使用 GRE 和 IPsec 解決 IPv4 分段、MTU、MSS 和 PMTUD 問題

目錄

簡介

本文件說明 IPv4 分段和路徑最大傳輸單位探索 (PMTUD) 如何運作。

背景資訊

另外還討論了與IPv4通道的不同組合結合時涉及PMTUD行為的案例。

IPv4 分段和重組

雖然 IPv4 資料包的最大長度為 65535，但大多數傳輸連結會強制採用較小的封包長度上限，稱為

MTU。MTU值取決於傳輸連結。

IPv4的設計可容納MTU差異，因為此設計允許路由器根據需要對IPv4資料包進行分段。

接收站負責將片段重組為原始的完整大小IPv4資料包。

IPv4分段會將資料包分解成多個部分，稍後再重新組裝。

IPv4來源、目的地、識別、總長度和片段位移欄位，以及IPv4標頭中的「more fragments」(MF)和「do not fragment」(DF)旗標，用於IPv4分段和重組。

有關 IPv4 分段和重組機制的詳細資訊，請參閱 RFC 791。

此圖說明 IPv4 標頭的配置。

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

16位元的識別碼是IPv4資料包的傳送者指定的值。這麼做可協助重組資料包的片段。

片段位移為 13 位元，指出片段屬於原始 IPv4 資料包的哪個位置。此值是8位元組的倍數。

IPv4標頭的標誌欄位中，有3位元用於控制標誌。「不分段」(DF)位元決定是否允許將封包分段。

系統會保留第0位元，且一律設為0。

第1位元是DF位元（0 =「可分段」，1 =「不分段」）。

第 2 位元是 MF 位元（0 =「最後片段」，1 =「還有片段」）。

價值	保留第 0 位元	第 1 位元 DF	第 2 位元 MF
0	0	可以	最後
1	0	不要	還有

如果增加了IPv4片段的長度，值會比原始IPv4資料包長度多出60。

總長度多出 60 是因為建立了三個額外的 IPv4 標頭，第一個片段之後的每個片段各一個。

第一個片段的位移為0，此片段的長度為1500；其中包括經過微幅修改的原始IPv4標頭的20位元組。

第二個片段的位移為185(185 x 8 = 1480)；此片段的資料部分在原始IPv4資料包中的起始位置為1480位元組。

此片段的長度為1500；其中包括為此片段建立的額外IPv4標頭。

第三個片段的位移為370(370 x 8 = 2960)；此片段的資料部分在原始IPv4資料包中的起始位置為2960位元組。

此片段的長度為1500；其中包括為此片段建立的額外IPv4標頭。

第四個片段的位移為 555 (555 x 8 = 4440)，這表示此片段的資料部分在原始 IPv4 資料包中的起始位置為 4440 位元組。

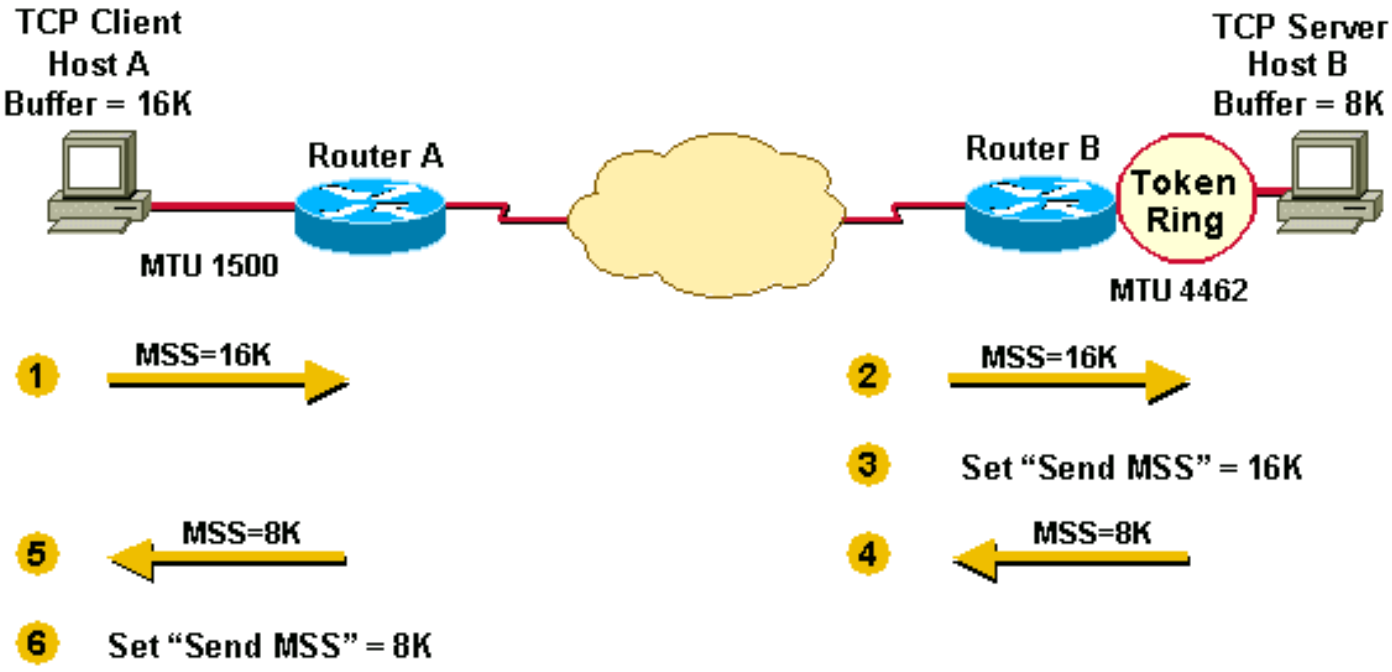
此片段的長度為700位元組；其中包括為此片段建立的額外IPv4標頭。

只有在收到最後一個片段時，才能確定原始 IPv4 資料包的大小。

最後一個片段 (555) 中的片段位移，為原始 IPv4 資料包帶來 4440 位元組的資料位移。

最後一個片段的資料位元組總和(680 = 700 - 20)產生5120位元組，這是原始IPv4資料包的資料部分。

IPv4標頭新增20位元組時，等於圖中所示的原始IPv4資料包大小(4440 + 680 + 20 = 5140)。



IPv4 分段問題

IPv4分段會導致將IPv4資料包分段的CPU和記憶體額外負荷些微增加。此情況適用於傳送者，以及傳送者與接收者之間路徑中的路由器。

建立片段的過程包括建立片段標頭以及將原始資料包複製到片段中。

此程式能以高效率完成，因為建立片段所需的資訊立即可用。

重組片段時，分段會導致接收者的額外負荷增加，因為接收者必須為到達的片段配置記憶體，並在收到所有片段後將它們合併回一個資料包。

在主機上進行重組不會構成問題，因為主機擁有用於此工作所需的時間和記憶體資源。

但是在路由器上進行重組的效率非常低，因為其主要工作是儘快轉送封包。

路由器的用途並非不限時間長度保留封包。

進行重組的路由器會選擇最大的可用緩衝區(18K)，因為在收到最後一個片段之前無法決定原始IPv4封包的大小。

另一個分段問題與遭捨棄片段的處理方式有關。

如果捨棄IPv4資料包的一個片段，那麼必須存在整個原始IPv4資料包，且也會將其分段。

網路檔案系統(NFS)會出現這種情況。NFS的讀取和寫入塊大小為8192。

因此，NFS IPv4/UDP資料包大約為8500位元組（包括NFS、UDP和IPv4標頭）。

連線到乙太網路(MTU 1500)的傳送站，必須將8500位元組資料包分段為六(6)部分；五(5)個1500位元組片段和一(1)個1100位元組片段。

如果由於連結擁塞而捨棄了六個片段中的任何一個，就必須重新傳輸整個原始資料包。這會產生另外6個片段要建立。

如果此連結捨棄六個封包中的其中一個，則幾乎無法透過此連結傳輸任何NFS資料，因為每個NFS 8500位元組原始IPv4資料包中至少會有一個IPv4片段被捨棄。

根據第4層(L4)到第7層(L7)資訊過濾或操縱封包的防火牆，無法正確處理IPv4片段。

如果IPv4片段的順序打亂，防火牆會封鎖非初始片段，因為這些片段不含與封包過濾器相符的資訊。

這表示接收主機無法重組原始IPv4資料包。

如果防火牆設定為允許包含不充足資訊的初始片段與過濾器正確相符，則可能會發生通過防火牆的非初始片段攻擊。

網路裝置（例如內容交換機引擎）根據L4至L7資訊來定向資料包，如果資料包跨越多個片段，則裝置在強制執行其策略時將會出現問題。

避免IPv4分段：TCP MSS的運作方式

傳輸控制通訊協定(TCP)最大片段大小(MSS)定義了主機接受單個TCP/IPv4資料包中的最大資料量。

此TCP/IPv4資料包可能在IPv4層進行分段。MSS 值只會在 TCP SYN 區段中作為 TCP 標頭選項加以傳送。

TCP 連線的兩端都會將其 MSS 值報告給另一端。主機之間不會協商MSS值。

傳送主機需要將單個 TCP 區段中的資料大小限制為小於或等於接收主機所報告的 MSS 值。

最初 MSS 指的是需在接收站上配置多少緩衝區大小（大於或等於 65496 位元組），才能儲存包含在單個 IPv4 資料包中的 TCP 資料。

MSS是TCP接收者要接受的最大資料區段。此TCP區段最大可為64K，且在IPv4層分段，以便傳輸到接收主機。

