de temporisateur KA de la part de l'autre côté, la connexion DEVRAIT être considérée comme perdue.

4.7 Fermeture de PEP/PDP

Finalement, les messages Client-Fermé sont utilisés pour s'opposer aux effets des messages Client-Ouvert correspondants, en notifiant à l'autre côté que le type de client spécifié n'est plus pris en charge/actif. Lorsque le PEP détecte qu'une connexion est perdue à cause d'une condition de fin de temporisation de Garder-en-vie, il DEVRAIT envoyer explicitement un message Client-Fermé pour chaque type de client ouvert en spécifiant un code d'erreur Défaillance de la communication. Le PEP PEUT alors procéder à la clôture de la connexion avec le PDP et tenter de se reconnecter ou essayer un PDP de sauvegarde/de remplacement. Lorsque le PDP ferme, il DEVRAIT aussi envoyer explicitement un Client-Fermé à tous les PEP connectés pour chaque type de client, peut-être en spécifiant un PDP à utiliser en remplacement.

5. Considérations pour la sécurité

Le protocole COPS fournit un objet Intégrité qui peut réaliser l'authentification, l'intégrité du message, et la prévention de répétition. Toutes les mises en œuvre de COPS DOIVENT mettre en œuvre l'objet COPS Intégrité et ses mécanismes, comme décrits dans le présent document. Pour s'assurer que le client (PEP) communique avec le serveur de politique correct (PDP) il faut l'authentification du PEP et du PDP en utilisant un secret partagé, et une preuve cohérente que la connexion reste valide. Le secret partagé exige au moins une configuration manuelle de clés (identifiée par un identifiant de clé) partagée entre le PEP et son PDP. La clé est utilisée en conjonction avec le contenu d'un message COPS pour calculer un résumé de message qui fait partie de l'objet Intégrité. L'objet Intégrité est alors utilisé pour valider tous les messages COPS envoyés sur la connexion TCP entre un PEP et un PDP.

La gestion des clés sort du domaine d'application du présent document au delà des exigences spécifiques exposées au paragraphe 4.2. En général, il est de bonne pratique de changer régulièrement les clés pour entretenir la sécurité. De plus, il est de bonne pratique d'utiliser des clés localisées spécifiques d'un PEP particulier de telle sorte qu'un PEP volé ne compromette pas la sécurité d'un domaine administratif entier.

L'objet COPS Intégrité apporte aussi des numéros de séquence pour éviter les attaques en répétition. Le PDP choisit le numéro de séquence initial pour le PEP et le PEP choisit le numéro de séquence initial pour le PDP. Ces numéros initiaux sont alors incrémentés par chaque message ultérieur envoyé sur la connexion dans la direction correspondante. Les numéros de séquence initiaux DEVRAIENT être choisis de façon à être à croissance monotone et à ne jamais se répéter pour une clé particulière.

La sécurité entre le client (PEP) et le serveur (PDP) PEUT être fournie par la sécurité IP [RFC2401]. Dans ce cas, l'en-tête d'authentification (AH, *Authentication Header*) IPsec DEVRAIT être utilisé pour la validation de la connexion ; de plus, l'encapsulation de la charge utile de sécurité (ESP, *Encapsulation Security Payload*) IPsec PEUT être utilisée pour assurer à la fois la validation et le secret.

La sécurité de la couche transport (TLS, *Transport Layer Security*) [RFC2246] PEUT être utilisée à la fois pour la validation de niveau connexion et pour la confidentialité.

6. Considérations relatives à l'IANA

Le type de client identifie l'application client de politique à laquelle se réfère un message. Les valeurs de type de client dans la gamme 0x0001 à 0x3FFF sont réservées pour l'état Spécification exigée comme défini dans la [RFC2434]. Ces valeurs DOIVENT être enregistrées auprès de l'IANA et leur comportement et leur applicabilité DOIVENT être décrits dans un document d'extension COPS.

Les valeurs de type de client dans la gamme 0x4000 à 0x7FFF sont réservées pour utilisation privée comme défini dans la [RFC2434]. Ces types de client ne sont pas suivis par l'IANA et ne sont pas à utiliser dans les normes ou les produits mis à la disposition du public, car leur unicité ne peut être garantie.

