解决 GRE 和 IPsec 中的 IPv4 分段、MTU、MSS 和 PMTUD 问题

目录

简介

本文档介绍 IPv4 分段和路径最大传输单元发现 (PMTUD) 的工作方式。

背景信息

此外，还讨论了当结合不同的IPv4隧道组合时，涉及PMTUD行为的场景。

IPv4 分段和重组

尽管 IPv4 数据报的最大长度为 65535 字节，但大多数传输链路强制执行更小的最大数据包长度限

制（即 MTU）。MTU值取决于传输链路。

IPv4的设计可以容纳MTU差异，因为它允许路由器根据需要对IPv4数据报进行分段。

接收站负责将分段重组为原始的全尺寸IPv4数据报。

IPv4分段将数据报划分为多个片段，稍后再重新组合。

IPv4报头中的IPv4 source、destination、identification、total length和fragment offset字段以及“更多分段” (MF)和“不分段” (DF)标志用于IPv4分段和重组。

有关 IPv4 分段和重组机制的更多信息，请参阅 RFC 791。

下图显示了 IPv4 报头的布局。

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

标识是16位，是由IPv4数据报的发送方分配的值。这有助于重组数据报的片段。

分段偏移量字段长 13 位，表示分段在原始 IPv4 数据报中所属的位置。此值为8字节的倍数。

IPv4报头的标志字段中有3个控制标志位。“不分段” (DF)位确定是否允许对数据包进行分段。

位0被保留，并且始终设置为0。

位1是DF位(0 = "can fragment"， 1 = "do not fragment")。

位 2 为 MF 位（0 =“最后一个分段”，1 =“更多分段”）。

价值	位 0（保留）	位 1 (DF)	位 2 (MF)
0	0	5 月	最后一页
1	0	不分段	更多

如果添加IPv4分段的长度，则值会比原始IPv4数据报长度大60。

总长度之所以增加 60 字节是因为为第一个分段后的另外三个分段多创建了三个 IPv4 报头，每个分段一个。

第一个分段的偏移量为0，此分段的长度为1500；这包括20个字节，表示经过略微修改的原始IPv4报头。

第二个分段的偏移量为185 (185 x 8 = 1480)；此分段的数据部分从原始IPv4数据报的1480个字节开始。

此分段的长度为1500；其中包括为此分段创建的其他IPv4报头。

第三个分段的偏移量为370 (370 x 8 = 2960)；此分段的数据部分从原始IPv4数据报的2960个字节开始。

此分段的长度为1500；其中包括为此分段创建的其他IPv4报头。

