Résolution des enjeux liés à la fragmentation d’IPv4, à MTU, à MSS et à PMTUD avec GRE et IPsec.

Table des matières

Introduction

Informations générales

Fragmentation et réassemblage IPv4

- Problèmes avec la fragmentation IPv4
- Éviter la fragmentation IPv4 : Fonctionnement de TCP MSS
  - Exemple 1
  - Exemple 2

Qu’est-ce que la PMTUD

- Exemple 3
- Exemple 4
- Problèmes survenant avec la découverte PMTUD
- Topologies réseau courantes nécessitant la découverte PMTUD

Tunnel

- Considérations concernant les interfaces de tunnel
- Routeur participant à la PMTUD au point d’extrémité d’un tunnel
  - Exemple 5
  - Exemple 6

Mode tunnel IPsec pur

- Exemple 7
- Exemple 8

Utilisation combinée de IPv4sec et de GRE

- Exemple 9
- Exemple 10

Autres recommandations

Informations connexes

Introduction

Ce document décrit le fonctionnement de la fragmentation IPv4 et de PMTUD (Path Maximum Transmission Unit Discovery).

Informations générales

Cette section traite également des scénarios qui impliquent le comportement de PMTUD lorsqu'il est associé à différentes combinaisons de tunnels IPv4.

Fragmentation et réassemblage IPv4

Bien que la longueur maximale d'un datagramme IPv4 soit de 65 535, la plupart des liens de transmission imposent une limite de longueur de paquet inférieure appelée MTU (Maximum Transmission Unit). La valeur MTU dépend de la liaison de transmission.

La conception d'IPv4 prend en compte les différences de MTU, car elle permet aux routeurs de fragmenter les datagrammes IPv4 selon les besoins.

La station réceptrice (dans ce contexte) est responsable du réassemblage des fragments dans le datagramme IPv4 d'origine de taille réelle.

La fragmentation IPv4 divise un datagramme en éléments qui seront réassemblés ultérieurement.

Les champs source, destination, identification, longueur totale et décalage de fragment IPv4, ainsi que les indicateurs « more fragments » (MF) et « do not fragment » (DF) dans l'en-tête IPv4, sont utilisés pour la fragmentation et le réassemblage IPv4.

Pour de plus amples renseignements concernant les aspects mécaniques de la fragmentation et du réassemblage des paquet IPv4, consultez le document RFC 791.

L'illustration suivante décrit la structure d'un en-tête IPv4.

Original IP Datagram

Sequence	Identifier	Total Length	DF May / Don't	MF Last / More	Fragment Offset
0	345	5140	0	0	0

IP Fragments (Ethernet)

Sequence	Identifier	Total Length	DF May / Don't	MF Last / More	Fragment Offset
0-0	345	1500	0	1	0
0-1	345	1500	0	1	185
0-2	345	1500	0	1	370
0-3	345	700	0	0	555

L'identification est de 16 bits et est une valeur attribuée par l'expéditeur d'un datagramme IPv4. Cela facilite le réassemblage des fragments d'un datagramme.

Le champ Décalage de fragment mesure 13 bits; il indique la position du fragment dans le datagramme IPv4 original. Cette valeur est un multiple de 8 octets.

Le champ des indicateurs de l'en-tête IPv4 contient 3 bits pour les indicateurs de contrôle. Le bit « Ne pas fragmenter » (DF) détermine si un paquet peut ou non être fragmenté.

Le bit 0 est réservé et toujours défini sur 0.

Le bit 1 est le bit DF (0 = "peut fragmenter", 1 = "ne pas fragmenter").

Le bit 2 représente le bit MF (0 = « dernier fragment », 1 = « autres fragments »).

Valeur	Bit 0 réservé	Bit 1 DF	Bit 2 MF
0	0	Peut	Dernier
1	0	Ne faites pas	Plus

Si les longueurs des fragments IPv4 sont ajoutées, la valeur dépasse de 60 la longueur du datagramme IPv4 d'origine.

Cette augmentation de 60 tient au fait que trois en-têtes IPv4 supplémentaires ont été créés, une pour chaque fragment succédant au premier fragment.

Le premier fragment a un décalage de 0, la longueur de ce fragment est 1500 ; Ce nombre comprend les 20 octets de l’en-tête IPv4 original qui a été légèrement modifié.

Le deuxième fragment a un décalage de 185 (185 x 8 = 1480) ; la partie données de ce fragment commence à 1 480 octets dans le datagramme IPv4 d'origine.

La longueur de ce fragment est 1500 ; Ce nombre inclut l’en-tête IPv4 supplémentaire créé pour ce fragment.

Le troisième fragment a un décalage de 370 (370 x 8 = 2960) ; la partie données de ce fragment commence à 2 960 octets dans le datagramme IPv4 d'origine.

La longueur de ce fragment est 1500 ; Ce nombre inclut l’en-tête IPv4 supplémentaire créé pour ce fragment.

Le quatrième fragment a un décalage de 555 (555 x 8 = 4 440), ce qui signifie que la partie données de ce fragment commence à 4 440 octets du début du datagramme IPv4 original.

La longueur de ce fragment est de 700 octets ; Ce nombre inclut l’en-tête IPv4 supplémentaire créé pour ce fragment.

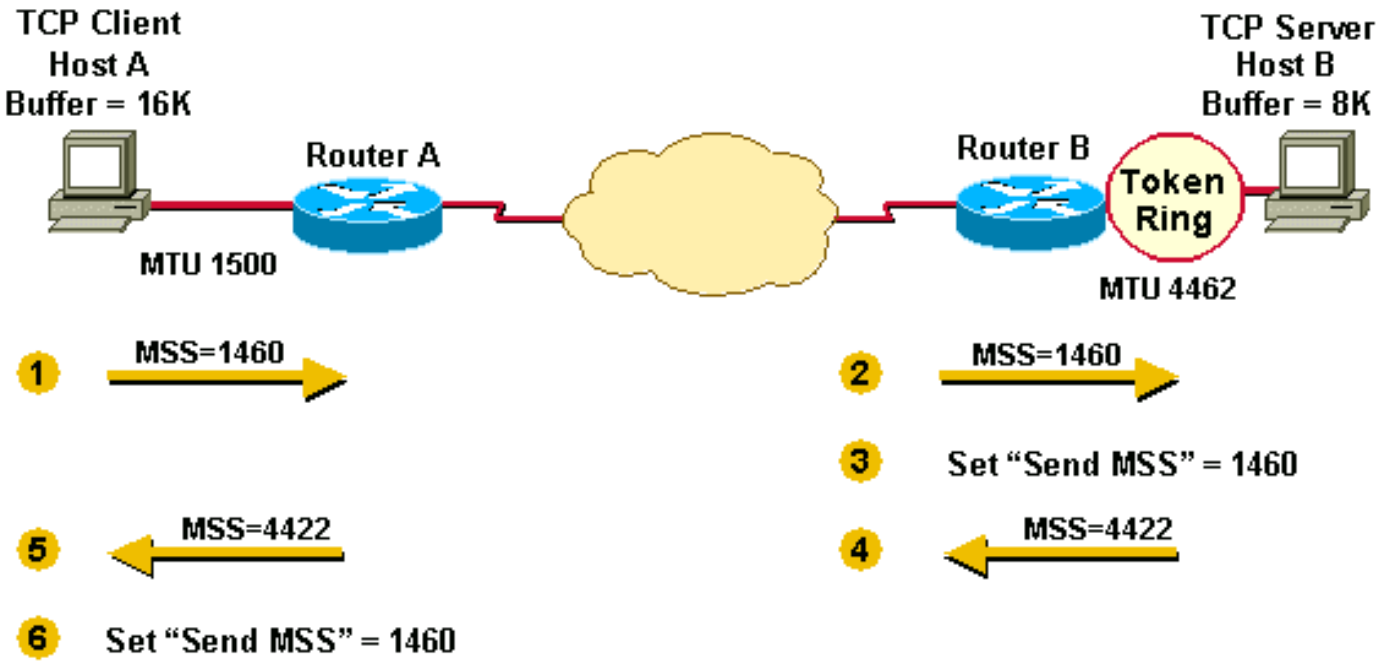
La taille originale d’un datagramme IPv4 ne peut être déterminée que lorsque le dernier fragment a été reçu.

Le champ Décalage de fragment du dernier fragment (555) fait état d’un décalage de 4 440 octets dans le datagramme IPv4 original.

La somme des octets de données du dernier fragment (680 = 700 - 20) donne 5 120 octets, qui est la partie données du datagramme IPv4 d'origine.

L'ajout de 20 octets pour un en-tête IPv4 équivaut à la taille du datagramme IPv4 d'origine (440 +

680 + 20 = 5140) comme indiqué dans les images.



Problèmes avec la fragmentation IPv4

La fragmentation IPv4 se traduit par une légère augmentation de la charge CPU et mémoire pour fragmenter un datagramme IPv4. Cela est vrai pour l’expéditeur et pour un routeur sur le chemin entre un expéditeur et un destinataire.

La création de fragments implique la création d'en-têtes de fragments et copie le datagramme d'origine dans les fragments.

Ceci est fait efficacement parce que les informations nécessaires pour créer les fragments sont immédiatement disponibles.

La fragmentation entraîne davantage de surcharge pour le récepteur lors du réassemblage des fragments, car le récepteur doit allouer de la mémoire pour les fragments en entrée et les fusionner de nouveau dans un datagramme une fois tous les fragments reçus.

Le réassemblage sur un hôte n'est pas considéré comme un problème, parce que l'hôte possède les ressources temporelles et de mémoire suffisantes pour l'exécution de cette tâche.

Cependant, le réassemblage est inefficace sur un routeur dont la tâche principale est de transférer les paquets aussi rapidement que possible.

Les routeurs ne sont pas conçus pour la rétention de paquets pendant une durée indéterminée.

Un routeur qui effectue le réassemblage choisit la plus grande mémoire tampon disponible (18 Ko), car il n'a aucun moyen de déterminer la taille du paquet IPv4 d'origine jusqu'à la réception du dernier fragment.

Un autre problème relatif à la fragmentation concerne la manipulation des fragments ignorés.

Si un fragment d'un datagramme IPv4 est abandonné, l'intégralité du datagramme IPv4 d'origine doit être présente et il est également fragmenté.

Cela est visible avec le système de fichiers réseau (NFS). NFS a une taille de bloc de lecture et d'écriture de 8192.

Par conséquent, un datagramme NFS IPv4/UDP contient environ 8 500 octets (ce qui inclut les en-têtes NFS, UDP et IPv4).

Une station émettrice connectée à un réseau Ethernet (MTU 1500) doit fragmenter le datagramme de 8 500 octets en six (6) parties ; Cinq (5) fragments de 1 500 octets et un (1) fragment de 1 100 octets.

Si l'un des six fragments est abandonné en raison d'une liaison encombrée, le datagramme d'origine complet doit être retransmis. Il en résulte six fragments supplémentaires à créer.

Si cette liaison abandonne un paquet sur six, les chances sont faibles que des données NFS soient transférées sur cette liaison, car au moins un fragment IPv4 serait abandonné de chaque datagramme IPv4 d'origine NFS de 8 500 octets.

Les pare-feu qui filtrent ou manipulent les paquets en fonction des informations de couche 4 (L4) à 7 (L7) ont des difficultés à traiter correctement les fragments IPv4.

Si les fragments IPv4 sont dans le désordre, un pare-feu bloque les fragments non initiaux, car ils ne transportent pas les informations correspondant au filtre de paquets.

Cela signifie que le datagramme IPv4 d'origine n'a pas pu être réassemblé par l'hôte récepteur.

Si le pare-feu est configuré pour autoriser les fragments non initiaux avec des informations insuffisantes pour correspondre correctement au filtre, une attaque de fragment non initial à travers le pare-feu est possible.

Les périphériques réseau, tels que les moteurs de commutation de contenu, dirigent les paquets en fonction des informations de couches 4 à 7. Si un paquet s'étend sur plusieurs fragments, le périphérique a des difficultés à appliquer ses politiques.

Éviter la fragmentation IPv4 : Fonctionnement de TCP MSS

La taille maximale de segment (MSS) du protocole TCP (Transmission Control Protocol) définit la quantité maximale de données qu'un hôte accepte dans un datagramme TCP/IPv4 unique.

Ce datagramme TCP/IPv4 est probablement fragmenté au niveau de la couche IPv4. La valeur MSS est envoyée comme en-tête TCP uniquement dans les segments TCP SYN.

Chaque côté d'une connexion TCP indique sa valeur MSS à l'autre côté. La valeur MSS n'est pas négociée entre les hôtes.

L'hôte expéditeur doit limiter la taille des données dans les segments TCP à une valeur inférieure ou égale au MSS indiqué par l'hôte de destination.

Initialement, la taille maximale d'un segment (MSS) représentait la taille de la mémoire tampon (supérieure ou égale à 65 496 octets) qui devait être prévue par l'hôte destinataire pour emmagasiner les données TCP contenues dans un seul datagramme IPV4.

MSS était le segment de données maximal que le récepteur TCP allait accepter. Ce segment TCP peut atteindre 64 Ko et être fragmenté au niveau de la couche IPv4 afin d'être transmis à l'hôte récepteur.

L'hôte destinataire devrait réassembler le datagramme IPv4 avant de transmettre le segment TCP entier à la couche TCP.

Comment les valeurs MSS sont définies et utilisées pour limiter la taille des segments TCP et des datagrammes IPv4.

