Groupe de travail Réseau
Request for Comments : 879
Traduction Claude Brière de L'Isle

J. Postel, ISI
novembre 1983

Taille maximum de segment TCP et problèmes qui s'y rapportent

Le présent mémoire discute de l'option TCP Taille maximum de segment et des questions connexes. Il a pour objet de préciser certains aspects de TCP et de son interaction avec IP. Le présent mémoire apporte des précisions à la spécification de TCP, et contient des informations qui peuvent être considérées comme un "avis aux mises en œuvre".

1. Introduction

Le présent mémoire discute de la taille maximum de segment TCP et de sa relation avec la taille maximum de datagramme IP. TCP est spécifié dans la référence [1]. IP est spécifié dans les références [2] et [3].

Cette discussion est nécessaire parce que la spécification actuelle de cette option TCP est ambiguë.

Une grande partie de la difficulté de compréhension de ces tailles et de leurs relations est due à la taille variable des en-têtes IP et TCP.

Certaines hypothèses ont été faites sur l'utilisation de tailles autres que celle par défaut pour les datagrammes avec des résultats malheureux.

Les hôtes ne doivent pas envoyer de datagrammes de plus de 576 octets sauf si ils savent spécifiquement que l'hôte de destination est prêt à accepter des datagrammes plus grands.

Cette règle est établie depuis longtemps.

Pour résoudre l'ambiguïté de la définition de la taille de segment maximum de TCP, la règle suivante est établie :
La taille maximum de segment TCP est la taille maximum de datagramme IP MOINS quarante.
La raille maximum par défaut de datagramme IP est 576.
La taille maximum par défaut de segment TCP est 536.

2. Taille maximum de datagramme IP

Les hôtes ne sont pas obligés de réassembler des datagrammes IP infiniment grands. La taille maximum de datagramme que tous les hôtes sont obligés d'accepter ou réassembler à partir de fragments est de 576 octets. La taille maximum de mémoire tampon de réassemblage que doivent avoir tous les hôtes est de 576 octets. Il est permis aux hôtes d'accepter de plus grands datagrammes et de réassembler des fragments en de plus grands datagrammes, les hôtes peuvent avoir des mémoires tampon de la taille qui leur plait.

Les hôtes ne doivent pas envoyer des datagrammes plus grands que 576 octets sauf si ils savent de façon spécifique que l'hôte de destination est prêt à accepter de plus grands datagrammes.

3. Option Taille maximum de segment TCP

TCP fournit une option qui peut (seulement) être utilisée au moment de l'établissement d'une connexion pour indiquer la taille maximum de segment TCP qui peut être acceptée sur cette connexion. Cette annonce de taille maximum de segment (MSS, *Maximum Segment Size*) (souvent appelée à tort une négociation) est envoyée du receveur des données à l'envoyeur des données et dit "Je peux accepter des segments TCP jusqu'à la taille X". La taille (X) peut être supérieure ou inférieure à celle par défaut. La MSS peut être utilisée de façon complètement indépendante dans chaque direction du flux de données. Cela peut donner des tailles maximum assez différentes dans les deux directions.

La MSS ne compte que les octets de données dans le segment, elle ne compte pas l'en-tête TCP ou l'en-tête IP.

Note : La valeur de la MSS ne compte que les octets de données, et donc, elle ne compte pas les bits de contrôle TCP SYN et FIN même si SYN et FIN consomment bien des numéros de séquence TCP.

4. Relations entre segments TCP et datagrammes IP

Les segments TCP sont transmis comme les données dans les datagrammes IP. La correspondance entre les segments TCP et les datagrammes IP doit être biunivoque. Cela parce que TCP s'attend à trouver exactement un segment TCP complet dans chaque bloc de données qui lui sont retournés par IP, et IP doit retourner un bloc de données pour chaque datagramme reçu (ou complètement réassemblé).

5. Mise en couche et modularité

TCP est un protocole de flux de données fiable de bout en bout avec contrôle d'erreur, contrôle de flux, etc. TCP se souvient de beaucoup de choses sur l'état d'une connexion.

IP est un protocole de datagramme au coup par coup. IP n'a pas de mémoire des datagrammes transmis. Il n'est pas approprié que IP conserve des informations sur la taille maximum de datagramme qu'un hôte de destination particulier pourrait être capable d'accepter.