第四个分段的偏移量为 555 (555 x 8 = 4440)，这意味着此分段的数据部分从原始 IPv4 数据报的第 4440 字节后开始。

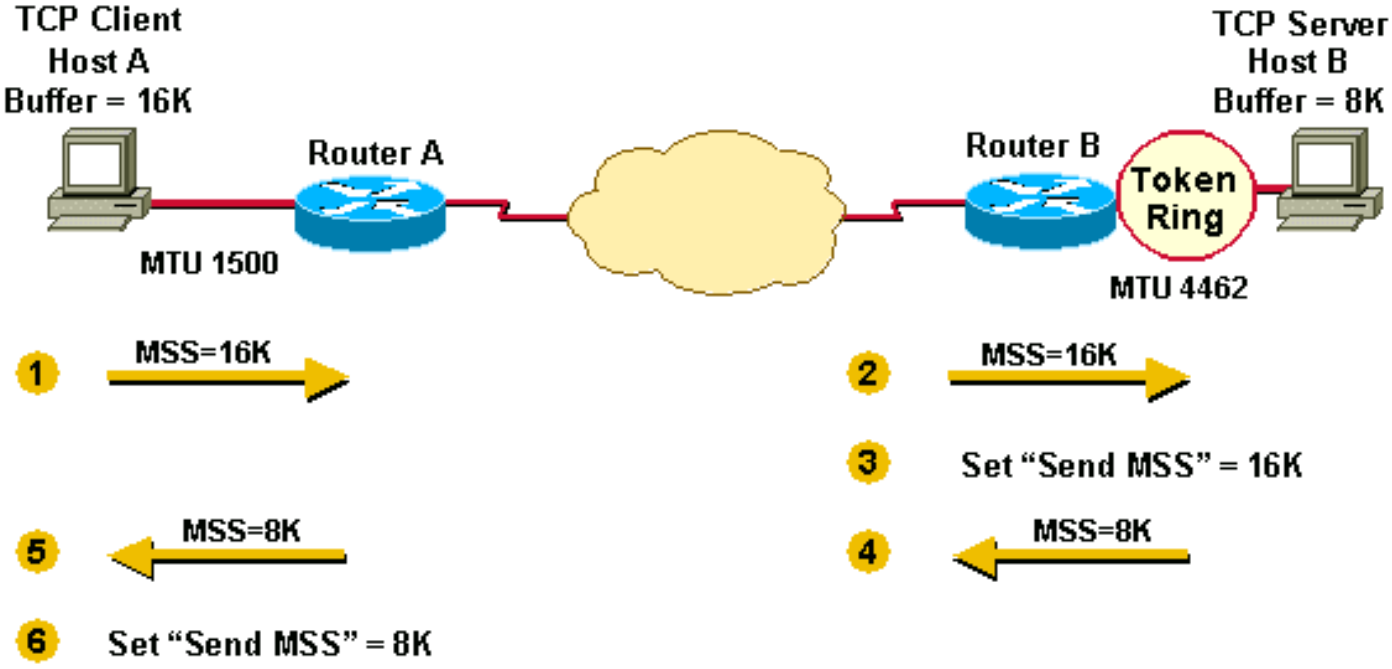
此分段的长度为700个字节；其中包括为此分段创建的其他IPv4报头。

只有当收到最后一个分段时，才可以确定原始 IPv4 数据报的大小。

最后一个分段中的分段偏移量 (555) 表示数据偏移量为原始 IPv4 数据报的 4440 字节。

最后一个分段的数据字节总和(680 = 700 – 20)可产生5120个字节，这是原始IPv4数据报的数据部分。

IPv4报头增加20个字节等于原始IPv4数据报的大小(4440 + 680 + 20 = 5140)，如图所示。



IPv4 分段相关问题

IPv4分段会小幅增加CPU和内存开销，从而对IPv4数据报进行分段。对于发送方以及发送方和接收方之间路径中的路由器，情况也是如此。

分段的创建包括创建分段报头并将原始数据报复制到分段中。

由于创建分段所需的信息立即可用，因此可以高效地完成此操作。

重组分段时，分段会导致接收者产生更大开销，原因是接收者必须为到达的分段分配内存，然后在收到所有分段后将它们重新组合为一个数据报。

在主机上进行重组不会带来任何问题，因为主机拥有完成此任务所需的时间和内存资源。

但是，如果路由器的主要任务是尽快转发数据包，则重组效率会非常低。

路由器不是为了将数据包保存任意时长而设计的。

执行重组的路由器会选择可用的最大缓冲区(18K)，因为在收到最后一个分段之前，它无法确定原始IPv4数据包的大小。

另一个分段问题涉及到已丢弃分段的处理方式。

如果IPv4数据报的一个分段被丢弃，则整个原始IPv4数据报必须存在，并且它也是分段的。

网络文件系统(NFS)中可看到此情况。NFS的读写块大小为8192。

因此，NFS IPv4/UDP数据报约为8500字节（包括NFS、UDP和IPv4报头）。

连接到以太网(MTU 1500)的发送站必须将8500字节的数据报分段为六(6)段；五(5)个1500字节的分段和一(1)个1100字节的分段。

如果由于链路拥塞而丢弃了六个分段中的任何一个，则必须重新传输完整的原始数据报。这会导致创建另外六个分段。

如果此链路丢弃六分之一的数据包，则通过此链路传输任何NFS数据的可能性较低，因为每个NFS 8500字节的原始IPv4数据报至少会丢弃一个IPv4分段。

防火墙根据从第4层(L4)到第7层(L7)的信息过滤或处理数据包，无法正确处理IPv4分段。

如果IPv4分段顺序混乱，防火墙会阻止非初始分段，因为它们不携带与数据包过滤器匹配的信息。

这意味着接收主机无法重组原始IPv4数据报。