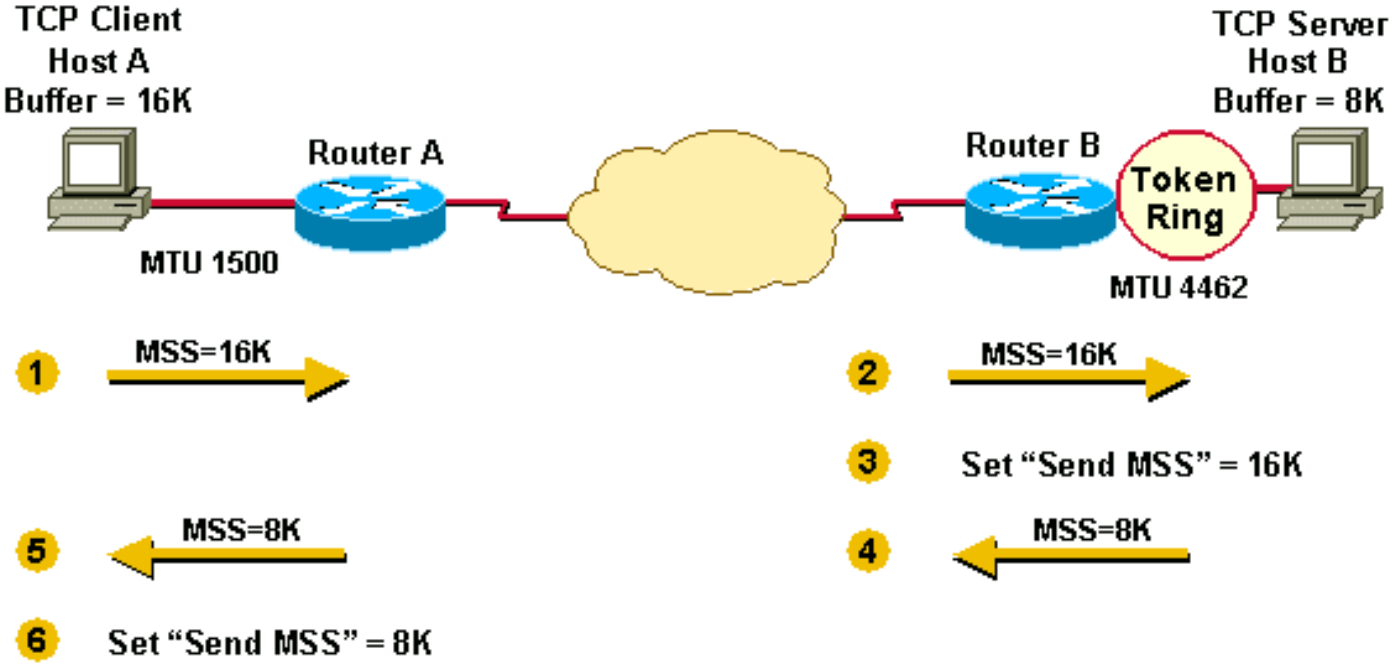
L'exemple 1 illustre la façon dont MSS a été implémenté pour la première fois.

L'hôte A a une mémoire tampon de 16K et l'hôte B une mémoire tampon de 8K. Ils envoient et reçoivent leurs valeurs MSS et ajustent leur envoi MSS en vue de l'envoi réciproque de données.

L'hôte A et l'hôte B doivent fragmenter les datagrammes IPv4 qui sont plus grands que la MTU de l'interface, mais inférieurs à la MSS d'envoi, car la pile TCP transmet 16 000 ou 8 000 octets de données vers IPv4.

Dans le cas de l'hôte B, les paquets sont fragmentés pour atteindre le réseau local Token Ring et à nouveau pour atteindre le réseau local Ethernet.

Exemple 1



1. L'hôte A envoie sa valeur MSS de 16K à l'hôte B.
2. L'hôte B reçoit la valeur 16K MSS envoyée par l'hôte A.
3. L'hôte B définit sa valeur MSS à 16K.

4. L'hôte B envoie sa valeur MSS de 8K à l'hôte A.
5. L'hôte A reçoit la valeur 8K MSS envoyée par l'hôte A.
6. L'hôte A définit sa valeur MSS à 8K.

Afin d'éviter la fragmentation IPv4 aux points d'extrémité de la connexion TCP, la valeur MSS a été remplacée par la taille de tampon minimale et la MTU de l'interface de sortie (- 40).

Les numéros MSS sont inférieurs de 40 octets aux numéros MTU, car MSS (la taille des données TCP) n'inclut pas l'en-tête IPv4 de 20 octets et l'en-tête TCP de 20 octets.

MSS est basé sur les tailles d'en-tête par défaut ; la pile de l'expéditeur doit soustraire les valeurs appropriées pour l'en-tête IPv4 et l'en-tête TCP dépend des options TCP ou IPv4 utilisées.

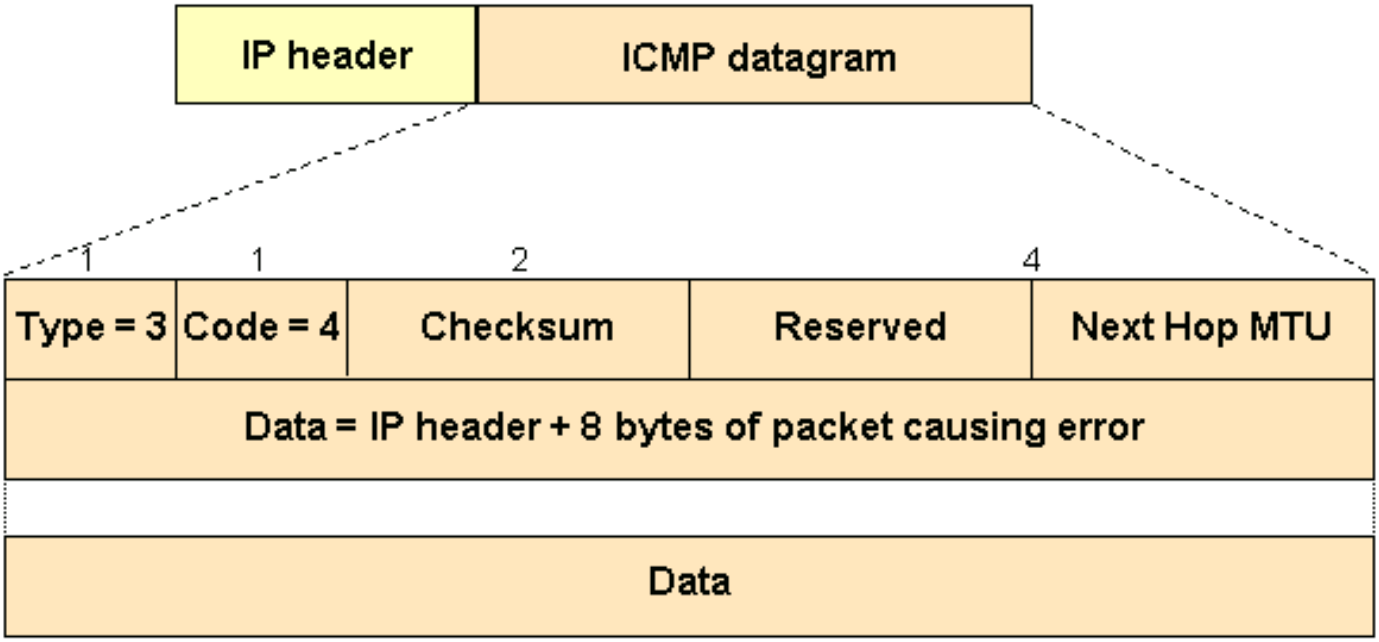
Actuellement, MSS fonctionne de telle sorte que chaque hôte compare d'abord son MTU d'interface sortante avec sa propre mémoire tampon et choisit la valeur la plus faible comme MSS à envoyer.

Les hôtes comparent ensuite la taille MSS reçue à leur propre MTU d'interface et choisissent à nouveau la valeur la plus faible des deux.

L'exemple 2 illustre cette étape supplémentaire effectuée par l'expéditeur afin d'éviter la fragmentation sur les fils locaux et distants.

Le MTU de l'interface sortante est pris en compte par chaque hôte avant que les hôtes ne s'envoient leurs valeurs MSS. Cela permet d'éviter la fragmentation.

Exemple 2



1. L'hôte A compare sa mémoire tampon MSS (16K) et son MTU (1500 - 40 = 1460) et utilise la valeur la plus basse comme MSS (1460) pour l'envoyer à l'hôte B.
2. L'hôte B reçoit la MSS d'envoi (1460) de l'hôte A et la compare à la valeur de son interface

sortante MTU - 40 (4422).
3. L’hôte B désigne une valeur inférieure de MSS (1 460) afin d’envoyer des datagrammes IPv4 à l’hôte A.
4. L'hôte B compare sa mémoire tampon MSS (8K) et son MTU (4462-40 = 4422), puis utilise 4422 comme MSS à envoyer à l'hôte A.
5. L'hôte A reçoit la MSS d'envoi (4422) de l'hôte B et la compare à la valeur de son interface sortante MTU -40 (1460).
6. L’hôte A désigne une valeur inférieure de MSS (1 460) afin d’envoyer des datagrammes IPv4 à l’hôte B.

1460 est la valeur choisie par les deux hôtes comme valeur MSS à s'envoyer réciproquement. Souvent, les valeurs MSS d'envoi sont les mêmes à chaque extrémité d'une connexion TCP.

Dans l'exemple 2, la fragmentation ne se produit pas aux points d'extrémité d'une connexion TCP, car les deux MTU d'interface sortantes sont prises en compte par les hôtes.

Les paquets deviennent toujours fragmentés dans le réseau entre les routeurs A et B s'ils rencontrent une liaison avec une MTU inférieure à celle de l'interface sortante de l'un des hôtes.

Qu’est-ce que la PMTUD

TCP MSS traite la fragmentation au niveau des deux points d'extrémité d'une connexion TCP, mais ne gère pas les cas où il y a une liaison MTU plus petite au milieu entre ces deux points d'extrémité.

La PMTUD a été mise au point pour éviter la fragmentation dans le chemin entre les deux points d’extrémité. Il est utilisé pour déterminer dynamiquement le MTU le plus bas le long du chemin entre une source de paquets et sa destination.

Remarque : La PMTUD est uniquement prise en charge par les protocoles TCP et UDP. Les autres protocoles ne la prennent pas en charge. Si la PMTUD est activée sur un hôte, tous les paquets TCP et UDP de l'hôte ont le bit DF défini.

Lorsqu’un hôte envoie un paquet de données MSS complet avec le bit DF, la PMTUD réduit la valeur MSS d’envoi pour la connexion si elle reçoit de l’information indiquant que le paquet doit être fragmenté.

Un hôte enregistre la valeur MTU pour une destination, car il crée une entrée d’hôte (/32) dans sa table de routage avec cette valeur MTU.

Si un routeur tente de transférer un datagramme IPv4 (avec le bit DF défini) sur une liaison dont la MTU est inférieure à la taille du paquet, il abandonne le paquet et renvoie un message ICMP (Internet Control Message Protocol) « Destination Unreachable » à la source du datagramme IPv4 avec le code indiquant « fragmentation requise et DF défini » (type 3, code 4).

Lorsque la station source reçoit le message ICMP, elle diminue la valeur MSS envoyée et lorsque TCP retransmet le segment, elle utilise la taille de segment la plus petite.

Voici un exemple de message ICMP « fragmentation requise et DF défini » affiché sur un routeur après l'activation de la **debug ip icmp** commande :

```
ICMP: dst (10.10.10.10) frag. needed and DF set
unreachable sent to 10.1.1.1
```

Ce diagramme montre le format de l'en-tête ICMP d'un message « fragmentation requise et bit DF » et « destination inaccessible ».

Plateau	MTU	Comments	Reference
	65535	Official maximum MTU	RFC 791
	65535	Hyperchannel	RFC 1044
65535			
32000		Just in case	
	17914	16Mb IBM Token Ring	ref. [6]
17914			
	8166	IEEE 802.4	RFC 1042
8166			
	4464	IEEE 802.5 (4Mb max)	RFC 1042
	4352	FDDI (Revised)	RFC 1188
4352 (1%)			
	2048	Wideband Network	RFC 907
	2002	IEEE 802.5 (4Mb recommended)	RFC 1042
2002 (2%)			
	1536	Exp. Ethernet Nets	RFC 895
	1500	Ethernet Networks	RFC 894
	1500	Point-to-Point (default)	RFC 1134
	1492	IEEE 802.3	RFC 1042
1492 (3%)			
	1006	SLIP	RFC 1055
	1006	ARPANET	BBN 1822
1006			
	576	X.25 Networks	RFC 877
	544	DEC IP Portal	ref. [10]
	512	NETBIOS	RFC 1088
	508	IEEE 802/Source-Rt Bridge	RFC 1042
	508	ARCNET	RFC 1051
508 (13%)			
	296	Point-to-Point (low delay)	RFC 1144
296			
68		Official minimum MTU	RFC 791

Comme indiqué dans le document RFC 1191, un routeur qui retourne un message ICMP indiquant qu'une fragmentation est nécessaire, mais que le fanion DF est activé doit inclure la MTU du nœud suivant dans les 16 bits inférieurs du champ d'en-tête supplémentaire du message ICMP étiqueté « unused » (non utilisé) dans le document de spécifications RFC 792.

Les premières mises en oeuvre de RFC 1191 n'ont pas fourni les informations MTU de saut suivant. Même lorsque ces informations ont été fournies, certains hôtes les ignorent.