TCP et IP sont des couches distinctes dans l'architecture de protocole, et sont souvent mis en œuvre dans des modules de programme distincts.

Certains semblent penser qu'il ne doit y avoir aucune communication entre les couches de protocole ou les modules de programmes. Il doit y avoir une communication entre les couches et les modules, mais elle devrait être spécifiée et contrôlée avec soin. Un problème pour comprendre la vision correcte de la communication entre couches de protocoles ou modules de programme en général, ou entre TCP et IP en particulier est que les documents sur les protocoles ne sont pas très clairs sur ce point. C'est souvent parce que les documents sont sur les échanges de protocole entre les machines, et non sur l'architecture de programme au sein d'une machine, et du désir de permettre de nombreuses architectures de programmes avec différentes organisations des tâches en modules.

6. Exigences des informations IP

Il n'y a pas d'exigence générale que IP conserve des informations hôte par hôte.

IP doit prendre une décision quant à l'adresse de réseau directement rattaché à laquelle envoyer chaque datagramme. C'est simplement la transposition d'une adresse IP en une adresse de réseau directement rattaché.

Il y a deux cas à considérer : la destination est sur le même réseau, et la destination est sur un réseau différent.

Même réseau : Pour certains réseaux, l'adresse du réseau directement rattaché peut être calculée à partir de l'adresse IP de l'hôte de destination sur le réseau directement rattaché.
Pour d'autres réseaux, la transposition doit être faite par une recherche dans un tableau (cependant, le tableau est initialisé et entretenu, par exemple, [4]).

Réseau différent : L'adresse IP doit être transposée en adresse de réseau directement rattaché d'un routeur. Pour les réseaux avec un routeur pour le reste de l'Internet, l'hôte a seulement besoin de déterminer et de se souvenir de l'adresse du routeur et de l'utiliser pour envoyer tous les datagrammes aux autres réseaux.

Pour les réseaux avec plusieurs routeurs avec le reste de l'Internet, l'hôte doit décider quel routeur utiliser pour chaque datagramme envoyé. Il a seulement besoin de vérifier le réseau de destination de l'adresse IP et conserver l'information sur quel routeur utiliser pour chaque réseau.

IP conserve bien, dans certains cas, les informations d'acheminement d'hôte pour les autres hôtes sur le réseau directement rattaché. IP conserve bien, dans certains cas, les informations d'acheminement par réseau.

Cas particulier :
Il y a deux messages ICMP qui portent des informations sur des hôtes particuliers. Ce sont des sous-types des messages ICMP Destination injoignable et Redirection. Ces messages ne sont attendus que dans des circonstances très inhabituelles. Pour faire une utilisation efficace de ces messages, l'hôte receveur va devoir conserver les informations sur les hôtes spécifiques sur lesquels il est fait rapport. Parce que ces messages sont assez rares, il est vivement recommandé que cela soit fait par un mécanisme exceptionnel plutôt que en ayant IP qui tient des tableaux par hôte pour tous les hôtes.

7. Relations entre tailles de datagramme IP et de segment TCP

La relation entre la valeur de la taille maximum de datagramme IP et la taille maximum de segment TCP est obscure. Le problème est que la longueur de l'en-tête IP et l'en-tête TCP peuvent toutes deux varier. L'option TCP Taille maximum de segment (MSS) est définie comme spécifiant le nombre maximum d'octets de données dans un segment TCP qui exclut l'en-tête TCP (ou IP).

Pour notifier à l'envoyeur des données le plus grand segment TCP qu'il est possible de recevoir, le calcul de la valeur de MSS à envoyer est : MSS = MTU – taille de(en-têteTCP) - taillede(en-têteIP)

À réception de l'option MSS, le calcul de la taille du segment qui peut être envoyé est :

TailleMaxSegEnvoi = MIN((MTU - taille de(en-têteTCP) - taille de(en-têteIP)), MSS)

où MSS est la valeur dans l'option, et MTU est l'unité de transmission maximum (ou la taille maximum de paquet) permise sur le réseau directement rattaché.

