Équipe d'ingénierie de l'Internet (IETF)
Request for Comments : 8166
RFC rendue obsolète : 5666
Catégorie : Sur la voie de la normalisation
ISSN: 2070-1721

C. Lever, éditeur, Oracle
W. Simpson, Red Hat
T. Talpey, Microsoft
juin 2017
Traduction Claude Brière de L'Isle

Transport d'accès direct à la mémoire distante pour l'appel de procédure à distance, version 1

Résumé
Le présent document spécifie un protocole pour porter les messages d'appel de procédure distante (RPC, *Remote Procedure Call*) sur des transports physiques capables d'accès direct à la mémoire distante (RDMA, *Remote Direct Memory Access*). Ce protocole est appelé protocole RPC sur RDMA version 1 dans le présent document. Il n'exige pas de révision des protocoles d'application de RPC ou du protocole RPC lui-même. Le présent document rend obsolète la RFC 5666.

Statut de ce mémoire
Ceci est un document de l'Internet sur la voie de la normalisation.

Le présent document a été produit par l'équipe d'ingénierie de l'Internet (IETF). Il représente le consensus de la communauté de l'IETF. Il a subi une révision publique et sa publication a été approuvée par le groupe de pilotage de l'ingénierie de l'Internet (IESG). Tous les documents approuvés par l'IESG ne sont pas candidats à devenir une norme de l'Internet ; voir la Section 2 de la RFC5741.

Les informations sur le statut actuel du présent document, tout errata, et comment fournir des réactions sur lui peuvent être obtenues à http://www.rfc-editor.org/info/rfc8166

Table des matières

1. Introduction

Le présent document spécifie le protocole RPC sur RDMA version 1, sur la base des mises en œuvre existantes de la RFC 5666 et de l'expérience obtenue par son déploiement. Le présent document rend obsolète la RFC 5666.

La présente spécification précise le texte qui a été l'objet de multiples interprétations et supprime la prise en charge des éléments du protocole RPC sur RDMA version 1 qui n'ont pas été mis en œuvre. Il précise le rôle des liens de couche supérieure (ULB, *Upper-Layer Binding*) et décrit ce qu'ils doivent contenir.

De plus, le présent document décrit les pratiques courantes d'utilisation de RPCSEC_GSS [RFC7861] sur les transports RDMA.

Le numéro de version du protocole n'a pas été changé parce que le protocole spécifiée dans le présent document interopère pleinement avec les mises en œuvre du protocole RPC sur RDMA version 1 spécifié dans la [RFC5666].

1.1 RPC sur transports RDMA

RDMA [RFC5040] [RFC5041] [IBARCH] est une technique pour le déplacement efficace de données entre nœuds d'extrémités. En dirigeant les données dans les mémoires tampons de destination lorsque elles sont envoyées sur un réseau, et en les plaçant via un accès direct en mémoire par le matériel, le double avantage de transferts plus rapides et de frais d'hébergement réduits est obtenu.

Le protocole d'appel de procédure à distance (RPC, *Remote Procedure Call*) de calcul de réseau ouvert (ONC, *Open Network Computing*) (ONC RPC, ou simplement, RPC) [RFC5531] est un protocole d'invocation de procédure à distance qui fonctionne sur divers transports. La plupart des mises en œuvre de RPC utilisent aujourd'hui UDP [RFC768] ou TCP [RFC793]. Sur UDP, les messages RPC sont encapsulés dans des datagrammes, tandis que sur un flux d'octets TCP, les messages RPC sont séparés par un protocole de marquage d'enregistrements. Un transport RDMA convoie aussi les messages RPC d'une façon spécifique qui doit être pleinement décrite si les mises en œuvre de client et de serveur doivent interopérer.

Les transports RDMA présentent une sémantique qui diffère de UDP ou de TCP. Ils conservent les séparations de message comme UDP mais fournissent un transfert de données fiable et en séquence comme TCP. Ils fournissent aussi un service de transfert en vrac hors charge non fourni par UDP ou TCP. Les transports RDMA sont donc vus à juste titre comme un nouveau type de transport par RPC.

Dans ce contexte, les protocoles de système de fichier réseau (NFS, *Network File System*) comme décrits dans les [RFC1094], [RFC1813], [RFC7530], [RFC5661], et les futures versions mineures de NFSv4, sont tous les bénéficiaires évidents des transports RDMA. Une déclaration de problème complète est discutée dans la [RFC5532]. De nombreux autres protocoles fondés sur RPC peuvent aussi en bénéficier.

Bien que le transport RDMA décrit ici fournisse un support relativement transparent pour toute application RPC, le présent document décrit aussi des mécanismes qui peuvent encore plus optimiser le transfert de données, quand les applications de RPC veulent exploiter la capacité d'utilisation de RDMA comme transport.

2. Terminologie

2.1 Langage des exigences

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119], [RFC8174] quand, et seulement quand ils apparaissent tout en majuscules, comme montré ci-dessus.

2.2 RPC

Cette section met en lumière les éléments clés des protocoles RPC [RFC5531] et de représentation des données externes (XDR, *External Data Representation*) [RFC4506] sur lesquels la version 1 de RPC sur RDMA est construite. Une bonne connaissance de ces protocoles est recommandée avant de lire le présent document.

2.2.1 Protocoles de couche supérieure

Les RPC sont une abstraction utilisée pour mettre en œuvre les opérations d'un protocole de couche supérieure (ULP, *Upper-Layer Protocol*). "ULP" se réfère à un couple de programme et version RPC, qui est un ensemble de versions d'appels de procédure qui comporte une seule API bien définie. Un exemple d'ULP est le système de fichier réseau version 4.0 [RFC7530].

Dans le présent document, le terme de "consommateur RPC" se réfère à une mise en œuvre d'un ULP fonctionnant sur un point d'extrémité de client RPC.

2.2.2 Demandeurs et répondeurs

Comme un appel de procédure local, chaque procédure RPC a un ensemble "d'arguments" et un ensemble de "résultats". Un contexte d'appel invoque une procédure, lui passant des arguments, et la procédure retourne ensuite un ensemble de résultats. À la différence d'un appel de procédure local, la procédure appelée est exécutée à distance plutôt que dans le contexte d'exécution de l'application locale.

Le protocole RPC décrit dans la [RFC5531] est fondamentalement un protocole de passation de messages entre un ou plusieurs clients (où fonctionnent des consommateurs RPC) et un serveur (où un contexte d'exécution distante est disponible pour traiter les transactions RPC au nom de ces consommateurs).

Les transactions ONC RPC sont composées de deux types de messages :

CALL : Un "message d'appel RPC" demande qu'un travail soit fait. Ce type de message est désigné par la valeur zéro (0) dans le champ msg_type du message. Une valeur unique arbitraire est placée dans le champ XID du message afin de confronter ce message d'appel RPC à un message de réponse RPC correspondant.

REPLY : un "message de réponse RPC" rapporte le résultat du travail demandé par un message d'appel RPC. Un message de réponse RPC est désigné par la valeur un (1) dans le champ msg_type du message. La valeur contenue dans un champ XID d'un message de réponse RPC est copiée du message d'appel RPC dont le résultat est rapporté.

Le point d'extrémité de client RPC agit comme "demandeur". Il met à la suite les arguments de la procédure et les apporte à un point d'extrémité de serveur via un message d'appel RPC. Ce message contient un en-tête de protocole RPC, qui décrit l'opération de couche supérieure demandée, et tous les arguments.

Le point d'extrémité de serveur RPC agit comme "répondeur". Il désérialise les arguments et traite l'opération demandée. Il sérialise ensuite le résultat de l'opération dans un autre flux d'octets. Ce flux d'octets est retransporté au demandeur via un message de réponse RPC. Ce message contient un en-tête de protocole RPC, qui décrit la réponse de couche supérieure, et tous les résultats.

Le demandeur désérialise le résultat et permet à l'appelant original de poursuivre. À ce point, la transaction RPC désignée par le XID dans le message d'appel RPC est achevée, et le XID est retiré.

En résumé, les messages d'appel RPC sont envoyés par les demandeurs aux répondeurs pour initier des transactions RPC. Les messages de réponse RPC sont envoyés par les répondeurs aux demandeurs pour achever le traitement sur une transaction RPC.

2.2.3 Transports RPC

Le rôle d'un "transport RPC" est de servir d'intermédiaire dans l'échange de messages RPC entre demandeurs et répondeurs. Un transport RPC bouche le trou entre l'abstraction de message RPC et les opérations natives d'un transport de réseau particulier.

RPC sur RDMA est un transport RPC en mode connexion. Quand un transport en mode connexion est utilisé, les clients initient des connexions de transport, tandis que les serveurs attendent passivement les demandes de connexion entrantes.

2.2.4 Représentation des données externes

On ne peut pas supposer que tous les demandeurs et répondeurs représentent les objets de données de la même façon en interne. RPC utilise la représentation de données externes (XDR, *External Data Representation*) pour traduire les types de données natives et mettre à la suite les arguments et résultats [RFC4506].

Le protocole XDR code les données indépendamment de leur "boutiennetée" ou de la taille des types de données de l'hôte d'origine, permettant un décodage sans ambiguïté des données à l'extrémité de réception. Les programmes RPC sont spécifiés en écrivant une définition XDR de leurs procédures, types de données d'argument, et types de données de résultat.

XDR suppose que le nombre de bits dans un octet et leur ordre sont les mêmes sur les deux points d'extrémité et sur le réseau physique. La plus petite unité indivisible de codage XDR est un groupe de quatre octets. XDR aplatit aussi les listes, les matrices, et autres types de données complexes afin qu'ils puissent être portés comme un flux d'octets.

Un flux d'octets mis en série qui est le résultat du codage XDR est appelé un "flux XDR". Un point d'extrémité envoyeur code les données natives dans un flux XDR et transmet ensuite ce flux à un receveur. Un point d'extrémité receveur décode les flux d'octets XDR entrants dans leur format de représentation des données natif.

2.2.4.1 Données XDR opaques

Parfois, un élément de données doit être transféré tel quel : sans codage ni décodage. Le contenu d'un tel élément de données est appelé "données opaques". Le codage XDR place le contenu des éléments de données opaques directement dans un flux XDR sans l'altérer d'aucune façon. Les ULP ou les applications effectuent dans ce cas toutes les traductions de données nécessaires. Des exemples d'éléments de données opaques incluent le contenu de fichiers ou de chaîne d'octets génériques.

2.2.4.2 Arrondi XDR

Le nombre d'octets dans un élément de données de longueur variable précède cet élément dans un flux XDR. Si la taille d'un élément de données codé n'est pas un multiple de quatre octets, des octets contenant des zéros sont ajoutés après la fin de l'élément ; c'est pour que le prochain élément de données codé dans le flux XDR commence sur une frontière de quatre octets. La taille codée de l'élément n'est pas changée par l'ajout des octets supplémentaires. Ces octets supplémentaires ne

sont jamais exposés aux ULP.

Cette technique est appelée "arrondi XDR", et les octets supplémentaires sont appelés le "bourrage d'arrondi XDR".

2.3 RDMA

Les demandeurs et répondeurs RPC peuvent être rendus plus efficaces si de grands messages RPC sont transférés par un tiers, comme un matériel d'interface réseau intelligente (déchargement du mouvement des données) et placés dans la mémoire du receveur afin qu'aucun ajustement supplémentaire de l'alignement des données ne soit à faire (placement direct des données (DDP, *direct data placement*). Les transports RDMA permettent les deux optimisations.

2.3.1 DDP

Normalement, les mises en œuvre RPC copient le contenu des messages RPC dans une mémoire tampon avant leur envoi. Une mise en œuvre efficace de RPC envoie les données en vrac sans les copier d'abord dans une mémoire tampon d'envoi séparée.

Cependant, les mises en œuvre de RPC fondées sur une prise sont souvent incapables de recevoir directement les données dans leur place finale en mémoire. Les receveurs ont souvent besoin de copier les données entrantes pour finir une opération RPC : parfois seulement pour ajuster l'alignement des données.

Dans le présent document, "RDMA" se réfère au mécanisme physique qu'un transport RDMA utilise quand il déplace les données. Bien que cela puisse n'être pas efficace, avant un transfert RDMA, un envoyeur peut copier les données dans une mémoire tampon intermédiaire. Après un transfert RDMA, un receveur peut copier à nouveau ces données dans leur destination finale.

Dans le présent document, le terme "DDP" se réfère à tout transfert de données optimisé où il n'est pas nécessaire que la CPU d'un hôte receveur copie les données transférées dans une autre localisation après leur réception.

Comme le faisait la [RFC5666], le présent document se concentre sur l'utilisation des opérations RDMA Read et Write pour réaliser le déchargement du mouvement des données et le DDP. Cependant, tous les transferts de données fondés sur RDMA ne se qualifient pas comme DDP, et DDP peut être réalisé en utilisant des mécanismes non RDMA.

2.3.2 Exigences du transport RDMA

Pour réaliser de bonnes performances durant les opérations de réception, les transports RDMA exigent que les consommateurs RDMA provisionnent des ressources à l'avance de la réception des messages entrants.

Un consommateur RDMA pourrait fournir des mémoires tampons de réception à l'avance en envoyant une demande de travail de RDMA Receive pour chaque RDMA Send attendu d'un homologue distant. Ces mémoires tampons sont fournies avant que l'homologue distant envoie les demandes de travail RDMA Send ; donc, celles-ci sont souvent appelées des mémoires tampons "pré allouées".

Une demande de travail RDMA Receive reste en instance jusqu'à ce que le matériel la confronte à une opération Send entrante. Les ressources associées à ce Receive doivent être conservées dans la mémoire de l'hôte, ou "épinglées", jusqu'à ce que le Receive s'achève.

Étant donnés ces éléments de base de l'opération de transport RDMA, le protocole RPC sur RDMA version 1 suppose que chaque transport fournit les opérations abstraites qui suivent. Une discussion plus complète de ces opérations se trouve dans la [RFC5040].

Mémoire enregistrée : la mémoire enregistrée est une région de mémoire à qui est allouée une étiquette de pilotage qui permet temporairement l'accès par le fournisseur RDMA pour effectuer les opérations de transfert des données. Le protocole RPC sur RDMA version 1 suppose que chaque région de mémoire enregistrée DOIT être identifiée par une étiquette de pilotage de pas moins de 32 bits et des adresses de mémoire de jusqu'à 64 bits de long.

RDMA Send : le fournisseur RDMA prend en charge une opération RDMA Send, avec l'achèvement signalé à l'homologue receveur après que les données ont été placées dans une mémoire tampon pré-allouée. Les envois se terminent chez le receveur dans l'ordre de leur production chez l'envoyeur. La quantité de données transférée par une seule opération

RDMA Send est limitée seulement par la taille des mémoires tampons pré allouées de l'homologue distant.