Dans ce cas, RFC 1191 contient également une table qui répertorie les valeurs suggérées par lesquelles le MTU est abaissé pendant PMTUD.

Il est utilisé par les hôtes afin d'arriver plus rapidement à une valeur raisonnable pour le MSS d'envoi et comme indiqué dans cet exemple.

La PMTUD est exécutée en continu sur tous les paquets, car le chemin entre l'expéditeur et le destinataire peut changer dynamiquement.

Chaque fois qu'un expéditeur reçoit un message ICMP « Cannot Fragment », il met à jour les informations de routage (où il stocke la PMTUD).

Deux situations peuvent se produire pendant le PMTUD :

1. Le paquet peut atteindre le destinataire sans être fragmenté.



Remarque : Pour qu'un routeur puisse protéger le processeur contre les attaques DoS, il limite le nombre de messages ICMP qu'il envoie à deux par seconde. Par conséquent, dans ce contexte, si vous avez un scénario réseau dans lequel vous vous attendez à ce que le routeur doive répondre avec plus de deux messages ICMP (type = 3, code = 4) par seconde (il peut s'agir d'hôtes différents), désactivez la limitation des messages ICMP à l'aide de la **no ip icmp rate-limit unreachable [df] interface** commande.

---

2. L'expéditeur reçoit des messages ICMP « Cannot Fragment » à partir des sauts situés sur le chemin menant au destinataire.

PMTUD est exécuté indépendamment pour les deux directions d'un flux TCP.

Dans certains cas, PMTUD dans une direction d'un flux déclenche l'une des stations d'extrémité pour réduire le MSS d'envoi et l'autre station d'extrémité conserve le MSS d'envoi d'origine, car elle n'a jamais envoyé de datagramme IPv4 suffisamment grand pour déclencher PMTUD.

La connexion HTTP illustrée dans l'exemple 3 en est un exemple. Le client TCP envoie des petits paquets et le serveur des paquets volumineux.

Dans ce cas, seuls les gros paquets provenant du serveur (supérieurs à 576 octets) déclenchent la PMTUD.

Les paquets provenant du client sont petits (moins de 576 octets) et ne déclenchent pas PMTUD car ils ne nécessitent pas de fragmentation pour traverser la liaison MTU 576.

Exemple 3



L'exemple 4 présente un exemple de routage asymétrique où l'un des chemins a une MTU minimale inférieure à l'autre.

Le routage asymétrique se produit lorsque des chemins différents sont utilisés pour transmettre et recevoir des données entre deux points d'extrémité.

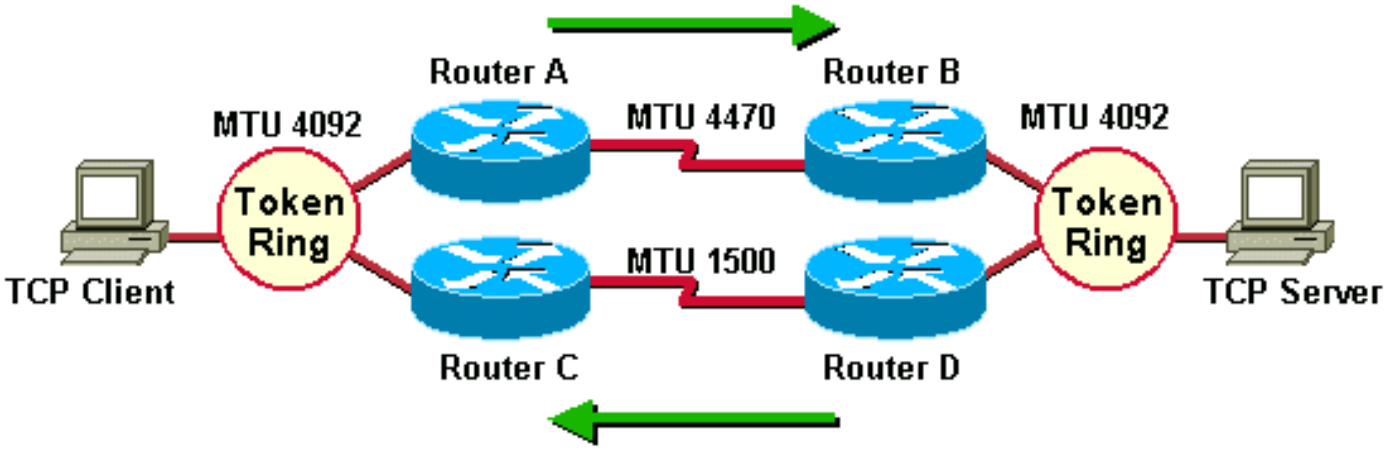
Dans cet exemple, PMTUD déclenche l'abaissement de la MSS d'envoi uniquement dans une direction d'un flux TCP.

Le trafic du client TCP vers le serveur passe par les routeurs A et B, tandis que le trafic de retour qui provient du serveur au client traverse les routeurs D et C.

Lorsque le serveur TCP envoie des paquets au client, PMTUD déclenche le serveur pour diminuer la valeur MSS d'envoi, car le routeur D doit fragmenter les paquets de 4 092 octets avant de pouvoir les envoyer au routeur C.

Inversement, le client ne reçoit jamais de message ICMP « Destination Unreachable » avec le code indiquant « fragmentation requise et DF défini », car le routeur A n'a pas à fragmenter les paquets lorsqu'il les envoie au serveur via le routeur B.

Exemple 4



Remarque : La commande ip tcp path-mtu-discovery sert à activer laTCP MTU path discovery pour les connexions TCP initiées par des routeurs (BGP et Telnet par exemple).

Problèmes survenant avec la découverte PMTUD

Ce sont des choses qui peuvent casser la PMTUD.

- Un routeur abandonne un paquet et n'envoie pas de message ICMP. (Rare)
- Un routeur génère et envoie un message ICMP, mais le message ICMP est bloqué par un routeur ou un pare-feu entre ce routeur et l'expéditeur. (Commun)
- Un routeur génère et envoie un message ICMP, mais l'expéditeur ignore le message. (Rare)

La première et la dernière des trois puces ici sont généralement le résultat d'une erreur, mais la puce du milieu décrit un problème courant.

Ceux qui implémentent des filtres de paquets ICMP ont tendance à bloquer tous les types de messages ICMP plutôt que seulement certains types de messages ICMP.

Il est possible pour le filtre de paquets de bloquer tous les types de messages ICMP à l'exception de ceux qui sont « inaccessibles » ou « dépassés dans le temps ».

La réussite ou l'échec de PMTUD dépend des messages ICMP inaccessibles qui parviennent à l'expéditeur d'un paquet TCP/IPv4.

Les messages ICMP « time-exceeded » (délai expiré) sont importants pour traiter d'autres problèmes IPv4.

Voici un exemple de filtre de paquets appliqué à un routeur.

```
access-list 101 permit icmp any any unreachable
access-list 101 permit icmp any any time-exceeded
access-list 101 deny icmp any any
access-list 101 permit ip any any
```

Il existe d'autres techniques qui peuvent être utilisées pour résoudre le problème d'un ICMP complètement bloqué.

- Effacez le bit DF sur le routeur et autorisez la fragmentation. (Ce n'est pas une bonne idée, cependant. Pour plus d'informations, voir Problèmes de fragmentation IP .
- Manipulez la valeur MSS de l'option TCP MSS à l'aide de la commande interface **ip tcp adjust-mss <500-1460>.**

Dans l'exemple suivant, les routeurs A et B se trouvent dans le même domaine administratif. Le routeur C est inaccessible et bloque le protocole ICMP, donc la PMTUD ne fonctionne pas.

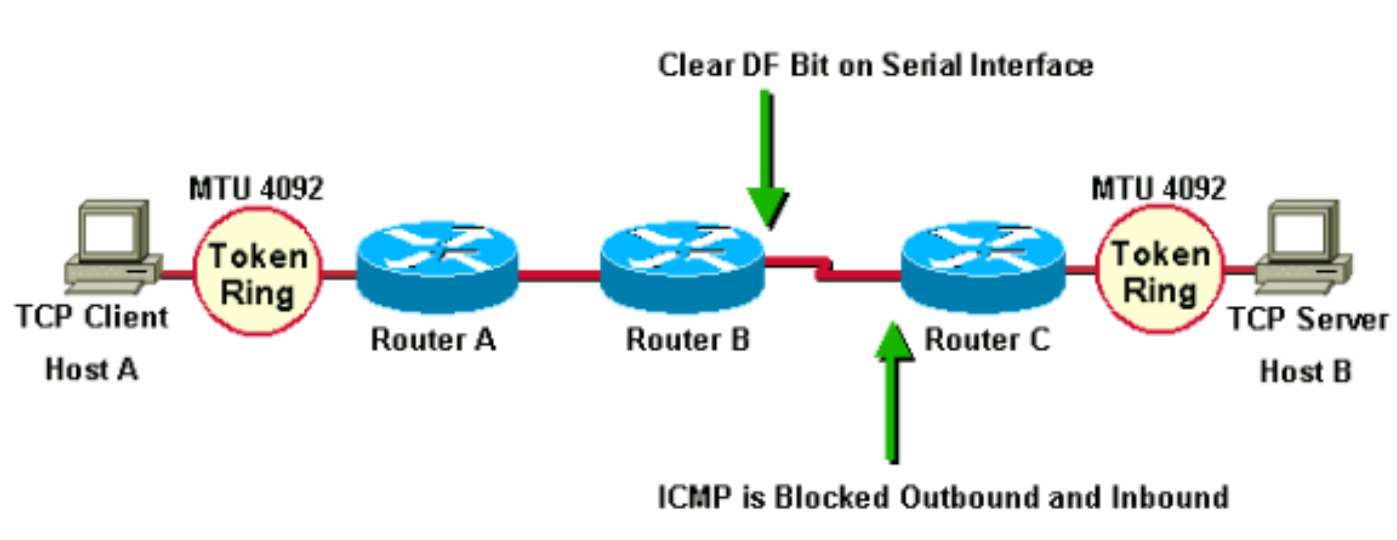
Une solution de contournement pour cette situation consiste à désactiver le bit DF dans les deux directions sur le routeur B afin de permettre la fragmentation. Cela peut être accompli avec la politique de routage.

La syntaxe à utiliser pour effacer le bit DF est disponible dans Cisco IOS® 12.1(6) et versions

ultérieures.

```
interface serial0
...
ip policy route-map clear-df-bit
route-map clear-df-bit permit 10
    match ip address 111
    set ip df 0

access-list 111 permit tcp any any
```



Une autre option consiste à changer la valeur de l'option TCP MSS des paquets SYN qui transitent par le routeur (disponible sur les appareils Cisco IOS® versions 12.2(4)T et plus récente).

Ceci réduit la valeur de l'option MSS dans le paquet TCP SYN de sorte qu'elle soit inférieure à la valeur (1460) dans la **ip tcp adjust-mss**commande.

Il en résulte que l'expéditeur TCP n'envoie pas de segments supérieurs à cette valeur.

La taille de paquet IPv4 est supérieure de 40 octets (1 500) à la valeur MSS (1 460 octets) afin de prendre en compte l'en-tête TCP (20 octets) et l'en-tête IPv4 (20 octets).

Vous pouvez ajuster la MSS des paquets TCP SYN à l'aide de la **ip tcp adjust-mss** commande. Cette syntaxe réduit la valeur MSS sur les segments TCP à 1460.

Cette commande permet de traiter le trafic entrant et sortant sur l'interface serial0.

```
int s0
ip tcp adjust-mss 1460
```

Les problèmes de fragmentation IPv4 sont plus répandus depuis que les tunnels IPv4 gagnent en popularité.

Les tunnels entraînent une fragmentation accrue, car l'encapsulation de tunnel ajoute une « surcharge » à la taille d'un paquet.

Par exemple, l'ajout de l'encapsulation de routeur générique (GRE) ajoute 24 octets à un paquet et, après cette augmentation, le paquet doit être fragmenté car il est plus grand que le MTU sortant.

Topologies réseau courantes nécessitant la découverte PMTUD

PMTUD est nécessaire dans les situations réseau dans lesquelles des liaisons intermédiaires possèdent des MTU plus faibles que le MTU des liaisons finales. Les raisons de ces valeurs plus faibles de MTU peuvent notamment être les suivantes :

- eToken Ring (ou FDDI) - hôtes d'extrémité connectés avec une connexion Ethernet entre elles. La MTU des anneaux à jeton (ou FDDI) aux extrémités est plus élevée que la MTU Ethernet au milieu.
- PPPoE (souvent utilisé avec ADSL) a besoin de 8 octets pour son en-tête. Ceci ramène le MTU efficace d'Ethernet à 1492 (1500 - 8).

Les protocoles de tunnel tels que GRE, IPv4sec et L2TP ont également besoin d'espace pour leurs en-têtes et en-queues respectifs. Cela réduit également le MTU effectif de l'interface sortante.

Tunnel

Un tunnel est une interface logique sur un routeur Cisco qui fournit une méthode d'encapsulation de paquets passagers au sein d'un protocole de transport.

Il s'agit d'une architecture conçue pour offrir des services permettant d'implémenter un schéma d'encapsulation point à point. Les interfaces de tunnel ont ces trois composants principaux :

- Protocole de passager (passenger) (AppleTalk, Banyan VINES, CLNS, DECnet, IPv4 ou IPX)
- Protocole de transport – l'un de ces protocoles d'encapsulation :
    - GRE : protocole de porteuse multiprotocole Cisco. Voir RFC 2784 et RFC 1701 pour plus d'informations.
    - IPv4 dans les tunnels IPv4 – Consultez le document RFC 2003 pour de plus amples renseignements.
- Protocole de transport – Le protocole utilisé pour transporter le protocole encapsulé.