如果防火墙配置为允许信息不足的非初始分段与过滤器正确匹配，则可能通过防火墙进行非初始分段攻击。

网络设备（如内容交换引擎）根据L4到L7信息转发数据包，如果数据包跨越多个分段，设备将难以实施其策略。

避免IPv4分段：TCP MSS的工作原理

传输控制协议(TCP)最大分段大小(MSS)定义主机在单个TCP/IPv4数据报中接受的最大数据量。

此TCP/IPv4数据报可能在IPv4层分段。MSS 值仅作为 TCP SYN 数据段中的一个 TCP 报头选项发送。

TCP 连接的每一端都会向另一端报告其 MSS 值。主机之间不协商MSS值。

发送主机需要将单个 TCP 数据段中的数据大小限制为小于或等于接收主机报告的 MSS 的值。

最初，MSS 决定着接收站上分配用于存储单个 IPv4 数据报所含 TCP 数据的缓冲区的大小（大于等于 65496 字节）。

MSS是TCP接收方将要接受的最大数据段。此TCP数据段可能高达64K，在IPv4层进行分段以便传输到接收主机。

在将完整的 TCP 分段传送至 TCP 层之前，接收主机会重组 IPv4 数据报。

如何设置MSS值并用于限制TCP数据段和IPv4数据报大小。

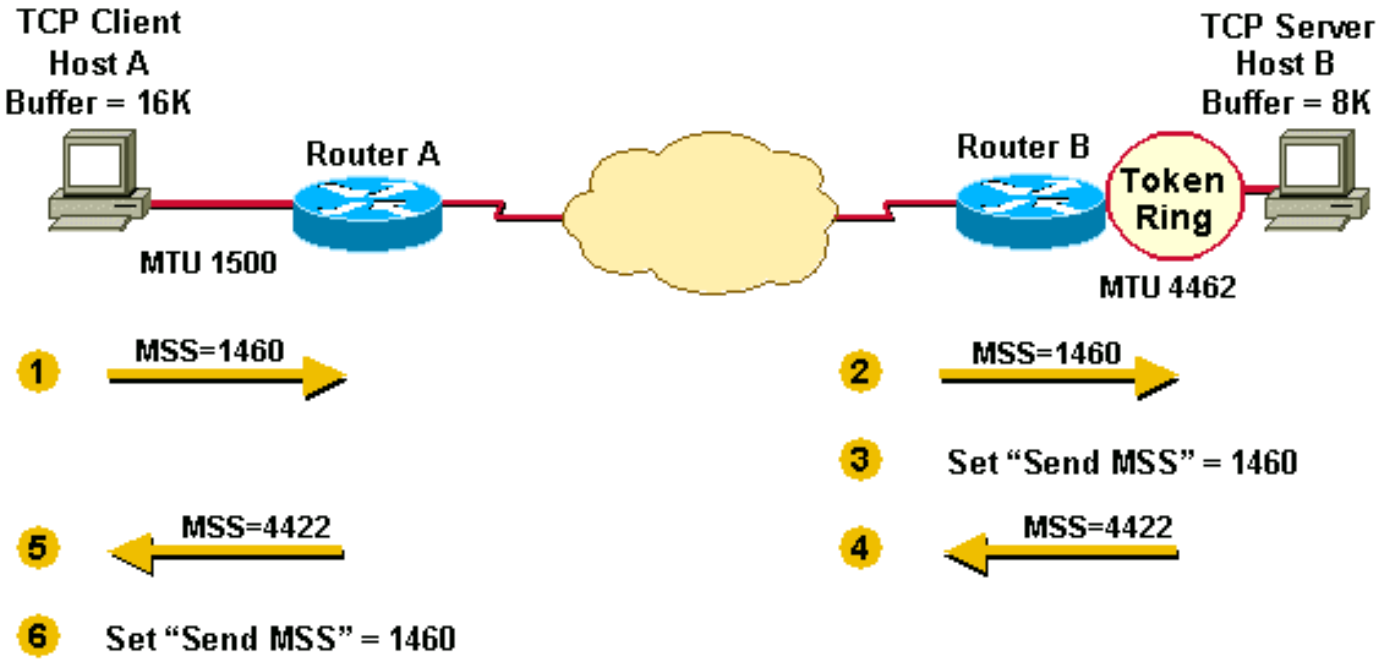
示例1说明了MSS的首次实施方式。

主机 A 的缓冲区为 16K，主机 B 的缓冲区为 8K。这些主机发送和接收其各自的 MSS 值，并调整其发送 MSS 以便彼此发送数据。

主机A和主机B必须对大于接口MTU但小于发送MSS的IPv4数据报进行分段，因为TCP堆栈会将16K或8K字节的数据沿堆栈向下传递到IPv4。

对于主机B，对数据包进行分段以进入令牌环LAN，然后再次进入以太网LAN。

示例 1



1. 主机 A 将其 MSS 值 16K 发送到主机 B。
2. 主机 B 收到来自主机 A 的 MSS 值 16K。
3. 主机 B 将其发送 MSS 值设置为 16K。
4. 主机 B 将其 MSS 值 8K 发送到主机 A。
5. 主机 A 收到来自主机 B 的 MSS 值 8K。
6. 主机 A 将其发送 MSS 值设置为 8K。

为帮助避免TCP连接终端的IPv4分段，MSS值的选择已更改为最小缓冲区大小和传出接口的MTU(-40)。

MSS编号比MTU编号小40字节，因为MSS（TCP数据大小）不包括20字节的IPv4报头和20字节的TCP报头。

MSS基于默认报头大小；发送方堆栈必须减去IPv4报头的相应值，TCP报头取决于使用的TCP或IPv4选项。

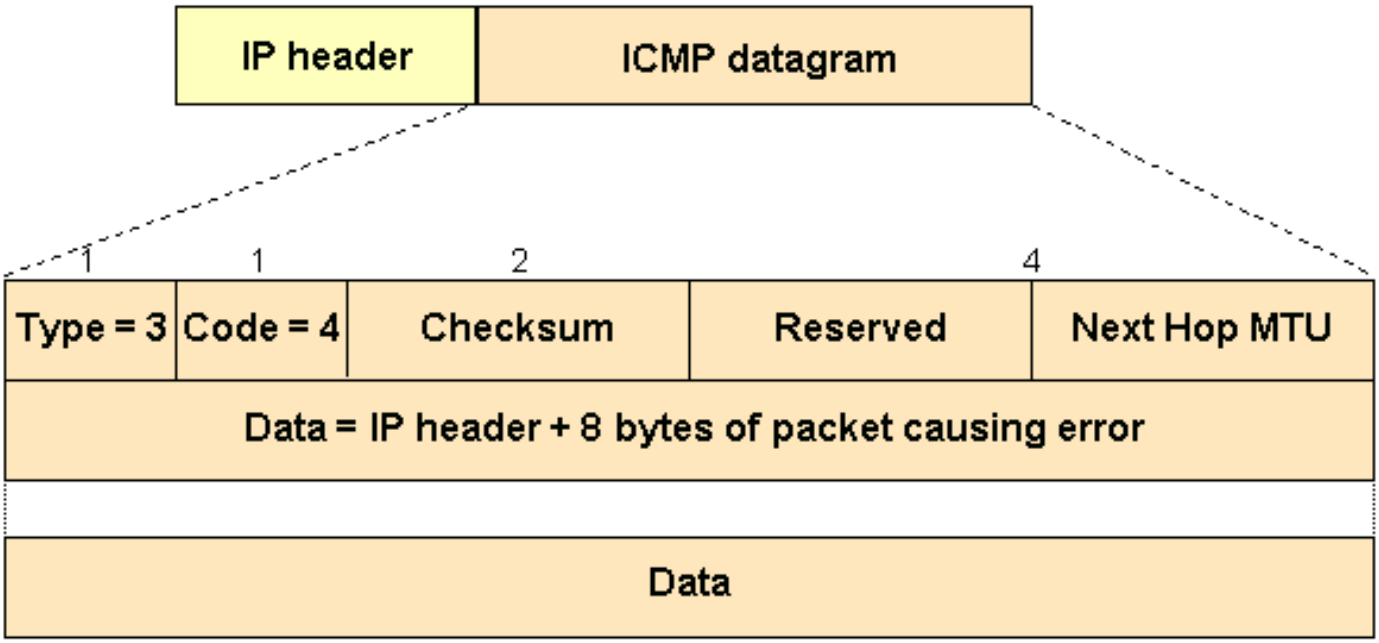
MSS当前的工作方式是，每台主机首先将其传出接口MTU与自己的缓冲区进行比较，并选择最低值作为MSS发送。