Les valeurs de type de client dans la gamme 0x8000 à 0xFFFF sont au premier utilisateur, comme défini dans la [RFC2434]. Ces types de client sont suivis par l'IANA mais leur publication dans des documents qui décrivent leur utilisation n'est pas obligatoire. L'IANA assure simplement leur unicité.

Les objets du protocole COPS sont identifiés par leurs valeurs de C-Num et de C-Type. Le consensus IETF, comme défini dans la [RFC2434] est requis pour introduire de nouvelles valeurs pour ces numéros, et donc de nouveaux objets dans le protocole COPS de base.

Des objets de contexte supplémentaires R-Types, Reason-Codes, Report-Types, des objets de décision Command-Codes/Flags, et des codes d'erreur PEUVENT être définis pour être utilisés avec de futurs types de client, mais de tels ajouts exigent le consensus de l'IETF, comme défini dans la [RFC2434].

Des objets de contexte M-Types, des sous-codes de cause, et des sous-codes d'erreur PEUVENT être définis par rapport à un type de client particulier suivant les considérations de l'IANA sur leur type de client respectif.

7. Références

[IANA] http://www.isi.edu/in-notes/iana/assignments/port-numbers

[RFC1321] R. Rivest, "Algorithme de résumé de message MD5", avril 1992. *(Information)*

[RFC2104] H. Krawczyk, M. Bellare et R. Canetti, "HMAC : Hachage de clés pour l'authentification de message", février 1997.

[RFC2205] R. Braden, éd., L. Zhang, S. Berson, S. Herzog, S. Jamin, "Protocole de réservation de ressource (RSVP) -- version 1, spécification fonctionnelle", septembre 1997. (*MàJ par* RFC2750, RFC3936, RFC4495) *(P.S.)*

[RFC2209] R. Braden, L. Zhang, "Protocole de réservation de ressource (RSVP) -- version 1 : règles de traitement de message", septembre 1997. *(Information)*

[RFC2215] S. Shenker, J. Wroclawski, "Paramètres généraux de caractérisation pour éléments de réseau à intégration de service", septembre 1997. *(P.S.)*

[RFC2246] T. Dierks et C. Allen, "Protocole TLS version 1.0", janvier 1999.

[RFC2401] S. Kent et R. Atkinson, "Architecture de sécurité pour le protocole Internet", novembre 1998. (*Obsolète, voir* RFC4301)

[RFC2434] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, octobre, 1998. *(Rendue obsolète par la RFC5226)*

[RFC2608] E. Guttman et autres, "Protocole de localisation de service, version 2", juin 1999. (*MàJ par* RFC3224) *(P.S.)*

[RFC2753] R. Yavatkar, D. Pendarakis, R. Guerin, "Cadre pour le contrôle d'admission fondé sur la politique", janvier 2000. *(Info.)*

8. Informations sur les auteurset remerciements

Des remerciement particuliers à Andrew Smith et Timothy O'Malley, les présidents de nos groupes de travail, et à Raj Yavatkar, Russell Fenger, Fred Baker, Laura Cunningham, Roch Guerin, Ping Pan, et Dimitrios Pendarakis. pour leurs précieuses contributions.

Jim Boyle
Level 3 Communications
1025 Eldorado Boulevard
Broomfield, CO 80021
téléphone : 720.888.1192
mél : jboyle@Level3.net

Ron Cohen
CISCO Systems
4 Maskit St.
Herzeliya Pituach 46766 Israel
téléphone : 972.9.9700064
mél : ronc@cisco.com

David Durham
Intel
2111 NE 25th Avenue
Hillsboro, OR 97124
téléphone : 503.264.6232
mél : David.Durham@intel.com

Raju Rajan
AT&T Labs Research
180 Park Ave., P.O. Box 971
Florham Park, NJ 07932
téléphone : 973.360.7229
mél : raju@research.att.com

Shai Herzog
IPHighway, Inc.
55 New York Avenue
Framingham, MA 01701
téléphone : 508.620.1141
mél : herzog@iphighway.com

Arun Sastry
Cisco Systems
4 The Square
Stockley Park
Uxbridge, Middlesex UB11 1BN
UK
téléphone : +44-208-756-8693
mél : asastry@cisco.com

9. Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'éditeur des RFC est actuellement fourni par la Internet Society.