Les paquets présentés dans cette section illustrent les concepts de tunnellisation IPv4 où GRE est

le protocole d’encapsulation et IPv4 est le protocole de transport.

Le protocole passager est également IPv4. Dans ce cas, IPv4 est à la fois le protocole de transport et le protocole passager.

Paquet normal

IPv4	TCP	Telnet

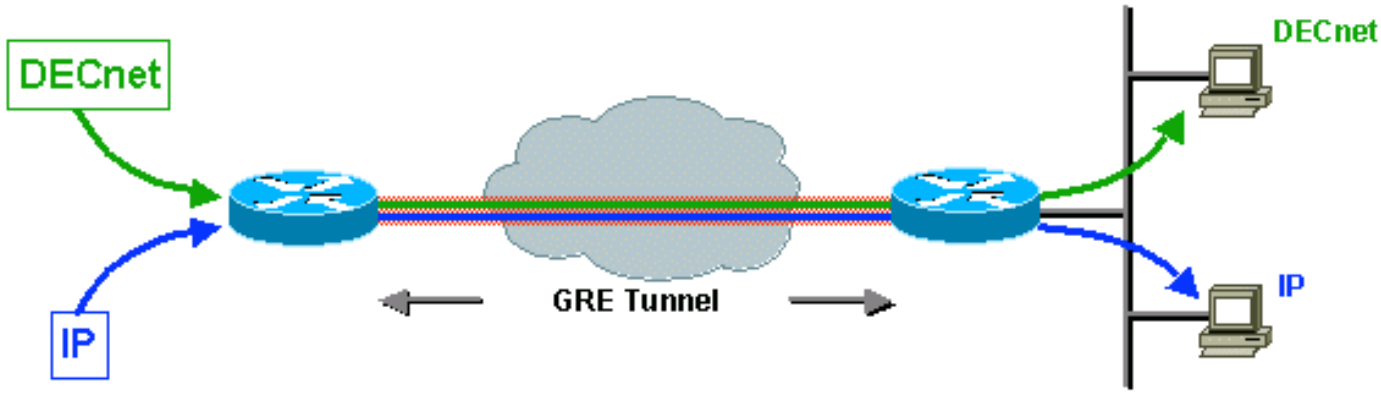
Paquet tunnel

IPv4	GRE	IPv4	TCP	Telnet

- IPv4 est le protocole de transport.
- GRE est le protocole d'encapsulation.
- IPv4 est le protocole de passager.

L’exemple suivant représente l’encapsulation de IPv4 et DECnet comme des protocoles de passagers avec GRE agissant comme protocole de transport.

Cela illustre la possibilité que les protocoles de l’opérateur encapsulent plusieurs protocoles passagers, comme illustré dans l’image.



Un administrateur réseau considère la transmission tunnel dans une situation où il existe deux réseaux non IPv4 discontinus séparés par un réseau fédérateur IPv4.

Si les réseaux discontinus exécutent DECnet, l'administrateur peut choisir de les connecter ensemble (ou non) en configurant DECnet dans le backbone.

L'administrateur ne souhaite pas autoriser le routage DECnet à consommer la bande passante du réseau fédérateur, car cela pourrait nuire aux performances du réseau IPv4.

Une alternative viable consiste à « tunneliser » DECnet dans le réseau dorsal IPv4. La solution de tunnel encapsule les paquets DECnet à l'intérieur d'IPv4, et les envoie à travers le backbone au point d'extrémité du tunnel où l'encapsulation est supprimée et les paquets DECnet sont routés vers leur destination via DECnet.

Il existe des avantages à encapsuler le trafic dans un autre protocole :

- Les terminaux utilisent des adresses privées (RFC 1918) et le réseau fédérateur ne prend pas en charge le routage de ces adresses.
- Autorisez les VPN (Virtual Private Network) sur les WAN ou sur Internet.
- Joignez les réseaux multiprotocole non contigus sur un circuit principal à protocole unique.
- Cryptez le trafic sur le circuit principal ou sur Internet.

Par la suite, IPv4 est utilisé comme protocole passager et IPv4 comme protocole de transport.

Considérations concernant les interfaces de tunnel

Ce sont des points à prendre en considération lors de la tunnellisation.

- La commutation rapide des tunnels GRE a été introduite dans la version 11.1 de Cisco IOS® et la commutation CEF a été introduite dans la version 12.0.
- La commutation CEF pour les tunnels GRE multipoints a été introduite dans la version 12.2(8)T.
- L’encapsulation et la décapsulation aux points de terminaison du tunnel étaient des opérations lentes dans les versions antérieures de Cisco iOS® , lorsque seule la commutation de processus était prise en charge.
- Lors de la transmission tunnel de paquets, des problèmes de sécurité et de topologie se posent. Les tunnels peuvent ignorer les listes de contrôle d'accès (ACL) et les pare-feu.
- Si vous effectuez un tunnel à travers un pare-feu, vous contournez le protocole passager en cours de tunnellisation. Par conséquent, il est recommandé d’inclure la fonctionnalité de pare-feu aux points de terminaison du tunnel afin d’appliquer n’importe quelle politique de traitement sur les protocoles passager.
- La transmission tunnel crée des problèmes avec les protocoles de transport qui ont des compteurs limités (par exemple, DECnet) en raison d'une latence accrue.
- La transmission tunnel entre des environnements avec des liaisons à vitesse différente, comme des anneaux FDDI rapides et des lignes téléphoniques lentes à 9 600 bits/s, pose des problèmes de réorganisation des paquets. Certains protocoles passagers fonctionnent mal dans des réseaux de supports mixtes.
- Les tunnels point à point consomment de la bande passante sur une liaison physique. Sur plusieurs tunnels point à point, chaque interface de tunnel a une bande passante et l'interface physique sur laquelle le tunnel s'exécute a une bande passante. Par exemple, définissez la bande passante du tunnel à 100 Ko si 100 tunnels sont exécutés sur une liaison de 10 Mo. La bande passante par défaut pour un tunnel est 9Kb.
- Les protocoles de routage préfèrent un tunnel à une liaison réelle, car le tunnel semble être

une liaison à un saut avec le chemin de moindre coût, bien qu'il implique plus de sauts et donc plus coûteux qu'un autre chemin. Ceci est atténué par une configuration appropriée du protocole de routage. Envisagez d'exécuter un protocole de routage différent sur l'interface de tunnel que le protocole de routage exécuté sur l'interface physique.

- Les problèmes de routage récursif sont évités en configurant des routes statiques appropriées vers la destination du tunnel. Une route est dite récursive lorsque le meilleur chemin d'accès à la destination du tunnel est par le tunnel lui-même. Cette situation pousse l'interface de tunnel à rebondir de haut en bas. Cette erreur se produit en cas de problème de routage récursif.

```
%TUN-RECURDOWN Interface Tunnel 0
temporarily disabled due to recursive routing
```

Routeur participant à la PMTUD au point d'extrémité d'un tunnel

Le routeur a deux rôles PMTUD différents à jouer lorsqu'il représente le point d'extrémité d'un tunnel.

- Le premier rôle du routeur est d'acheminer le paquet d'un hôte. Pour le traitement PMTUD, le routeur doit contrôler le bit DF et la taille de paquet du paquet de données d'origine, puis exécuter l'action appropriée, le cas échéant.
- Le second rôle est invoqué après que le routeur a encapsulé le paquet IPv4 original dans un paquet adapté à la tunnelisation. À cette étape, le routeur agit comme hôte en rapport avec la PMTUD et avec le paquet IPv4 adapté à la tunnelisation.

Lorsque le routeur joue le premier rôle (un routeur qui transfère les paquets IPv4 de l'hôte), ce rôle entre en jeu avant que le routeur n'encapsule le paquet IPv4 de l'hôte à l'intérieur du paquet du tunnel.

Si le routeur participe en tant que redirecteur d'un paquet hôte, il effectue les actions suivantes :

- Vérifier si le bit DF a été défini.
- Vérifier la taille de paquet utilisable par le tunnel.
- Fragmenter (si le paquet est trop volumineux et si le bit DF n'est pas défini), encapsuler et envoyer les fragments ; ou
- Supprimer le paquet (si le paquet est trop volumineux et si le bit DF est défini) et envoyer un message ICMP à l'expéditeur.
- Effectuer l'encapsulation (si le paquet n'est pas trop volumineux) et l'envoi.

De manière générique, on a le choix entre l'encapsulation suivie de la fragmentation (transmission

de deux fragments d’encapsulation), ou la fragmentation suivie de l’encapsulation (transmission de deux fragments encapsulés).

Deux exemples qui montrent l'interaction de PMTUD et des paquets qui traversent des exemples de réseaux sont détaillés dans cette section.

Le premier exemple montre ce qui arrive à un paquet lorsque le routeur (à la source du tunnel) agit comme routeur d’expédition.

Pour traiter la PMTUD, le routeur doit vérifier le bit DF et la taille de paquet du paquet de données d’origine et prendre les mesures appropriées.

Ces exemples utilisent l'encapsulation GRE pour le tunnel. GRE effectue la fragmentation avant l'encapsulation.

Les exemples suivants montrent les scénarios dans lesquels la fragmentation est faite après l'encapsulation.

Dans le premier exemple, le bit DF n’est pas activé (DF = 0) et la MTU du tunnel GRE IPv4 est de 1 476 (1 500 - 24).

Exemple 1

1. Le routeur de transfert (à la source du tunnel) reçoit un datagramme de 1 500 octets avec le bit DF effacé (DF = 0) de l’hôte émetteur.

Ce datagramme se compose d'un en-tête IP de 20 octets et d'une charge utile TCP de 1480 octets.

IPv4	TCP 1480 octets + données

2. Comme le paquet est trop volumineux pour la MTU IPv4 après l'ajout de la surcharge GRE (24 octets), le routeur de transfert divise le datagramme en deux fragments de 1 476 (en-tête IPv4 de 20 octets + charge utile IPv4 de 1 456 octets) et 44 octets (en-tête IPv4 de 20 octets + charge utile IPv4 de 24 octets)

Une fois l'encapsulation GRE ajoutée, le paquet n'est pas plus grand que le MTU de l'interface physique sortante.

IP_0	1 456 octets TCP + données
IP_1	Données 24 octets

3. Le routeur de transfert ajoute l’encapsulation GRE, qui inclut un en-tête GRE de 4 octets plus un en-tête IPv4 de 20 octets, à chaque fragment du datagramme IPv4 d’origine.

Ces deux datagrammes IPv4 ont maintenant une longueur de 1 500 et de 68 octets et sont considérés comme des datagrammes IPv4 distincts, et non pas comme des fragments.

IPv4	GRE	IP_0	1 456 octets TCP + données
IPv4	GRE	IP_1	Données 24 octets

4. Le routeur de destination du tunnel supprime l’encapsulation GRE de chaque fragment du datagramme d’origine, ce qui laisse deux fragments IPv4 de 1 476 et 24 octets.

Ces fragments de datagrammes IPv4 sont acheminés séparément par ce routeur vers l’hôte destinataire.

IP_0	1 456 octets TCP + données
IP_1	Données 24 octets

5. L’hôte récepteur réassemble ces deux fragments dans le datagramme d’origine.

IPv4	TCP 1480 octets + données

L'exemple 2 décrit le rôle du routeur de transfert dans le contexte d'une topologie de réseau.

Le routeur joue le même rôle que le routeur de transfert, mais cette fois, le bit DF est défini (DF = 1).

Exemple 2

1. Le routeur de transfert à la source du tunnel reçoit un datagramme de 1 500 octets avec DF = 1 de l’hôte émetteur.

IPv4	TCP 1480 octets + données

2. Comme le bit DF est défini et que la taille du datagramme (1 500 octets) est supérieure à la MTU IPv4 du tunnel GRE (1 476), le routeur abandonne le datagramme et envoie un message « ICMP fragmentation required but DF bit set » à la source du datagramme.

Le message ICMP avertit l'expéditeur que le MTU est 1476.

IPv4	ICMP MTU 1476

3. L’hôte émetteur reçoit le message ICMP et, lorsqu’il renvoie les données d’origine, il utilise un datagramme IPv4 de 1 476 octets.

IPv4	1 456 octets TCP + données

4. Cette longueur de datagramme IPv4 (1 476 octets) est désormais égale en valeur à la MTU IPv4 du tunnel GRE, de sorte que le routeur ajoute l'encapsulation GRE au datagramme IPv4.

IPv4	GRE	IPv4	1 456 octets TCP + données

5. Le routeur récepteur (à la destination du tunnel) supprime l’encapsulation GRE du datagramme IPv4 et l’envoie à l’hôte récepteur.

IPv4	1 456 octets TCP + données

C'est ce qui se produit lorsque le routeur agit dans le deuxième rôle en tant qu'hôte émetteur par

rapport à PMTUD et par rapport au paquet IPv4 du tunnel.

Ce rôle entre en jeu une fois que le routeur a encapsulé le paquet IPv4 d'origine à l'intérieur du paquet de tunnel.

Remarque : Par défaut, un routeur n'exécute pas PMTUD sur les paquets de tunnel GRE qu'il génère. La commande **tunnel path-mtu-discovery** peut être utilisée pour activer PMTUD pour les paquets de tunnel GRE-IPv4.

L’exemple 3 montre ce qui se produit lorsque l’hôte envoie des datagrammes IPv4 assez petits pour respecter la MTU IPv4 de l’interface de tunnel GRE.