RDMA Receive : le fournisseur RDMA prend en charge une opération RDMA Receive pour recevoir des données portées par les opérations RDMA Send entrantes. Pour réduire la quantité de mémoire qui doit rester épinglée en attendant les envois entrants, la quantité de mémoire pré allouée est limitée. Le contrôle de flux pour empêcher de submerger les ressources du receveur sont fournies par le consommateur RDMA (dans ce cas, le protocole RPC sur RDMA version 1).

RDMA Write : le fournisseur RDMA prend en charge une opération RDMA Write pour placer directement les données dans une région de mémoire tampon. L'hôte local initie un RDMA Write, et l'achèvement lui est signalé. Aucun achèvement n'est signalé sur l'homologue distant. L'hôte local fournit une étiquette de pilotage, une adresse de mémoire, et la longueur de la région de mémoire de l'homologue distant. Les RDMA Write ne sont pas ordonnés les uns par rapport aux autres, mais sont ordonnés par rapport aux RDMA Send. L'achèvement d'un RDMA Send suivant obtenu de l'initiateur de l'écriture garantit que les données du RDMA Write antérieur ont bien été placées dans la mémoire de l'homologue distant.

RDMA Read : le fournisseur RDMA prend en charge une opération RDMA Read pour placer directement les données de source de l'homologue dans la mémoire de l'initiateur de la lecture. Un RDMA Read est initié par l'hôte local et l'achèvement lui est signalé. L'hôte local fournit des étiquettes de pilotage, des adresses de mémoire, et une longueur pour la région de mémoire de source distante et de destination locale. L'hôte local signale l'achèvement de la lecture à l'homologue distant au titre d'un message RDMA Send suivant. L'homologue distant peut alors libérer les étiquettes de pilotage et ensuite les régions de mémoires de source associées.

Le protocole RPC sur RDMA version 1 est destiné à être porté sur les transports RDMA qui prennent en charge les opérations abstraites ci-dessus. Ce protocole porte à l'homologue RPC des informations suffisantes pour qu'un fournisseur RDMA effectue les transferts contenant les données de RPC et communique leurs résultats.

3. Cadre du protocole RPC sur RDMA

3.1 Modèles de transfert

Un "modèle de transfert" désigne quel point d'extrémité expose sa mémoire et lequel est chargé d'initier le transfert des données. Pour permettre les opérations RDMA Read et Write, par exemple, un point d'extrémité expose d'abord les régions de sa mémoire à un point d'extrémité distant, qui initie ces opérations sur la mémoire exposée.

Read-Read : les demandeurs exposent leur mémoire au répondeur, et le répondeur expose sa mémoire aux demandeurs. Le répondeur lit, ou tire, les arguments RPC ou les appels RPC entiers provenant de chaque demandeur. Les demandeurs tirent les résultats RPC ou tous les restes RPC provenant du répondeur.

Write-Write : les demandeurs exposent leur mémoire au répondeur, et le répondeur expose sa mémoire aux demandeurs. Les demandeurs écrivent, ou poussent, les arguments RPC ou les appels RPC entiers au répondeur. Le répondeur pousse les résultats RPC ou tous les restes RPC à chaque demandeur.

Read-Write : les demandeurs exposent leur mémoire au répondeur, mais le répondeur n'expose pas sa mémoire. Le répondeur tire les arguments RPC ou les appels RPC entiers de chaque demandeur. Le répondeur pousse les résultats RPC ou tous les restes RPC à chaque demandeur.

Write-Read : le répondeur expose sa mémoire aux demandeurs, mais les demandeurs n'exposent pas leur mémoire. Les demandeurs poussent les arguments RPC ou les appels RPC entiers au répondeur. Les demandeurs poussent les résultats RPC ou tous les restes RPC provenant du répondeur.

3.2 Tramage de message

Sur un transport RPC sur RDMA, chaque message RPC est encapsulé par un message RPC sur RDMA. Un message RPC sur RDMA consiste en deux flux XDR.

Flux de charge utile RPC : le "flux de charge utile" contient le message RPC encapsulé en cours de transfert par ce message RPC sur RDMA. Ce flux commence toujours par le champ Identifiant de transaction (XID, *Transaction ID*) du message RPC encapsulé.

Flux de transport : le "flux de transport" contient un en-tête qui décrit et contrôle le transfert du flux de charge utile dans ce message RPC sur RDMA. Cet en-tête est analogue au marquage d'enregistrement utilisé pour RPC sur les prises TCP mais est plus coûteux, car les transports RDMA prennent en charge plusieurs modes de transfert de données.

Dans sa forme la plus simple, un message RPC sur RDMA consiste en un flux de transport suivi immédiatement par un flux de charge utile, portés ensemble dans un seul RDMA Send. Pour transmettre de grands messages RPC, une combinaison d'une opération RDMA Send et d'une ou plusieurs autres opérations RDMA est employée.

Le tramage RPC sur RDMA remplace tous les autres tramages RPC (comme le marquage d'enregistrement TCP) quand il est utilisé par dessus une association RPC sur RDMA, même quand le protocole RDMA sous-jacent peut lui-même être mis en couche par dessus un transport avec un tramage RPC défini (comme TCP).

Cependant, il est possible que RPC sur RDMA soit activé de façon dynamique dans le cours d'une négociation de l'utilisation de RDMA via un échange d'ULP. Parce que le tramage RPC délimite une demande ou réponse RPC entière, le glissement de tramage résultant doit se produire entre des messages RPC distincts, et de concert avec le transport sous-jacent.

3.3 Gestion des ressources du receveur

Il est critique de fournir le contrôle de flux de RDMA Send pour une connexion RDMA. Si une des mémoires tampons de réception pré allouées sur la connexion n'est pas assez grande pour accepter un envoi RDMA entrant, ou si une mémoire tampon de réception pré allouée n'est pas disponible pour accepter un envoi RDMA entrant, la connexion RDMA peut être terminée. Ceci est différent du réseautage TCP/IP conventionnel, dans lequel les mémoires tampons sont allouées dynamiquement lorsque des messages sont reçus.

La longévité d'une connexion RDMA rend obligatoire que les points d'extrémité envoyeurs respectent les limites de ressources des homologues receveurs. Pour assurer que les messages peuvent être envoyés et reçus fiablement, il y a deux paramètres opérationnels pour chaque connexion.

3.3.1 Crédits RPC sur RDMA

Le contrôle de flux pour les opérations d'envoi RDMA dirigées sur le répondeur est mis en œuvre comme un simple protocole de demande/allocation dans l'en-tête RPC sur RDMA associé à chaque message RPC.

Un crédit RPC sur RDMA version 1 est la capacité de traiter une transaction RPC sur RDMA. Chaque message RPC sur RDMA envoyé de demandeur à répondeur demande un certain nombre de crédits de la part du répondeur. Chaque message RPC sur RDMA envoyé du répondeur au demandeur informe le demandeur du nombre de crédits que le répondeur a accordé. Les valeurs demandées et accordées sont portées dans le champ rdma_credit de chaque message RPC sur RDMA (voir le paragraphe 4.2.3).

En pratique, la valeur critique est la valeur accordée. Un demandeur NE DOIT PAS envoyer des demandes non acquittées au delà de la limite de crédit accordée par le répondeur. Si la valeur accordée est dépassée, la couche RDMA peut signaler une erreur, terminant éventuellement la connexion. La valeur accordée NE DOIT PAS être zéro, car une telle valeur pourrait résulter en une impasse.

Les appels RPC s'achèvent dans n'importe quel ordre, mais la limite de crédit accordé en cours au répondeur est connue du demandeur à partir des propriétés d'ordre de RDMA Send. Le nombre de nouvelles demandes admises que le demandeur peut envoyer est alors la plus faible des valeurs de crédit demandé et alloué en cours, moins le nombre de demandes en vol. Les valeurs de crédit annoncées ne sont pas altérées quand des RPC individuels commencent ou se terminent.

Les valeurs de crédit demandées et allouées PEUVENT être ajustées pour correspondre aux besoins ou aux politiques en vigueur sur l'un ou l'autre homologue. Par exemple, un répondeur peut réduire la valeur de crédit accordé pour s'accommoder des ressources disponibles dans une file d'attente de réception partagée. Le répondeur DOIT s'assurer qu'une augmentation des ressources de réception est effectuée avant que le prochain message de réponse RPC soit envoyé.

Un demandeur DOIT maintenir assez de ressources de réception pour s'accommoder des réponses attendues. Les répondeurs doivent être prêts à ce qu'il n'y ait pas de ressources de réception disponibles chez les demandeurs qui n'ont pas de transaction RPC en instance.

Certaines mises en œuvre de RDMA peuvent imposer des restrictions de contrôle de flux supplémentaires, comme des limites sur les opérations RDMA Read en cours chez le répondeur. S'accommoder de telles restrictions est considéré comme étant de la responsabilité de chaque mise en œuvre de RPC sur RDMA version 1.

3.3.2 Seuil en ligne

Une valeur de "seuil en ligne" est la plus grande taille de message (en octets) qui peut être convoyée dans une direction entre des mises en œuvre d'homologues utilisant des RDMA Send et Receive. La valeur de seuil en ligne est le plus petit du plus grand nombre d'octets que l'envoyeur peut envoyer via une seule opération RDMA Send et le plus grand nombre d'octets que le receveur peut accepter via une seule opération RDMA Receive. Chaque connexion a deux valeurs de seuil en ligne : une pour les messages qui s'écoulent du demandeur au répondeur (appelée le "seuil d'appel en ligne") et une pour les messages qui s'écoulent de répondeur à demandeur (appelée le "seuil de réponse en ligne").

À la différence des limites de crédit, les valeurs de seuil en ligne ne sont pas annoncées aux homologues via le protocole RPC sur RDMA version 1, et il n'y a aucune disposition pour changer les valeurs de seuil en ligne pendant la durée de vie d'un RPC sur une connexion RDMA version 1.

3.3.3 État de connexion initial

La première fois qu'une connexion est établie, les homologues peuvent ne pas savoir quelles ressources de réception a l'autre, ni quels sont les seuils en ligne de l'autre homologue.

Comme base d'un échange initial de demandes RPC, chaque connexion RPC sur RDMA version 1 fournit la capacité d'échanger au moins un message RPC à la fois, dont les messages RPC d'appel et de réponse ne font pas plus de 1024 octets. Un répondeur PEUT excéder ce niveau de base de configuration, mais un demandeur NE DOIT PAS supposer que plus d'un crédit est disponible et DOIT recevoir une réponse valide du répondeur portant le nombre réel de crédits disponibles, avant d'envoyer sa demande suivante.

Les mises en œuvre de receveur DOIVENT prendre en charge des seuils en ligne de 1024 octets mais PEUVENT prendre en charge de plus grandes valeurs de seuils en ligne. Un mécanisme indépendant pour découvrir les seuils en ligne d'un homologue avant l'établissement d'une connexion peut être utilisé pour optimiser l'utilisation des opérations RDMA Send et Receive. En l'absence d'un tel mécanisme, les envoyeurs et les receveurs DOIVENT supposer que les seuils en ligne sont de 1024 octets.

3.4 Codage XDR avec tronçons

Quand une capacité DDP est disponible, le transport place le contenu d'un ou plusieurs éléments de données XDR directement dans la mémoire du receveur, séparément du transfert des autres parties du flux XDR contenant.

3.4.1 Réduction d'un flux XDR

RPC sur RDMA version 1 fournit un mécanisme pour déplacer une partie d'un message RPC via un transfert de données distinct d'une paire Send/Receive RDMA. L'envoyeur supprime un ou plusieurs éléments de données XDR du flux de charge utile. Ils sont portés via d'autres mécanismes, comme une ou plusieurs opérations RDMA Read ou Write. Lorsque le receveur décode un message entrant, il saute directement aux éléments de données placés.

La portion d'un flux XDR qui est partagée et déplacée séparément est appelée un "tronçon". Dans certains contextes, les données dans un en-tête RPC sur RDMA qui décrivent ces régions séparées de mémoire peuvent aussi être appelées des "tronçons".

Un flux de charge utile après que les tronçons ont été supprimés est appelé un flux de charge utile "réduit". De même, un élément de données qui a été retiré d'un flux de charge utile pour être transféré séparément est appelé un élément de données "réduit".

3.4.2 Éligibilité à DDP

Tous les éléments de données XDR ne bénéficient pas de DDP. Par exemple, les petits éléments de données ou les éléments de données qui exigent un changement de l'ordre XDR par le receveur ne bénéficient pas de DDP. De plus, il est

impraticable aux receveurs de préparer chaque élément de données XDR possible dans un protocole à être transféré dans un tronçon.

Pour maintenir l'interopérabilité sur un transport RPC sur RDMA, on doit déterminer quels éléments de données XDR dans chaque ULP peuvent utiliser DDP.

Ceci est fait par des spécifications supplémentaires qui décrivent comment les ULP emploient DDP. Une "spécification d'ULB" identifie quels éléments de données XDR individuels spécifiques dans un ULP PEUVENT être transférés via DDP. De tels éléments de données sont appelés "éligibles à DDP". Tous les autres éléments de données XDR NE DOIVENT PAS être réduits.

Les exigences détaillées pour les ULB sont données à la Section 6.

3.4.3 Segments RDMA

Quand il code un flux de charge utile qui contient un élément de données éligible à DDP, un envoyeur peut choisir de réduire cet élément de données. Quand il choisit de le faire, l'envoyeur ne place pas l'élément dans le flux de charge utile. À la place, l'envoyeur enregistre dans l'en-tête RPC sur RDMA la localisation et la taille de la région de mémoire contenant cet élément de données.

Le demandeur fournit les informations de localisation pour les éléments de données éligibles à DDP dans les messages RPC d'appel et de réponse. Le répondeur utilise ces informations pour restituer les arguments contenus dans la région spécifiée de la mémoire du demandeur ou place les résultats dans cette région de mémoire.

Un "segment RDMA", ou "segment complet", est un objet de données d'en-tête de transport RPC sur RDMA qui contient les coordonnées précises d'une région de mémoire contiguë qui doit être convoyée séparément du flux de charge utile. Les segments complets contiennent les informations suivantes :

Bride : étiquette de pilotage (STag, *Steering tag*) ou R_key générée en enregistrant cette mémoire auprès du fournisseur RDMA.

Longueur : longueur de la région de mémoire du segment RDMA, en octets. Un "segment vide" est un segment RDMA avec la valeur zéro (0) dans son champ de longueur.

Décalage : le décalage ou le début de l'adresse de mémoire de la région de mémoire du segment RDMA.

Voir la discussion dans la [RFC5040].

3.4.4 Tronçons

Dans RPC sur RDMA version 1, un "tronçon" se réfère à une portion du flux de charge utile qui est déplacée indépendamment de l'en-tête de transport RPC sur RDMA et du flux de charge utile. Les données de tronçon sont supprimées du flux de charge utile de l'envoyeur, transférées via des opérations séparées, et ensuite réinsérées dans le flux de charge utile du receveur pour former un message RPC complet.