接收主機會重組 IPv4 資料包，然後再將完整的 TCP 區段傳遞到 TCP 層。

MSS值的設定方式及用於限制TCP區段和IPv4資料包大小的方式。

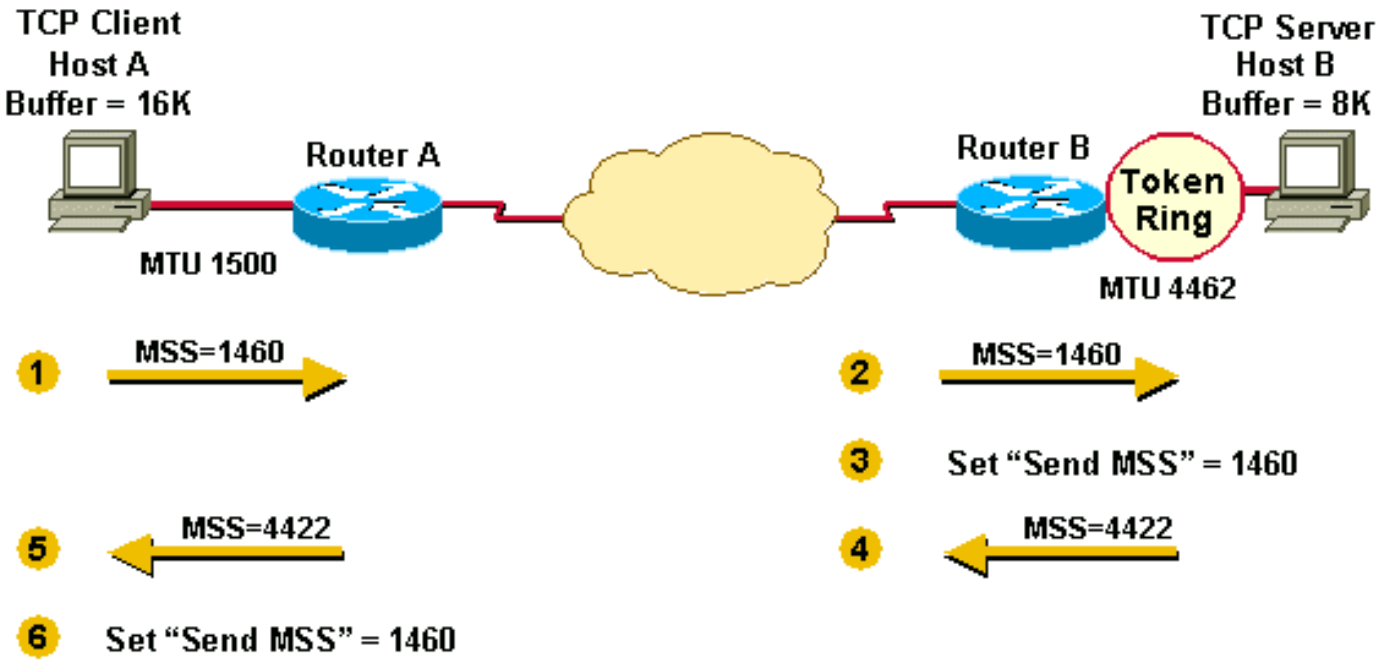
範例1說明首次實作MSS的方式。

主機 A 的緩衝區為16K，主機 B 的緩衝區為 8K。它們傳送和接收 MSS 值，並調整其傳送 MSS 以互相傳送資料。

主機A和主機B必須將IPv4資料包分段，該等資料包大於介面MTU，但小於傳送MSS，因為TCP堆疊會將16K或8K位元組資料從堆疊傳送到IPv4。

在主機B的情況下，封包會被分段以進入權杖環LAN，然後再次分段以進入乙太網路LAN。

範例 1



1. 主機 A 將其 MSS 值 16K 傳送給主機 B。
2. 主機 B 收到主機 A 傳來的 MSS 值 16K。
3. 主機 B 將其傳送 MSS 值設定為 16K。
4. 主機 B 將其 MSS 值 8K 傳送給主機 A。
5. 主機 A 收到主機 B 傳來的 MSS 值 8K。
6. 主機 A 將其傳送 MSS 值設定為 8K。

為了協助避免TCP連線的端點進行IPv4分段，將選擇的MSS值變更為最小緩衝區大小和傳出介面的MTU(-40)。

MSS數比MTU數小40位元組，因為MSS（TCP資料大小）不包括20位元組的IPv4標頭和20位元組的TCP標頭。

MSS取決於預設標頭大小；傳送者堆疊必須減去IPv4標頭的適當值，而TCP標頭取決於所使用的TCP或IPv4選項。

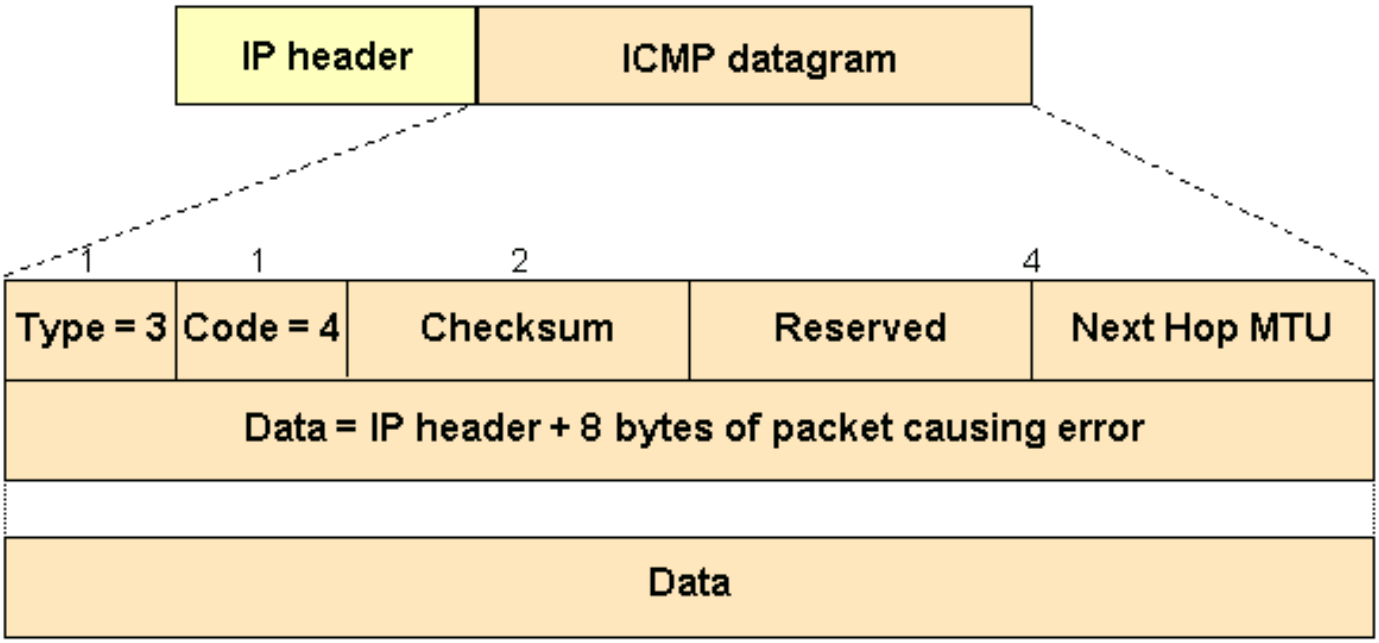
MSS目前的工作方式如下：每部主機先比較其傳出介面MTU和自己的緩衝區，並選擇最小值作為MSS加以傳送。

然後，主機將收到的MSS大小與自己的介面MTU進行比較，並再次選擇兩個值中的較低者。

範例2說明傳送者執行的這個額外步驟，其目的是避免在本機和遠端線路上進行分段。

在主機互相傳送其MSS值之前，每個主機都會考慮傳出介面的MTU。這有助於避免分段。

範例 2



1. 主機 A 比較其 MSS 緩衝區 (16K) 及 MTU (1500 - 40 = 1460) ，並使用較小的值作為 MSS (1460) 傳送給主機 B。
2. 主機B收到主機A的傳送MSS(1460)，並將其與傳出介面MTU - 40(4422)的值進行比較。
3. 主機 B 將 MSS 設定為較小的值 (1460)，以便將 IPv4 資料包傳送至主機 A。

4. 主機 B 比較其 MSS 緩衝區 (8K) 及 MTU (4462-40 = 4422)，並使用 4422 作為 MSS 傳送給主機 A。
5. 主機A收到主機B的傳送MSS(4422)，並將其與傳出介面MTU - 40(1460)的值進行比較。
6. 主機 A 將 MSS 設定為較小的值 (1460)，以便將 IPv4 資料包傳送至主機 B。

1460 是兩個主機為對方所選擇作為傳送 MSS 的值。通常TCP連線兩端的傳送MSS值相同。

在範例2中，不會在TCP連線的端點進行分段，因為主機會考慮兩個傳出介面MTU。

如果封包遇到的連結MTU小於其中一個主機的傳出介面MTU，則這些封包仍會在路由器A和路由器B之間的網路中進行分段。

什麼是PMTUD

TCP MSS會處理TCP連線的兩個端點上的分段，但不會處理這兩個端點之間的中間有較小MTU連結的情況。

之所以開發出 PMTUD，是為了避免在端點之間的路徑中進行分段。它用於以動態方式判斷從封包來源到目的地沿途間的最小MTU。

註：只有TCP和UDP支援PMTUD，其他通訊協定不支援。如果在主機上啟用了PMTUD，則來自該主機的所有TCP和UDP封包都會設定DF位元。

主機傳送已設定 DF 位元的完整 MSS 資料封包時，如果收到封包可能需要進行分段的資訊，PMTUD 便會降低連線的傳送 MSS 值。

主機記錄目的地的MTU值，因為它會在其路由表中建立含有此MTU值的主機(/32)專案。

如果路由器嘗試將IPv4資料包（已設定DF位元）轉送到MTU小於封包大小的連結，那麼路由器將會捨棄該封包，並將網際網路控制訊息通訊協定(ICMP)「無法到達目的地」訊息傳回到IPv4資料包來源，顯示代表「需要分段和設定DF」的代碼（型別3，代碼4）。

來源站收到ICMP訊息時會降低傳送MSS，而當TCP重新傳輸區段時，會使用較小的區段大小。

以下範例顯示開啟指令後，路由器上看到的ICMP「需要分段和設**debug ip icmp**定DF」訊息：

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

此圖顯示「需要分段和設定 DF」、「無法到達目的地」訊息的 ICMP 標頭格式。

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

根據 RFC 1191，若路由器傳回表示「需要分段和設定 DF」的 ICMP 訊息，則該路由器必須在 ICMP 規範 RFC 792 中標記為「未使用」的 ICMP 額外標頭欄位內，於低位 16 位元中包含下個躍點網路的 MTU。

RFC 1191 的早期實作沒有提供下個躍點 MTU 資訊。即使提供了此資訊，某些主機也會將其忽略。

在本案例中，RFC 1191也包含一張表，列出在PMTUD期間降低MTU的建議值。

主機使用它來更快達到傳送MSS的合理值，如本範例所示。

由於傳送者和接收者之間的路徑可以透過動態方式變更，因此會在所有封包上持續執行PMTUD。

每當傳送者收到「無法分段」ICMP訊息，便會更新路由資訊（儲存PMTUD的位置）。

進行 PMTUD 期間可能會發生兩種可能的情況：

1.封包可以直接傳給接收者，不需要進行分段。

註：為了讓路由器保護CPU免受DoS攻擊，會將其可傳送的ICMP無法到達訊息數量限製為每秒兩次。因此在這種情況下，如果您的網路預期路由器將需要以每秒（可為不同主機）超過兩個ICMP訊息（型別= 3，代碼= 4）進行回應，請使用**no ip icmp rate-limit unreachable [df] interface**命令停用限制ICMP訊息數的功能。

2.傳送者收到來自通向接收者的路徑上躍點的ICMP「不能分段」訊息。

TCP 資料流兩個方向的 PMTUD 是獨立進行的。

在某些情況下，資料流的一個方向上的PMTUD會觸發其中一個終端站，使其降低傳送MSS，而另一個終端站會保留原始傳送MSS，因為它從未傳送過大小足以觸發PMTUD的IPv4資料包。