Dans ce cas, le bit DF peut être soit défini, soit effacé (1 ou 0).

La commande n'est pas configurée sur l'interface de tunnel GRE, **tunnel path-mtu-discovery** de sorte que le routeur ne meurt pas en PMTUD sur le paquet GRE-IPv4.

Exemple 3

1. Le routeur de transfert à la source du tunnel reçoit un datagramme de 1 476 octets de l’hôte émetteur.

IPv4	1 456 octets TCP + données

2. Ce routeur encapsule le datagramme IPv4 de 1 476 octets dans GRE pour obtenir un datagramme IPv4 GRE de 1 500 octets.

Le bit DF de l'en-tête GRE IPv4 est effacé (DF = 0). Ce routeur achemine ensuite ce paquet vers la destination du tunnel.

IPv4	GRE	IPv4	1 456 octets TCP + données

3. Supposons qu’il existe un routeur entre la source et la destination du tunnel avec un MTU de liaison de 1400.

Ce routeur fragmente le paquet du tunnel puisque le bit DF est libre (DF = 0).

N'oubliez pas que cet exemple fragmente l'IPv4 le plus à l'extérieur, de sorte que les en-têtes GRE, IPv4 interne et TCP n'apparaissent que dans le premier fragment.

IP_0	GRE	IP	TCP 1352 octets + données
IP_1	Données 104 octets		

4. Le routeur de destination du tunnel doit réassembler le paquet de tunnel GRE.

IP	GRE	IP	1 456 octets TCP + données

5. Une fois le paquet de tunnel GRE réassemblé, le routeur supprime l'en-tête IPv4 GRE et envoie le datagramme IPv4 d'origine sur son chemin.

IPv4	TCP 1456 octets + données

L'exemple 4 montre ce qui se passe lorsque le routeur joue le rôle d'hôte émetteur par rapport à la PMTUD et au paquet IPv4 du tunnel.

Cette fois, le bit DF est défini (DF = 1) dans l'en-tête IPv4 d'origine et la **tunnel path-mtu-discovery** commande a été configurée pour que le bit DF soit copié de l'en-tête IPv4 interne vers l'en-tête externe (GRE + IPv4).

Exemple 4

1. Le routeur de transfert à la source du tunnel reçoit un datagramme de 1 476 octets avec DF = 1 de l'hôte émetteur.

IPv4	1 456 octets TCP + données

2. Ce routeur encapsule le datagramme IPv4 de 1 476 octets dans GRE pour obtenir un datagramme IPv4 GRE de 1 500 octets.

Cet en-tête IPv4 GRE a le bit DF défini (DF = 1) puisque le datagramme IPv4 d'origine avait le bit DF défini.

Ce routeur achemine ensuite ce paquet vers la destination du tunnel.

IPv4	GRE	IPv4	TCP 1456 octets

3. À nouveau, supposons qu'il existe un routeur entre la source et la destination du tunnel avec une MTU de liaison de 1400.

Ce routeur ne fragmente pas le paquet de tunnel car le bit DF est défini (DF=1).

Ce routeur doit abandonner le paquet et envoyer un message d'erreur ICMP au routeur source du tunnel, car il s'agit de l'adresse IPv4 source du paquet.

IPv4	ICMP MTU 1400

4. Le routeur de transfert à la source du tunnel reçoit ce message d'erreur « ICMP » et abaisse la MTU IPv4 du tunnel GRE à 1376 (1400 - 24).

La prochaine fois que l'hôte émetteur retransmet les données dans un paquet IPv4 de 1 476 octets, ce paquet peut être trop volumineux et ce routeur envoie alors un message d'erreur « ICMP » à l'expéditeur avec une valeur MTU de 1 376.

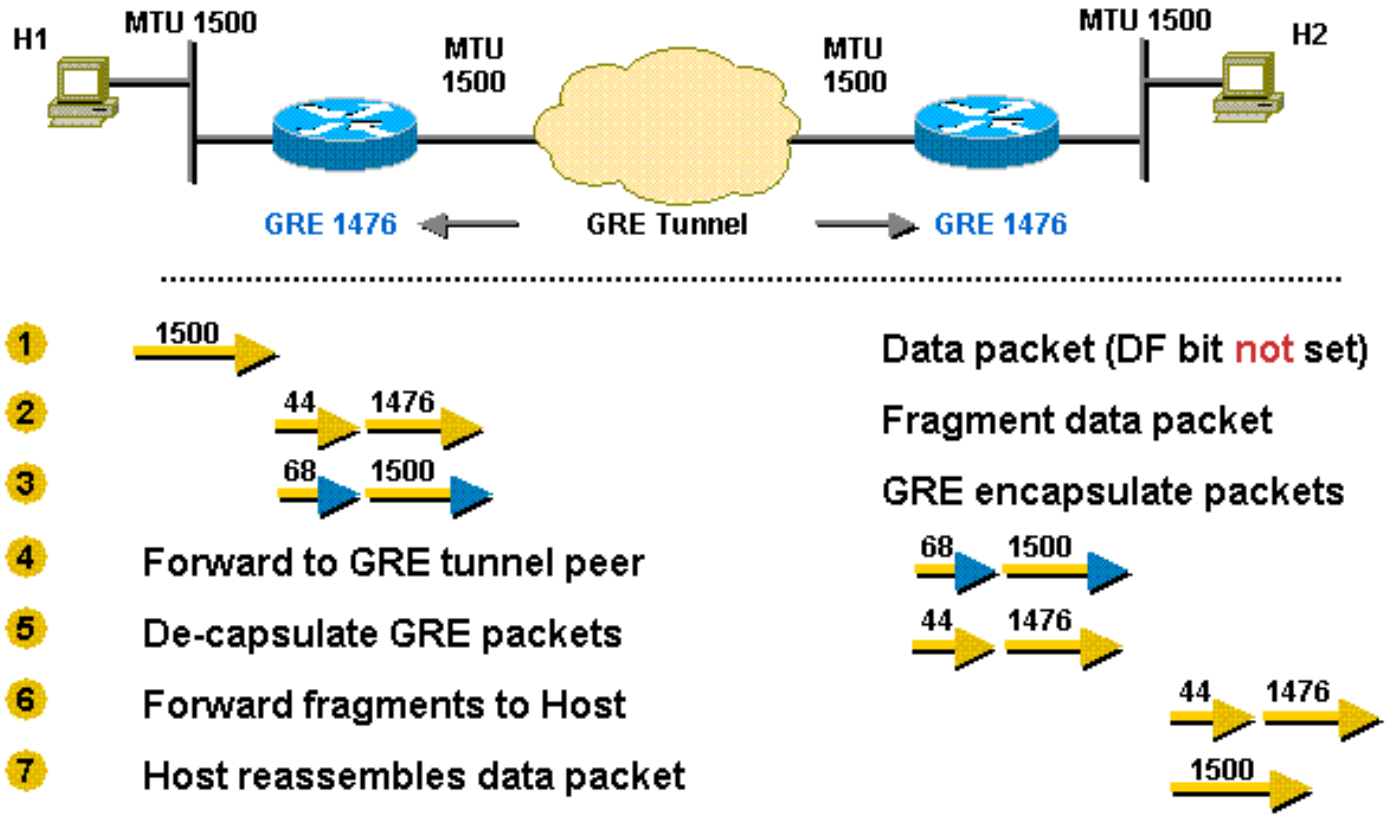
Lorsque l'hôte émetteur retransmet les données, il les envoie dans un paquet IPv4 de 1 376 octets et ce paquet passe par le tunnel GRE jusqu'à l'hôte récepteur.

Exemple 5

Cet exemple illustre la fragmentation GRE. Fragmentez avant l'encapsulation pour GRE, puis effectuez PMTUD pour le paquet de données et le bit DF n'est pas copié lorsque le paquet IPv4

est encapsulé par GRE.

Le bit DF n'est pas défini. La MTU IPv4 de l'interface de tunnel GRE est, par défaut, inférieure de 24 à la MTU IPv4; donc la MTU IPv4 de l'interface GRE est de 1 476, comme illustré.



1. L'expéditeur envoie un paquet de 1 500 octets (en-tête IP de 20 octets + 1 480 octets de charge utile TCP).
2. Comme le MTU du tunnel GRE est de 1476, le paquet de 1500 octets est divisé en deux fragments IPv4 de 1476 et 44 octets, chacun en prévision des 24 octets supplémentaires de l'en-tête GRE.
3. Les 24 octets de l'en-tête GRE sont ajoutés à chaque fragment IPv4. Maintenant les fragments sont de 1500 (1476 + 24) et 68 (44 + 24) octets chacun.
4. Les paquets GRE + IPv4 qui contiennent les deux fragments IPv4 sont transmis au routeur homologue du tunnel GRE.
5. Le routeur partenaire de tunnel GRE retire les en-têtes GRE des deux packets.
6. Ce routeur achemine les deux paquets vers l'hôte de destination.
7. L'hôte destinataire réassemble les fragments IPv4 afin de reconstruire le datagramme IPv4 original.

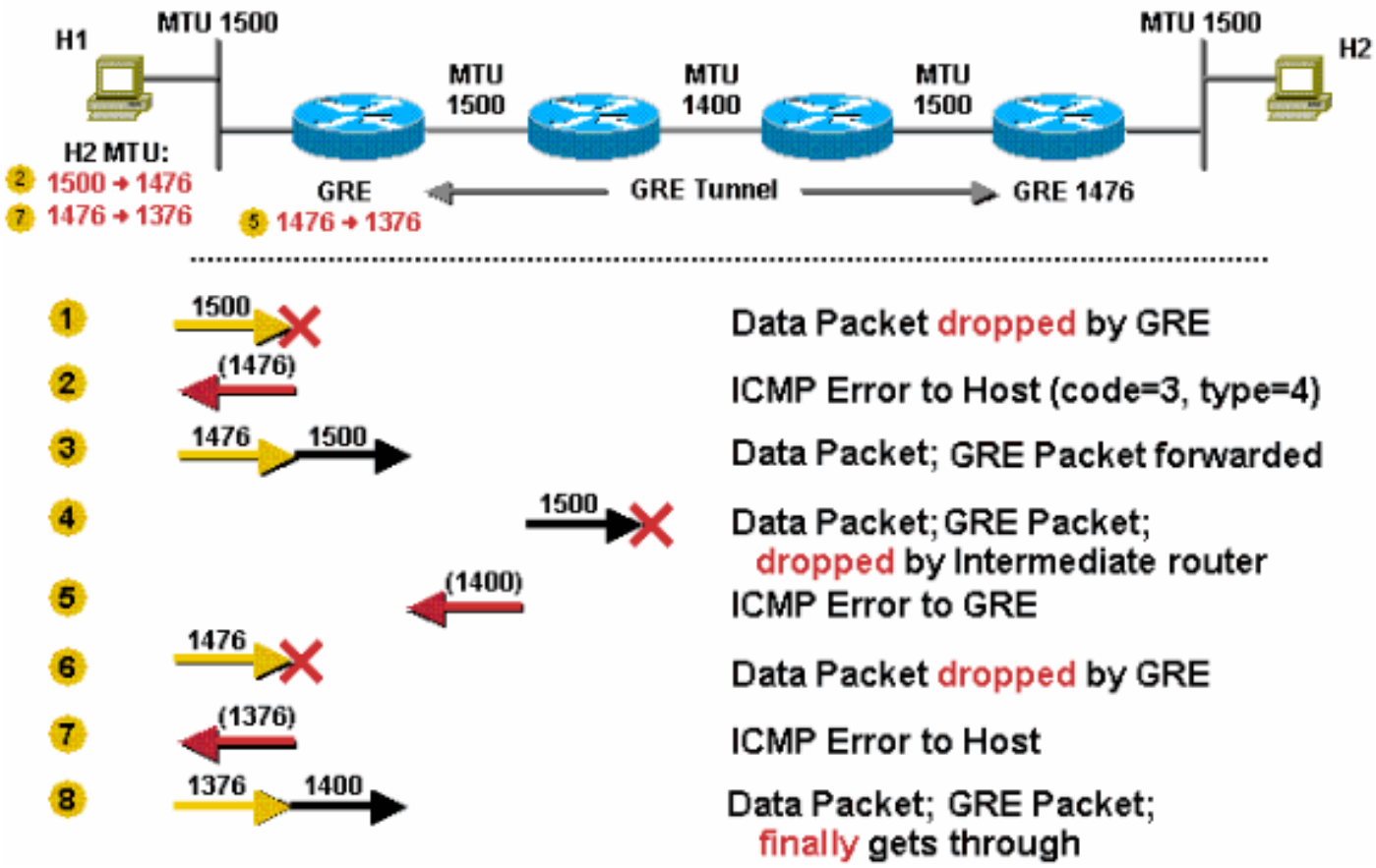
Exemple 6

Cet exemple est similaire à l'exemple 5, mais cette fois, le bit DF est défini. Le routeur est configuré pour effectuer la PMTUD sur les paquets de tunnel GRE + IPv4 avec la **tunnel path-mtu-discovery**commande, et le bit DF est copié de l'en-tête IPv4 d'origine vers l'en-tête IPv4 GRE.

Si le routeur reçoit une erreur ICMP pour le paquet GRE + IPv4, il réduit la MTU IPv4 de l'interface

de tunnel GRE.

Le MTU IPv4 du tunnel GRE est défini sur 24 octets de moins que le MTU de l'interface physique par défaut, de sorte que le MTU IPv4 GRE est ici de 1476. Il y a une liaison MTU 1400 dans le chemin du tunnel GRE comme illustré dans l'image.