Cela pose cependant la question : Quelle valeur devrait être utilisée pour la "taillede(en-têteTCP)" et pour la "taillede(en-têteIP)" ?

Il y a trois positions raisonnables à prendre : la prudente, la modérée, et la libérale.

La position prudente ou pessimiste suppose le pire – que l'en-tête IP et l'en-tête TCP ont tous deux la taille maximum, c'est-à-dire, 60 octets chacun.

MSS = MTU - 60 - 60 = MTU - 120
Si la MTU est de 576, alors MSS = 456

La position modérée suppose que IP est de taille maximum (60 octets) et que l'en-tête TCP est de taille minimum (20 octets) parce qu'il n'y a pas d'option d'en-tête TCP actuellement définie qui serait normalement envoyée en même temps que des segments de données.

MSS = MTU - 60 - 20 = MTU - 80
Si la MTU est de 576, alors MSS = 496

La position libérale ou optimiste suppose le meilleur – que les en-têtes IP et TCP sont tous deux de taille minimum, c'est-à-dire, de 20 octets chacun.

MSS = MTU - 20 - 20 = MTU - 40
Si la MTU est de 576, alors MSS = 536

Si rien n'est dit sur MSS, l'envoyeur des données peut bourrer autant que possible un datagramme de 576 octets, et si le datagramme a l'en-tête minimum (ce qui est très probable) le résultat sera de 536 octets de données dans le segment TCP. La règle qui met en relation la MSS avec la taille maximum de datagramme devrait être cohérente avec cela.

Un point pratique est soulevé en faveur de la position libérale. Comme l'utilisation des en-têtes minimum IP et TCP est très probable dans un très large pourcentage de cas, il semble inopportun de limiter le segment de données TCP à beaucoup moins que ce qui pourrait être transmis en une seule fois, spécialement si c'est moins de 512 octets.

Par comparaison, 536/576 est 93 % des données, 496/576 est 86 % des données, 456/576 est 79 % des données.

8. Taille maximum de paquet

Chaque réseau a une taille maximum de paquet, ou une unité de transmission maximum (MTU). Il y a en fin de compte des limites qui sont imposées par la technologie, mais souvent la limite est un choix d'ingénierie ou même un choix administratif. Des installations différentes du même produit réseau n'ont pas à utiliser la même taille maximum de paquet. Même au sein d'une installation, tous les hôte ne doivent pas utiliser la même taille de paquet (ce qui paraît un peu fou cependant).

Certaines mises en œuvre IP ont supposé que tous les hôtes sur le réseau directement rattaché seront les mêmes ou au moins auront la même mise en œuvre. C'est une supposition dangereuse. Il s'est souvent produit qu'après qu'un petit ensemble homogène d'hôtes sont devenus opérationnels, des hôtes supplémentaires de types différents sont introduits dans

l'environnement. Et il s'est souvent trouvé qu'on désire utiliser une copie de la mise en œuvre dans un environnement non homogène différent.

Les concepteurs de routeurs devraient être prêts au fait que les routeurs réussis seront copiés et utilisés dans d'autres situations et installations. Les routeurs doivent être prêts à accepter des datagrammes aussi grands qu'il peut être envoyé dans les paquets maximums des réseaux directement rattachés. Les mises en œuvre de routeurs devraient être facilement configurées pour être installées dans des circonstances différentes.

Note : MTU de certains réseaux populaires (noter que la limite actuelle de certaines installations peut être réglée plus bas par la politique administrative) :
ARPANET, MILNET = 1007
Ethernet (10 Mbits) = 1500
Proteon PRONET = 2046

9. Fragmentation à la source

Un hôte de source ne va normalement pas créer de datagramme fragmenté. Dans des circonstances normales, il n'y a de datagrammes fragmentés que lorsque un routeur doit envoyer un datagramme dans un réseau qui a une plus petite taille maximum de paquet que le datagramme. Dans ce cas, le routeur doit fragmenter le datagramme (sauf si il est marqué "Ne pas fragmenter", auquel cas il est éliminé, avec l'option d'envoyer un message ICMP à la source pour faire rapport du problème).