例如，示例3中描述的HTTP連線。TCP 用戶端會傳送小型封包，伺服器則傳送大型封包。

在此案例中，只有來自伺服器的大型封包（大於576位元組）會觸發PMTUD。

來自使用者端的封包較小（小於576位元組），不會觸發PMTUD，因為它們不需進行分段即可穿越576 MTU連結。

範例 3

範例4顯示非對稱路由範例，其中一個路徑的最小MTU小於另一個。

經由不同路徑在兩個端點之間傳送和接收資料時，會發生非對稱路由。

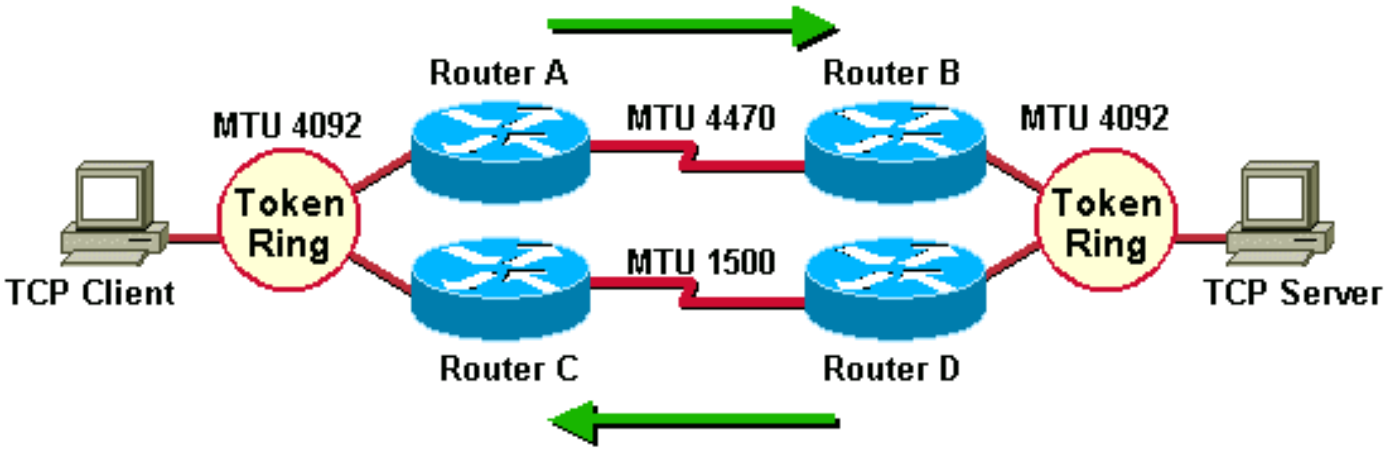
在本範例中，PMTUD只會在TCP資料流的一個方向上觸發降低傳送MSS。

從 TCP 用戶端傳到伺服器的流量流經路由器 A 和路由器 B，而從伺服器傳到用戶端的返回流量則流經路由器 D 和路由器 C。

當TCP伺服器將封包傳送到使用者端時，PMTUD會觸發伺服器降低傳送MSS，因為路由器D必須將4092位元組的封包分段，才能將其傳送到路由器C。

反之，使用者端永遠不會收到含有表示「需要分段和設定DF」的代碼之ICMP「無法到達目的地」訊息，因為路由器A在透過路由器B將封包傳送到伺服器時，不需要對封包進行分段。

範例 4



註:ip tcp path-mtu-discovery指令用於為路由器（例如BGP和Telnet）所起始的TCP連線啟用TCP MTU路徑探索功能。 ip tcp path-mtu-discovery指令

PMTUD 問題

以下情況可能會破壞PMTUD。

- 路由器捨棄封包而不傳送ICMP訊息。（不常見）
- 路由器產生並傳送了ICMP訊息，但該路由器和傳送者之間的路由器或防火牆封鎖了ICMP訊息。（常見）
- 路由器產生並傳送了ICMP訊息，但傳送者忽略該訊息。（不常見）

這三項中的第一個和最後一個通常是錯誤導致的結果，但中間那項描述的則是很常見的問題。

實作ICMP封包過濾器的那些裝置往往會封鎖所有ICMP訊息型別，而不是只封鎖特定ICMP訊息型別。

封包過濾器可以封鎖除了「無法到達」或「超過時間」等型別以外的所有ICMP訊息型別。

PMTUD的成敗取決於ICMP無法到達訊息能否確實傳給TCP/IPv4封包的傳送者。

ICMP 超過時間訊息對其他 IPv4 問題來說非常重要。

此處範例顯示在路由器上實作的這種封包過濾器。

```
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
```

還有其他一些技巧可用於緩解ICMP遭完全封鎖的問題。

- 清除路由器上的DF位元並允許分段。(不過，這不是個好主意。如需詳細資訊，請參閱 IP 分段問題）。
- 使用interface指令操縱TCP MSS選項值**ip tcp adjust-mss <500-1460>**MSS。

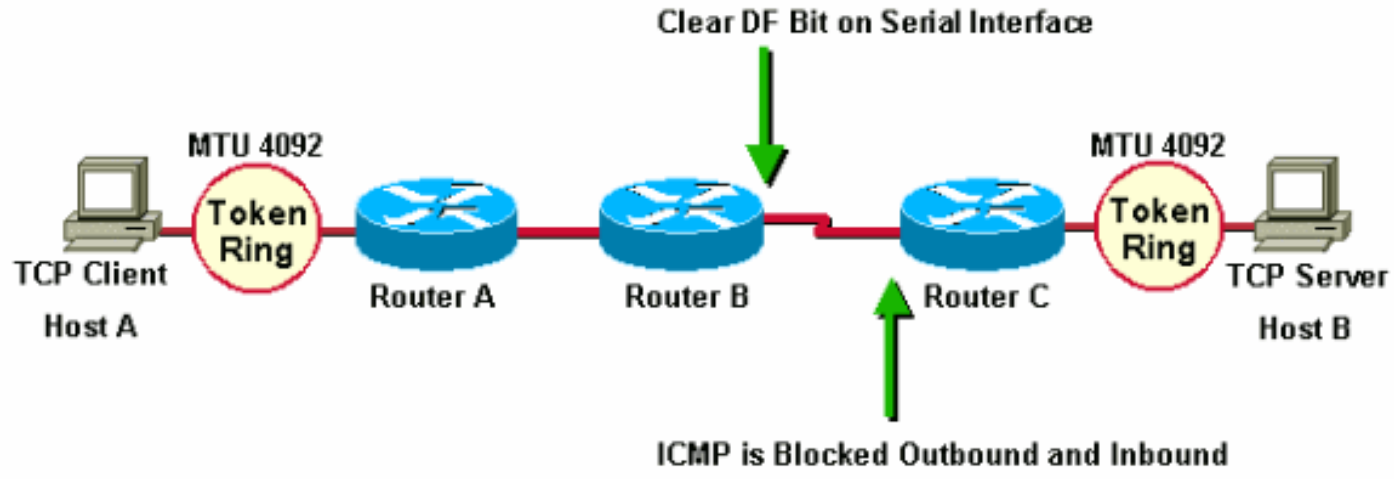
在下一個示例中，路由器A和路由器B位於同一個管理域中。路由器 C 無法連線且封鎖了 ICMP，因此破壞了 PMTUD。

這種情形的解決方法是清除路由器 B 兩個方向的 DF 位元，以便允許分段。可以使用原則路由來完成此操作。

Cisco IOS® 軟體版本 12.1(6) 及更新版本提供用於清除 DF 位元的語法。

```
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
```



另一種方法是在經過路由器的 SYN 封包上變更 TCP MSS 選項值（Cisco IOS® 12.2(4)T 及更新版本提供此功能）。

這會降低TCP SYN封包中的MSS選項值，使其小於命令中的值(1460**ip tcp adjust-mss**)。

結果是TCP傳送者會傳送不大於此值的區段。

IPv4封包大小大於MSS值（1460位元組）40位元組(1500)，以便為容納TCP標頭（20位元組）和IPv4標頭（20位元組）。

您可以使用指令調整TCP SYN封包的**ip tcp adjust-mss**MSS。此語法會將TCP區段上的MSS值降低到1460。

這個指令會影響介面 serial0 上的傳入和傳出流量。

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

因為 IPv4 通道的部署越來越廣泛，IPv4 分段問題隨之變得更加普遍。

通道之所以會導致進行更多分段，是因為通道封裝會對封包大小增加「額外負荷」。

例如，若新增通用路由器封裝(GRE)，會讓封包增加24位元組，增加後需要將封包分段，因為它大於傳出MTU。

需要 PMTUD 的常見網路拓撲

如果網路上發生中繼連結的 MTU 比終端連結的 MTU 還小的情形，就需要使用 PMTUD。存在這些較小 MTU 連結的幾個常見原因如下：

- 連接權杖環（或 FDDI）的終端主機之間有乙太網路連線。兩端的權杖環（或 FDDI）MTU 大於中間的乙太網路 MTU。
- PPPoE（通常搭配 ADSL 一起使用）需要 8 位元組用於其標頭。這會將乙太網路的有效 MTU 降低到 1492 (1500 - 8)。

GRE、IPv4sec和L2TP等通道通訊協定也需有各自的標頭和標尾的空間。這也降低了傳出介面的有效MTU。

通道

通道是思科路由器上的邏輯介面，可用來封裝傳輸通訊協定中的乘客封包。

這個架構的目的是提供服務以實作點對點封裝配置。通道介面具有以下三個主要元件：

- 乘客通訊協定（AppleTalk、Banyan VINES、CLNS、DECnet、IPv4 或 IPX）
- 載體通訊協定 - 以下封裝通訊協定中的其中一個：
  - GRE — 思科多重通訊協定載體通訊協定。如需詳細資訊，請參閱 RFC 2784和 RFC 1701。
  - IPv4 內 IPv4 通道 - 如需詳細資訊，請參閱 RFC 2003。

- 傳輸通訊協定 - 用於承載封裝通訊協定的通訊協定。

本節顯示的封包解釋了 IPv4 通道概念，其中 GRE 是封裝通訊協定，IPv4 是傳輸通訊協定。

乘客通訊協定也是 IPv4。在這種情況下，IPv4 既是傳輸通訊協定，也是乘客通訊協定。

正常封包

IPv4	TCP	Telnet

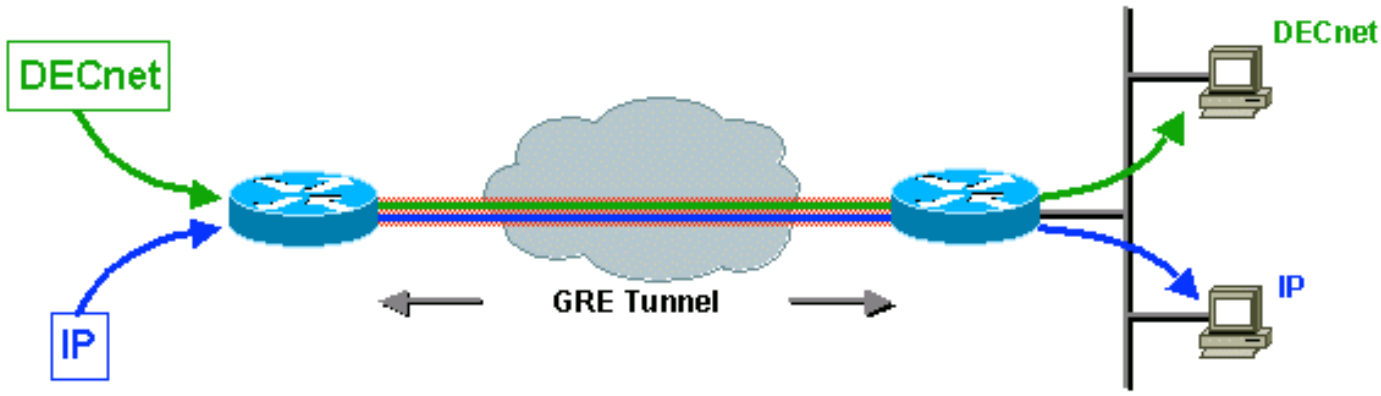
通道封包

IPv4	GRE	IPv4	TCP	Telnet

- IPv4 是傳輸通訊協定。
- GRE 是封裝通訊協定。
- IPv4 是乘客通訊協定。

下一個範例以 IPv4 作為封裝通訊協定、DECnet 作為乘客通訊協定，並將 GRE 作為載體。

如圖所示，這說明了載體通訊協定封裝多個乘客通訊協定的可能性。



網路管理員考慮在IPv4主幹分隔開有兩個不連續非IPv4網路的情況下進行通道化。

如果不連續的網路執行DECnet，管理員可以選擇將兩者連線在一起（或不連線），方法是在骨幹中設定DECnet。

管理員不希望允許DECnet路由消耗主幹頻寬，因為這可能干擾IPv4網路的效能。

可行的替代方案是透過 IPv4 骨幹來以通道傳輸 DECnet 流量。通道解決方案將DECnet封包封裝在IPv4中，並將其傳送到已移除封裝的通道端點，且會透過DECnet將DECnet封包路由到其目的地。