然后，主机将收到的MSS大小与其自己的接口MTU进行比较，并再次选择两个值中较小的一个。

示例2说明了发送方为避免本地和远程线路上的分段而执行的这一附加步骤。

在主机相互发送其MSS值之前，每台主机都会考虑传出接口的MTU。这有助于避免分段。

示例 2



1. 主机 A 比较其 MSS 缓冲区 (16K) 和其 MTU (1500 - 40 = 1460)，并使用较小的值作为 MSS (1460)，以便发送到主机 B。
2. 主机B收到来自主机A的发送MSS (1460)，并将其与出站接口MTU - 40 (4422)的值进行比较。
3. 主机 B 将较低的值 (1460) 设置为 MSS，以便向主机 A 发送 IPv4 数据报。

4. 主机 B 比较其 MSS 缓冲区 (8K) 和其 MTU (4462 - 40 = 4422)，并使用 4422 作为 MSS，以便发送到主机 A。
5. 主机A从主机B接收发送MSS (4422)，并将其与出站接口MTU -40 (1460)的值进行比较。
6. 主机 A 将较低的值 (1460) 设置为 MSS，以便向主机 B 发送 IPv4 数据报。

1460 便是两台主机为彼此选择的发送 MSS 的值。通常，TCP连接的每一端的发送MSS值相同。

在示例2中，TCP连接的终端不会发生分段，因为主机会考虑两个传出接口MTU。

如果路由器A和路由器B之间链路的MTU低于任一主机的出站接口，则数据包仍然会在网络中分段。

什么是PMTUD

TCP MSS解决TCP连接的两个端点上的分段，但不处理这两个端点之间中间有较小的MTU链路的情况。

为了避免在终端之间的路径上出现分段，开发了 PMTUD。它用于动态确定从数据包源到其目的地的路径中的最低MTU。

注意：PMTUD仅受TCP和UDP支持。其他协议不支持 PMTUD。如果在主机上启用 PMTUD，则来自该主机的所有TCP和UDP数据包都会设置DF位。

如果主机发送设置了 DF 位的完整 MSS 数据包，则在收到数据包需要分段的信息时，PMTUD 将减少连接的发送 MSS 值。

主机记录目标的MTU值，因为它会在路由表中创建具有此MTU值的主机(/32)条目。

如果路由器尝试将IPv4数据报（设置了DF位）转发到MTU低于数据包大小的链路上，路由器会丢弃该数据包，并将互联网控制消息协议(ICMP)“目标不可达”(Destination Unreachable)消息返回到IPv4数据报源，消息中带有指示“需要分段并设置DF”（类型3，代码4）的代码。

当源站点收到ICMP消息时，它降低发送MSS，当TCP重新传输数据段时，它使用较小的数据段大小。

以下示例说明了在启用 **debug ip icmp** 命令后，在路由器上看到的ICMP“fragmentation needed and DF set”消息：

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

下图显示了“需要分段和设置 DF”、“无法到达目标”消息的 ICMP 报信头的格式。

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

根据 RFC 1191，如果某路由器返回 ICMP 消息“需要分段但 DF 已设置”，则该路由器必须在根据 ICMP 规范 RFC 792 标记为“未使用”的低阶 16 位 ICMP 附加报头字段中，使用下一跳网络的 MTU。

RFC1191 的早期实现未提供下一跳 MTU 信息。即使提供了此信息，某些主机也会忽略它。

在本例中，RFC 1191还包含一个表，其中列出了PMTUD期间MTU降低的建议值。

主机使用它来更快地为发送MSS得出一个合理的值，如本例所示。

由于发送方和接收方之间的路径可以动态更改，因此会对所有数据包持续执行PMTUD。

发送方每次收到“Cannot Fragment”ICMP消息时，都会更新路由信息（存储PMTUD的位置）。

执行 PMTUD 时，可能会发生以下两种情况：

1. 数据包可以一直传输到接收方，而不会被分段。

注意：为了使路由器保护CPU免受DoS攻击，它会将发送的ICMP不可达消息数限制为每秒两条。因此，在这种情况下，如果您的网络方案预期路由器每秒钟将需要响应超过两个ICMP消息（类型= 3，代码= 4）（可以是不同主机），请使用 **no ip icmp rate-limit unreachable [df] interface**命令禁用ICMP消息限制。

---

2. 发送方从通往接收方的路径上的每一跳收到ICMP“无法分段”消息。

PMTUD 在 TCP 流量的两个方向上独立执行。

在某些情况下，流量一个方向的PMTUD会触发其中一个终端站降低发送MSS，而另一个终端站会保留原始发送MSS，因为它从未发送足够大的IPv4数据报来触发PMTUD。