Chaque tronçon est composé de segments RDMA. Chaque segment RDMA représente une seule pièce contiguë de ce tronçon. Un demandeur PEUT diviser un tronçon en segments RDMA en utilisant les frontières qui conviennent. La longueur d'un tronçon est la somme des longueurs des segments RDMA qui le composent.

Le protocole de transport RPC sur RDMA version 1 ne fixe pas de limite à la taille de tronçon. Cependant, chaque ULP peut encadrer la quantité de données qui peut être transférée par un seul RPC (par exemple, NFS a "rsize" et "wsize", qui restreignent la taille de charge utile des opérations NFS READ et WRITE). Le répondeur peut utiliser de telles limites pour effectuer des vérifications de taille de tronçon avant de les utiliser dans les opérations RDMA.

3.4.4.1 Dispositifs comptés

Si un tronçon contient un type de données d'un dispositif compté, le compte des éléments du dispositif DOIT rester dans le flux de charge utile, tandis que les éléments du dispositif DOIVENT être déplacés au tronçon. Par exemple, quand on code un dispositif opaque d'octets comme un tronçon, le compte d'octets reste dans le flux de charge utile, tandis que les octets

dans le dispositif sont retirés du flux de charge utile et transférés dans le tronçon.

Les éléments individuels de dispositif apparaissent dans un tronçon dans leur entièreté. Par exemple, quand on code un dispositif de dispositifs comme un tronçon, le compte des éléments dans le dispositif enclosant reste dans le flux de charge utile, mais chaque dispositif enclos, incluant son compte d'éléments, est transféré au titre du tronçon.

3.4.4.2 Données facultatives

Si un tronçon contient un type de données facultatif, le champ "is present" DOIT rester dans le flux de charge utile, tandis que les données, si elles sont présentes, DOIVENT être déplacées au tronçon.

3.4.4.3 Unions XDR

Un type de données union NE DOIT PAS être rendu éligible à DDP, mais une ou plusieurs de ses branches PEUVENT être éligibles à DDP, sous réserve des autres exigences de ce paragraphe.

3.4.4.4 Arrondi de tronçon

Sauf dans des cas particuliers (traités au paragraphe 3.5.3) un tronçon DOIT contenir exactement un élément de données XDR. Cela rend directe la réduction des éléments de données de longueur variable sans affecter l'alignement XDR des éléments de données dans le flux de charge utile.

Quand un élément de données XDR de longueur variable est réduit, l'envoyeur DOIT supprimer le bourrage d'arrondi XDR pour cet élément de données du flux de charge utile afin que ces éléments de données restants dans le flux de charge utile commencent sur un alignement de quatre octets.

3.4.5 Tronçons en lecture

Un "tronçon en lecture" représente un élément de données XDR qui est à tirer du demandeur au répondeur.

Un tronçon en lecture est une liste d'un ou plusieurs segments RDMA en lecture. Un segment RDMA en lecture consiste en un champ Position suivi par un segment complet. Voir les détails au paragraphe 4.1.2.

Position : le décalage d'octets dans le flux de charge utile non réduit où le receveur réinsère l'élément de données porté dans un tronçon. La valeur de Position DOIT être calculée à partir du début du flux de charge utile non réduit, qui commence à la position zéro. Tous les segments RDMA en lecture qui appartiennent au même segment Read ont la même valeur dans leur champ Position.

Quand il construit un message d'appel RPC, un demandeur enregistre les régions de mémoire qui contiennent les données à transférer via les opérations RDMA Read. Il annonce les coordonnées de ces régions dans l'en-tête de transport RPC sur RDMA du message RPC Call.

Après la réception d'un message RPC Call envoyé via une opération RDMA Send, un répondeur transfère les données du tronçon provenant du demandeur en utilisant des opérations RDMA Read. Le répondeur reconstruit les données du tronçon transféré en enchaînant le contenu de chaque segment RDMA, dans l'ordre de la liste, dans le flux de charge utile reçu à la valeur de Position enregistrée dans ce segment RDMA.

Dit autrement, le répondeur insère le premier segment RDMA dans un tronçon Read dans le flux de charge utile au décalage d'octet indiqué par son champ Position. Les segments RDMA dont la valeur du champ Position correspond à ce décalage d'octet sont ensuite enchaînés, jusqu'à ce qu'il n'y ait plus de segment RDMA à cette valeur de Position.

Le champ Position dans un segment de lecture indique où commence le tronçon Read contenant dans le flux de charge utile. La valeur dans ce champ DOIT être un multiple de quatre. Tous les segments dans le même tronçon Read partagent la même valeur de Position, même si un ou plusieurs des segments RDMA ont une longueur non alignée sur quatre octets.

3.4.5.1 Décodage des tronçons en lecture

Lorsque il décode un flux de charge utile reçu, chaque fois que le décalage XDR dans le flux de charge utile correspond à

celui d'un tronçon Read, le répondeur initie un RDMA Read pour tirer le contenu de données du tronçon dans la mémoire locale enregistrée.

Le répondeur accuse réception de l'achèvement de l'utilisation des mémoires tampons de source du tronçon Read quand il envoie un message RPC Reply au demandeur. Le demandeur peut alors libérer les tronçons Read annoncés dans la demande.

3.4.5.2 Arrondi de tronçon en lecture

Quand il réduit un élément de données d'argument de longueur variable, le demandeur NE DEVRAIT PAS inclure le bourrage d'arrondi XDR de l'élément de données dans le tronçon. La longueur d'un tronçon Read est déterminée comme suit :

- Si le demandeur choisit d'inclure du bourrage d'arrondi dans un tronçon Read, la longueur totale du tronçon DOIT être la somme de la longueur codée de l'élément de données et de la longueur du bourrage d'arrondi. La longueur de l'élément de données qui a été codé dans le flux de charge utile reste inchangée. L'envoyeur peut accroître la longueur du tronçon en ajoutant un autre segment RDMA contenant seulement le bourrage d'arrondi, ou il peut le faire en étendant le segment final RDMA dans le tronçon.

- Si l'envoyeur choisit de ne pas inclure de bourrage d'arrondi dans le tronçon, la longueur totale du tronçon DOIT être la même que la longueur codée de l'élément de données.

3.4.6 Tronçons en écriture

Lorsque il construit un message RPC Call, un demandeur prépare des régions de mémoire dans lesquelles recevoir les éléments de données de résultat éligibles à DDP. Un "tronçon en écriture" représente un élément de données XDR qui doit être poussé d'un répondeur à un demandeur. Il est constitué d'un dispositif de zéro, un ou plusieurs segments complets.

Les tronçons en écriture sont provisionnés par un demandeur longtemps avant que le répondeur ait préparé le flux de charge utile de réponse. Un demandeur ne connaît souvent pas la longueur réelle des éléments de données de résultat à retourner, car le résultat n'existe pas encore. Donc, il DOIT enregistrer des tronçons en écriture assez longs pour s'accommoder de la taille maximum possible de chaque élément de données retourné.

De plus, la position XDR des éléments de données éligibles à DDP dans le flux de charge utile de la réponse n'est pas prévisible quand un demandeur construit un message d'appel RPC. Donc, les segments RDMA dans un tronçon en écriture n'ont pas de champ Position.

Pour chaque tronçon en écriture fourni par un demandeur, le répondeur pousse un élément de données au demandeur, remplissant le tronçon de façon contiguë et dans l'ordre du dispositif de segments jusqu'à ce que cet élément de données ait été complètement écrit chez le demandeur. Le répondeur DOIT copier le compte de segments et tous les segments à partir du tronçon en écriture fourni par le demandeur dans l'en-tête de transport du message de réponse RPC. Lorsque il fait cela, le répondeur met à jour chaque champ de longueur de segment pour refléter la quantité réelle de données qui est retournée dans ce segment. Le répondeur envoie alors le message de réponse RPC via une opération RDMA Send.

Un "tronçon en écriture vide" est un tronçon en écriture avec un compte de segment de zéro. Par définition, la longueur d'un tronçon en écriture vide est zéro. Un "tronçon en écriture inutilisé" a un compte de segment non de zéro, mais tous ses segments sont vides.

3.4.6.1 Décodage des tronçons en écriture

Après la réception du message de réponse RPC, le demandeur reconstruit les données transférées en enchaînant le contenu de chaque segment, dans l'ordre du dispositif, dans le flux XDR du message RPC Reply à la position XDR connue de l'élément associé de données de résultat éligible à DDP.

3.4.6.2 Arrondi de tronçon en écriture

Quand il provisionne un tronçon en écriture pour un élément de données de résultat de longueur variable, le demandeur NE DEVRAIT PAS inclure d'espace supplémentaire pour le bourrage d'arrondi XDR. Un répondeur NE DOIT PAS écrire de

bourrage d'arrondi XDR dans un tronçon en écriture, même si le demandeur a rendu de l'espace disponible pour lui. Donc, quand on retourne un seul élément de données de résultat de longueur variable, la longueur totale d'un tronçon en écriture retourné DOIT être la même que la longueur codée de l'élément de données de résultat.

3.5 Taille de message

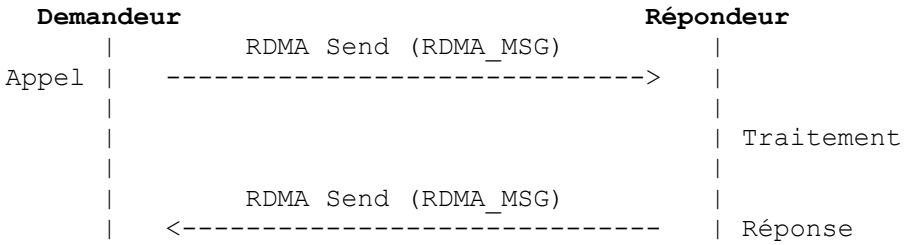
Il est exigé par RDMA qu'un receveur d'opérations RDMA Send ait préalablement alloué une ou plusieurs mémoires tampons de taille adéquate. Les économies de mémoire sont réalisées sur les demandeurs et les répondeurs en allouant de petites mémoires tampons de réception. Cependant, tous les messages RPC ne sont pas petits. RPC sur RDMA version 1 fournit plusieurs mécanismes qui permettent de transporter efficacement des messages de toutes tailles.

3.5.1 Messages courts

Les messages RPC sont fréquemment plus petits que les seuils en ligne normaux. Par exemple, l'opération GETATTR de NFS version 3 fait seulement 56 octets : 20 octets d'en-tête RPC, un argument de bride de fichier de 32 octets, et 4 octets pour sa longueur. La réponse à cette demande courante est d'environ 100 octets.

Comme tous les messages RPC convoyés via RPC sur RDMA exigent une opération RDMA Send, la façon la plus efficace d'envoyer un message RPC qui est plus petit que le seuil en ligne est d'ajouter le flux de charge utile directement au flux de transport. Un en-tête RPC sur RDMA avec un petit message d'appel ou réponse RPC suivant immédiatement est transféré en utilisant une seule opération RDMA Send. Aucune autre opération n'est nécessaire.

Transaction RPC sur RDMA utilisant des messages courts :

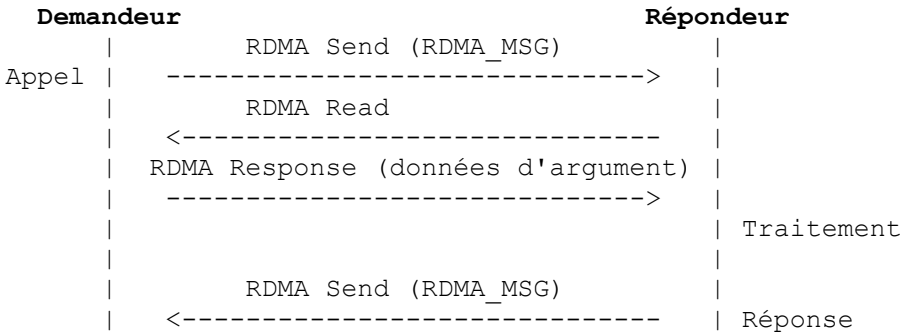


3.5.2 Tronçons de messages

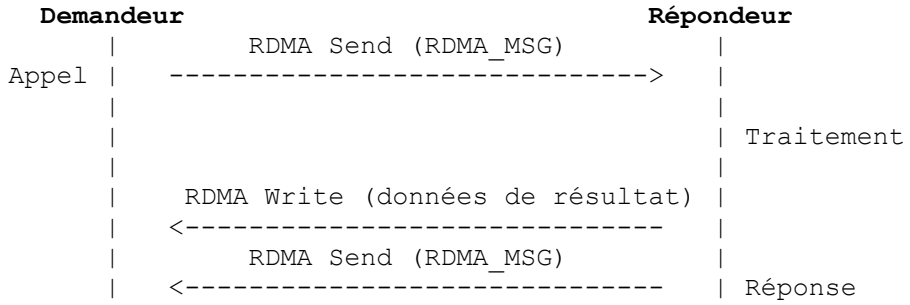
Si des éléments de données éligibles à DDP sont présents dans un flux de charge utile, un envoyeur PEUT réduire certains ou tous ces élements en les retirant du flux de charge utile. L'envoyeur utilise un mécanisme séparé pour transférer les éléments de données réduits. Le flux de transport avec le flux de charge utile réduit suivant immédiatement est alors transféré en utilisant une seule opération RDMA Send.

Après la réception des flux de transport et de charge utile d'un message d'appel RPC accompagné de tronçons en lecture, le répondeur utilise des opérations RDMA Read pour déplacer les éléments de données réduits dans les tronçons en lecture. Avant d'envoyer les flux de transport et de charge utile d'un message de réponse RPC contenant des tronçons en écriture, le répondeur utilise des opérations RDMA Write pour déplacer les éléments de données réduits dans les tronçons en écriture et de réponse.

Transaction RPC sur RDMA avec un tronçon en lecture :



Transaction RPC sur RDMA avec un tronçon en écriture :



3.5.3 Messages longs

Quand un flux de charge utile est plus long que le seuil en ligne du receveur, le flux de charge utile est réduit en supprimant des éléments de données éligibles à DDP et en les plaçant dans des tronçons à déplacer séparément. Si il n'y a pas d'éléments de données éligibles à DDP dans le flux de charge utile, ou si le flux de charge utile est encore trop grand après sa réduction, le transport RDMA DOIT utiliser des opérations RDMA Read ou Write pour convoyer le flux de charge utile lui-même. Ce mécanisme est appelé un "message long".

Pour transmettre un message long, l'envoyeur transporte seulement le flux de transport avec une opération RDMA Send. Le flux de charge utile n'est pas inclus dans la mémoire tampon d'envoi dans cette instance. À la place, le demandeur fournit des tronçons que le répondeur utilise pour déplacer le flux de charge utile.