封裝另一種通訊協定中流量有一些優點：

- 端點使用私人位址 (RFC 1918)，而骨幹不支援路由這些位址。
- 允許跨 WAN 或網際網路的虛擬私人網路 (VPN)。

- 透過單一通訊協定骨幹將不連續的多通訊協定網路連接在一起。
- 透過骨幹或網際網路加密流量。

之後，IPv4用作乘客通訊協定，IPv4用作傳輸通訊協定。

有關通道介面的注意事項

以下是進行通道傳輸時的注意事項。

- Cisco IOS® 11.1 版引入了 GRE 通道的快速交換功能，而 12.0 版引入 CEF 交換功能。
- 12.2(8)T 版引入了多點 GRE 通道的 CEF 交換功能。
- 只支援處理序交換的舊版 Cisco IOS® 中，通道端點的封裝和解除封裝作業執行速度緩慢。
- 透過通道傳輸封包時存在安全和拓撲問題。通道可以繞過存取控制清單 (ACL) 和防火牆。
- 如果要透過防火牆進行通道傳輸，則會繞過正在通道傳輸的乘客通訊協定。因此，建議在通道端點加入防火牆功能，以便對乘客通訊協定強制執行任何原則。
- 由於延遲增加，通道會造成具備有限計時器的傳輸通訊協定（例如DECnet）出現問題。
- 若跨越具有不同速度連結的環境進行通道傳輸（例如快速FDDI環和穿過慢速9600-bps電話線路），就會帶來封包重新排序問題。部分乘客通訊協定在混合媒體網路中的運作效果不佳。
- 點對點通道會消耗實體連結上的頻寬。透過多個點對點通道，每個通道介面都有一個頻寬，且執行中通道所在的實體介面也有一個頻寬。例如，如果通過10 Mb鏈路運行100個隧道，請將隧道頻寬設定為100 Kb。通道的預設頻寬為 9Kb。
- 路由通訊協定偏好使用通道而非實際連結，因為通道看起來就像是具有最低成本路徑的單躍點連結，但實際上卻包含更多躍點，且成本比另一種路徑更高。正確設定路由通訊協定便可緩解此情形。請考慮透過通道介面執行不同的路由通訊協定，而不是在實體介面上執行的路由通訊協定。
- 設定通往通道目的地的適當靜態路由，便可避免遞迴路由問題。當通往通道目的地的最佳路徑是經由通道本身，便稱為遞迴路由。這種情況會導致通道介面上下跳動。遞迴路由問題存在時會顯示此錯誤。

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

在通道端點擔任 PMTUD 參與者的路由器

當路由器是通道的端點時，需扮演兩個不同的 PMTUD 角色。

- 路由器的第一個角色是主機封包的轉送者。處理 PMTUD 時，路由器需要檢查原始資料封包

的 DF 位元和封包大小，並在必要時執行適當的動作。

- 路由器將原始 IPv4 封包封裝到通道封包中後，第二個角色便開始發揮作用。這個階段中，在 PMTUD 和通道 IPv4 封包方面，路由器的作用更像是主機。

當路由器扮演第一個角色（轉送主機IPv4封包的路由器）時，路由器將主機IPv4封包封裝到通道封包中之前，此角色便開始發揮作用。

如果路由器擔任主機封包的轉送者，它將會完成以下動作：

- 檢查 DF 位元是否已設定.
- 檢查通道可容納的封包大小.
- 分段（如果封包過大且未設定DF位元）、封裝片段並傳送片段；或
- 捨棄封包（如果封包過大且已設定 DF 位元），並傳送 ICMP 訊息給傳送者.
- 封裝（如果封包沒有過大）並傳送.

一般情況下，可以選擇先封裝再分段（傳送兩個封裝片段），或是先分段再封裝（傳送兩個經過封裝的片段）。

本節詳細介紹兩個顯示PMTUD和穿越範例網路的資料包互動的範例。

第一個範例顯示當路由器（於通道來源）負責轉送路由器時，封包會發生什麼情況。

處理PMTUD時，路由器需要檢查原始資料封包的DF位元和封包大小，並執行適當的動作。

此範例針對通道使用 GRE 封裝。GRE先分段再進行封裝。

稍後的範例會說明先封裝後分段的案例。

在範例 1 中，並未設定 DF 位元 (DF = 0)，而 GRE 通道 IPv4 MTU 為 1476 (1500 - 24)。

範例 1

1.轉送路由器（位於通道來源）收到傳送主機傳來的1500位元組資料包，且其DF位元已清除(DF = 0)。

這個資料包由 20 位元組的 IP 標頭加上 1480 位元組的 TCP 負載組成。

IPv4	1480 位元組 TCP + 資料

2.由於增加GRE額外負荷（24位元組）後，封包對於IPv4 MTU而言過大，因此轉送路由器將資料包分割為兩個片段，分別為1476位元組（20位元組IPv4標頭+ 1456位元組IPv4負載）和44位元組（20位元組的IPv4標頭+ 24位元組的IPv4負載）

新增GRE封裝後，封包不會大於傳出實體介面MTU。

IP_0	1456 位元組 TCP + 資料
IP_1	24 位元組資料

3.轉送路由器為原始IPv4資料包的每個片段新增了GRE封裝，其中包括4位元組的GRE標頭加上20位元組的IPv4標頭。

現在這兩個 IPv4 資料包的長度為 1500 和 68 位元組，且被視為個別的 IPv4 資料包而不是片段。

IPv4	GRE	IP_0	1456 位元組 TCP + 資料
IPv4	GRE	IP_1	24 位元組資料

4.通道目的地路由器從原始資料包的每個片段中移除GRE封裝，這讓兩個IPv4片段的長度分別變為1476和24位元組。

這些 IPv4 資料包片段由此路由器分別轉送至接收主機。

IP_0	1456 位元組 TCP + 資料
IP_1	24 位元組資料

5.接收主機將這兩個片段重組為原始資料包。

IPv4	1480 位元組 TCP + 資料

範例2說明轉送路由器在網路拓撲環境中的作用。

路由器的作用與轉送路由器相同，但這一次已設定DF位元(DF = 1)。

範例 2

1.通道來源的轉送路由器收到傳送主機傳來的1500位元組資料包，其DF = 1。

IPv4	1480 位元組 TCP + 資料

2.由於已設定DF位元，且資料包大小（1500位元組）大於GRE通道IPv4 MTU(1476)，因此路由器捨棄資料包，並向資料包的來源傳送「需要ICMP分段但DF位元已設定」訊息。

ICMP訊息警示傳送者MTU為1476。

IPv4	ICMP MTU 1476

3.傳送主機收到ICMP訊息，但重新傳送原始資料時，會使用1476位元組的IPv4資料包。

IPv4	1456 位元組 TCP + 資料

4.此IPv4資料包長度（1476位元組）現在等於GRE通道IPv4 MTU的值，因此路由器將GRE封裝新增到IPv4資料包。

IPv4	GRE	IPv4	1456 位元組 TCP + 資料

5.接收路由器（位於通道目的地）移除IPv4資料包的GRE封裝，並將其傳送至接收主機。

IPv4	1456 位元組 TCP + 資料

這是路由器扮演第二個角色，針對PMTUD及通道IPv4封包發揮傳送主機的作用時會發生的情況。

路由器將原始IPv4封包封裝到通道封包中後，此角色便開始發揮作用。

注意:預設情況下，路由器不會在其產生的GRE通道封包上執行PMTUD。**tunnel path-mtu-discovery**指令可用於為GRE-IPv4通道封包開啟PMTUD。

範例 3 顯示當主機傳送的 IPv4 資料包夠小，GRE 通道介面上的 IPv4 MTU 足以容納時會發生的情況。

在此案例中，DF 位元可為已設定或已清除（1 或 0）。

GRE通道介面未設定命令**tunnel path-mtu-discovery**，因此路由器不會在GRE IPV4封包上宕機。

範例 3

1.位於通道來源的轉送路由器收到來自傳送主機的1476位元組資料包。

IPv4	1456 位元組 TCP + 資料

2.此路由器將1476位元組的IPv4資料包封裝到GRE中，得到1500位元組的GRE IPv4資料包。

GRE IPv4標頭中的DF位元已清除(DF = 0)。接著此路由器將這個封包轉送到通道目的地。

IPv4	GRE	IPv4	1456 位元組 TCP + 資料

3.假設通道來源和目的地之間有一個連結MTU為1400的路由器。

此路由器將通道封包分段，因為DF位元已清除(DF = 0)。

請記得，此範例將最外部的IPv4片段，因此GRE、內部IPv4和TCP標頭只會顯示在第一個片段中。

IP_0	GRE	IP	1352 位元組 TCP + 資料
IP_1	104 位元組資料		

4.通道目的地路由器必須重組GRE通道封包。

IP	GRE	IP	1456 位元組 TCP + 資料

5.重組GRE通道封包後，路由器移除GRE IPv4標頭，並在傳送過程中傳送原始IPv4資料包。

IPv4	1456 位元組 TCP + 資料

範例4顯示路由器針對PMTUD及通道IPv4封包發揮傳送主機的作用時，發生了什麼情況。

這一次，已在原始IPv4標頭中設定DF位元(DF = 1)，且已設定命令，因此會將DF位元從內部IPv4標頭複製到外部(GRE + IPv4)標頭**tunnel path-mtu-discovery**。