例如，示例3中描述的HTTP连接。TCP 客户端发送小数据包，服务器发送大数据包。

在这种情况下，只有来自服务器的大数据包（大于576字节）触发PMTUD。

来自客户端的数据包较小（小于576字节），并且不触发PMTUD，因为它们不需要分段即可通过576 MTU链路。

示例 3

示例4显示了一个非对称路由示例，其中一条路径的最小MTU小于另一条路径。

采用不同的路径在两个终端之间发送和接收数据时，将会出现非对称路由。

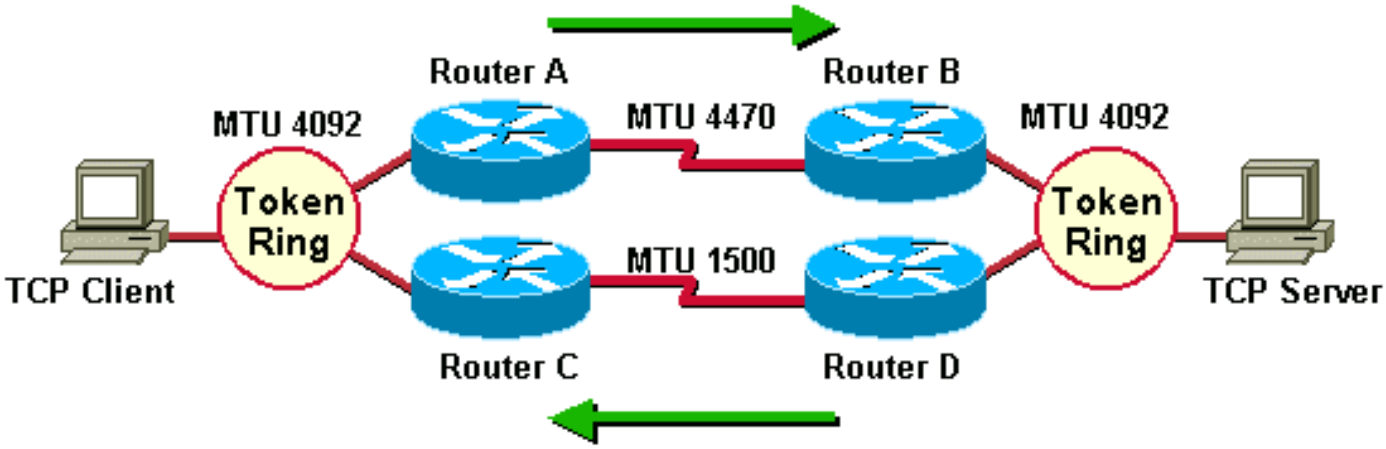
在本示例中，PMTUD仅在TCP流的一个方向上触发发送MSS的下降。

从 TCP 客户端到服务器的流量会经过路由器 A 和路由器 B，而从服务器到客户端的返回流量会经过路由器 D 和路由器 C。

当TCP服务器向客户端发送数据包时，PMTUD会触发服务器降低发送MSS，因为路由器D必须对4092字节的数据包进行分段，然后才能将其发送到路由器C。

相反，客户端永远不会收到代码为“fragmentation needed and DF set”的ICMP“Destination Unreachable”消息，因为路由器A在通过路由器B向服务器发送数据包时不必对数据包进行分段。

示例 4



注意：可以使用**ip tcp path-mtu-discovery**命令对路由器启动的TCP连接（例如BGP和Telnet）启用TCP MTU路径发现。

PMTUD 问题

这些东西可以破坏PMTUD。

- 路由器丢弃数据包，且不发送ICMP消息。（不常见）
- 路由器生成并发送ICMP消息，但此路由器和发送器之间的路由器或防火墙阻止该ICMP消息。（常见）
- 路由器生成并发送ICMP消息，但发送方忽略该消息。（不常见）

这里的三项要点中的第一项、最后一项通常是由错误导致的，但中间一项描述的是一个常见问题。

实施ICMP数据包过滤器的那些设备倾向于阻止所有ICMP消息类型，而不是只阻止某些ICMP消息类型。

数据包过滤器可以阻止所有ICMP消息类型，但“无法到达”或“超时”消息除外。

PMTUD的成功或失败取决于到达TCP/IPv4数据包发送方的ICMP不可达消息。

ICMP 超时消息对于其他的 IPv4 问题非常重要。

下面显示了在路由器上实现的此类数据包过滤器示例。

```
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
```