Appel long : pour envoyer un message Appel long, le demandeur fournit un tronçon Read spécial qui contient le flux de charge utile du message d'appel RPC. Chaque segment de lecture RDMA dans ce tronçon DOIT contenir zéro dans son champ Position. Donc, ce tronçon est appelé "tronçon en lecture de position zéro".

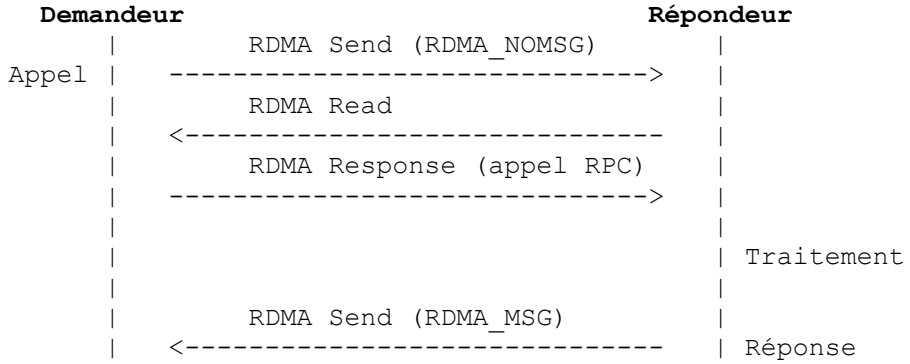
Réponse longue : pour envoyer une réponse longue, le demandeur fournit un seul tronçon Write spécial à l'avance, appelé le "tronçon de réponse", qui va contenir le flux de charge utile du message de réponse RPC. Le demandeur dimensionne le tronçon de réponse de façon à s'accommoder de la taille maximum attendue pour cette opération de couche supérieure.

Bien que l'objet d'un message long soit de traiter les grands messages RPC, les demandeurs PEUVENT utiliser un message long à tout moment pour convoyer un message d'appel RPC.

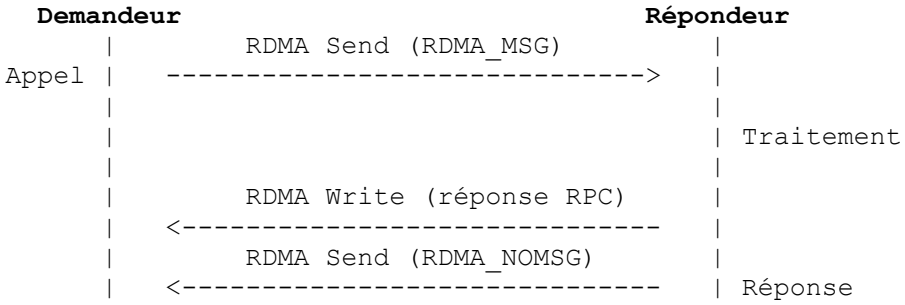
Un répondeur choisit la forme de réponse à utiliser sur la base des tronçons fournis par le demandeur. Si des tronçons en écriture ont été fournis et si le répondeur a un résultat éligible à DDP, il réduit d'abord le flux de charge utile de réponse. Si un tronçon de réponse a été fourni et si le flux de charge utile réduit est plus grand que le seuil en ligne de réponse, le répondeur DOIT utiliser le tronçon de réponse fourni par le demandeur pour la réponse.

Des éléments de données XDR peuvent apparaître dans ces tronçons spéciaux sans égard à leur éligibilité à DDP. Comme ces tronçons contiennent un flux de charge utile, ils DOIVENT inclure le bourrage d'arrondi XDR approprié pour maintenir l'alignement XDR de leur contenu.

Transaction RPC sur RDMA utilisant un appel long :



Transaction RPC sur RDMA utilisant une réponse longue :



4. Fonctionnement de RPC sur RDMA

Chaque message RPC sur RDMA version 1 a un en-tête qui inclut une copie de l'identifiant de transaction du message, des données pour gérer les crédits de contrôle de flux RDMA, et des listes de segments RDMA décrivant les tronçons. Tout le contenu d'en-tête RPC sur RDMA est contenu dans le flux de transport ; donc, il DOIT être codé en XDR.

La disposition du message RPC est inchangée par rapport à celle décrite dans la [RFC5531] excepté pour la réduction possible des éléments de données qui sont déplacés par des opérations séparées.

Le protocole RPC sur RDMA passe les messages RPC sans égard à leur type (CALL ou REPLY). À part les restrictions imposées par les ULB, chaque point d'extrémité d'une connexion PEUT envoyer des types d'en-tête de message RDMA_MSG ou RDMA_NOMSG à tout moment (sous réserve des limites de crédit).

4.1 Définition du protocole XDR

Ce paragraphe contient la description des caractéristiques centrales du protocole RPC sur RDMA version 1, exprimées dans le langage XDR [RFC4506].

Cette description est fournie d'une façon qui rend simple de l'extraire sous une forme prête à compiler. Le lecteur peut appliquer le script suivant à ce document pour produire une description XDR lisible par la machine du protocole RPC sur RDMA version 1.

<DÉBUT DU CODE>

```
#!/bin/sh
grep '^ *///' | sed 's?^ /// ??' | sed 's?^ *///$??'
```

<FIN DU CODE>

C'est-à-dire que si le script ci-dessus est mémorisé dans un fichier appelé "extract.sh" et si ce document est dans un fichier appelé "spec.txt", alors le lecteur peut faire ce qui suit pour extraire un fichier de description XDR :

<DÉBUT DU CODE>

```
sh extract.sh < spec.txt > rpcrdma_corev1.x
```

<FIN DU CODE>

4.1.1 Licence de composant de code

Les composants de code extraits du présent document doivent inclure le texte de licence suivant. Quand le code XDR extrait est combiné avec d'autre code XDR complémentaire, qui lui-même a une licence identique, une seule copie du texte de licence doit être préservée.

<DÉBUT DU CODE>

```
/// /*
///  * Copyright (c) 2010-2017 IETF Trust et les personnes identifiées comme auteurs du code. Tous droits réservés.
///  *
///  * Les auteurs du code sont B. Callaghan, T. Talpey, et C. Lever.
```

La redistribution et l'utilisation en forme source et binaire, avec ou sans modification, sont permises pourvu que les conditions suivantes soient satisfaites :

- Les redistributions de code source doivent conserver la notice de droits de reproduction ci-dessus, cette liste de conditions et le déclinatoire de responsabilité qui suit.
- Les redistributions en forme binaire doivent reproduire la notice de droits de reproduction ci-dessus, cette liste de conditions et le déclinatoire de responsabilité qui suit dans la documentation et/ou autres matériaux fournis avec la distribution.
- Ni le nom de la Internet Society, de l'IETF ou de l'IETF Trust, ni les noms des contributeurs spécifiques, ne peuvent être utilisés pour avaliser ou promouvoir des produits dérivés de ce logiciel sans permission écrite spécifique préalable.

Ce logiciel est fourni par les détenteurs du droit de reproduction et les contributeurs "TEL QUEL" et toutes garanties expresses ou implicites, incluant, sans s'y limiter, les garanties implicites de commercialisabilité et de convenance à un objet particulier sont déclinées. En aucun cas les détenteurs du droit de reproduction et les contributeurs ne seront responsables de dommages directs, indirects, incidents, spéciaux, exemplaires, ou consécutifs (incluant, sans s'y limiter, la fourniture de biens ou services de substitution, perte d'usage, données, ou profits, ou interruption d'affaires) cependant causés, et aucune théorie de responsabilité, contractuelle, stricte, ou pour tort (incluant la négligence ou autrement) survenant d'une façon quelconque de l'utilisation de ce logiciel, même si on est avisé de la possibilité de tels dommages.

```
///
```

<FIN DU CODE>

4.1.2 XDR de RPC sur RDMA version 1

Les éléments de données XDR définis dans ce paragraphe codent le flux d'en-tête de transport dans chaque message RPC sur RDMA version 1. Les commentaires identifient les éléments qui ne peuvent pas être changés dans les versions suivantes.

<DÉBUT DU CODE>

```
/// /*
///  * Segment RDMA complet (paragraphe 3.4.3)
///  */
/// struct xdr_rdma_segment {
///    uint32 handle;                    /* Bride de mémoire enregistrée */
///    uint32 length;                    /* Longueur du tronçon en octets */
///    uint64 offset;                    /* Adresse virtuelle ou décalage du tronçon */
/// };
///
/// /*
///  * Segment RDMA en lecture (paragraphe 3.4.5)
///  */
/// struct xdr_read_chunk {
///    uint32 position;                  /* Position dans le flux XDR */
///    struct xdr_rdma_segment target;
/// };
///
/// /*
///  * Liste de lecture (paragraphe 4.3.1)
///  */
/// struct xdr_read_list {
///         struct xdr_read_chunk entry;
///         struct xdr_read_list  *next;
/// };
```

```
///
/// /*
///  * Tronçon en écriture (paragraphe 3.4.6)
///  */
/// struct xdr_write_chunk {
///         struct xdr_rdma_segment target<>;
/// };
///
/// /*
///  * Liste d'écriture (paragraphe 4.3.2)
///  */
/// struct xdr_write_list {
///         struct xdr_write_chunk entry;
///         struct xdr_write_list  *next;
/// };
///
/// /*
///  * Listes de tronçons (paragraphe 4.3)
///  */
/// struct rpc_rdma_en-tête {
///    struct xdr_read_list   *rdma_reads;
///    struct xdr_write_list  *rdma_writes;
///    struct xdr_write_chunk *rdma_reply;
///                                                  /* le corps rpc suit */
/// };
///
/// struct rpc_rdma_header_nomsg {
///    struct xdr_read_list   *rdma_reads;
///    struct xdr_write_list  *rdma_writes;
///    struct xdr_write_chunk *rdma_reply;
/// };
///
/// /* À ne pas utiliser */
/// struct rpc_rdma_header_padded {
///    uint32                 rdma_align;
///    uint32                 rdma_thresh;
///    struct xdr_read_list   *rdma_reads;
///    struct xdr_write_list  *rdma_writes;
///    struct xdr_write_chunk *rdma_reply;
///                                                  /* le corps rpc suit */
/// };
///
/// /*
///  * Erreur de traitement (paragraphe 4.5)
///  */
/// enum rpc_rdma_errcode {
///    ERR_VERS = 1,                                 /* Valeur fixée pour toutes les versions */
///    ERR_CHUNK = 2
/// };
///
/// /* Structure fixée pour toutes les versions */
/// struct rpc_rdma_errvers {
///    uint32 rdma_vers_low;
///    uint32 rdma_vers_high;
/// };
///
/// union rpc_rdma_error switch (rpc_rdma_errcode err) {
///    cas ERR_VERS:
///      rpc_rdma_errvers range;
///    cas ERR_CHUNK:
///      void;
```

```
/// };
///
/// /*
///  * Procédures (paragraphe 4.2.4)
///  */
/// enum rdma_proc {
///    RDMA_MSG = 0,                    /* Valeur fixée pour toutes les versions */
///    RDMA_NOMSG = 1,                  /* Valeur fixée pour toutes les versions */
///    RDMA_MSGP = 2,                   /* À ne pas utiliser */
///    RDMA_DONE = 3,                   /* À ne pas utiliser */
///    RDMA_ERROR = 4                   /* Valeur fixée pour toutes les versions */
/// };
///
/// /* La position du champ discriminateur de procédure est fixée pour toutes les versions */
/// union rdma_body switch (rdma_proc proc) {
///    cas RDMA_MSG:
///      rpc_rdma_header rdma_msg;
///    cas RDMA_NOMSG:
///      rpc_rdma_header_nomsg rdma_nomsg;
///    cas RDMA_MSGP:                   /* À ne pas utiliser */
///      rpc_rdma_header_padded rdma_msgp;
///    cas RDMA_DONE:                   /* À ne pas utiliser */
///      void;
///    cas RDMA_ERROR:
///      rpc_rdma_error rdma_error;
/// };
///
/// /*
///  * Champs d'en-tête fixes (paragraphe 4.2)
///  */
/// struct rdma_msg {
///    uint32    rdma_xid;              /* Position fixée pour toutes les versions */
///    uint32    rdma_vers;             /* Position fixée pour toutes les versions */
///    uint32    rdma_credit;           /* Position fixée pour toutes les versions */
///    rdma_body rdma_body;
/// };
```

<FIN DU CODE>

4.2 Champs d'en-tête fixes

L'en-tête RPC sur RDMA commence par quatre champs fixes de 32 bits qui contrôlent l'interaction RDMA.

Les trois premiers mots sont des champs individuels dans la structure rdma_msg. Le quatrième mot est le premier mot de l'union rdma_body, qui agit comme discriminant pour l'union commutée. Le contenu de ce champ est décrit au paragraphe 4.2.4.

Ces quatre champs doivent rester avec la même signification et les mêmes positions dans toutes les versions suivantes du protocole RPC sur RDMA.

4.2.1 Identifiant de transaction (XID)

Le XID généré pour les messages d'appel et réponse RPC. Avoir le XID à une position fixe dans l'en-tête rend facile au receveur d'établir le contexte aussitôt qu'arrive chaque message RPC sur RDMA. Ce XID DOIT être le même que le XID dans le message RPC. Le receveur PEUT effectuer son traitement seulement sur la base du XID dans l'en-tête RPC sur RDMA, et ainsi ignorer le XID dans le message RPC, si c'est son choix.

4.2.2 Numéro de version

Pour RPC sur RDMA version 1, ce champ DOIT contenir la valeur un (1). Les règles concernant les changements du numéro de version de ce protocole de transport se trouvent à la Section 7.

4.2.3 Valeur de crédit

Quand elle est envoyée avec un message d'appel RPC, la valeur de crédit demandée est fournie. Quand elle est envoyée avec un message de réponse RPC, la valeur de crédit accordée est retournée. La discussion de la façon de déterminer la valeur de crédit se trouve au paragraphe 3.3.

4.2.4 Numéro de procédure

RDMA_MSG = 0 indique que des listes de tronçons et un flux de charge utile suivent. Le format des listes de tronçons est discuté ci-dessous.

RDMA_NOMSG = 1 indique qu'après les listes de tronçons il n'y a pas de flux de charge utile. Dans ce cas, les listes de tronçons donnent les informations qui permettent au répondeur de transférer le flux de charge utile en utilisant des opérations RDMA explicites.

RDMA_MSGP = 2 est réservé.

RDMA_DONE = 3 est réservé.

RDMA_ERROR = 4 est utilisé pour signaler une erreur de codage dans l'en-tête RPC sur RDMA.

Une procédure RDMA_MSG convoie le flux de transport et le flux de charge utile via une opération Send RDMA. Le flux de transport contient les quatre champs fixes suivis par les listes de tronçons Read et Write et le tronçon Reply, bien qu'un, ou tous les trois, PUISSE être marqué comme non présent. Le flux de charge utile suit alors, commençant par son champ XID. Si une liste de tronçons Read ou Write est présente, une portion du flux de charge utile a été réduite et est convoyée via des opérations séparées.