範例 4

1.位於通道來源的轉送路由器收到來自傳送主機的1476位元組資料包，其DF = 1。

IPv4	1456 位元組 TCP + 資料

2.此路由器將1476位元組的IPv4資料包封裝到GRE中，得到1500位元組的GRE IPv4資料包。

此GRE IPv4標頭已設定DF位元(DF = 1)，因為原始IPv4資料包已設定DF位元。

接著此路由器將這個封包轉送到通道目的地。

IPv4	GRE	IPv4	1456 位元組 TCP

3.同樣地，假設通道來源和目的地之間有一個連結MTU為1400的路由器。

此路由器不會將通道封包分段，因為DF位元已設定(DF=1)。

此路由器必須捨棄封包並向通道來源路由器傳送ICMP錯誤訊息，因為此封包上的來源IPv4位址。

IPv4	ICMP MTU 1400

4.通道來源的轉送路由器收到這個「ICMP」錯誤訊息，並將GRE通道IPv4 MTU降低到1376(1400 - 24)。

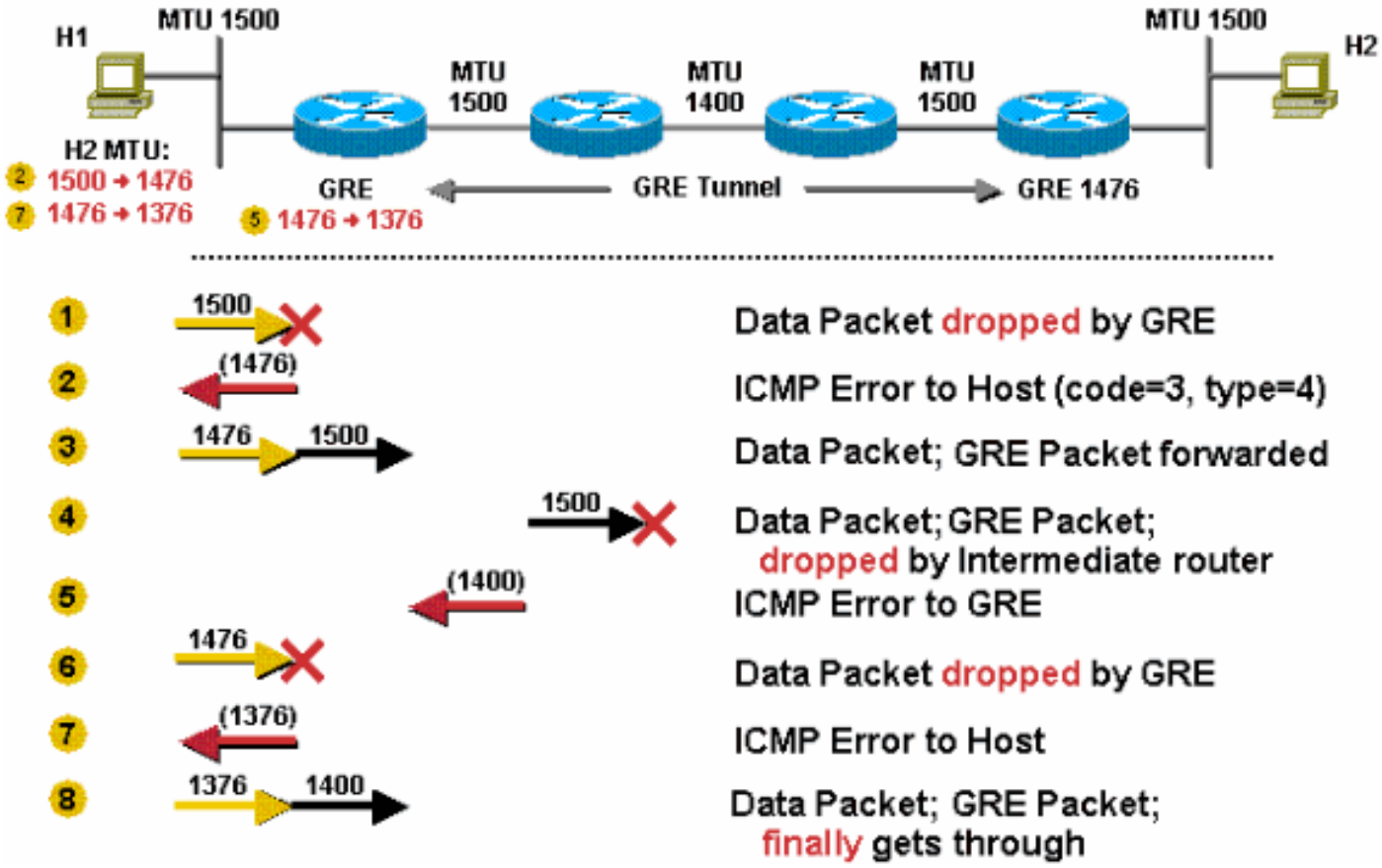
下次傳送主機以1476位元組的IPv4封包重新傳輸資料時，此封包可能過大，因此此路由器會向傳送者傳送MTU值為1376的「ICMP」錯誤訊息。

傳送主機重新傳輸資料時，會以1376位元組的IPv4封包傳送資料，此封包將穿過GRE通道到達接收主機。

範例 5

此範例說明GRE分段。先分段再進行GRE封裝，然後為資料封包執行PMTUD，且若IPv4封包是由GRE封裝，系統就不會複製DF位元。

未設定DF位元。依照預設，GRE 通道介面 IPv4 MTU 會比實體介面 IPv4 MTU 小 24 位元組，因此 GRE 介面 IPv4 MTU 是 1476，如圖所示。



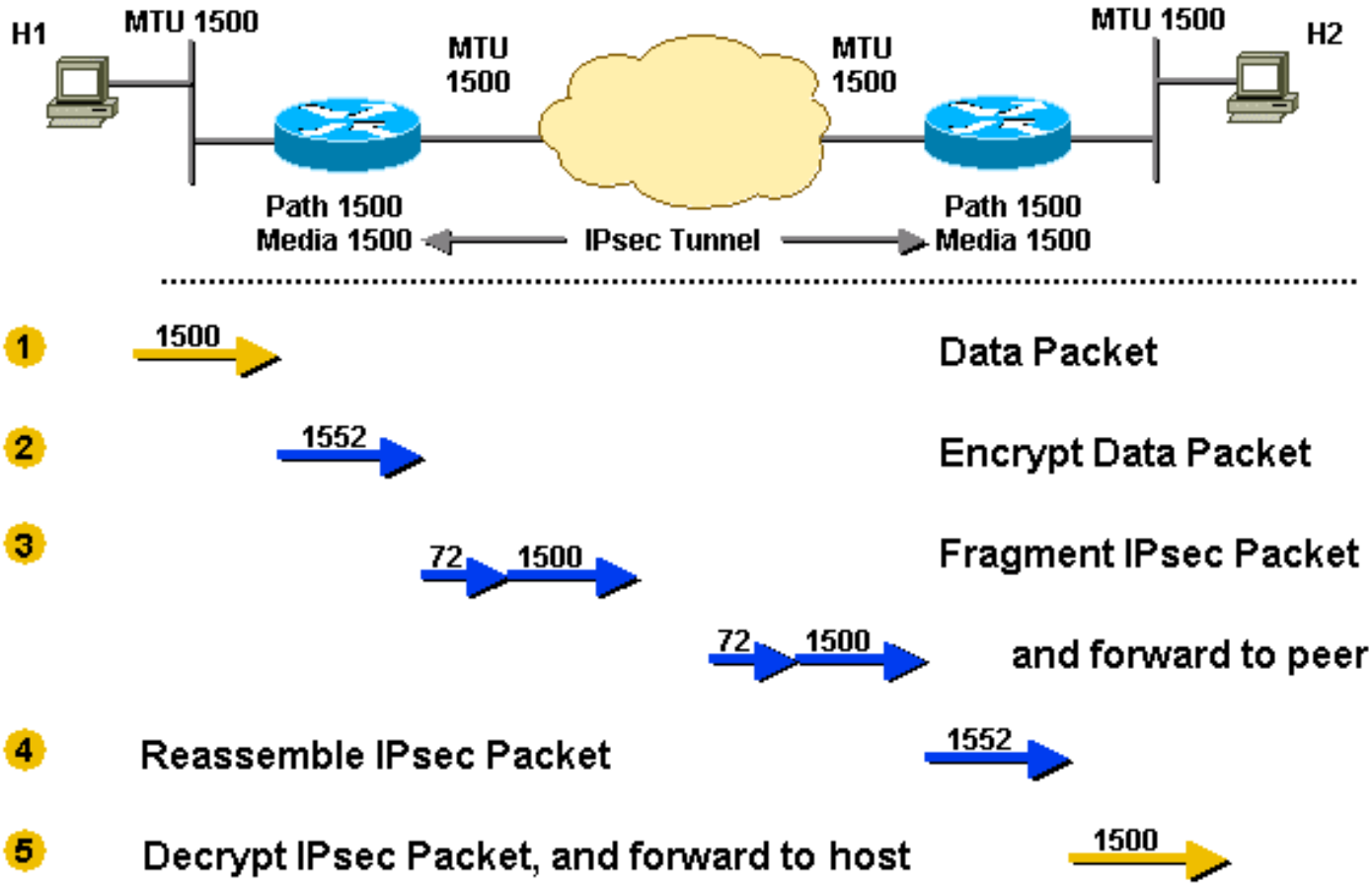
1. 傳送者傳送一個 1500 位元組的封包（20 位元組 IPv4 標頭 + 1480 位元組的 TCP 負載）。
2. 由於GRE通道的MTU為1476，因此1500位元組封包被分割為兩個IPv4片段，分別為1476和44位元組，各自預計會多出額外24位元組的GRE標頭。
3. 將 24 位元組的 GRE 標頭新增到各個 IPv4 分段。現在兩個片段分別是 1500 (1476 + 24) 和 68 (44 + 24) 位元組。
4. 包含兩個 IPv4 片段的 GRE + IPv4 封包將轉送到 GRE 通道對等路由器。
5. GRE 通道對等路由器從兩個封包中移除 GRE 標頭。
6. 此路由器將兩個封包轉送到目的地主機。
7. 目的主機將 IPv4 片段重組回原始的 IPv4 資料包。

範例 6

此範例與範例5類似，但這一次已設定DF位元。路由器設定為使用指令對GRE + IPv4通道封包執行PMTUD，並將DF位元從原始IPV4標頭複製到tunnel path-mtu-discoveryGRE IPv4標頭。

如果路由器收到 GRE + IPv4 封包的 ICMP 錯誤，便會降低 GRE 通道介面的 IPv4 MTU。

預設情況下，GRE通道IPv4 MTU設定為比實體介面MTU小24位元組，因此GRE IPv4 MTU為1476。GRE通道路徑中有一個1400 MTU連結，如圖所示。



1. 路由器收到一個 1500 位元組封包（20 位元組 IPv4 標頭 + 1480 TCP 負載）並將該封包捨棄。封包大於 GRE 通道介面的 IPv4 MTU (1476)，因此路由器將其捨棄。
2. 路由器傳送一個 ICMP 錯誤給傳送者，告知下一個躍點的 MTU 為 1476。主機記錄此資訊，通常會記錄為其路由表中目的地的主機路由。
3. 傳送主機重新傳送資料時使用 1476 位元組的封包大小。GRE 路由器新增了 24 位元組的 GRE 封裝，並將 1500 位元組的封包傳出。
4. 1500 位元組的封包無法周遊 1400 位元組的連結，因此中繼路由器將其捨棄。
5. 中繼路由器向 GRE 路由器傳送 ICMP（類型 = 3，代碼 = 4），告知下一個躍點的 MTU 為 1400。GRE 路由器將此值降低到 1376 (1400 - 24)，並在 GRE 介面上設定內部 IPv4 MTU 值。只有當您使用命令時才會看到此**debug tunnel** 更改；它不能顯示在命令的輸出中**show ip interface tunnel**<#>顯示。
6. 下次主機重新傳送1476位元組封包時，GRE路由器將會捨棄該封包，因為它大於GRE通道介面上目前的IPv4 MTU(1376)。
7. GRE路由器傳送另一個ICMP（型別= 3，代碼= 4）給傳送者，告知下一個躍點的MTU為1376，而主機會使用新值更新其目前的資訊。
8. 主機再次重新傳送資料，但現在GRE以較小的1376位元組封包加入24位元組的封裝並繼續轉送。這一次，封包會順利到達GRE通道對等路由器，在此解除封裝並傳送至目的地主機。

註:在此案例中，如果沒有在轉送路由器上設定**tunnel path-mtu-discovery**，且已在透過GRE通道轉送的封包中設定了DF位元，那麼主機1仍會成功將TCP/IPv4封包傳送到主機2，但會在1400 MTU連結的中間將這些封包分段。此外，GRE通道對等路由器必須再次重組這些封包，才能將其解除封裝並進行轉送。