还有其他技术可用于缓解ICMP完全阻塞的问题。

- 清除路由器上的DF位并允许分段。(不过，这不是个好主意。有关详细信息，请参阅 IP 分段问题) 。

- 使用接口命令 **ip tcp adjust-mss <500-1460>**调整TCP MSS选项值MSS。

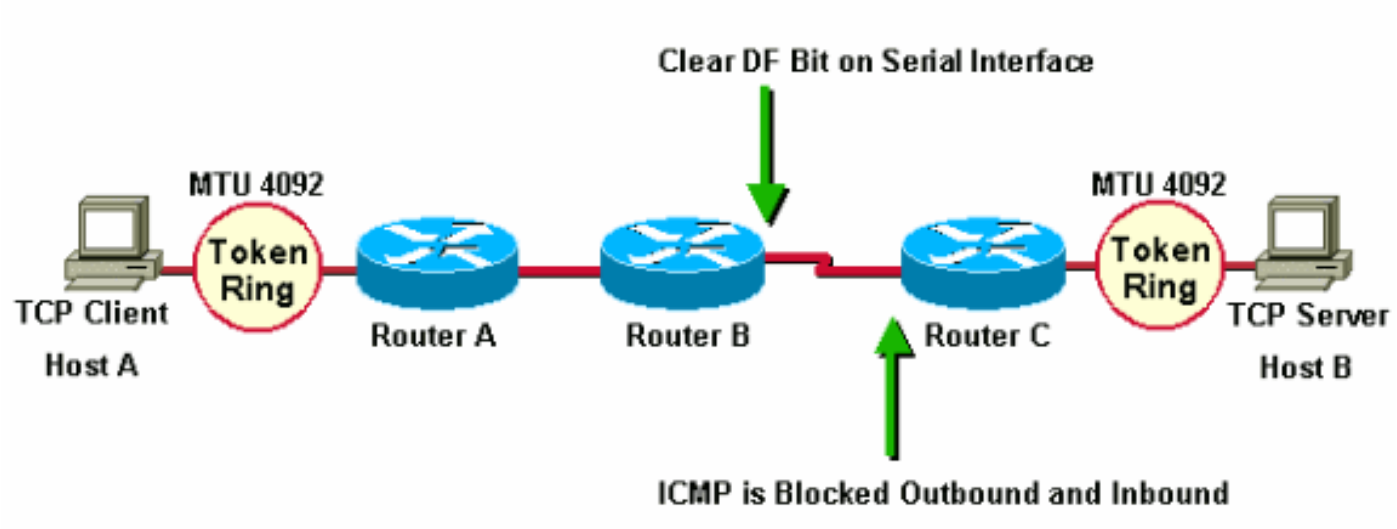
在下一个示例中，路由器A和路由器B位于同一个管理域中。路由器 C 不可访问并且会阻止 ICMP，因此，PMTUD 将中断。

在这种情况下，解决方案是在路由器 B 的两个方向上清除 DF 位，从而允许分段。这可以通过策略路由来完成。

Cisco IOS® 软件版本 12.1(6) 和更高版本中提供了用于清除 DF 位的语法。

```
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
```



另一个选择是更改流经路由器的 SYN 数据包的 TCP MSS 选项值（在思科 IOS® 12.2(4)T 及更高版本中可用）。

这会减小TCP SYN数据包中的MSS选项的值，使其小于 **ip tcp adjust-mss**命令中的值(1460)。

结果是TCP发送方发送的数据段不大于此值。

IPv4数据包的大小比MSS值（1460字节）大40字节(1500)，以说明TCP报头（20字节）和IPv4报头（20字节）。

您可以使用 **ip tcp adjust-mss** 命令调整TCP SYN数据包的MSS。此语法将TCP数据段的MSS值减少到1460。

此命令将影响 serial0 接口上的入站和出站流量。

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

随着 IPv4 隧道部署越来越广泛，IPv4 分段问题也变得越来越普遍。

隧道会导致更多分段，因为隧道封装会增加数据包大小的“开销”。

例如，增加通用路由器封装(GRE)会向数据包中添加24个字节，增加之后，数据包需要分段，因为它大于出站MTU。

需要 PMTUD 的常见网络拓扑

在网络环境中，如果中间链路的 MTU 小于终端链路的 MTU，则需要 PMTUD。存在这些较小的 MTU 链路的一些常见原因是：

- 与令牌环（或 FDDI）相连的终端主机之间存在以太网连接。两端的令牌环（或 FDDI）MTU 大于中间的以太网 MTU。
-

PPPoE（通常与 ADSL 配合使用）的报头需要 8 个字节。这使得以太网的有效 MTU 减小至 1492 (1500 - 8)。

GRE、IPv4sec和L2TP等隧道协议还需要用于各自报头和报尾的空间。这也会降低出站接口的有效MTU。

隧道

隧道是 Cisco 路由器上的一个逻辑接口，它提供了一种将乘客数据包封装在传输协议内的方法。

这种架构旨在为点对点封装方案的实施提供服务。隧道接口有以下三个主要组件：

- 乘客协议（AppleTalk、Banyan VINES、CLNS、DECnet、IPv4 或 IPX）
- 运载载波协议 - 以下封装协议之一：
  - GRE -思科多协议运营商协议。有关详细信息，请参阅 RFC 2784 和 RFC 1701。
  - IPv4-in-IPv4 隧道；有关详细信息，请参阅 RFC 2003。
- 传输协议 - 用于传输经过封装的协议。