Il pourrait être souhaitable pour l'hôte de source d'envoyer un datagramme fragmenté si la taille maximum de segment (par défaut ou négociée) permise par le receveur des données était supérieure à la taille maximum de paquet permise par le réseau directement rattaché. Cependant, de tels fragments de datagramme ne doivent pas se combiner à une taille supérieure à celle permise par l'hôte de destination.

Par exemple, si le TCP receveur a annoncé qu'il acceptera des segments jusqu'à 5000 octets (en coopération avec l'IP receveur) le TCP envoyeur pourrait alors donner un segment aussi grand à l'IP envoyeur pourvu que l'IP envoyeur l'envoie dans un datagramme fragmenté qui tient dans les paquets du réseau directement rattaché.

Il y a certaines conditions où l'hôte de source devra fragmenter.
Si l'hôte est rattaché à un réseau qui a une petite taille de paquet (par exemple 256 octets) et si il prend en charge une application définie pour envoyer des messages de taille fixée supérieure à cette taille de paquet (par exemple TFTP [5]).
Si l'hôte reçoit un message Écho ICMP avec des données, il est obligé d'envoyer un message ICMP Réponse d'écho avec les mêmes données. Si la quantité de données dans l'écho est supérieure à la taille de paquet du réseau directement rattaché, les étapes suivantes peuvent être nécessaires : (1) recevoir les fragments, (2) réassembler le datagramme, (3) interpréter l'écho, (4) créer une Réponse d'écho, (5) la fragmenter, et (6) envoyer les fragments.

10. Fragmentation par le routeur

Les routeurs doivent être prêts à effectuer la fragmentation. Ce n'est pas un dispositif facultatif pour un routeur.

Les routeurs n'ont pas d'information sur la taille des datagrammes que les hôtes de destination sont prêts à accepter. Il serait inapproprié qu'un routeur tente de conserver de telles informations.

Les routeurs doivent être prêts à accepter les plus grands datagrammes qui sont permis sur chacun des réseaux auxquels ils sont directement rattachés, même si ils font plus de 576 octets.

Les routeurs doivent être prêts à fragmenter les datagrammes pour les faire tenir dans les paquets du prochain réseau, même si c'est inférieur à 576 octets.

Si un hôte de source pense tirer parti de la capacité du réseau local à porter de plus grands datagrammes mais qu'il n'a pas la plus petite idée de si l'hôte de destination peut accepter des datagrammes plus grands que la taille par défaut et s'il s'attend à ce que le routeur fragmente le datagramme en fragments de la taille par défaut, l'hôte de source est alors dans l'erreur. Si bien sûr, l'hôte de destination ne peut accepter de plus grands datagrammes que ceux de la taille par défaut, il ne peut probablement pas non plus les réassembler. Si le routeur passe entier le grand datagramme ou le fragmente en fragments de la taille par défaut, la destination ne les acceptera pas. Donc, ce mode de comportement des hôtes de source doit être proscrit.

Un datagramme supérieur à la taille par défaut ne peut arriver à un routeur que parce que l'hôte de source sait que l'hôte de destination peut traiter d'aussi grands datagrammes (probablement parce que l'hôte de destination l'a annoncé à l'hôte de source dans une option MSS de TCP). Donc, le routeur devrait passer ce grand datagramme en un seul morceau ou dans les plus grands fragments qui tiennent dans le prochain réseau.

Il est intéressant de noter que mêmes si le routeur peut connaître la règle des 576 octets, elle n'est pas pertinente pour lui.

11. Communication entre couches

Le pilote réseau (ND, *Network Driver*) ou l'interface devrait connaître l'unité de transmission maximum (MTU) du réseau directement rattaché.

IP devrait demander au pilote réseau l'unité de transmission maximum.

TCP devrait demander à IP la taille maximum de datagramme de données (MDDS, *Maximum Datagram Data Size*). C'est la MTU moins la longueur de l'en-tête IP (MDDS = MTU – LgEn-têteIP).

Lors de l'ouverture d'une connexion, TCP peut envoyer une option MSS avec la valeur égale à MDDS – LgEn-têteTCP.

TCP devrait déterminer la taille maximum de segment de données (MSDS, *Maximum Segment Data Size*) pour la valeur par défaut ou la valeur reçue de l'option MSS.