Une procédure RDMA_NOMSG convoie le flux de transport via une opération Send RDMA. Le flux de transport contient les quatre champs fixes suivis par les listes de tronçons Read et Write et le tronçon Reply. Bien qu'un quelconque d'entre eux PUISSE être marqué comme non présent, il DOIT y en avoir un présent et il DOIT contenir le flux de charge utile pour ce message RPC sur RDMA. Si une liste de tronçons Read ou Write est présente, une portion du flux de charge utile a été réduite et est convoyée via des opérations séparées.

Une procédure RDMA_ERROR convoie le flux de transport via une opération Send RDMA. Le flux de transport contient les quatre champs fixes suivis par les informations d'erreur formatées. Aucun flux de charge utile n'est convoyé dans ce type de message RPC sur RDMA.

Un demandeur NE DOIT PAS envoyer un en-tête RPC sur RDMA avec la procédure RDMA_ERROR. Un répondeur DOIT éliminer en silence les procédures RDMA_ERROR.

Le flux de transport et le flux de charge utile peuvent être construits dans des mémoires tampons séparées. Cependant, la longueur totale des mémoires tampons rassemblées ne peut pas excéder le seuil en ligne.

4.3 Listes de tronçons

Les listes de tronçons dans un en-tête RPC sur RDMA version 1 sont trois champs de données facultatives XDR qui suivent les champs d'en-tête fixes dans les procédures RDMA_MSG et RDMA_NOMSG. Lire avec attention le paragraphe 4.19 de la [RFC4506] pour comprendre comment fonctionnent les champs de données facultatives. Des exemples de listes de tronçons codées en XDR sont fournis au paragraphe 4.7 pour aider à le comprendre.

Souvent, un message RPC sur RDMA n'a pas de tronçon associé. Dans ce cas, la liste Read, la liste Write, et le tronçon Reply sont tous marqués "non présent".

4.3.1 Liste Read

Chaque procédure RDMA_MSG ou RDMA_NOMSG a une "liste de lecture". La liste de lecture est une liste de zéro, un ou plusieurs segments de lecture RDMA, fournis par le demandeur, qui sont groupés par leur champ Position en tronçons de lecture. Chaque tronçon de lecture annonce la localisation des données d'argument que le répondeur va tirer du demandeur. Le demandeur a réduit les éléments de données dans ces tronçons à partir du flux de charge utile de l'appel.

Un demandeur peut transmettre le flux de charge utile d'un message d'appel RPC en utilisant un tronçon de lecture de position zéro. Si le message d'appel RPC n'a pas de données d'argument qui soient éligibles à DDP et si le tronçon de lecture de position zéro n'est pas utilisé, le demandeur laisse la liste de lecture vide.

Les répondeurs DOIVENT laisser la liste de lecture vide dans toutes les réponses.

4.3.1.1 Confrontation des tronçons en lecture aux arguments

Quand il réduit un élément de données d'argument éligible à DDP, un demandeur enregistre le décalage du flux XDR de cet élément de données dans le champ Position du tronçon en lecture. Le répondeur peut alors dire sans ambiguïté où ce tronçon est à réinsérer dans le flux de charge utile reçu pour former un message d'appel RPC complet.

4.3.2 Liste Write

Chaque procédure RDMA_MSG ou RDMA_NOMSG a une "liste d'écriture". La liste d'écriture est une liste de zéro, un, ou plusieurs tronçons en écriture, fournis par le demandeur. Chaque tronçon en écriture est un dispositif de segments complets ; donc, la liste d'écriture est une liste de dispositifs comptés.

Si un message de réponse RPC n'a pas d'élément de données de résultat éligible à DDP possible, le demandeur laisse la liste d'écriture vide. Quand un demandeur fournit une liste d'écriture, le répondeur DOIT pousser les données correspondant à ces éléments de données de résultat éligibles à DDP à la mémoire du demandeur référencée dans la liste d'écriture. Le répondeur supprime ces éléments de données du flux de charge utile de la réponse.

4.3.2.1 Confrontation des tronçons en écriture aux résultats

Un demandeur construit la liste d'écriture pour une transaction RPC avant que le répondeur ait formulé sa réponse. Quand il y a seulement un élément de données de résultat éligible à DDP, le demandeur insère seulement un tronçon en écriture dans la liste d'écriture. Si le tronçon d'écriture retourné n'est pas un tronçon d'écriture inutilisé, le demandeur sait avec certitude quel élément de données de résultat y est contenu.

Quand un demandeur a fourni plusieurs tronçons en écriture, le répondeur remplit chaque tronçon en écriture avec un résultat éligible à DDP jusqu'à ce qu'il n'y ait plus de résultat éligible à DDP ou plus de tronçon en écriture.

Le demandeur pourrait n'être pas capable de prédire à l'avance quel élément de données éligible à DDP va dans quel tronçon. Donc, le demandeur est chargé d'allouer et enregistrer des tronçons en écriture assez grands pour s'accommoder du plus grand élément de données de résultat qui pourrait être associé à chaque tronçon dans la liste d'écriture.

Lorsque un demandeur décode un flux de charge utile de réponse, il est clair d'après le contenu du message de réponse RPC quel tronçon en écriture contient quel élément de données de résultat.

4.3.2.2 Tronçons en écriture inutilisés

Il y a des occasions où un demandeur fournit un tronçon en écriture non vide mais où le répondeur n'est pas capable de l'utiliser. Par exemple, un ULP peut définir un résultat d'union où certaines branches de l'union contiennent un élément de données éligible à DDP tandis que d'autres branches n'en contiennent pas. Il est exigé du répondeur qu'il utilise des tronçons en écriture fournis par le demandeur dans ce cas, mais si le répondeur retourne un résultat qui utilise une branche de l'union qui n'a pas d'élément de données éligible à DDP, ce tronçon en écriture reste non consommé.

Si il y a un élément de données de résultat suivant éligible à DDP dans le message de réponse RPC, il DOIT être placé dans ce tronçon en écriture non consommé. Donc, le demandeur DOIT provisionner chaque tronçon en écriture de telle façon qu'il puisse être rempli avec le plus grand élément de données éligible à DDP qui peut y être placé.

Si c'est le dernier ou le seul tronçon en écriture disponible et qu'il reste non consommé, le répondeur DOIT retourner ce tronçon Write comme tronçon Write inutilisé (voir le paragraphe 3.4.6). Le répondeur règle le compte de segments à une valeur correspondant au tronçon en écriture fourni par le demandeur, mais retourne seulement des segments vides dans ce tronçon Write.

Les tronçons Write non utilisés, ou les octets inutilisés dans les segments de tronçon Write, sont retournés au consommateur RPC au titre de l'achèvement de RPC. Même si un répondeur indique qu'un tronçon Write n'est pas consommé, le répondeur peut avoir écrit des données dans un ou plusieurs segments avant de choisir de ne pas retourner cet élément de données. Le demandeur NE DOIT PAS supposer que les régions de mémoire qui soutiennent un tronçon en écriture n'ont pas été modifiées.

4.3.2.3 Tronçons en écriture vides

Pour forcer un répondeur à retourner un résultat en ligne éligible à DDP, un demandeur emploie le mécanisme suivant :

- Quand il y a seulement un élément de résultat éligible à DDP dans un message de réponse RPC, le demandeur fournit une liste d'écriture vide.
- Quand il y a plusieurs éléments de données de résultat éligibles à DDP et qu'un demandeur préfère qu'un élément de données soit retourné en ligne, le demandeur fournit un tronçon en écriture vide pour cet élément (voir le paragraphe 3.4.6). Le répondeur DOIT retourner l'élément de données de résultat en ligne correspondant et DOIT retourner un tronçon en écriture vide dans cette position de liste d'écriture dans le message de réponse RPC.

Comme toujours, un demandeur et un répondeur doivent être prêts à utiliser une réponse longue si la réponse RPC résultante serait trop grande pour être convoyée dans un RDMA Send.

4.3.3 Tronçon de réponse

Chaque procédure RDMA_MSG ou RDMA_NOMSG a un créneau "tronçon de réponse". Un demandeur DOIT fournir un tronçon de réponse chaque fois que la taille maximum possible des flux de transport et de charge utile du message de réponse RPC est supérieure au seuil en ligne pour les messages du répondeur au demandeur. Autrement, le demandeur marque le tronçon de réponse comme non présent.

Si le flux de transport et le flux de charge utile ensemble sont plus petits que le seuil de réponse en ligne, le répondeur PEUT retourner le message de réponse RPC comme un message court plutôt que d'utiliser le tronçon de réponse fourni par le demandeur.

Quand un demandeur fournit un tronçon de réponse dans un message d'appel RPC, le répondeur DOIT copier ce tronçon dans l'en-tête de transport du message de réponse RPC. Comme avec les tronçons d'écriture, le répondeur modifie le tronçon de réponse copié dans le message de réponse RPC pour refléter la quantité réelle de données qui est retournée dans le tronçon de réponse.

4.4 Enregistrement de mémoire

Le coût d'enregistrement et d'invalidation de la mémoire peut être une proportion significative du coût d'une transaction RPC sur RDMA. Donc, une importante considération de mise en œuvre est comment minimiser l'activité d'enregistrement sans exposer sans nécessité la mémoire du système.

4.4.1 Longévité d'enregistrement

Les données transférées via RDMA Read et Write peuvent résider dans une allocation de mémoire qui n'est pas sous le contrôle du transport RPC sur RDMA. Ces allocations de mémoire peuvent persister en dehors des limites d'une transaction RPC. Elles sont enregistrées et invalidées comme nécessaire, au titre de chaque transaction RPC.

Le point d'extrémité demandeur doit s'assurer que les régions de mémoire associées à chaque transaction RPC sont protégées de l'accès du répondeur avant de permettre l'accès des couches supérieures aux données qu'elles contiennent. De plus, le demandeur ne doit pas accéder à ces régions de mémoire quand le répondeur y a accès.

Cela inclut les régions de mémoire qui sont associées à des RPC annulés. Un répondeur ne peut pas savoir que le demandeur n'attend plus une réponse, et il pourrait procéder à la lecture ou même mettre à jour une mémoire que le

demandeur pourrait avoir libérée pour une autre utilisation.

4.4.2 Communication de l'éligibilité à DDP

L'interface par laquelle une mise en œuvre d'ULP communique l'éligibilité d'un élément de données localement à son point d'extrémité local RPC sur RDMA n'est pas décrite dans la présente spécification.

Selon la mise en œuvre et les contraintes imposées par les ULB, il est possible de mettre en œuvre la réduction de façon transparente pour les couches supérieures. De telles mises en œuvre peuvent conduite à de l'inefficacité, soit parce qu'elles exigent que la couche RPC effectue un enregistrement coûteux et l'invalidation de mémoire "au vol", soit parce qu'elles peuvent exiger d'utiliser des tronçons RDMA dans les messages de réponse RPC, avec les prises de contact supplémentaires résultantes avec l'homologue RPC sur RDMA.

Cependant, ces questions sont internes et généralement confinées à l'interface locale entre RPC et ses couches supérieures, dans lesquelles les mises en œuvre sont libres d'innover. La seule exigence, au delà des contraintes imposées par l'ULB, est que le protocole RPC sur RDMA résultant envoyé à l'homologue soit valide pour la couche supérieure.

4.4.3 Stratégies d'enregistrement

Le choix de la stratégie d'enregistrement de mémoire à employer est laissé aux mises en œuvre de demandeur et répondeur. Pour prendre en charge la plus large variété de mises en œuvre de RDMA, ainsi que le plus général schéma d'étiquette de pilotage, un champ Décalage est inclus dans chaque segment RDMA.

Alors que des schémas de décalage fondés sur le zéro sont disponibles dans de nombreuses mises en œuvre de RDMA, leur utilisation par RPC exige un enregistrement individuel de chaque région de mémoire. Pour ces mises en œuvre, cela peut être une surcharge significative. En fournissant un décalage dans chaque tronçon, de nombreux pré-enregistrements ou enregistrements fondés sur la région peuvent être directement pris en charge.

4.5 Traitement des erreurs

Un receveur effectue des vérifications de validité de base sur l'en-tête RPC sur RDMA et le contenu des tronçons avant de passer le message RPC à la couche RPC. Si un message RPC sur RDMA entrant n'est pas aussi long que la taille minimale d'en-tête RPC sur RDMA (28 octets) le receveur ne peut pas faire confiance à la valeur du champ XID ; donc, il DOIT éliminer en silence le message avant d'effectuer aucune analyse. Si d'autres erreurs sont détectées dans l'en-tête RPC sur RDMA d'un message d'appel RPC, un répondeur DOIT renvoyer un message RDMA_ERROR au demandeur. Si des erreurs sont détectées dans l'en-tête RPC sur RDMA d'un mesage de réponse RPC, un demandeur DOIT éliminer en silence le message.

Pour former une procédure RDMA_ERROR :
- o le champ rdma_xid DOIT contenir le même XID que dans le champ rdma_xid de la demande défaillante ;
- o le champ rdma_vers DOIT contenir la même version que dans le champ rdma_vers de la demande défaillante ;
- o le champ rdma_proc DOIT contenir la valeur RDMA_ERROR ; et
- o le champ rdma_err contient une valeur qui reflète le type d'erreur qui s'est produite, comme décrit ci-dessous.

Une procédure RDMA_ERROR indique une erreur permanente. La réception de cette procédure achève la transaction RPC associée à XID dans le champ rdma_xid. Un receveur DOIT éliminer en silence une procédure RDMA_ERROR qu'il ne peut pas décoder.

4.5.1 Discordance de version d'en-tête

Quand un répondeur détecte une version d'en-tête RPC sur RDMA qu'il ne prend pas en charge (actuellement, le présent document définit seulement la version 1) il DOIT répondre avec une procédure RDMA_ERROR et régler la valeur de rdma_err à ERR_VERS, fournissant aussi les numéros de version haut et bas inclus qu'il prend en charge.

4.5.2 Erreurs XDR

Un receveur pourrait rencontrer une erreur d'analyse XDR qui l'empêche de traiter le flux de transport entrant. Des exemples de telles erreurs incluent une valeur invalide dans le champ rdma_proc, un message RDMA_NOMSG où la liste

de lecture, la liste d'écriture, et le tronçon de réponse sont marqués comme non présents, ou la valeur du champ rdma_xid ne correspond pas à la valeur du champ XID dans le message RPC d'accompagnement. Si le champ rdma_vers contient une valeur reconnue, mais qu'une erreur d'analyse XDR survient, le répondeur DOIT répondre avec une procédure RDMA_ERROR et régler la valeur de rdma_err à ERR_CHUNK.

Quand un répondeur reçoit un en-tête RPC sur RDMA valide mais que la mise en œuvre d'ULP du répondeur ne peut pas analyser les arguments RPC dans le message d'appel RPC, le répondeur DEVRAIT retourner un message de réponse RPC avec l'état GARBAGE_ARGS, en utilisant une procédure RDMA_MSG. Ce type d'échec d'analyse pourrait être dû à des discordances entre les tailles de tronçon ou de décalages et le contenu du flux de charge utile, par exemple.