純 IPsec 通道模式

IPv4 安全 (IPv4sec) 通訊協定是一種以標準為基準的方法，可為透過 IPv4 網路傳輸的資訊提供隱私性、完整性和真實性。

IPv4sec 提供 IPv4 網路層加密。IPv4sec 藉由新增至少一個 IPv4 標頭（通道模式）來延長 IPv4 封包。

新增的標頭長度各異，取決於IPv4sec組態模式，但每個封包不超過~58位元組(封裝安全負載(ESP)和ESP驗證(ESPauth))。

IPv4sec 有兩種模式，分別為通道模式和傳輸模式。

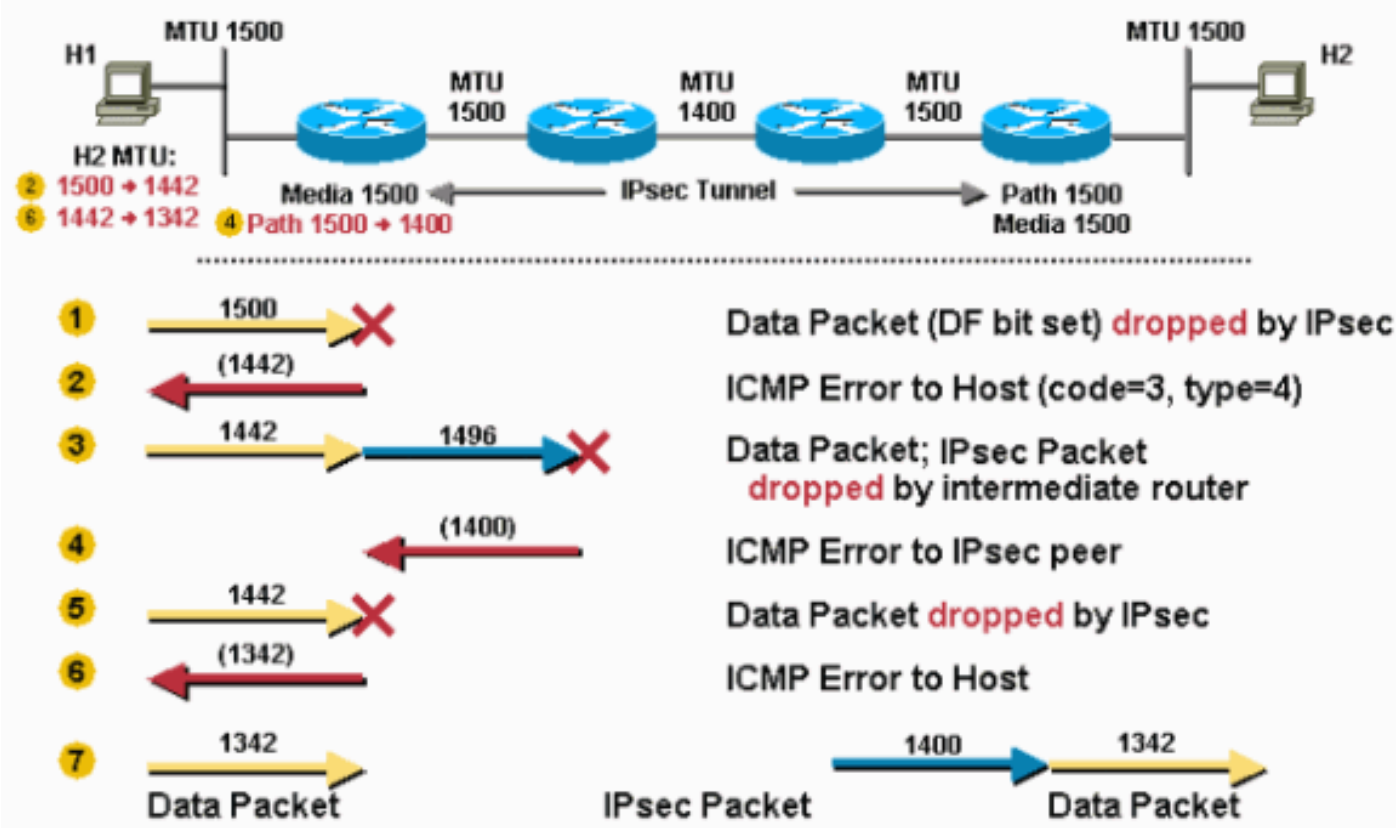
1. 通道模式是預設的模式。在通道模式下，整個原始 IPv4 封包都受到保護（經過加密、驗證或兩者均有），並以 IPv4sec 標頭和標尾封裝。接著在封包前面加上新的 IPv4 標頭，此標頭會指定 IPv4sec 端點（對等路由器）為來源和目的地。通道模式可與任何單點傳播 IPv4 流量搭配使用，且如果 IPv4sec 要保護來自 IPv4sec 對等路由器之後主機的流量，就必須使用通道模式。例如，通道模式可搭配虛擬私人網路 (VPN) 使用，其中一個受保護網路上的主機會透過一對 IPv4sec 對等路由器，將封包傳送到其他受保護網路上的主機。透過 VPN，IPv4sec「通道」藉由在 IPv4sec 對等路由器之間加密此流量來保護主機之間的 IPv4 流量。
2. 若使用傳輸模式(使用transform定義中的子命令設**mode transport**定)，則只有原始IPv4封包的負載會受到保護（經過加密、驗證或兩者均有）。負載會以 IPv4sec 標頭和標尾封裝。原始 IPv4 標頭會完整保留，不同之處在於 IPv4 通訊協定欄位會變更為 ESP (50)，而原始通訊協定值儲存在 IPv4sec 標尾中，以便在封包解密時復原。只有當要保護的 IPv4 流量是 IPv4sec 對等路由器本身之間的流量時，才會使用傳輸模式，封包上的來源和目的地 IPv4 位址與 IPv4sec 對等路由器位址相同。通常只有在使用另一個通道通訊協定（例如 GRE）來封裝 IPv4 封包，然後使用 IPv4sec 來保護 GRE 通道封包時，才會使用 IPv4sec 傳輸模式。

IPv4sec 一律會對資料封包和自己的封包執行 PMTUD。有一些 IPv4sec 組態指令可用於修改 IPv4sec IPv4 封包的 PMTUD 處理方式，IPv4sec 可以清除、設定 DF 位元，或將 DF 位元從資料封包 IPv4 標頭複製到 IPv4sec IPv4 標頭。此功能稱為「DF 位元覆寫功能」。

注意:使用IPv4sec進行硬體加密時，請避免封裝後進行分段。硬體加密可以為您提供約50 Mb的輸送量（視硬體而定），但如果將IPv4sec封包分段，則會損失百分之50到90的輸送量。這個損失是因為分段的 IPv4sec 封包需透過執行序交換進行重組，然後傳遞給硬體加密引擎進行解密。輸送量的損失可能使硬體加密輸送量下降到軟體加密 (2-10 Mbs) 的效能水準。

範例 7

此案例說明 IPv4sec 分段的實際運作情況。在此案例中，整個路徑中的 MTU 為 1500。在此案例中，並未設定 DF 位元。



1. 路由器收到要傳給主機 2 的 1500 位元組封包（20 位元組的 IPv4 標頭 + 1480 位元組 TCP 負載）。
2. 1500 位元組封包是以 IPv4sec 加密，且增加了 52 位元組的額外負荷（IPv4sec 標頭、標尾和其他 IPv4 標頭）。現在 IPv4sec 需要傳送 1552 位元組的封包。由於傳出MTU為1500，因此此封包必須分段。
3. 將 IPv4sec 封包分段後建立了兩個片段。分段期間，為第二個片段增加了額外的20位元組IPv4標頭，因此產生1500位元組的片段和72位元組的IPv4片段。
4. IPv4sec 通道對等路由器收到片段，去除額外的 IPv4 標頭，並將 IPv4 片段合併回原始 IPv4sec 封包。接著 IPv4sec 解密此封包。
5. 然後路由器將原始的 1500 位元組封包轉送至主機 2。