1. Le routeur reçoit un paquet de 1 500 octets (20 octets pour l'en-tête IPv4 + 1400 octets de charge utile (payload) TCP), il doit conséquemment abandonner le paquet. Le routeur abandonne le paquet parce qu'il est plus grand que la MTU IPv4 (1 476) de l'interface du tunnel GRE.
2. Le routeur envoie une erreur ICMP à l'expéditeur lui indiquant que le MTU du saut suivant est de 1476. L'hôte enregistre ces informations, généralement en tant que route hôte pour la destination dans sa table de routage.
3. L'hôte expéditeur utilise une taille de paquet de 1476 octets lorsqu'il envoie de nouveau les données. Le routeur GRE ajoute 24 octets d'encapsulation GRE et expédie un paquet de 1500 octets.
4. Le paquet de 1500 octets ne peut pas traverser la liaison de 1400 octets, il est donc supprimé par le routeur intermédiaire.
5. Le routeur intermédiaire envoie un message ICMP (type = 3, code = 4) au routeur GRE dans lequel la valeur MTU du prochain nœud est de 1 400. Le routeur GRE réduit la taille à 1 376 (1400 - 24) et configure une valeur de MTU IPv4 interne de l'interface GRE. Ce changement ne peut être vu que lorsque vous utilisez la **debug tunnel** commande ; il ne peut pas être vu dans le résultat de la **show ip interface tunnel**<#> commande.
6. La prochaine fois que l'hôte renverra le paquet de 1 476 octets, le routeur GRE abandonnera le paquet, car il est plus grand que la MTU IPv4 actuelle (1 376) sur l'interface de tunnel

GRE.
7. Le routeur GRE envoie un autre ICMP (type = 3, code = 4) à l'expéditeur avec un MTU de tronçon suivant de 1376 et l'hôte met à jour ses informations actuelles avec une nouvelle valeur.
8. L'hôte renvoie à nouveau les données, mais dans un paquet plus petit de 1 376 octets, GRE ajoute 24 octets d'encapsulation et les transmet. Cette fois, le paquet est envoyé à l'homologue de tunnel GRE, où le paquet est décapsulé et envoyé à l'hôte de destination.

Remarque : Si la **tunnel path-mtu-discovery** commande n'a pas été configurée sur le routeur de transfert dans ce scénario et que le bit DF a été défini dans les paquets transférés via le tunnel GRE, l'hôte 1 réussit toujours à envoyer des paquets TCP/IPv4 à l'hôte 2, mais ils sont fragmentés au milieu sur la liaison MTU 1400. L'homologue de tunnel GRE doit également les réassembler avant de pouvoir les décapsuler et les transférer.

Mode tunnel IPsec pur

Le protocole de sécurité IPv4 (IPv4sec) est une méthode basée sur des normes qui assure la confidentialité, l'intégrité et l'authenticité de l'information transférée sur des réseaux IPv4.

IPv4sec assure le chiffrement IPv4 au niveau de la couche réseau. IPv4sec augmente la longueur des paquets IPv4 en ajoutant au moins un en-tête IPv4 (mode de tunnelisation).

La longueur des en-têtes ajoutés varie en fonction du mode de configuration IPv4sec, mais ils ne dépassent pas ~58 octets (ESP (Encapsulating Security Payload) et ESP authentication (ESPauth)) par paquet.

IPv4sec comporte deux modes : le mode de tunnellisation et le mode de transport.

1. Le mode tunnel est le mode par défaut. Avec le mode de tunnellisation, le paquet IPv4 original est protégé en entier (chiffré, authentifié, ou les deux) et encapsulé par les champs d'en-tête et de terminaison IPv4sec. Puis, un nouvel en-tête IPv4 est ajouté au paquet pour indiquer les points d'extrémité (homologues) que sont la source et la destination. Le mode de tunnellisation peut être utilisé avec n'importe quel trafic IPv4 unicast et doit être utilisé si IPv4sec protège le trafic des hôtes au-delà des homologues IPv4sec. Par exemple, le mode de tunnellisation est utilisé sur des réseaux virtuels privés (VPN, Virtual Private Network) où les hôtes sur un réseau protégé envoient des paquets aux hôtes sur un réseau protégé différents grâce à des paires homologues IPv4sec. Avec les VPN, le « tunnel » IPv4sec protège le trafic IPv4 entre les hôtes en chiffrant ce trafic entre les routeurs homologues IPv4sec.
2. Avec le mode transport (configuré avec la sous-commande, **mode transport**, sur la définition de transformation), seule la charge utile du paquet IPv4 d'origine est protégée (chiffrée, authentifiée ou les deux). La charge utile est encapsulée par les champs d'en-tête et de terminaison IPv4sec. Les en-têtes IPv4 originaux demeurent intacts, sauf que le champ du protocole IPv4 est changé pour ESP (50) et la valeur initiale du protocole est enregistrée dans les champs de terminaison IPv4sec pour être restaurée lorsque le paquet est déchiffré. Le mode de transport est utilisé uniquement lorsque le trafic IPv4 à être protégé se situe

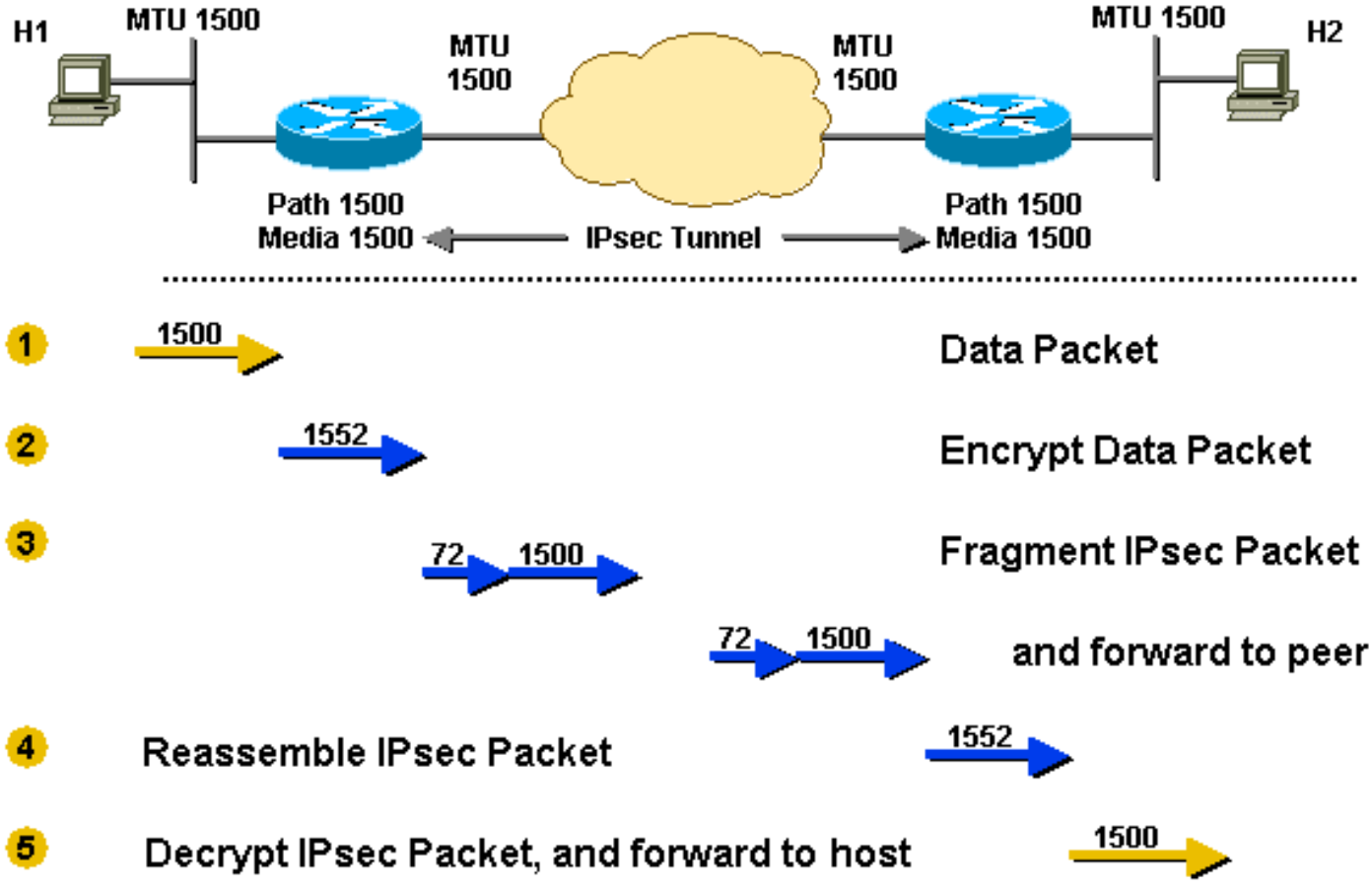
entre les homologues IPv4sec eux-mêmes, les adresses de source et de destination IPv4 sur le paquet sont identiques à celles des homologues IPv4sec. Normalement, le mode de transport IPv4sec est utilisé uniquement lorsqu'un autre protocole de tunnellisation (comme GRE) est utilisé au préalable pour encapsuler le paquet de données IPv4, puis IPv4sec est utilisé pour protéger les paquets de tunnellisation GRE.

IPv4sec effectue toujours une PMTUD pour les paquets de données et pour ses propres paquets. Des commandes de configuration IPv4sec permettent de modifier le traitement PMTUD pour le paquet IPv4 de IPv4sec IPv4; IPv4sec peut désactiver, activer ou copier le bit DF de l'en-tête du paquet de données IPv4 vers l'en-tête IPv4 de IPv4sec. Ceci s'appelle « la fonction de remplacement de bit DF ».

Remarque : Évitez la fragmentation après l'encapsulation lorsque le chiffrement matériel avec IPv4sec est effectué. Le cryptage matériel vous offre un débit d'environ 50 Mbits/s, selon le matériel, mais si le paquet IPv4sec est fragmenté, vous perdez 50 à 90 % du débit. Cette perte de performance est due au fait que les paquets IPv4sec fragmentés sont traités pour être réassemblés, puis passés au moteur de chiffrement matériel pour y être déchiffrés. Cette perte de débit peut mener le débit de cryptage matériel au niveau des performances de cryptage logiciel (2-10 Mbs).

Exemple 7

Ce scénario décrit la fragmentation IPv4sec fragmentation en action. Dans ce scénario, le MTU du chemin entier est 1500. Dans ce scénario, le bit DF n'est pas défini.



1. Le routeur reçoit un paquet de 1 500 octets (en-tête IPv4 de 20 octets + 1 480 octets de charge utile destiné à l'hôte 2.
2. Le paquet de 1 500 octets est chiffré par IPv4sec et 54 octets sont ajoutés (en-tête IPv4, terminaison, et en-tête IPv4 supplémentaire). IPv4sec doit maintenant acheminer un paquet de 1 552 octets. Puisque le MTU sortant est de 1500, ce paquet doit être fragmenté.
3. Le paquet IPv4 a donné lieu à la création de deux fragments. Lors de la fragmentation, un en-tête IPv4 de 20 octets supplémentaire est ajouté pour le deuxième fragment, ce qui donne un fragment de 1 500 octets et un fragment IPv4 de 72 octets.
4. Le routeur homologue du tunnel IPv4 reçoit les fragments, élimine l'en-tête IPv4 supplémentaire et concatène les fragments IPv4 pour reformer le paquet IPv4sec original. Finalement, IPv4sec déchiffre ce paquet.
5. Le routeur achemine ensuite le paquet de données de 1500 octets vers l'hôte 2.

Exemple 8

Cet exemple est similaire à l'exemple 6, à ceci près que dans ce cas, le bit DF est défini dans le paquet de données d'origine et qu'il y a une liaison dans le chemin entre les homologues de tunnel IPv4sec qui a une MTU inférieure à celle des autres liaisons.

Cet exemple montre comment le routeur homologue IPv4sec joue les deux rôles PMTUD, comme décrit dans la section Le routeur en tant que participant PMTUD au point d'extrémité d'un tunnel.