4.5.3 Erreurs de fonctionnement de répondeur RDMA

Dans RPC sur RDMA version 1, le répondeur initie des opérations RDMA Read et Write qui ciblent la mémoire du demandeur. Des problèmes pourraient survenir lorsque le répondeur tente d'utiliser des ressources fournies par le demandeur pour des opérations RDMA. Par exemple :

- Généralement, les tronçons peuvent être validés seulement en utilisant leur contenu pour effectuer des transferts de données. Si le contenu du tronçon est invalide (par exemple, une région de mémoire n'est plus enregistrée ou la longueur d'un tronçon dépasse la fin de la région de mémoire enregistrée) une erreur d'accès distant survient.

- Si la mémoire tampon d'un demandeur est trop petite, l'opération Send du répondeur s'achève avec une erreur de longueur locale.

- Si le tronçon de réponse fourni par le demandeur est trop petit pour s'accommoder d'un grand message de réponse RPC, une erreur d'accès distant survient. Un répondeur pourrait détecter ce problème avant de tenter d'écrire après la fin d'un tronçon de réponse.

Les erreurs de fonctionnement RDMA sont normalement fatales pour la connexion. Pour éviter une boucle de retransmission et des pertes répétées de connexion qui mettent la connexion dans une impasse, une fois que le demandeur a rétabli une connexion, le répondeur devrait envoyer une réponse RDMA_ERROR avec une valeur de rdma_err de ERR_CHUNK pour indiquer qu'aucune réponse n'est possible au niveau RPC pour cet XID.

4.5.4 Autres erreurs de fonctionnement

Quand un demandeur construit un message d'appel RPC, un problème irréparable pourrait survenir qui empêche le demandeur d'envoyer d'autres demandes de travaux RDMA au nom de ce message. Comme avec les autres transports, si un demandeur est incapable de construire et transmettre un message d'appel RPC, la transaction RPC associée échoue immédiatement.

Après qu'un demandeur a reçu une réponse, si il est incapable d'invalider une région de mémoire à cause d'un problème irréparable, le demandeur DOIT clore la connexion pour protéger cette mémoire contre un accès du répondeur avant que la transaction RPC associée soit achevée.

Quand un répondeur construit un message de réponse RPC ou un message d'erreur, un problème irréparable pourrait survenir qui empêche le répondeur d'envoyer d'autres demandes de travaux RDMA au nom de ce message. Si un répondeur est incapable de construire et transmettre une réponse RPC ou un message d'erreur RPC sur RDMA, le répondeur DOIT clore la connexion pour signaler au demandeur qu'une réponse a été perdue.

4.5.5 Erreurs de transport RDMA

La connexion RDMA et la liaison physique fournissent un certain degré de détection et retransmission d'erreur. La couche d'alignement sur la PDU de marqueur (MPA, *Marker PDU Aligned*) de iWARP (quand utilisée sur TCP) le protocole de transmission de commandes de flux (SCTP, *Stream Control Transmission Protocol*) ainsi que la couche de liaison InfiniBand [IBARCH] fournissent tous la protection du contrôle de redondance cyclique (CRC, *Cyclic Redundancy Check*) de la charge utile RDMA, et la protection de classe CRC est un attribut général de ces transports.

De plus, la couche RPC elle-même peut accepter des erreurs provenant du transport et récupérer via la retransmission. La récupération RPC peut traiter la perte complète et le rétablissement d'une connexion de transport.

Les détails du rapport et de la récupération des erreurs de couche de liaison RDMA sont décrits dans des API spécifiques de couche de liaison et des spécifications opérationnelles et sortent du domaine d'application de la présente spécification de protocole. Voir à la Section 8 la discussion de l'utilisation des schémas d'intégrité de niveau RPC pour détecter les erreurs.

4.6 Éléments de protocole qui ne sont plus pris en charge

Les éléments de protocole suivants ne sont plus pris en charge dans RPC sur RDMA version 1. Les valeurs d'énumération et les définitions de structure qui s'y rapportent restent dans le protocole RPC sur RDMA version 1 pour la rétro compatibilité.

4.6.1 RDMA_MSGP

La spécification de RDMA_MSGP au paragraphe 3.9 de la [RFC5666] est incomplète. Spécifier complètement RDMA_MSGP exigerait :

- de mettre à jour la définition de l'éligibilité à DDP pour inclure les éléments de données qui peuvent être transférés, avec bourrage, via les procédures RDMA_MSGP,
- d'ajouter les descriptions complètes des champs d'alignement et de seuil,
- de discuter comment les préférences d'alignement sont communiquées entre deux homologues sans utiliser CCP,
- de décrire le traitement des procédures RDMA_MSGP qui portent des tronçons Read ou Write.

Le type de message RDMA_MSGP n'est utile que quand la charge utile de données avec bourrage est à la fin de l'argument ou de la liste de résultats d'un message RPC. Ceci n'est pas normal pour les RPC NFSv4 COMPOUND, qui incluent souvent une opération GETATTR comme élément final du dispositif d'opération compound.

Sans une pleine spécification de RDMA_MSGP, il n'y a pas eu de prototype pleinement mis en œuvre. Sans un prototype complet de prise en charge de RDMA_MSGP, il est difficile de certifier que cet élément de protocole présente des avantages ou peut même être rendu interopérable.

Donc, les envoyeurs NE DOIVENT PAS envoyer de procédures RDMA_MSGP. Quand ils reçoivent une procédure RDMA_MSGP, les répondeurs DEVRAIENT répondre avec une procédure RDMA_ERROR, réglant le champ rdma_err à ERR_CHUNK ; les demandeurs DOIVENT éliminer le message en silence.

4.6.2 RDMA_DONE

Parce que aucune mise en œuvre de RPC sur RDMA version 1 n'utilise le modèle de transfert Read-Read, il n'y a jamais eu besoin d'envoyer une procédure RDMA_DONE.

Donc, les envoyeurs NE DOIVENT PAS envoyer de messages RDMA_DONE. Les receveurs DOIVENT éliminer en silence les messages RDMA_DONE.

4.7 Exemples de XDR

Les listes de tronçons de RPC sur RDMA sont des types de données complexes. Dans ce paragraphe, on donne des illustrations pour aider le lecteur à comprendre comment les listes de tronçons sont représentées dans un en-tête RPC sur RDMA.

Un segment complet est le plus simple composant, constitué d'une bride (H) de 32 bits, d'une longueur (L) de 32 bits, et d'un décalage (OO) de 64 bits. Une fois aplatis dans un flux XDR, les segments complets apparaissent comme HLOO.

Un segment RDMA read a un champ supplémentaire de position (P) de 32 bits. Les segments RDMA en lecture apparaissent comme PHLOO.

Un tronçon de lecture est une liste de segments RDMA read. Chaque segment RDMA read est précédé d'un mot de 32 bits contenant un un si un segment suit ou un zéro si il n'y a plus de segments dans la liste. En forme XDR, cela ressemblerait à 1 PHLOO 1 PHLOO 1 PHLOO 0 où P contiendrait la même valeur pour chaque segment RDMA read appartenant au même tronçon Read.

La liste Read est aussi une liste de segments RDMA read. En forme XDR, cela ressemblerait à un tronçon Read, sauf que

les valeurs P pourraient varier dans la liste. Une liste Read vide est codée comme un seul zéro de 32 bits.

Un tronçon Write est un dispositif compté de segments complets. En forme XDR, le compte apparaîtrait comme le premier mot de 32 bits, suivi par un HLOO pour chaque élément du dispositif. Par exemple, un tronçon Write avec trois éléments ressemblerait à 3 HLOO HLOO HLOO.

La liste Write est une liste de dispositifs comptés. En forme XDR, c'est une combinaison de données facultatives et de dispositifs comptés. Pour représenter une liste Write contenant un tronçon Write avec trois segments et un tronçon Write avec deux segments, XDR coderait 1 3 HLOO HLOO HLOO 1 2 HLOO HLOO 0.

Une liste Write vide est codée comme un seul zéro de 32 bits.

Le tronçon de réponse est un tronçon Write. Cependant, comme c'est un champ de données facultatives, il a devant un champ de 32 bits qui contient un un si le tronçon de réponse est présent ou un zéro si il ne l'est pas. Après codage, un tronçon de réponse avec deux segments ressemblerait à 1 2 HLOO HLOO.

Fréquemment, un demandeur ne fournit aucun tronçon. Dans ce cas, après les quatre champs fixes dans l'en-tête RPC sur RDMA-tête, il y a simplement trois champs de 32 bits qui contiennent zéro.

5. Paramètres RPC Bind

Pour établir une nouvelle connexion RDMA, la première action d'un demandeur est d'obtenir une adresse de transport pour le répondeur. Les moyens utilisés pour obtenir cette adresse, et pour ouvrir une connexion RDMA, dépendent du type de transport RDMA et sont de la responsabilité de chaque lien de protocole RPC et de sa mise en œuvre locale.

Les services RPC s'enregistrent normalement auprès d'un service portmap ou rpcbind [RFC1833], qui associe un numéro de programme RPC à une adresse de service. Cette politique n'est pas différente avec les transports RDMA. Cependant, une adresse de service différente et distincte (numéro d'accès) pourrait parfois être exigée pour l'opération ULP avec RPC sur RDMA.

Quand il est transposé par dessus le transport iWARP [RFC5040], [RFC5041], qui utilise l'adressage d'accès IP à cause de sa mise en couche sur TCP et/ou SCTP, la transposition d'accès est triviale et consiste simplement à produire l'accès dans le processus de connexion. L'adresse de service de protocole NFS/RDMA a reçu de l'IANA le numéro d'accès 20049, pour iWARP/TCP et iWARP/SCTP [RFC5667].

Quand il est transposé par dessus InfiniBand [IBARCH], qui utilise un schéma de dénomination de point d'extrémité de service fondé sur un identifiant de groupe (GID, *Group Identifier*), une traduction DOIT être faite. Une telle traduction est décrite dans les Annexes A3 (Identifiants spécifiques d'application) A4 (Protocole direct de prises (SDP, *Sockets Direct Protocol*)), et A11 (Service RDMA IP CM) de [IBARCH], qui est appropriée pour traduire l'adressage d'accès IP pour le réseau InfiniBand. Donc, dans ce cas, l'adressage d'accès IP peut être directement employé par la couche supérieure.

Quand une norme ou convention de transposition existe pour les accès IP sur une interconnexion RDMA, chaque couche supérieure a plusieurs possibilités à considérer :

- Une possibilité est que le répondeur enregistre son accès IP transposé au service rpcbind sous l'identifiant de réseau défini ici. Un demandeur à capacité RPC sur RDMA peut alors résoudre son service désiré en un accès transposable et procéder à la connexion. C'est l'approche la plus souple et compatible, pour les couches supérieures qui sont définies comme utilisant le service rpcbind.

- Une seconde possibilité est que le transposeur d'accès du répondeur s'enregistre lui-même sur l'interconnexion RDMA à une adresse de service "bien connue" (sur UDP ou TCP, cela correspond à l'accès 111). Un demandeur pourrait se connecter à cette adresse de service et utiliser le protocole portmap pour obtenir une adresse de service en réponse à un numéro de programme, par exemple, un numéro d'accès iWARP ou un GID InfiniBand.

- Autrement, le demandeur pourrait simplement se connecter à l'accès bien connu transposé pour le service lui-même, si il est défini de façon appropriée. Par convention, le service NFS/RDMA, quand il fonctionne par dessus un tel tissu InfiniBand, utilise la même allocation 20049 que pour iWARP.

Historiquement, différents protocoles RPC ont eu des approches différentes de leur allocation d'accès. Donc, la méthode

spécifique est laissé à chaque ULB à capacité RPC sur RDMA et n'est pas traitée dans le présent document.

À la Section 9, cette spécification définit deux nouvelles valeurs de netid, à utiliser pour l'enregistrement des couches supérieures par dessus iWARP [RFC5040], [RFC5041], et (quand un service de traduction d'accès convenable est disponible) InfiniBand [IBARCH]. Des réseaux à capacité RDMA supplémentaires PEUVENT définir leurs propres identifiants de réseau, ou si ils fournissent une traduction d'accès, ils PEUVENT partager celui défini dans ce document.

6. Spécifications d'ULB

Un ULP est normalement défini indépendamment de tout transport RPC particulier. Une spécification d'ULB fournit des lignes directrices qui aident l'ULP à interopérer correctement et efficacement sur un transport particulier. Pour RPC sur RDMA version 1, un ULB peut fournir :

- une taxonomie des éléments de données XDR qui sont éligibles pour DDP,
- les contraintes selon lesquelles les procédures de couche supérieure peuvent être réduites et sur combien de tronçons peuvent apparaître dans une seule demande RPC,
- une méthode pour déterminer la taille maximum du flux de charge utile de réponse pour toutes les procédures dans l'ULP,
- une allocation d'accès rpcbind pour le fonctionnement du programme et version RPC sur un transport RPC sur RDMA.

Chaque couple programme-version RPC qui utilise RPC sur RDMA version 1 a besoin d'avoir une spécification d'ULB.

6.1 Éligibilité à DDP

Un ULB désigne des éléments de données XDR comme éligibles à DDP. Comme un message RPC sur RDMA est formé, des éléments de données éligibles à DDP peuvent être supprimés du flux de charge utile et placés directement dans la mémoire du receveur.

Un élément de données XDR devrait être considéré pour l'éligibilité à DDP si il y a un clair avantage à déplacer le contenu de l'élément directement de la mémoire de l'envoyeur à la mémoire du receveur. Les critère de l'éligibilité à DDP incluent :

- L'élément de données XDR est fréquemment envoyé ou reçu, et sa taille est souvent bien supérieure aux seuils en ligne normaux.
- Si l'élément de données XDR est un résultat, sa taille maximale doit être prévisible par le demandeur.
- Le traitement de niveau transport de l'élément de données XDR n'est pas nécessaire. Par exemple, l'élément de données est un dispositif d'octets opaque, qui n'exige pas de codage et décodage XDR de son contenu.
- Le contenu de l'élément de données XDR est sensible à l'alignement d'adresse. Par exemple, une opération de copie de données serait exigée chez le receveur pour permettre l'analyse correcte du message, ou pour permettre l'accès à l'élément de données.
- L'élément de données XDR ne contient pas d'élément de données éligible à DDP.

En plus de définir l'ensemble des éléments de données qui sont éligibles à DDP, un ULB peut aussi limiter l'utilisation de tronçons à des procédures particulières de couche supérieure. Si plus d'un élément de données dans une procédure est éligible à DDP, l'ULB peut aussi limiter le nombre de tronçons qu'un demandeur peut fournir pour une procédure de couche supérieure particulière.