本部分中的数据包展示了以 GRE 为封装协议、以 IPv4 为传输协议的 IPv4 隧道概念。

乘客协议还是 IPv4。在这种情况下，IPv4 同时是传输协议和乘客协议。

正常数据包

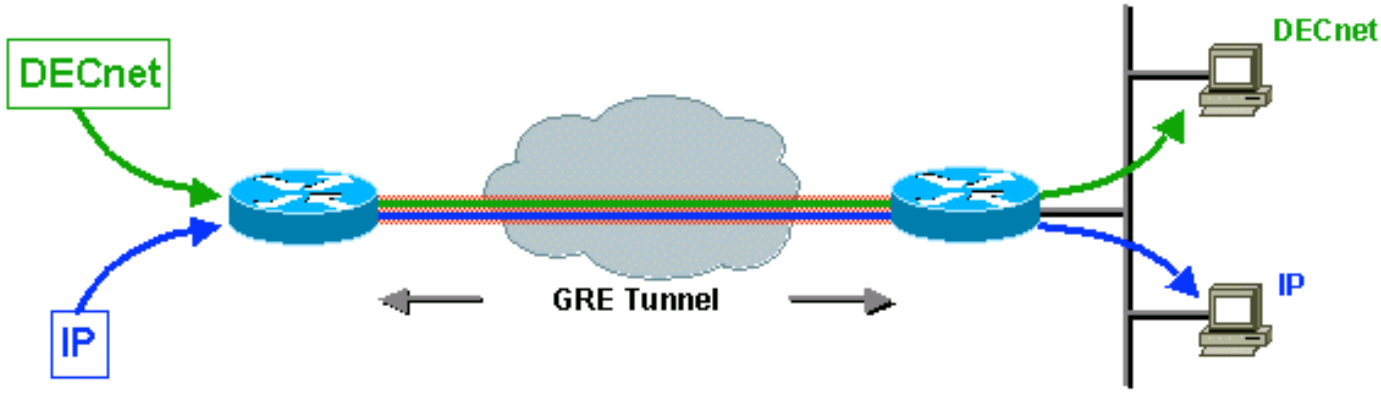
隧道数据包

IPv4	GRE	IPv4	TCP	Telnet

- IPv4 为传输协议。
- GRE 是封装协议。
- IPv4 为乘客协议。

在下一封装示例中，IPv4 和 DECnet 为乘客协议，GRE 为承载协议。

这说明运营商协议封装多个乘客协议的可能性，如图所示。



网络管理员考虑在存在两个不连续非IPv4网络（由IPv4主干分隔）的情况下建立隧道。

如果不连续网络运行DECnet，管理员可以选择通过配置主干网中的DECnet将它们连接在一起（或不连接）。

管理员不希望允许DECnet路由消耗主干带宽，因为这可能会干扰IPv4网络的性能。

一种可行的备选方案就是在 IPv4 主干网上通过隧道传输 DECnet。隧道解决方案将DECnet数据包封装在IPv4中，并通过主干网将它们发送到隧道终端，在那里删除封装，并且DECnet数据包通过DECnet路由到其目的地。

将流量封装在另一个协议中有一些优点：

- 终端使用私有地址(RFC 1918)，并且主干不支持路由这些地址。

- 允许在 WAN 或 Internet 中建立虚拟专用网络 (VPN)。
- 通过一个单一协议的骨干网将不连续多协议网络连接在一起。
- 通过骨干网或 Internet 对流量进行加密。

之后，IPv4用作乘客协议，IPv4用作传输协议。

有关隧道接口的注意事项

以下是建立隧道时的注意事项。

- Cisco IOS®版本11.1中引入了GRE隧道的快速交换，版本12.0中引入了CEF交换。
- 多点 GRE 隧道的 CEF 交换功能是在版本 12.2(8)T 中引入的。
- 在早期版本的Cisco IOS®中，如果仅支持进程交换，则隧道端点上的封装和解封操作会非常慢。
- 对数据包建立隧道时，存在安全和拓扑问题。隧道可以绕过访问控制列表 (ACL) 和防火墙。
- 如果通过防火墙建立隧道，则绕过通过隧道传输的乘客协议。因此，为了对乘客协议执行策略，建议在隧道终端部署防火墙功能。
-

由于延迟增加，隧道会对具有有限计时器（例如DECnet）的传输协议造成问题。

- 在具有不同速度链路的环境中（例如快速FDDI环和通过9600-bps慢速电话线）使用隧道会导致数据包重新排序问题。一些乘客协议在混合媒体网络中的运行效率非常低下。

- 点对点隧道会消耗物理链路上的带宽。在多个点对点隧道上，每个隧道接口都有一个带宽，并且运行隧道的物理接口也有一个带宽。例如，如果在10 Mb链路上运行了100个隧道，请将隧道带宽设置为100 Kb。隧道的默认带宽是 9 Kb。

- 路由协议更偏爱实际链路上的隧道，因为隧道看似是一跳链路，具有最低开销路径，但实际上它涉及的跳数更多，因此成本也高于另一条路径。通过正确配置路由协议可以缓解此问题。请考虑在隧道接口上运行与物理接口上运行的路由协议不同的路由协议。

- 通过配置通往隧道目的地的适当静态路由，可避免递归路由问题。递归路由指通往隧道目的地的最佳路径就是通过隧道本身。这种情况会导致隧道接口上下跳动。出现递归路由问题时，会出现此错误。

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

在隧道终端作为 PMTUD 参与者的路由器

当路由器作为隧道端点时，它将扮演两种不同的 PMTUD 角色。

- 路由器的第一个作用就是充当主机数据包的转发端。对于 PMTUD 处理，路由器需要检查原始数据包的 DF 位和数据包大小，并在必要时采取相应操作。

- 路由器将原始 IPv4 数据包封装到隧道数据包后，开始发挥第二个作用。在此阶段，路由器的作用对于 PMTUD 和隧道 IPv4 数据包来说更像主机。

当路由器充当第一个角色（转发主机IPv4数据包的路由器）时，该角色将在路由器将主机IPv4数据包封装到隧道数据包中之前发挥作用。

如果路由器作为主机数据包的转发器参与，它将完成以下操作：

- 检查是否已设置 DF 位。
- 检查隧道可容纳的数据包大小。
- 分段（如果数据包太大，但未设置DF位），封装分段并发送；或
- 丢弃数据包（如果数据包太大，且设置了 DF 位），并向发送者发送 ICMP 消息。
- 封装（如果数据包并不太大）并发送。