範例 8

此範例與範例6類似，不同之處在於，此案例是在原始資料封包中設定DF位元，而IPv4sec通道對等之間的路徑中有一個連結的MTU小於其他連結。

此範例展示IPv4sec對等路由器如何執行兩種PMTUD角色的功能，如在通道端點擔任PMTUD參與者的路由器一節所述。

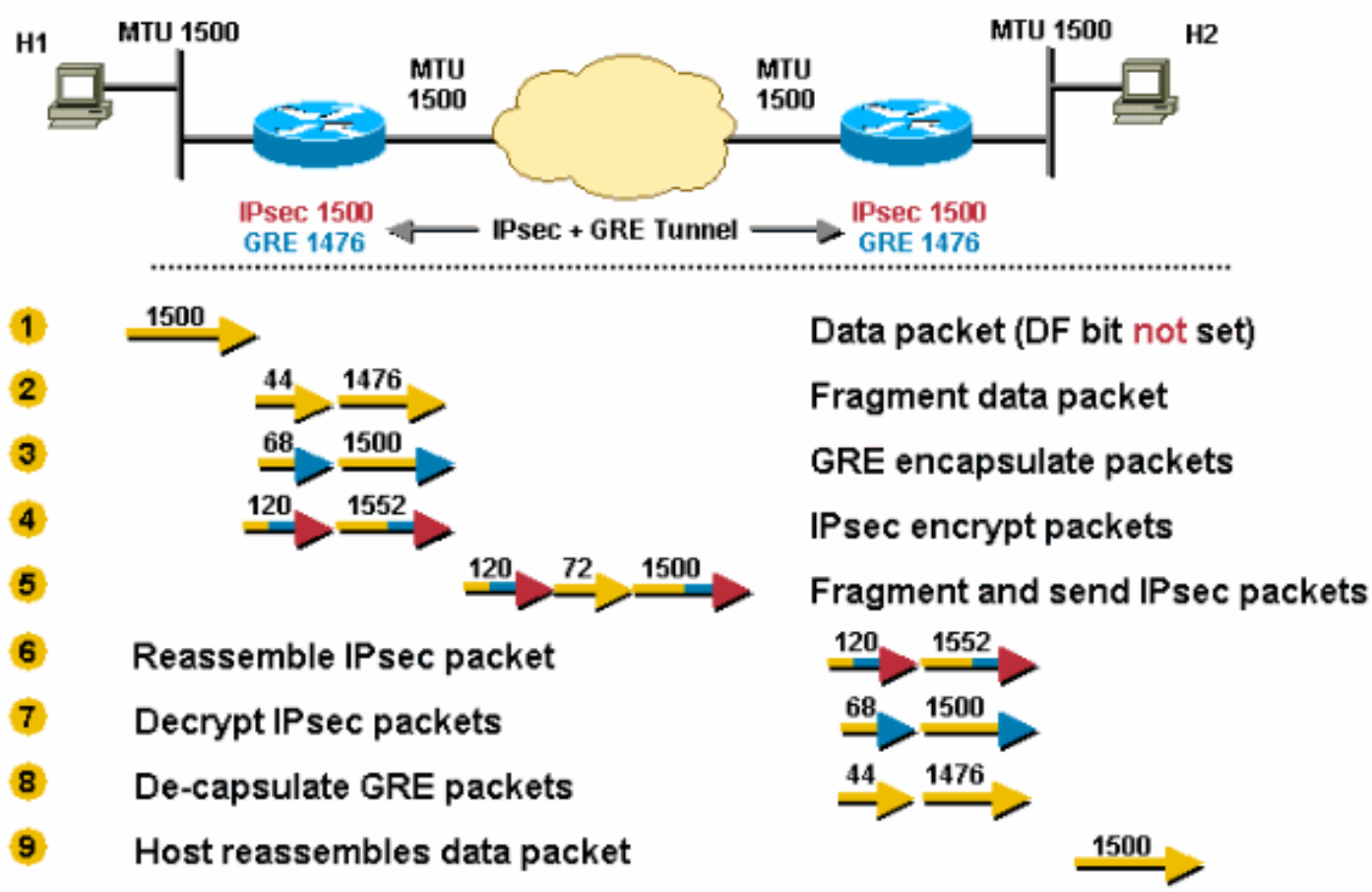
由於需要分段，IPv4sec PMTU會變更以降低值。

IPv4sec加密封包時，會將DF位元從內部IPv4標頭複製到外部IPv4標頭。

媒體 MTU 和 PMTU 值儲存在 IPv4sec 安全關聯 (SA) 中。

媒體 MTU 是以傳出路由器介面的 MTU 為依據，而 PMTU 是根據 IPv4sec 對等之間的路徑上出現的最小 MTU。

IPv4sec在嘗試將封包分段之前，先對封包進行封裝/加密，如圖所示。



1. 路由器收到1500位元組的封包並將其捨棄，因為若加上IPv4sec額外負荷，將使封包大於PMTU(1500)。
2. 路由器向主機 1 傳送一則 ICMP 訊息，告知下一個躍點的 MTU 為 1442 (1500 - 58 = 1442)。這58位元組是使用IPv4sec ESP和ESPauth時的最大IPv4sec額外負荷。實際IPv4sec額外負荷可能比此值小7位元組。主機 1 在其路由表中記錄此資訊，通常會記錄為目的地（主機 2）的主機路由。
3. 主機1將主機2的PMTU降低到1442，因此主機1會在將資料重新傳輸到主機2時，使用較小的封包（1442位元組）。路由器收到 1442 位元組封包，而 IPv4sec 增加 52 位元組的加密額外負荷，因此產生的 IPv4sec 封包為 1496 位元組。由於此封包在其標頭中設定了 DF 位元，因此含有 1400 位元組 MTU 連結的中間路由器將該封包捨棄。
4. 捨棄這個封包的中間路由器會向 IPv4sec 封包的傳送者（第一個路由器）傳送一則 ICMP 訊息，告知下一個躍點的 MTU 為 1400 位元組。此值記錄在 IPv4sec SA PMTU 中。
5. 下次主機1重新傳輸1442位元組封包時（它沒有收到相關的確認訊息），IPv4sec捨棄該封包。對封包增加IPv4sec額外負荷後，會使其大於PMTU(1400)，因此路由器捨棄該封包。
6. 路由器向主機1傳送一則ICMP訊息，告知下一個躍點的MTU現在為1342。(1400 - 58 = 1342)。主機1再次記錄此資訊。
7. 當主機1再次重新傳輸資料時，會使用大小較小的封包(1342)。此封包不需要分段，且已透過IPv4sec通道到達主機2。

GRE 和 IPv4sec 搭配使用

使用 IPv4sec 加密 GRE 通道時，分段和 PMTUD 會發生更複雜的互動。

以這種方式結合 IPv4sec 和 GRE，是因為 IPv4sec 不支援 IPv4 多點傳送封包，這表示無法透過 IPv4sec VPN 網路執行動態路由通訊協定。

而 GRE 通道支援多點傳送，因此 GRE 通道可用於先將動態路由通訊協定多點傳送封包封裝在 GRE IPv4 單點傳播封包中，然後 IPv4sec 便可加密該封包。

執行此作業時，通常會在GRE之上以傳輸模式部署IPv4sec，因為IPv4sec對等路由器和GRE通道端點（路由器）是相同的，而傳輸模式節省了20位元組的IPv4sec額外負荷。

將 IPv4 封包分割為兩個片段並由 GRE 封裝時的情況值得留意。

在此案例中，IPv4sec看到兩個獨立的GRE + IPv4封包。通常在預設組態中，其中一個封包會夠大，因此加密後需要將其分段。

IPv4sec對等路由器必須先重組此封包才能進行解密。在傳送路由器上，這種「雙重分段」（一次在 GRE 之前、一次在 IPv4sec 之後）會增加延遲並降低輸送量。

重組會進行處理序交換，因此每當發生這種情況，接收路由器上都會發生CPU命中。

將 GRE 通道介面上的「ip mtu」設定降到夠低，將 GRE 和 IPv4sec 的額外負荷都列入考慮（預設情況下，GRE 通道介面「ip mtu」設為傳出實際介面 MTU - GRE 額外負荷位元組），就可以避免這種情況。

下表列出假設傳出實體介面的MTU為1500的每個通道/模式組合的建議MTU值。

通道組合	需要的具體 MTU	建議的 MTU
GRE + IPv4sec（傳輸模式）	1440 位元組	1400 位元組
GRE + IPv4sec（通道模式）	1420 位元組	1400 位元組

注意:建議使用MTU值1400，因為此值涵蓋了最常見的GRE + IPv4sec模式組合。此外，允許額外的 20 或 40 位元組額外負荷，尚未出現可辨識的負面影響。記住並設定一個值較為輕鬆，而且此值幾乎適用於所有案例。

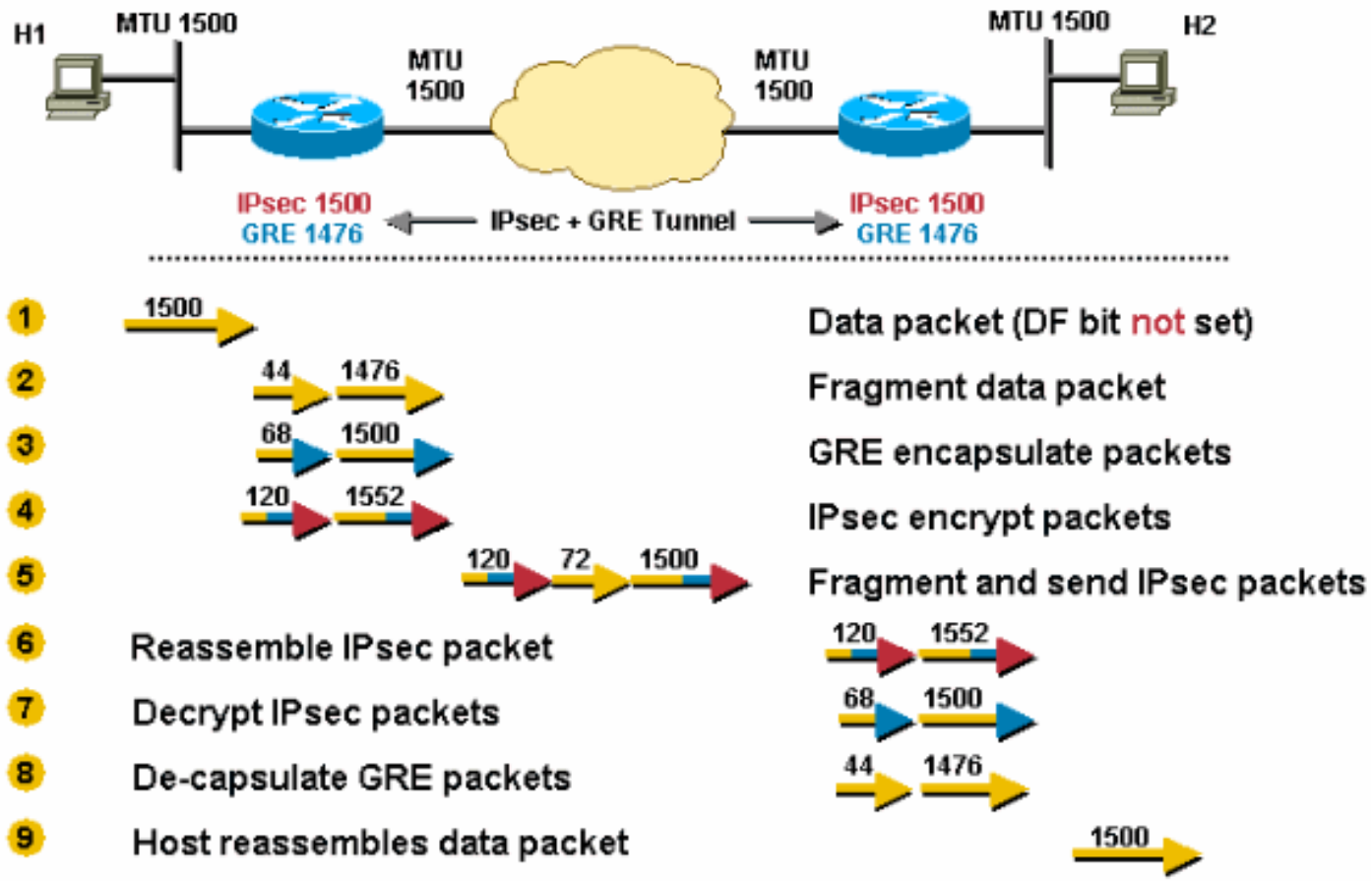
範例 9

IPv4sec 部署在 GRE 之上。傳出實體 MTU 為 1500，IPv4sec PMTU 為 1500，GRE IPv4 MTU 為 1476 (1500 - 24 = 1476)。

因此，TCP/IPv4封包分段兩次，一次在GRE之前，一次在IPv4sec之後。

封包在GRE封裝之前進行分段，其中一個GRE封包在IPv4sec加密後再次分段。

此案例中，在 GRE 通道上設定「ip mtu 1440」（IPv4sec 傳輸模式）或「ip mtu 1420」（IPv4sec 通道模式）會除去雙重分段的可能性。