Les envoyeurs NE DOIVENT PAS réduire les éléments de données qui ne sont pas éligibles à DDP. Ces éléments de données PEUVENT, cependant, être déplacés au titre d'un tronçon de lecture de position zéro ou d'un tronçon de réponse.

L'interface de programmation par lequel une mise en œuvre de couche supérieure indique au transport RPC l'éligibilité à DDP d'un élément de données n'est pas décrite dans la présente spécification. Les seules exigences sont que le receveur puisse ré-assembler le message RPC sur RDMA transmis dans un flux XDR valide, et que les règles d'éligibilité à DDP spécifiées par l'ULB soient respectées.

Il n'y a pas de disposition pour exprimer l'éligibilité à DDP dans le langage XDR. La seule spécification définitive de l'éligibilité à DDP est un ULB.

En général, une violation d'éligibilité à DDP se produit quand :

- Un demandeur réduit un élément de données d'argument non éligible à DDP. Le répondeur NE DOIT PAS traiter ce message d'appel RPC et DOIT rapporter la violation comme décrit au paragraphe 4.5.2.

- Un répondeur réduit un élément de données de résultat non éligible à DDP. Le demandeur DOIT terminer la transaction RPC en instance et rapporter une erreur permanente appropriée au consommateur RPC.
- Un répondeur ne réduit pas un élément de données de résultat éligible à DDP dans un tronçon d'écriture disponible. Le demandeur DOIT terminer la transaction RPC en instance et rapporter une erreur permanente appropriée au consommateur RPC.

6.2 Taille maximum de réponse

Un demandeur fournit des ressources pour un message d'appel RPC et son message de réponse RPC correspondant. Un demandeur forme le message d'appel RPC lui-même ; donc, le demandeur peut calculer les ressources exactes nécessaires.

Un demandeur doit allouer des ressources pour le message de réponse RPC (un crédit RPC sur RDMA, une mémoire tampon de réception, et éventuellement une liste d'écriture et un tronçon de réponse) avant que le répondeur ait formé la réponse réelle. Pour s'accommoder de toutes les réponses possibles pour la procédure dans le message d'appel RPC, un demandeur doit allouer les ressources de réponse sur la base de la taille maximum possible du message de réponse RPC attendu.

Si il y a des procédures dans l'ULP pour lesquelles il n'y a pas de taille maximum de réponse claire, l'ULB doit spécifier des moyens sûrs pour déterminer le maximum.

6.3 Considérations supplémentaires

Il peut y avoir d'autres détails fournis dans un ULB.

- Un ULB peut recommander des valeurs de seuil en ligne ou autres paramètres relatifs au transport pour les connexions RPC sur RDMA version 1 portant cet ULP.

- Un ULP peut fournir un moyen de communiquer ces paramètres relatifs au transport entre les homologues. Noter que RPC sur RDMA version 1 ne spécifie aucun mécanisme pour changer des paramètres relatifs au transport après l'établissement d'une connexion.

- Plusieurs ULP peuvent partager une seule connexion RPC sur RDMA version 1 quand leurs ULB permettent l'utilisation de RPC sur RDMA version 1 et que les allocations d'accès rpcbind pour les protocoles permettent le partage de connexion. Dans ce cas, les mêmes paramètres de transport (comme le seuil en ligne) s'appliquent à tous les protocoles utilisant cette connexion.

Chaque ULB doit être conçu pour permettre une inter opération correcte sans égard aux paramètres de transport réellement utilisés. De plus, les mises en œuvre des ULP doivent être conçues pour inter opérer correctement sans égard aux paramètres de connexion en vigueur sur une connexion.

6.4 Extensions d'ULP

Un couple de programme et version RPC peut être extensible. Par exemple, il peut y avoir un schéma de versions mineures qui n'est pas reflété dans le numéro de version RPC, ou l'ULP peut permettre que des caractéristiques supplémentaires soient spécifiées après la ratification de la spécification originale de programme RPC.

Les ULB sont fournis pour des programmes et versions RPC interopérables en étendant les ULB existants pour refléter les changements rendus nécessaires par chaque ajout à l'XDR existant.

7. Extensibilité du protocole

Le format d'en-tête RPC sur RDMA est spécifié en utilisant XDR, à la différence de l'en-tête de message utilisé avec RPC sur TCP. Pour maintenir un haut degré d'interopérabilité parmi les mises en œuvre de RPC sur RDMA, tout changement à ce XDR exige un changement de numéro de version de protocole. De nouvelles versions de RPC sur RDMA peuvent être publiées comme des spécifications de protocole séparées sans mettre à jour le présent document.

Les quatre premiers champs dans chaque en-tête RPC sur RDMA doivent rester alignés au même décalage fixe pour toutes

les versions du protocole RPC sur RDMA. Le numéro de version doit être dans un endroit fixe pour permettre aux mises en œuvre de détecter les discordances de version de protocole.

Pour que les discordances de version soient rapportées d'une façon que toutes les mises en œuvre de futures versions puissent décoder fiablement, le champ rdma_proc doit rester à un endroit fixe, la valeur de ERR_VERS doit toujours rester la même, et le placement de champ dans la structure rpc_rdma_errvers doit toujours rester la même.

7.1 Extensions conventionnelles

L'introduction de nouvelles capacités à RPC sur RDMA version 1 se limite à l'adoption de conventions qui utilisent le XDR existant (défini dans ce document) et des opérations RDMA abstraites permises. Parce qu'il n'existe aucun mécanisme pour détecter les caractéristiques facultatives dans RPC sur RDMA version 1, les mises en œuvre doivent s'appuyer sur les ULP pour communiquer l'existence de ces extensions.

Ces extensions doivent être spécifiées dans des RFC sur la voie de la normalisation avec les revues appropriées par le groupe de travail NFSv4 et l'IESG. Un exemple d'extension conventionnelle à RPC sur RDMA version 1 est la spécification de la prise en charge du message de direction arrière pour permettre les opérations de rappel NFSv4.1, décrites dans la [RFC8167].

8. Considérations sur la sécurité

8.1 Protection de la mémoire

Une considération majeure est la protection de l'intégrité et de la confidentialité de la mémoire locale par un transport RPC sur RDMA. L'utilisation d'un protocole de transport RPC sur RDMA NE DOIT PAS introduire de vulnérabilités dans le contenu de la mémoire du système ni dans la mémoire possédée par les processus d'utilisateur.

Il est EXIGÉ que tout fournisseur RDMA utilisé pour transporter RPC soit conforme aux exigences de la [RFC5042] afin de satisfaire ces protections. Ces protections sont fournies par les spécifications de la couche RDMA, et en particulier, leurs modèles de sécurité.

8.1.1 Domaines de protection

L'utilisation de domaines de protection pour limiter l'exposition de régions de mémoire à une seule connexion est critique. Toute tentative par un point d'extrémité ne participant pas à cette connexion de réutiliser les brides de mémoire doit nécessairement résulter en la défaillance immédiate de cette connexion. Parce que les mécanismes de sécurité d'ULP s'appuient sur cet aspect de comportement de connexion fiable, une forte authentification des points d'extrémité distants est recommandée.

8.1.2 Prévisibilité de bride

Des brides de mémoire imprévisibles devraient être utilisées pour toute opération exigeant l'annonce de régions de mémoire. L'annonce d'une région de mémoire enregistrée de façon continue permet à un hôte distant de lire ou écrire cette région même quand un RPC impliquant cette mémoire n'est pas en cours. Donc, les mises en œuvre devraient éviter d'annoncer une mémoire enregistrée de façon persistante.

8.1.3 Protection de mémoire

Les demandeurs devraient enregistrer les régions de mémoire pour l'accès à distance seulement quand ils vont être la cible d'une opération RPC qui implique un RDMA Read ou Write.

Les régions de mémoire enregistrées devraient être invalidées aussitôt que les opérations RPC qui les concernent sont achevées. L'invalidation et la détransposition DMA de régions de mémoire devrait être achevée avant la vérification de l'intégrité du message et avant qu'il soit permis au consommateur RPC de continuer l'exécution et l'utilisation ou l'altération du contenu d'une région de mémoire.

Une transaction RPC sur un demandeur pourrait être terminée avant qu'arrive une réponse si le consommateur RPC quitte de façon inattendue (par exemple, elle est signalée ou une faute de segmentation se produit). Quand un RPC se termine de

façon anormale, les régions de mémoire associées à ce RPC devraient être invalidées de façon appropriée avant que les régions soient libérées pour être réutilisées à d'autres fins chez le demandeur.

8.1.4 Déni de service

Une discussion détaillée des expositions au déni de service qui peuvent résulter de l'utilisation d'un transport RDMA se trouve au paragraphe 6.4 de la [RFC5042].

Un répondeur n'est pas obligé de tirer les tronçons en lecture qui sont d'une longueur déraisonnable. Le répondeur peut utiliser une réponse RDMA_ERROR pour terminer les RPC avec des tronçons de lecture illisibles. Si un répondeur transmet plus de données qu'un demandeur n'est prêt à recevoir dans un tronçon Write ou Reply, les cartes d'interface de réseau RDMA (RNIC, *RDMA Network Interface Card*) terminent normalement la connexion. Pour la discussion de ce point, voir le paragraphe 4.5. De telles erreurs de tronçon répétées de la part du demandeur fautif peuvent dénier le service aux autres utilisateurs qui partagent la connexion.

Une mise en œuvre de transport RPC sur RDMA n'est pas responsable de la réduction du taux de demandes RPC, en dehors de maintenir le nombre de transactions RPC concurrentes au nombre de crédits accordé par connexion, ou au dessous. Ceci est expliqué au paragraphe 3.3.1. Un envoyeur peut déclencher un auto déni de service en excédant de façon répétée le crédit accordé.

Quand un RPC a été annulé du fait d'un signal ou de la sortie prématurée d'un processus d'application, un demandeur peut invalider les tronçons Write et Reply du RPC. L'invalidation empêche l'arrivée ultérieure de la réponse du répondeur d'altérer les régions de mémoire associées à ces tronçons après la réutilisation de la mémoire.

Chez le demandeur, une application qui fonctionne mal ou un utilisateur malveillant peut créer une situation où les RPC sont continuellement initiés et ensuite interrompus, résultant en réponses du répondeur qui terminent de façon répétée la connexion RPC sur RDMA sous-jacente. De telles situations peuvent dénier le service aux autres utilisateurs qui partagent la connexion avec ce demandeur.

8.2 Sécurité du message RPC

ONC RPC fournit la sécurité cryptographique via le cadre RPCSEC_GSS [RFC7861]. RPCSEC_GSS met en œuvre l'authentification de message (rpc_gss_svc_none) la vérification d'intégrité par message (rpc_gss_svc_integrity) et la confidentialité par message (rpc_gss_svc_privacy) dans la couche au dessus de RPC sur RDMA. Les deux derniers services exigent un calcul et un mouvement de données significatifs sur chaque hôte de point d'extrémité. Certains avantages de performances réalisés par les transports RDMA peuvent être perdus.

8.2.1 Protection RPC sur RDMA aux couches inférieures

Pour tout transport RPC, utiliser les services RPCSEC_GSS d'intégrité ou de confidentialité a des implications de performances. La protection en dessous du transport RPC est souvent plus appropriée dans les déploiements sensibles aux performances, en particulier si elle aussi peut être téléchargée. Certaines configurations de IPsec peuvent être co-localisées dans le matériel RDMA, par exemple, sans changement pour les consommateurs RDMA et peu de perte d'efficacité du mouvement des données. De tels arrangements peuvent aussi fournir un haut degré de confidentialité en cachant l'identité du point d'extrémité ou en altérant la fréquence à laquelle les messages sont échangés, au prix des performances.

L'utilisation de la protection dans une couche inférieure PEUT être négociée par l'utilisation d'une nuance de sécurité RPCSEC_GSS définie dans la [RFC7861] en conjonction avec le mécanisme de lien de canal [RFC5056] et le verrouillage de connexion de canal IPsec [RFC5660]. L'utilisation de ces mécanismes est EXIGÉE lorsque l'intégrité ou la confidentialité est désirée et lorsque l'efficacité est requise.

8.2.2 RPCSEC_GSS sur transports RPC sur RDMA

Tous les appareils et tissus RDMA ne prennent pas en charge les mécanismes de protection ci-dessus. Aussi, l'authentification par message est encore requise sur les clients NFS où plusieurs utilisateurs accèdent aux fichiers NFS. Dans ces cas, RPCSEC_GSS peut protéger le trafic NFS convoyé sur les connexions RPC sur RDMA.

RPCSEC_GSS étend le protocole ONC RPC [RFC5531] sans changer le format des messages RPC. En observant les

conventions décrites dans ce paragraphe, un transport RPC sur RDMA peut convoyer de façon inter opérable les messages RPC protégés par RPCSEC_GSS.

Au titre du protocole ONC RPC, les éléments de protocole de RPCSEC_GSS qui apparaissent dans le flux de charge utile d'un message RPC sur RDMA (comme les messages de contrôle échangés au titre de l'établissement ou de la suppression d'un contexte de sécurité ou d'éléments de données qui font partie du matériel d'authentification de RPCSEC_GSS) NE DOIVENT PAS être réduits.

8.2.2.1 Négociation de contexte RPCSEC_GSS

Certaines mises en œuvre de client NFS utilisent une connexion séparée pour établir un contexte de service générique de sécurité (GSS, *Generic Security Service*) pour les opérations NFS. Ces clients utilisent TCP et l'accès NFS standard (2049) pour l'établissement de contexte. Pour permettre l'utilisation de RPCSEC_GSS avec NFS/RDMA, un serveur NFS DOIT aussi fournir un service NFS fondé sur TCP à l'accès 2049.

8.2.2.2 RPC sur RDMA avec authentification RPCSEC_GSS

Le service d'authentification RPCSEC_GSS n'a pas d'impact sur l'éligibilité à DDP des éléments de données dans un ULP.

Cependant, le matériel d'authentification RPCSEC_GSS qui apparaît dans un en-tête de message RPC peut être plus grand que, disons, un authentifiant AUTH_SYS. En particulier, quand une pseudo nuance RPCSEC_GSS est utilisée, un demandeur doit s'accommoder d'un accréditif RPC plus grand quand il réarrange les messages d'appel RPC et a besoin de fournir un vérificateur de taille maximum de RPCSEC_GSS quand il alloue les mémoires tampons de réponse et les tronçons de réponse.

Les messages RPC, et donc par suite les flux de charge utile, sont rendus plus grands. Les opérations d'ULP qui tiennent dans un message court quand une forme plus simple d'authentification est utilisée pourraient devoir être réduites, ou convoyées via un message long, quand l'authentification RPCSEC_GSS est utilisée. Il est plus probable qu'un demandeur fournisse à la fois une liste de lecture et un tronçon de réponse dans le même en-tête RPC sur RDMA pour convoyer un appel long et provisionne un réceptacle pour une réponse longue. Une utilisation plus fréquente de messages longs peut impacter l'efficacité du transport.