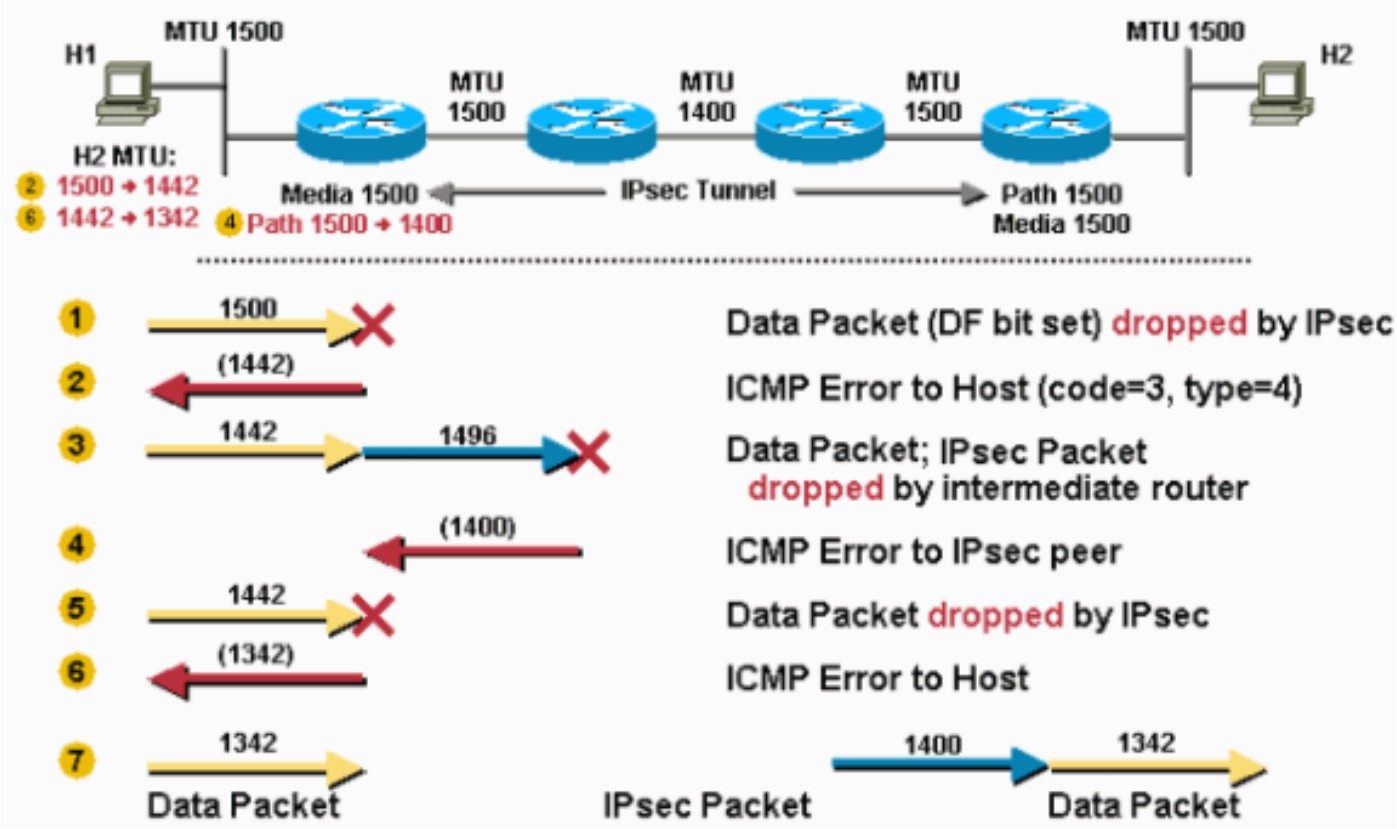
La PMTU IPv4sec passe à une valeur inférieure en raison du besoin de fragmentation.

Le bit DF est copié de l'en-tête IPv4 interne vers l'en-tête IPv4 externe lorsque IPv4sec chiffre un paquet.

Les valeurs MTU et PMTU du média sont enregistrées dans le champ Sécurité Association (SA) de IPv4sec.

La MTU du média est déterminée par la MTU de l'interface du routeur émetteur et la PMTU est déterminée par la MTU minimale observée sur le chemin entre les homologues IPv4.

IPv4sec encapsule/chiffre le paquet avant qu'il ne tente de le fragmenter, comme indiqué dans l'image.



1. Le routeur reçoit un paquet de 1 500 octets et l'abandonne, car la surcharge IPv4sec, lorsqu'elle est ajoutée, rend le paquet plus volumineux que la PMTU (1 500).
2. Le routeur envoie un message ICMP à l'hôte 1, en indiquant que le MTU du prochain saut est 1442 (1500 - 58 = 1442). Ces 58 octets représentent la surcharge IPv4sec maximale lorsque vous utilisez IPv4sec ESP et ESPauth. La surcharge IPv4sec réelle est probablement inférieure de 7 octets à cette valeur. L'hôte 1 enregistre cette information, habituellement sous forme de route hôte pour la destination (hôte 2), dans sa table de routage.
3. L'hôte 1 abaisse sa PMTU pour l'hôte 2 à 1 442, de sorte que l'hôte 1 envoie des paquets plus petits (1 442 octets) lorsqu'il retransmet les données à l'hôte 2. Le routeur reçoit le paquet de 1 442 octets et IPv4sec ajoute 52 octets de surcharge de chiffrement, de sorte que le paquet IPv4sec résultant est de 1 496 octets. Puisque le bit DF de ce paquet est défini dans son en-tête, il est supprimé par le routeur du milieu, avec la liaison MTU 1400 octets.
4. Le routeur au centre qui a abandonné le paquet envoie un message ICMP à l'expéditeur du paquet IPv4sec (le premier routeur), lui indiquant que la MTU du prochain noeud est de 1 400 octets. Cette valeur est enregistrée dans la PMTU SA de IPv4sec.
5. La prochaine fois que l'hôte 1 retransmettra le paquet de 1 442 octets (il n'a pas reçu d'accusé de réception), IPv4sec abandonnera le paquet. Le routeur abandonne le paquet parce que la surcharge IPv4sec, lorsqu'elle est ajoutée au paquet, le rend plus grand que la PMTU (1400).
6. Le routeur envoie un message ICMP à l'hôte 1, en indiquant que le MTU du prochain saut est désormais 1342. (1400 - 58 = 1342). L'hôte 1 enregistre à nouveau ces informations.
7. Lorsque l'hôte 1 retransmet les données, il utilise le paquet de plus petite taille (1342). Ce paquet ne nécessite pas de fragmentation et le transite par le tunnel IPv4sec vers l'hôte 2.

Utilisation combinée de IPv4sec et de GRE

Des interactions plus complexes impliquant la fragmentation et la PMTUD se produisent lorsque IPv4sec est utilisé pour chiffrer les tunnels GRE.

IPv4sec et GRE sont combinés de cette façon parce que IPv4sec ne prend pas en charge les paquets IPv4 multidestinataire (multicast), ce qui signifie que vous ne pouvez pas exécuter un protocole de routage dynamique sur le réseau VPN IPv4sec.

Les tunnels GRE prennent en charge les paquets multidestinataire, donc un tunnel GRE peut être utilisé pour d'abord encapsuler le paquet à protocole de routage dynamique multidestinataire (dynamic routing protocol multicast packet) dans un paquet iPv4 GRE monodestinataire (unicast) qui peut alors être chiffré par IPv4sec.

Dans ce cas, IPv4sec est souvent déployé en mode transport sur GRE, car les homologues IPv4sec et les points d'extrémité du tunnel GRE (les routeurs) sont identiques, et le mode transport économise 20 octets de surcharge IPv4sec.

Un cas intéressant survient lorsqu'un paquet IPv4 a été divisé en deux fragments puis encapsulé par GRE.

Dans ce cas, IPv4sec voit deux paquets GRE + IPv4 indépendants. Souvent, dans une configuration par défaut, l'un de ces paquets est suffisamment grand pour qu'il doive être fragmenté après avoir été chiffré.

L'homologue IPv4sec doit réassembler ce paquet avant le déchiffrement. Cette « double fragmentation » (une fois avant GRE et de nouveau après IPv4sec) sur le routeur émetteur augmente la latence et diminue le débit.

Le réassemblage est commuté par processus, de sorte qu'il y a une réponse du processeur sur le routeur récepteur chaque fois que cela se produit.

Cette situation peut être évitée en désignant une valeur « ip mtu » sur l'interface de tunnel GRE assez faible pour prendre en compte les ajouts de GRE et de IPv4sec (par défaut, la valeur du « ip mtu » de l'interface de tunnel GRE est réglée sur la taille de l'ajout en octets de l'interface MTU - GRE sortante réelle).

Ce tableau répertorie les valeurs de MTU suggérées pour chaque combinaison tunnel/mode qui suppose que l'interface physique sortante a une MTU de 1500.

Combinaison de tunnel	MTU spécifique requis	MTU recommandé
GRE + IPv4sec (mode de transport)	1440 bytes	1400 bytes
GRE + IPv4sec (mode de tunnelisation)	1420 bytes	1400 bytes



Remarque : Une valeur MTU de 1 400 est recommandée, car elle convient aux situations les plus usuelles de combinaisons de GRE + IPv4sec. En outre, il n'y a aucun inconvénient visible à permettre une surcharge supplémentaire de 20 ou 40 octets. Il est plus facile de se

rappeler et de définir une valeur et cette valeur couvre presque tous les scénarios.

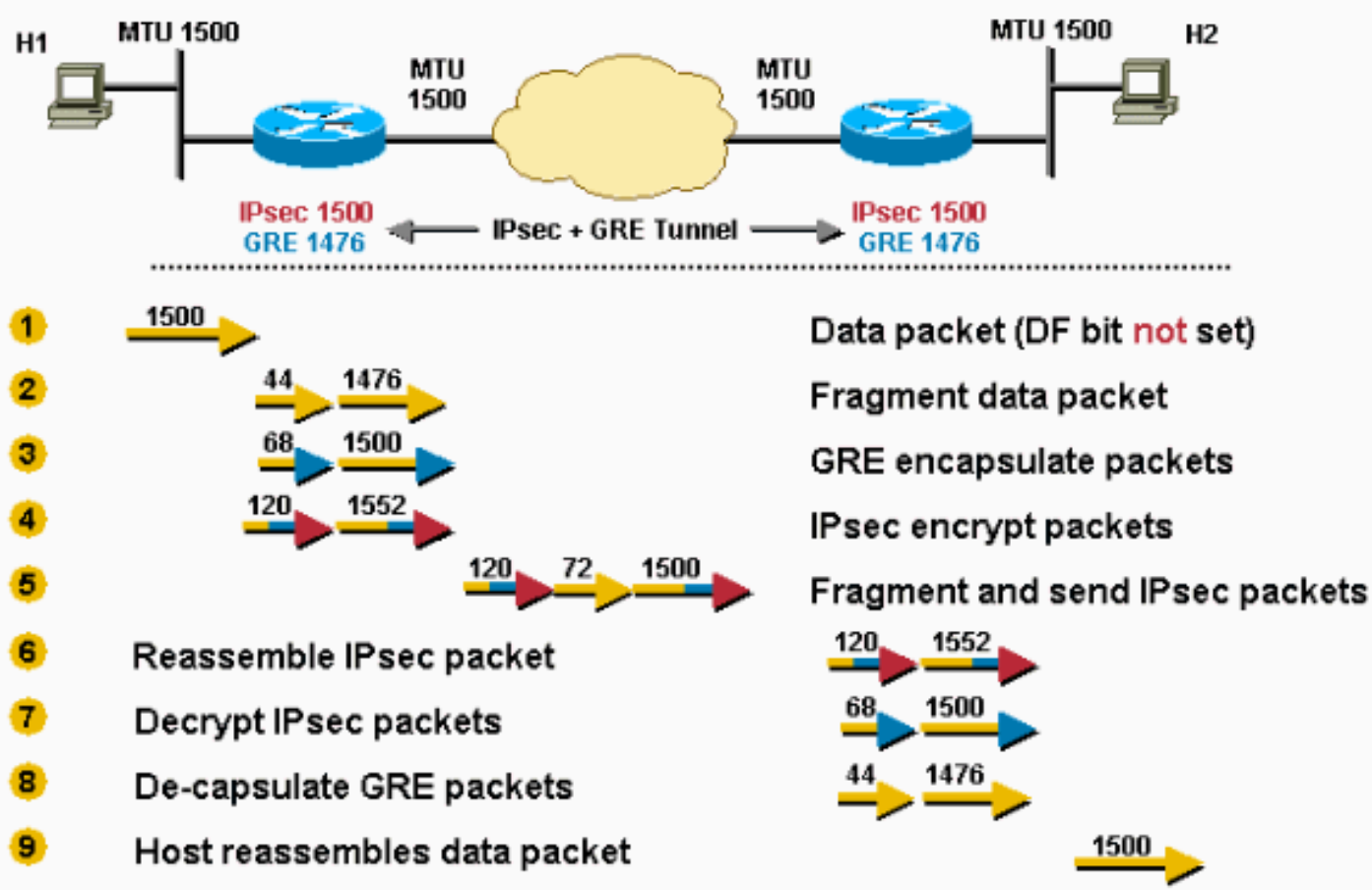
Exemple 9

IPv4sec est déployé en surcouche sur GRE. La MTU physique sortante est de 1 500 , la PMTU IPv4sec est de 1 500, et la MTU IPv4 GRE est de 1 476 (1 500 - 24 = 1 476).

Les paquets TCP/IPv4 sont donc fragmentés deux fois, une fois avant GRE et une fois après IPv4sec.

Le paquet est fragmenté avant l'encapsulation GRE et l'un de ces paquets GRE est à nouveau fragmenté après le chiffrement IPv4sec.

La configuration de « ip 1 440» (mode transport de IPv4sec) ou « ip mtu 1 420 »(mode de tunnelisation IPv4sec) sur le tunnel GRE fera en sorte d'éviter la double fragmentation dans ce scénario.