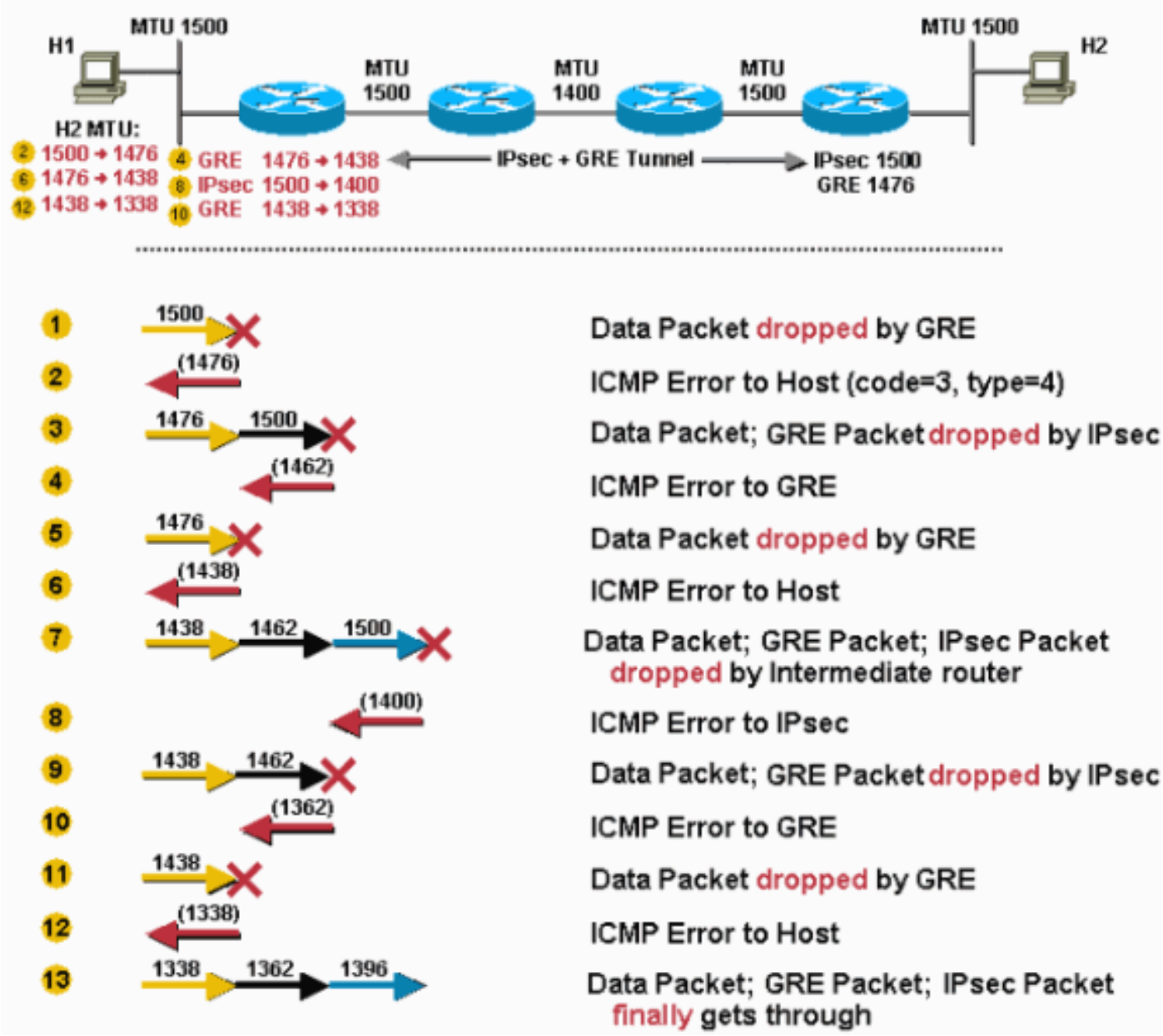


1. 路由器收到 1500 位元組的資料包。
2. 進行封裝之前，GRE 將 1500 位元組的封包分段為兩個部分，分別為 1476 位元組 (1500 - 24 = 1476) 和 44 位元組（24 位元組資料 + 20 位元組 IPv4 標頭）。
3. GRE 封裝 IPv4 片段，為每個封包增加 24 位元組。這會產生兩個 GRE + IPv4sec 封包，分別為 1500 (1476 + 24 = 1500) 和 68 (44 + 24) 位元組。
4. IPv4sec對兩個封包進行加密，為各封包增加了52位元組（IPv4sec通道模式）的封裝額外負荷，以便提供1552位元組和120位元組的封包。
5. 1552 位元組 IPv4sec 封包大於傳出 MTU (1500)，因此路由器將其分段。1552 位元組的封包被分割為多個部分：一個 1500 位元組的封包和一個 72 位元組的封包（52 位元組「負載」加上第二個片段的額外 20 位元組 IPv4 標頭）。將三個封包（分別為 1500 位元組、72 位元組和 120 位元組）轉送到 IPv4sec + GRE 對等路由器。
6. 接收路由器重組兩個 IPv4sec 片段（1500 位元組和 72 位元組），以取得原始 1552 位元組的 IPv4sec + GRE 封包。不需要對 120 位元組 IPv4sec + GRE 封包執行任何動作。
7. IPv4sec 將 1552 位元組和 120 位元組的 IPv4sec + GRE 封包解密，以取得 1500 位元組和 68 位元組的 GRE 封包。
8. GRE 將 1500 位元組和 68 位元組的 GRE 封包解除封裝，以取得 1476 位元組和 44 位元組的 IPv4 封包片段。這些 IPv4 封包片段會被轉送至目的地主機。
9. 主機 2 會重組這些 IPv4 片段，以取得原始的 1500 位元組 IPv4 資料包。

案例 10 與案例 8 類似，只是通道路徑中存在 MTU 較小的連結。對於從主機 1 傳送到主機 2 的第一個封包而言，這是最糟糕的情況。進行這個案例的最後一個步驟之後，主機1為主機2設定了正確的PMTU，同時也為主機1和主機2之間的TCP連線進行設定。主機1和其他主機之間的TCP資料流（可透過IPv4sec + GRE通道連線）只需經過案例10的最後三個步驟。

在此案例中，**tunnel path-mtu-discovery**於GRE通道上設定命令，並在來自主機1的TCP/IPV4封包上設定DF位元。

範例 10



- 路由器收到 1500 位元組的封包。GRE 捨棄此封包，因為在已設定 DF 位元的情況下，GRE 不能分段或轉送封包，而且增加 GRE 額外負荷（24 位元組）後，封包大小會超過傳出介面「ip mtu」。
- 路由器向主機 1 傳送一則 ICMP 訊息，告知下一個躍點的 MTU 為 1476 (1500 - 24 = 1476)。
- 主機 1 將主機 2 的 PMTU 變更為 1476，並在重新傳輸封包時傳送較小的大小。GRE 將其封裝，並將 1500 位元組的封包傳給 IPv4sec。IPv4sec 捨棄封包，因為 GRE 已從內部 IPv4 標頭複製 DF 位元（已設定），且因為含有 IPv4sec 額外負荷（最多 38 位元組）導致封包過大，無法從實體介面轉送出去。
- IPv4sec向GRE傳送一則ICMP訊息，說明下一個躍點的MTU為1462位元組（因為為加密和IPv4額外負荷增加了最多38位元組）。GRE 將值 1438 (1462 - 24) 記錄為通道介面上的「ip mtu」。

- 註：此值的更改儲存在內部，在命令輸出中無法看**show ip interface tunnel<#>**到。只有改用命令時才會看到此更**debug tunnel**改。

- 下次主機 1 重新傳輸 1476 位元組封包時，GRE 將該封包捨棄。
- 路由器向主機 1 傳送一則 ICMP 訊息，表示下一個躍點的 MTU 是 1438。
- 主機 1 降低主機 2 的 PMTU，並重新傳輸 1438 位元組的封包。這一次，GRE 接受封包，將其封裝並傳給 IPv4sec 進行加密。
- IPv4sec 封包被轉送到中繼路由器並遭捨棄，因為其傳出介面 MTU 為 1400。
- 中繼路由器向 IPv4sec 傳送一則 ICMP 訊息，告知下一個躍點的 MTU 為 1400。IPv4sec 將此值記錄在相關 IPv4sec SA 的 PMTU 值中。
- 主機 1 重新傳輸 1438 位元組封包時，GRE 將其封裝並將傳給 IPv4sec。IPv4sec 捨棄封包，因為它已將自己的 PMTU 變更為 1400。
- IPv4sec 向 GRE 傳送一個 ICMP 錯誤訊息，表示下一個躍點的 MTU 為 1362，而 GRE 在內部記錄值 1338。
- 主機 1 重新傳輸原始封包時（因為它沒有收到相關的確認訊息），GRE 捨棄該封包。
- 路由器向主機 1 傳送一則 ICMP 訊息，表示下一個躍點的 MTU 為 1338（1362 - 24 位元組）。主機 1 將主機 2 的 PMTU 降為 1338。
- 主機 1 重新傳輸 1338 位元組的封包，這一次終於可以順利到達主機 2。

更多建議

在通道**tunnel path-mtu-discovery** 介面上設定命令，可協助同一個路由器上設定的GRE和IPv4sec進行互動。

如果沒有設定**tunnel path-mtu-discovery**命令，系統一律會清除GRE IPV4標頭中的DF位元。

這會允許對 GRE IPv4 封包進行分段，即使封裝資料 IPv4 標頭設定了 DF 位元（這樣通常不允許對封包進行分段）也一樣。

如果在**tunnel path-mtu-discovery**GRE通道介面上設定了命令：

1. GRE將DF位元從資料IPv4標頭複製到GRE IPv4標頭。
2. 如果已在GRE IPv4標頭中設定DF位元，且在對實體傳出介面上的IPv4 MTU進行IPv4sec加密後，封包為「過大」，則IPv4sec捨棄封包，並通知GRE通道降低其IPv4 MTU大小。
3. IPv4sec對自己的封包執行PMTUD時，如果IPv4sec PMTU變更（若降低），IPv4sec並不會立即通知GRE，但當有另一個較大的封包傳入時，則會發生步驟2中的程式。
4. GRE IPv4 MTU現在變小了，因此它會捨棄已設定DF位元但現在過大的所有資料IPv4封包，並傳送ICMP訊息給傳送主機。

此命**tunnel path-mtu-discovery**令可協助GRE介面以動態方式設定其IPv4 MTU，而不是使用命令以靜態方**ip mtu**式設定。實際上建議同時使用這兩個指令。

此命**ip mtu**令用於為GRE和IPv4sec額外負荷提供空間（相對於本機實體傳出介面IPv4 MTU）。

如果**tunnel path-mtu-discovery**IPv4sec對等路由器之間的路徑中存在IPv4 MTU較小的連結，此指令可用來進一步降低GRE通道IPv4 MTU。

在已設定 GRE + IPv4sec 通道的網路中，如果 PMTUD 出現問題，可以執行以下操作。

此清單從最理想的解決方案開始列出。

1. 修正 PMTUD 無法運作的問題，這通常是因為封鎖 ICMP 的路由器或防火牆所導致。
2. 在通道**ip tcp adjust-mss**介面上使用命令，以便路由器降低TCP SYN封包中的TCP MSS值。這麼做可協助兩個終端主機（TCP傳送者和接收者）使用夠小而不需要執行PMTUD的封包。
3. 在路由器的輸入介面上使用原則路由，並設定一個路由映像，用於在進入 GRE 通道介面之前清除資料 IPv4 標頭中的 DF 位元。這允許在GRE封裝之前對資料IPv4封包進行分段。
4. 提高 GRE 通道介面上的「ip mtu」，使其等於傳出介面 MTU。這允許資料IPv4封包進行GRE封裝，而不是先將其分段。接著，GRE封包會進行IPv4sec加密，然後分段以傳出實體介面。在這種情況下，不需在**tunnel path-mtu-discovery**GRE通道介面上設定命令。這可能會大幅降低輸送量，因為 IPv4sec 對等路由器上的 IPv4 封包重組作業會在執行序交換模式下完成。

相關資訊

- IP 路由支援頁面
- IPSec（IP 安全通訊協定）支援頁面
- RFC 1191 路徑 MTU 探索
- RFC 1063 IP MTU 探索選項
- RFC 791 網際網路通訊協定
- RFC 793 傳輸控制通訊協定
- RFC 879 TCP 最大區段大小和相關主題
- RFC 1701 通用路由封裝 (GRE)
- RFC 1241 用於網際網路封裝通訊協定的配置
- RFC 2003 IP 內 IP 封裝
- 技術支援與文件 - Cisco Systems

關於此翻譯

思科已使用電腦和人工技術翻譯本文件，讓全世界的使用者能夠以自己的語言理解支援內容。請注意，即使是最佳機器翻譯，也不如專業譯者翻譯的內容準確。Cisco Systems, Inc. 對這些翻譯的準確度概不負責，並建議一律查看原始英文文件（提供連結）。