8.2.2.3 RPC sur RDMA avec intégrité ou confidentialité RPCSEC_GSS

Le service d'intégrité de RPCSEC_GSS permet aux points d'extrémité de détecter en vol la modification des messages RPC. Le service de confidentialité de RPCSEC_GSS empêche tout receveur autre que celui prévu de voir le contenu en clair des arguments et résultats RPC. Les services d'intégrité et de confidentialité de RPCSEC_GSS sont de bout en bout. Ils protègent les arguments et résultats RPC de l'application au point d'extrémité de serveur et retour.

Les services d'intégrité et de chiffrement RPCSEC_GSS opèrent sur les messages RPC entiers après qu'ils ont été codés en XDR pour la transmission, et avant qu'ils aient été décodés de XDR après réception. Les deux points d'extrémité d'envoyeur et de receveur utilisent des mémoires tampons intermédiaires pour empêcher l'exposition des données chiffrées ou de données en clair non vérifiées aux consommateurs RPC. Après que la vérification, le chiffrement, et l'enveloppement du message ont été accomplis, la couche transport PEUT utiliser le transfert de données RDMA entre ces mémoires tampons intermédiaires.

Le processus de réduction d'un élément de données éligible à DDP supprime l'élément de données et son bourrage XDR du flux XDR codé. Le bourrage XDR d'un élément de données réduit n'est pas transféré dans un message RPC sur RDMA. Après réduction, le flux de charge utile contient moins d'octets que le flux XDR complet n'en avait avant. Les octets de bourrage XDR sont souvent des octets de zéros, mais ils ne sont pas obligés de l'être. Donc, réduire les éléments éligibles à DDP affecte le résultat de la vérification d'intégrité ou le chiffrement du message.

Donc, un envoyeur NE DOIT PAS réduire un flux de charge utile quand les services d'intégrité ou de chiffrement RPCSEC_GSS sont utilisés. Effectivement, aucun élément de données n'est éligible à DDP dans cette situation, et les tronçons de messages ne peuvent pas être utilisés. Dans ce mode, un transport RPC sur RDMA opère de la même manière qu'un transport qui ne prend pas en charge DDP.

Quand un service d'intégrité ou de confidentialité RPCSEC_GSS est utilisé, un demandeur fournit une liste de lecture et un tronçon de réponse dans le même en-tête RPC sur RDMA pour convoyer un appel long et provisionne un réceptacle pour

une réponse longue.

8.2.2.4 Protection des en-têtes de transport RPC sur RDMA

Comme les champs de base dans un message ONC RPC (XID, direction d'appel, et ainsi de suite) le contenu d'un flux de transport de message RPC sur RDMA n'est pas protégé par RPCSEC_GSS. Cela expose les XID, les limites de crédit de connexion, et les listes de tronçons (mais pas le contenu des éléments de données auxquels ils se réfèrent) à des comportements malveillants, qui pourraient rediriger les données qui sont transférées par le message RPC sur RDMA, résulter en retransmissions parasites, ou déclencher une perte de connexion.

En particulier, si un attaquant altère les informations contenues dans les listes de tronçons d'un en-tête RPC sur RDMA, les données contenues dans ces tronçons peuvent être redirigées sur d'autres régions de mémoire enregistrées chez les demandeurs. Un attaquant pourrait altérer les arguments des opérations RDMA Read et RDMA Write sur le réseau avec un effet similaire. Si de telles altérations se produisent, l'utilisation des services d'intégrité ou confidentialité de RPCSEC_GSS permet à un demandeur de détecter le matériel inattendu dans un message RPC reçu.

Le chiffrement aux couches inférieures, comme décrit au paragraphe 8.2.1, protège le contenu du flux de transport. Pour traiter les attaques contre les protocoles RDMA eux-mêmes, les mises en œuvre de transport RDMA devraient se conformer à la [RFC5042].

9. Considérations relatives à l'IANA

Un ensemble d'identifiants de réseau RPC (netid) pour résoudre les services RPC sur RDMA est spécifié par le présent document. Ceci n'est pas changé par rapport à la [RFC5666].

Le transport RPC sur RDMA a eu un netid RPC alloué, qui est une chaîne rpcbind [RFC1833] utilisée pour décrire le protocole sous-jacent afin que RPC choisisse le tramage de transport approprié, ainsi que le format des adresses et accès de service.

Les chaînes du registre de netid suivantes sont définies à cette fin :

NC_RDMA "rdma"
NC_RDMA6 "rdma6"

Le netid "rdma" est à utiliser quand l'adressage IPv4 est employé par le transport sous-jacent, et "rdma6" pour l'adressage IPv6. La politique et le registre d'allocation de netid sont définis dans la [RFC5665].

Ces netids PEUVENT être utilisés pour tout réseau RDMA qui satisfait les exigences du paragraphe 2.3.2 et qui est capable d'identifier les points d'extrémité de service en utilisant l'adressage d'accès IP, éventuellement en utilisant un service de traduction comme décrit à la Section 5.

L'utilisation du protocole RPC sur RDMA n'a pas d'effet sur les numéros de programme RPC ou les numéros d'accès enregistrés existants. Cependant, de nouveaux numéros d'accès PEUVENT être enregistrés pour être utilisés par des services à capacité RPC sur RDMA, comme approprié pour les nouveaux réseaux sur lesquels les services vont opérer.

Par exemple, le service NFS/RDMA défini dans la [RFC5667] a reçu l'accès 20049 dans le "Registre des noms de service et des numéros d'accès de protocole de transport". Ceci est distinct du numéro d'accès défini pour NFS sur TCP, à qui est alloué l'accès 2049 dans le même registre. Les clients NFS utilisent le même numéro de programme RPC pour NFS (100003) quand ils utilisent l'un ou l'autre transport [RFC5531] (voir le registre "Numéros de programme d'appel de procédure à distance (RPC)").

10. Références

10.1 Références normatives

[RFC1833] R. Srinivasan, "Protocoles de liaison pour RPC ONC version 2", août 1995. *(P.S.) (MàJ par* RFC5665)

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. DOI 10.17487/RFC2119, *(MàJ par* RFC8174)

[RFC4506] M. Eisler, éd., "XDR : norme de représentation des données externes", mai 2006. (STD0067)

[RFC5042] J. Pinkerton, E. Deleganes, "Sécurité du protocole de placement direct des données (DDP) / protocole d'accès direct à une mémoire distante (RDMAP)", octobre 2007. *(P.S. ; MàJ par* RFC7146*)*

[RFC5056] N. Williams, "Sur l'utilisation des liens de canaux pour sécuriser les canaux", novembre 2007. *(P.S.)*

[RFC5531] R. Thurlow, "RPC : version 2 de la spécification du protocole d'appel de procédure à distance", mai 2009. *(P.S. ; remplace* RFC1831 ; *MàJ par* RFC9289*)*

[RFC5660] N. Williams, "Canaux IPsec : verrouillage de connexion**",** octobre 2009. *(P. S.)*

[RFC5665] M. Eisler, "Considérations de l'IANA pour les formats d'identifiant de réseau et d'adresse universelle dans les appels de procédure distante (RPC) ", janvier 2010. (*MàJ* RFC1833) *(P.S.)*

[RFC7861] W. Adamson, N. Williams, "Sécurité de l'appel de procédure distante (RPC) version 3", DOI 10.17487/RFC7861, novembre 2016. *(P.S. ; MàJ* RFC5403*)*

[RFC8174] B. Leiba, "Ambiguïté des mots clés en majuscules ou minuscules dans la RFC2119", DOI 10.17487/RFC8174, mai 2017. BCP14. *(MàJ RFC*2119)

10.2 Références pour information

[IBARCH] InfiniBand Trade Association, "InfiniBand Architecture Specification Volume 1", Release 1.3, mars 2015, <http://www.infinibandta.org/content/pages.php?pg=technology_download>.

[RFC0768] J. Postel, "Protocole de datagramme d'utilisateur (UDP)", (STD 6), DOI 10.17487/RFC0768, août 1980.

[RFC0793] J. Postel (éd.), "Protocole de commande de transmission – Spécification du protocole du programme Internet DARPA", STD 7, DOI 10.17487/RFC0793, septembre 1981. (*Remplacée par* RFC9293)

[RFC1094] Sun Microsystems, "NFS : Spécification du protocole de système de fichiers réseau (NFS)", mars 1989.

[RFC1813] B. Callaghan, B. Pawloowski et P. Staubach, "Spécification du protocole NFS version 3", juin 1995. *(Information)*

[RFC5040] R. Recio et autres, "Spécification d'un protocole d'accès direct à une mémoire distante", octobre 2007. *(P.S. ; MàJ par* RFC7146 *)*

[RFC5041] H. Shah et autres, "Placement direct des données sur transports fiables", octobre 2007. *(P.S. ; MàJ par RFC* 7146*)*

[RFC5532] T. Talpey, C. Juszczak, "Position du problème de l'accès direct en mémoire distante (RDMA) de système de fichier réseau (NFS)", mai 2009. *(Information)*

[RFC5661] S. Shepler, M. Eisler, D. Noveck, "Système de fichiers réseau (NFS) version 4.1 : Protocole", janvier 2010. *(P.S. ; MàJ par* RFC8178, RFC8434 *; remplacée par* RFC8881)

[RFC5662] S. Shepler, M. Eisler, D. Noveck, "Système de fichiers réseau (NFS) version 4.1 : Description de la norme de représentation des données externes (XDR)", DOI 10.17487/RFC5662, janvier 2010. *(P.S.)*

[RFC5666] T. Talpey, B. Callaghan, "Transport de l'accès direct en mémoire distante pour les appels de procédure distante", janvier 2010. *(P.S. ; remplacée par* RFC8166)

[RFC5667] T. Talpey, B. Callaghan, "Placement des données directes dans le système de fichiers réseau (NFS)", janvier

2010. *(P.S. ; remplacée par* RFC8267)

[RFC7530] T. Haynes, D. Noveck, "Protocole du système de fichier réseau (NFS) version 4", DOI 10.17487/RFC7530, mars 2015. *(P.S. ; Remplace* 3580, *MàJ par* RFC7931, RFC8587)

[RFC8167] C. Lever, "Appel de procédure distante bidirectionnel sur transport RPC sur RDMA", DOI 10.17487/RFC8167, juin 2017. *(P.S.)*

Appendice A. Changements par rapport à la RFC 5666

A.1 Changements à la spécification

Les altérations suivantes ont été faites à la spécification de RPC sur RDMA version 1. Les numéros de section ci-dessous se réfèrent à la [RFC5666].

- La Section 2 a été expansée pour introduire et expliquer la terminologie de RPC [RFC5531], XDR [RFC4506], et RDMA [RFC5040]. Ces termes sont maintenant utilisés de façon cohérente tout au long de la spécification.
- La Section 3 a été réorganisée et partagée en paragraphes pour aider le lecteur à localiser les exigences et définitions spécifiques.
- Les Sections 4 et 5 ont été combinées pour améliorer l'organisation de ces informations.
- Le protocole facultatif de configuration de connexion n'a jamais été mis en œuvre. La spécification de CCP a été supprimée de cette spécification.
- Une section consolidant les exigences pour les ULB a été ajoutée.
- Un mécanisme d'extraction de XDR est fourni, avec les droits de reproduction complets, correspondant à l'approche utilisée dans la [RFC5662].
- La section "Considérations sur la sécurité" a été étendue pour inclure une discussion de comment la sécurité de RPC sur RDMA dépend des caractéristiques du transport RDMA sous-jacent.
- Un paragraphe décrivant l'utilisation de RPCSEC_GSS [RFC7861] avec RPC sur RDMA version 1 a été ajouté.

A.2 Changements au protocole

Bien que le protocole décrit ici inter opère avec les mises en œuvre existantes de la [RFC5666], les changements suivants ont été faits par rapport au protocole qui y est décrit :

- La prise en charge du modèle de transfert Read-Read a été supprimée. Read-Read est un modèle de transfert plus lent que Read-Write. Par suite, les mises en œuvre ont choisi de ne pas le prendre en charge. La suppression de Read-Read simplifie le texte explicatif, et la procédure RDMA_DONE ne fait plus partie du protocole.
- La spécification de RDMA_MSGP dans la [RFC5666] n'est pas adéquate, bien que certaines mises en œuvre incomplètes existent. Même si une spécification adéquate était fournie et si une mise en œuvre était produite, l'avantage pour des protocoles comme NFSv4.0 [RFC7530] serait douteux. Donc, le type de message RDMA_MSGP n'est plus pris en charge.
- Des problèmes techniques à l'égard du traitement des erreurs d'en-tête RPC sur RDMA ont été corrigés.
- Des exigences spécifiques relatives à l'arrondi implicite XDR et aux types complexes de données XDR ont été ajoutées.
- Des lignes directrices explicites sont fournies sur le dimensionnement des tronçons en écriture, la gestion de plusieurs tronçons dans la liste d'écriture, et le traitement des tronçons en écriture non utilisés.
- Des lignes directrices claires sur les tailles de mémoire tampon d'envoi et de réception ont été introduites. Cela permet de meilleures décisions sur quand un tronçon de réponse doit être fourni.

Remerciements

L'éditeur exprime sa gratitude pour les travaux de Brent Callaghan et Tom Talpey sur la spécification originale de RPC sur RDMA version 1 [RFC5666].

Dave Noveck a fourni une excellente relecture, des suggestions constructives, et des conseils de navigation cohérents tout au long du processus d'élaboration du présent document. Dave a aussi beaucoup contribué à l'organisation et au contenu de la Section 7 et aidé les auteurs à comprendre les complexités de l'extensibilité de XDR.

Les commentaires et contributions de Karen Deitke, Dai Ngo, Chunli Zhang, Dominique Martinet, et Mahesh Siddheshwar ont été acceptés avec tous nos remerciements. L'éditeur souhaite aussi remercier Bill Baker, Greg Marsden, et Matt Benjamin de leur soutien à ce travail.

L'extrait de script .sh shell et les conventions de formatage ont d'abord été décrits par les auteurs de la spécification XDR de NFSv4.1 [RFC5662].

Des remerciements particuliers au directeur de la zone Transport, Spencer Dawkins, au président du groupe de travail NFSV4 et berger du document Spencer Shepler, et au secrétaire du groupe de travail NFSV4 Thomas Haynes pour leur soutien.

Adresse des auteurs

Charles Lever (éditeur)
Oracle Corporation
1015 Granger Avenue
Ann Arbor, MI 48104
United States of America
téléphone : +1 248 816 6463
mél : chuck.lever@oracle.com

William Allen Simpson
Red Hat
1384 Fontaine
Madison Heights, MI 48071
United States of America
mél : william.allen.simpson@gmail.com

Tom Talpey
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052
United States of America
téléphone : +1 425 704-9945
mél : ttalpey@microsoft.com