TCP devrait déterminer si la fragmentation de source est possible (en demandant à IP) et désirable.Si il en est ainsi, TCP peut traiter les segments IP (y compris l'en-tête TCP) jusqu'à MSDS + LgEn-têteTCP.
Sinon, TCP peut traiter les segments IP (y compris l'en-tête TCP) jusqu'au plus petit de (MSDS + LgEn-têteTCP) et de MDDS.

IP vérifie la longueur des données passées par TCP. Si la longueur est inférieure ou égale à MDDS, IP rattache l'en-tête IP et le passe au ND. Autrement, IP doit faire la fragmentation de source.

12. Quelle est la MSS par défaut ?

Une autre façon de poser cette question est "Quelle valeur transmise pour la MSS a exactement le même effet que de ne pas transmettre du tout l'option ?".

Dans les termes de la section précédente :
L'hypothèse par défaut est que l'unité maximum de transmission est 576 octets : MTU = 576

La taille maximum de datagramme de données (MDDS) est la MTU moins la longueur de l'en-tête IP.

MDDS = MTU – LgEn-têteIP = 576 - 20 = 556

Lors de l'ouverture d'une connexion, TCP peut envoyer une option MSS avec la valeur égale à MDDS – LgEn-têteTCP.

MSS = MDDS – LgEn-têteTCP = 556 - 20 = 536

TCP devrait déterminer la taille maximum de segment de données (MSDS) de la valeur par défaut ou de la valeur reçue de l'option MSS.

MSS par défaut = 536, alors MSDS = 536

TCP devrait déterminer si la fragmentation de source est possible et désirable.
Si oui, TCP peut passer à IP des segments (y compris l'en-tête TCP) jusqu'à MSDS + LgEn-têteTCP (536 + 20 = 556).
Sinon, TCP peut passer à IP des segments (y compris l'en-tête TCP) jusqu'au plus petit de
(MSDS + LgEn-têteTCP (536 + 20 = 556)) et MDDS (556).

13. La vérité

La règle concernant la taille maximum de datagramme IP et la taille maximum de segment TCP est :

Taille maximum de segment TCP = taille maximum de datagramme IP - 40

La règle doit correspondre au cas par défaut.

Si l'option Taille maximum de segment TCP n'est pas transmise, l'envoyeur des données est alors autorisé à envoyer des datagrammes IP de la taille maximum (576) avec un en-tête IP minimum (20) et un en-tête TCP minimum (20) et être par là capable de glisser 536 octets de données dans chaque segment TCP.

La définition de l'option MSS peut être formulée ainsi :
Le nombre maximum d'octets de données qui peuvent être reçus par l'envoyeur de cette option TCP dans les segments TCP sans option d'en-tête TCP transmise dans les datagrammes IP sans option d'en-tête IP.

14. Conséquences

Lorsque TCP est utilisé dans une situation où les en-têtes soit IP soit TCP ne sont pas minimums et où le datagramme IP maximum qui peut être reçu reste de 576 octets, l'option Taille de segment maximum TCP doit alors être utilisée pour réduire la limite sur les octets de données permise dans un segment TCP.

Par exemple, si l'option Sécurité IP (11 octets) est utilisée et si la taille maximum de datagramme IP reste à 576 octets, TCP devrait alors envoyer la MSS avec une valeur de 525 (536-11).

15. Références

[1] [RFC0793] J. Postel (éd.), "Protocole de commande de transmission – Spécification du protocole du programme Internet DARPA", STD 7, septembre 1981.

[2] [RFC0791] J. Postel, éd., "Protocole Internet - Spécification du protocole du programme Internet", STD 5, septembre 1981.

[3] [RFC0792] J. Postel, "Protocole du message de contrôle Internet – Spécification du protocole du programme Internet DARPA", STD 5, septembre 1981. *(MàJ par la* RFC6633)

[4] [RFC0826] D. Plummer, "Protocole de résolution d'adresses Ethernet : conversion des adresses de protocole réseau en adresses Ethernet à 48 bits pour transmission sur un matériel Ethernet", STD 37, novembre 1982.

[5] [RFC0783] K. Sollins, "Protocole TFTP (révision 2)", juin 1981. *(rendue obsolète par la* RFC1350*)*