1. Le routeur reçoit un datagramme de 1500 octets.
2. Avant l'encapsulation, GRE ceint le paquet de 1 500 octets en deux fragments : 1 476 (1 500 - 24 = 1476) et 44 (24 octets de données + 20 octets pour l'en-tête IPv4).
3. GRE encapsule les fragments IPv4, ce qui ajoute 24 octets à chaque paquet. Ceci produit finalement deux paquets GRE + IPv4sec : un paquet de 1 500 octets (1 476 + 24 = 1 500) et un paquet de 68 octets (44 + 24) octets.
4. IPv4sec chiffre les deux paquets, qui ajoutent 52 octets (mode tunnel IPv4sec) de surcharge

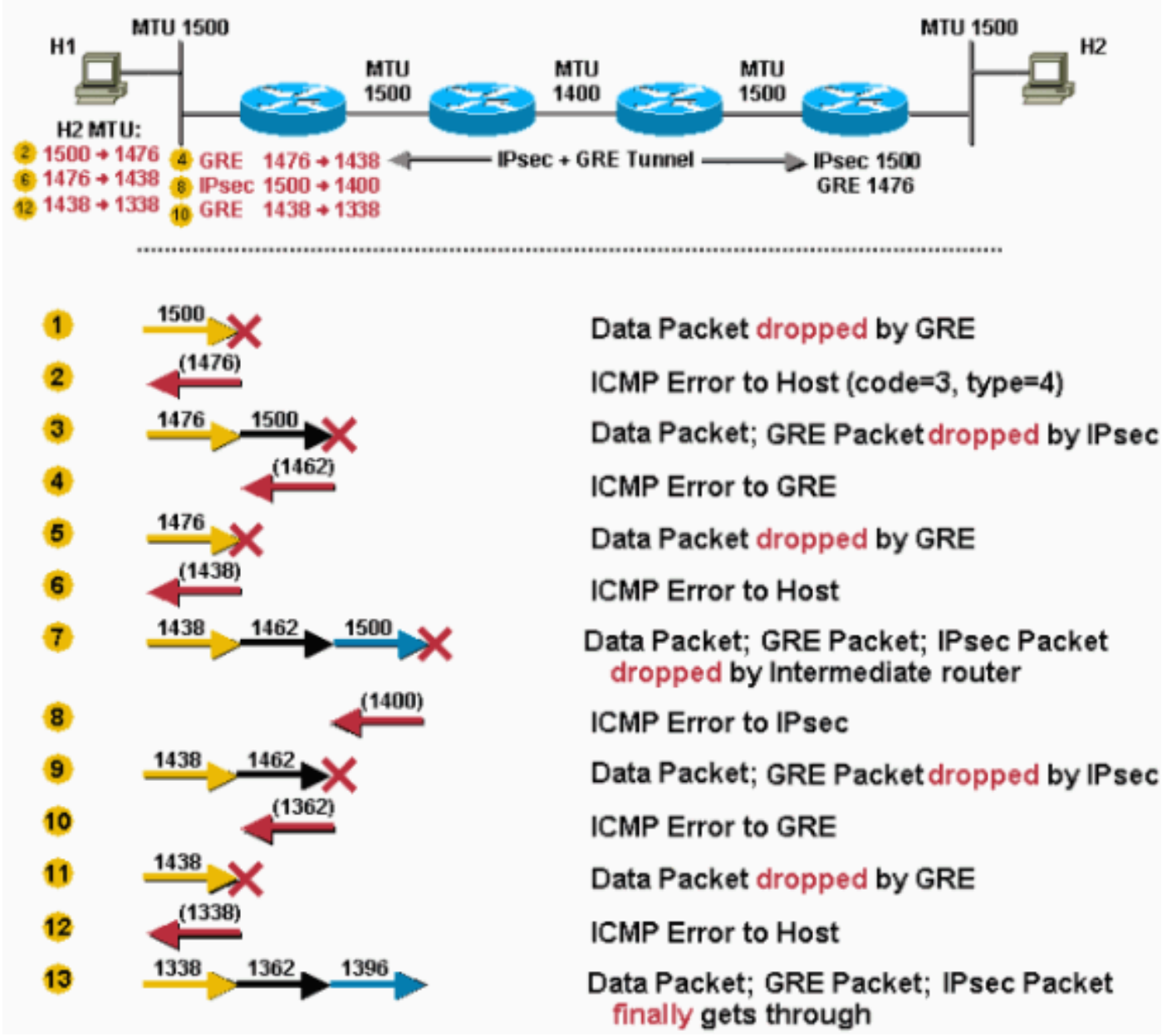
d'encapsulation à chacun, afin de donner un paquet de 1 552 octets et un paquet de 120 octets.

5. Le paquet IPsec de 1 552 octets est fragmenté par le routeur parce qu'il est plus grand que la valeur MTU sortante (1 500). Le paquet de 1 552 octets est segmenté : un paquet de 1 500 octets et un paquet de 72 octets (52 octets, plus l'ajout d'un en-tête IPv4 de 24 octets au second fragment). Les trois paquets de 1 500 octets, 72octets et 120 octets sont acheminés vers le routeur homologue IPv4sec + GRE.
6. Le routeur destinataire réassemble les deux fragments IPv4 du paquet (1 500 et 72 octets) pour reconstituer le paquet IPv4sec + GRE original de 1 552 octets. Aucun traitement n'est prévu pour le paquet IPv4sec + GRE de 120 octets.
7. IPv4sec déchiffre les paquets IPsec et GRE de 1 552 et de 120 octets afin d'obtenir des paquets GRE de 1 500 et de 68 octets.
8. GRE désencapsule les paquets GRE de 1 500 et de 68 octets afin d'obtenir des fragments de paquet IP de 1 476 et de 44 octets. Ces fragments de paquets IPv4 sont transférés vers l'hôte de destination.
9. L'hôte 2 réassemble ces fragments IPv4 afin d'obtenir le datagramme IPv4 original de 1 500 octets.

Le scénario 10 est semblable au scénario 8, à l'exception qu'on note une liaison MTU inférieure dans le chemin de tunnel. Il s'agit d'un scénario du pire cas pour le premier paquet envoyé de l'hôte 1 à l'hôte 2. Après la dernière étape de ce scénario, l'hôte 1 définit la PMTU correcte pour l'hôte 2 et tout va bien pour les connexions TCP entre l'hôte 1 et l'hôte 2. Les flux TCP entre l'hôte 1 et les autres hôtes (accessibles via le tunnel IPv4sec + GRE) ne doivent passer que par les trois dernières étapes du scénario 10.

Dans ce scénario, la **tunnel path-mtu-discovery** commande est configurée sur le tunnel GRE et le bit DF est défini sur les paquets TCP/IPv4 provenant de l'hôte 1.

Exemple 10



1. Le routeur reçoit un paquet de 1500 octets. Ce paquet est supprimé par GRE, parce que GRE ne peut pas fragmenter ou expédier le paquet car le bit DF est défini et la taille du paquet dépasse l'interface de sortie « ip mtu » après l'ajout du temps système GRE (24 octets).
2. Le routeur envoie un message ICMP à l'hôte 1 afin de l'informer que la valeur MTU du prochain nœud est de 1476 (1 500 - 24 = 1 476).
3. L'hôte 1 modifie son PMTU pour l'hôte 2 en 1476 et envoie la plus petite taille lorsqu'il retransmet le paquet. GRE l'encapsule et passe les paquets de 1 500 octets à IPv4sec. IPv4sec abandonne le paquet parce que GRE a copié le bit DF (activé) de l'en-tête IPv4 interne, et avec l'espace supplémentaire utilisé par IPv4sec (maximum de 38 octets), le paquet est trop grand pour passer l'interface physique.
4. IPv4sec envoie un message ICMP à GRE qui indique que la MTU du tronçon suivant est de 1 462 octets (puisqu'un maximum de 38 octets est ajouté pour le chiffrement et la surcharge IPv4). GRE enregistre la valeur 1438 (1462 - 24) en tant que « ip mtu » sur l'interface de tunnel.

> Remarque : Ce changement de valeur est stocké en interne et ne peut pas être vu dans le résultat de la **show ip interface tunnel**<#>commande. Cette modification n'est visible que si vous activez la **debug tunnel** commande.

5. La prochaine fois que l'hôte 1 retransmettra le paquet de 1476 octets, GRE le supprimera.
6. Le routeur envoie un message ICMP à l'hôte 1 pour indiquer que la valeur MTU du prochain nœud est de 1 438.
7. L'hôte 1 diminue le PMTU pour l'hôte 2 et retransmet un paquet de 1438 octets. Cette fois, GRE accepte le paquet, l'encapsule et le transmet à IPv4sec pour le chiffrement.
8. Le paquet IPv4sec est acheminé au routeur intermédiaire et abandonné, car il est limité par une MTU de 1 400 par l'interface de sortie.
9. Le routeur intermédiaire envoie un message ICMP à IPv4sec afin de l'informer que la valeur MTU du prochain nœud est de 1 400. Cette valeur est enregistrée par IPv4sec dans la valeur PMTU de la SA IPv4sec associée.
10. Lors de l'hôte 1 retransmet le paquet de 1 438 octets, GRE l'encapsule et le transmet à IPv4sec. IPv4sec abandonne le paquet parce qu'il a changé la valeur de sa propre PMTU pour 1 400.
11. IPv4sec envoie un message d'erreur ICMP à GRE pour indiquer que la valeur MTU du prochain nœud est de 1 362 octets, et GRE enregistre la valeur 1 338 à l'interne.
12. Quand l'hôte 1 retransmet le paquet d'origine (car il n'a pas reçu d'accusé de réception), GRE le supprime.
13. Le routeur envoie un message ICMP à l'hôte 1 pour indiquer que la valeur MTU du prochain nœud est de 1 338 (1 362 - 24 octets). L'hôte 1 abaisse son PMTU pour l'hôte 2 à 1338.
14. L'hôte 1 retransmet un paquet de 1338 octets, et cette fois, il peut finalement accéder complètement à l'hôte 2.

Autres recommandations

La configuration de la **tunnel path-mtu-discovery** commande sur une interface de tunnel peut aider l'interaction GRE et IPv4sec lorsqu'elles sont configurées sur le même routeur.

Sans la **tunnel path-mtu-discovery** commande configurée, le bit DF serait toujours effacé dans l'en-tête GRE IPv4.

Cela permet la fragmentation du paquet IPv4 GRE, même si le bit DF est activé dans l'en-tête IPv4 des données encapsulées, ce qui normalement ne permettrait pas la fragmentation du paquet.

Si la **tunnel path-mtu-discovery** commande est configurée sur l'interface de tunnel GRE :

1. GRE copie le bit DF de l'en-tête IPv4 de données vers l'en-tête IPv4 GRE.
2. Si le bit DF est défini dans l'en-tête IPv4 GRE et que le paquet est « trop volumineux » après le chiffrement IPv4sec pour la MTU IPv4 sur l'interface physique sortante, IPv4sec abandonne le paquet et notifie le tunnel GRE de réduire sa taille de MTU IPv4.
3. IPv4sec effectue la PMTUD pour ses propres paquets et si la PMTU IPv4sec change (si elle est réduite), alors IPv4sec ne notifie pas immédiatement GRE, mais quand un autre paquet

plus grand arrive, alors le processus de l'étape 2 se produit.
4. Le MTU IPv4 GRE est désormais plus petit, de sorte qu'il abandonne tous les paquets IPv4 de données avec le bit DF défini qui sont maintenant trop grands et envoie un message ICMP à l'hôte émetteur.

La **tunnel path-mtu-discovery** commande aide l'interface GRE à définir sa MTU IPv4 de manière dynamique, plutôt que statique avec la **ip mtu** commande. Il est recommandé d'utiliser ces deux commandes.

La **ip mtu** commande est utilisée pour fournir de l'espace pour la surcharge GRE et IPv4sec par rapport à la MTU IPv4 de l'interface physique sortante locale.

La **tunnel path-mtu-discovery** commande permet de réduire davantage la MTU IPv4 du tunnel GRE s'il existe une liaison MTU IPv4 inférieure dans le chemin entre les homologues IPv4sec.

Voici quelques avenues à considérer si vous avez des problèmes avec la PMTUD dans un réseau où des tunnels GRE + IPv4 sont configurés.

Cette liste commence par la solution la plus souhaitable.

1. Résolvez le problème de non-fonctionnement de la PMTUD, qui est habituellement du à un routeur ou à un pare-feu qui bloque les messages ICMP.
2. Utilisez la **ip tcp adjust-mss** commande sur les interfaces de tunnel de sorte que le routeur réduise la valeur MSS TCP dans le paquet SYN TCP. Cela permet aux deux hôtes finaux (l'expéditeur et le destinataire TCP) d'utiliser des paquets suffisamment petits pour que la PMTUD ne soit pas nécessaire.
3. Mettez en place une politique de routage sur l'interface d'entrée du routeur et configurez un cheminement dans lequel le bit DF est désactivé dans l'en-tête IPv4 des données avant que ce paquet soit transmis à l'interface de tunnel de la GRE. Cela permet de fragmenter le paquet IPv4 de données avant l'encapsulation GRE.
4. Augmentez la valeur « ip mtu » sur l'interface de tunnel GRE pour qu'elle soit égale à la valeur MTU de l'interface de sortie. Cela permet d'encapsuler le paquet de données IPv4 GRE sans le fragmenter au préalable. Le paquet GRE est ensuite chiffré IPv4sec, puis fragmenté pour sortir de l'interface physique sortante. Dans ce cas, vous ne configureriez pas de **tunnel path-mtu-discovery** commande sur l'interface de tunnel GRE. Cela peut considérablement réduire le débit, car le réassemblage du paquet IPv4 sur l'homologue IPv4sec s'effectue en mode de commutation de processus.

Informations connexes

- Page de support pour le routage IP
- Page d'assistance d'IPSec (protocole de sécurité IP)
- Découverte de MTU RFC 1191
- Options de découverte RFC 1063 IP MTU
- Internet Protocol RFC 791
- RFC 793 Transmission Control Protocol (TCP)
- RFC 879 Taille maximale des segments TCP et rubriques connexes

- RFC 1701 Encapsulation de routage générique (GRE)
- RFC 1241 Plan pour protocole IEP (Internet Encapsulation Protocol)
- Encapsulation RFC 2003 IP dans IP
- Assistance et documentation techniques - Cisco Systems

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.