一般情况下，可以选择先封装再分段（发送两个封装分段）或者先分段再封装（发送两个已封装分段）。

本部分详细介绍两个示例，它们显示了PMTUD与通过示例网络的数据包的交互。

第一个示例展示的是在（隧道源中的）路由器充当转发路由器的角色时，数据包发生的情况。

要处理PMTUD，路由器需要检查原始数据包的DF位和数据包大小，并采取适当操作。

本示例对隧道使用 GRE 封装。GRE在封装之前执行分段。

后面的示例显示在封装后进行分段的方案。

在示例 1 中，DF 位未设置 (DF = 0)，GRE 隧道 IPv4 MTU 为 1476 (1500-24) 字节。

示例 1

1. 转发路由器（在隧道源上）从发送主机收到一个1500字节且清除DF位(DF = 0)的数据报。

此数据报由一个 20 字节的 IP 报头和一个 1480 字节的 TCP 负载组成。

IPv4	1480 字节 TCP + 数据

2. 由于在添加GRE开销（24字节）后，数据包对于IPv4 MTU而言太大，因此转发路由器将数据报分成1476（20字节IPv4报头+ 1456字节IPv4负载）和44字节（20字节IPv4报头+ 24字节IPv4负载）的两个分段

添加GRE封装后，数据包不会大于传出物理接口MTU。

IP_0	1456 字节 TCP + 数据
IP_1	24 字节数据

3. 转发路由器将GRE封装（包括4字节GRE报头和20字节IPv4报头）添加到原始IPv4数据报的每个分段。

这两个 IPv4 数据报的长度现在分别为 1500 字节和 68 字节，它们被视为单独的 IPv4 数据报，而不是分段。

IPv4	GRE	IP_0	1456 字节 TCP + 数据
IPv4	GRE	IP_1	24 字节数据

4. 隧道目标路由器从原始数据报的每个分段中删除GRE封装，从而留下两个长度为1476和24字节的IPv4分段。

路由器向接收主机单独转发这些 IPv4 数据报分段。

IP_0	1456 字节 TCP + 数据
IP_1	24 字节数据

5. 接收主机将这两个分段重组为原始数据报。

IPv4	1480 字节 TCP + 数据

示例2描述了转发路由器在网络拓扑环境中的角色。

路由器的作用与转发路由器的作用相同，但这次设置了DF位(DF = 1)。

示例 2

1. 位于隧道源的转发路由器从发送主机收到一个1500字节的数据报，其中DF = 1。

IPv4	1480 字节 TCP + 数据

2. 由于设置了DF位，并且数据报大小（1500字节）大于GRE隧道IPv4 MTU (1476)，因此路由器会丢弃数据报并向数据报的源发送“需要进行ICMP分段但DF位已设置”消息。

ICMP消息通知发送方MTU为1476。

IPv4	ICMP MTU 1476

3. 发送主机收到ICMP消息，在重新发送原始数据时，它使用1476字节的IPv4数据报。

IPv4	1456 字节 TCP + 数据

4. 此IPv4数据报长度（1476字节）的值现在等于GRE隧道IPv4 MTU，因此路由器会将GRE封装添加到IPv4数据报。

IPv4	GRE	IPv4	1456 字节 TCP + 数据

5. 接收路由器（位于隧道目标位置）删除IPv4数据报的GRE封装并将其发送到接收主机。

IPv4	1456 字节 TCP + 数据

当路由器充当与PMTUD和隧道IPv4数据包相关的第二个发送主机时，会发生这种情况。

路由器将原始IPv4数据包封装到隧道数据包中后，此角色开始发挥作用。

注意：默认情况下，路由器不会对其生成的GRE隧道数据包执行PMTUD。可以使用**tunnel path-mtu-discovery**命令对GRE-IPv4隧道数据包启用PMTUD。

示例 3 展示的是当主机发送的 IPv4 数据报小到足以适应 GRE 隧道接口上的 IPv4 MTU 时发生的情况。

在此情况下，可以设置或清除 DF 位（1 或 0）。

GRE隧道接口未配置 **tunnel path-mtu-discovery** 命令，因此路由器不会死机，也不会在GRE-IPv4数据包上执行PMTUD。

示例 3

1. 位于隧道源的转发路由器从发送主机接收一个1476字节的数据报。

IPv4	1456 字节 TCP + 数据

2. 此路由器将1476字节的IPv4数据报封装在GRE内，以获取1500字节的GRE IPv4数据报。

GRE IPv4报头中的DF位被清除(DF = 0)。然后，此路由器将此数据包转发到隧道目标。

IPv4	GRE	IPv4	1456 字节 TCP + 数据

3. 假设隧道源和目标之间有一个路由器，且链路MTU为1400。

由于DF位是空的(DF = 0)，因此此路由器对隧道数据包进行分段。

请记住，此示例对最外层的IPv4进行分段，因此GRE、内部IPv4和TCP报头仅显示在第一个分段中。

IP_0	GRE	IP	1352 字节 TCP + 数据
IP_1	104 字节数据		

4. 隧道目标路由器必须重组GRE隧道数据包。

IP	GRE	IP	1456 字节 TCP + 数据

5. 重组GRE隧道数据包后，路由器会删除GRE IPv4报头，并在途中发送原始IPv4数据报。

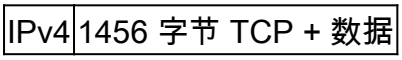
IPv4	1456 字节 TCP + 数据

示例4显示当路由器充当发送主机的角色时，对于PMTUD和隧道IPv4数据包会发生什么情况。

此时，在原始IPv4报头中设置了DF位(DF = 1)，并且配置了 **tunnel path-mtu-discovery** 命令，以便将DF位从内部IPv4报头复制到外部(GRE + IPv4)报头。

示例 4

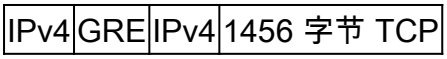
1. 位于隧道源的转发路由器从发送主机收到一个1476字节的数据报，其中DF = 1。



2. 此路由器将1476字节的IPv4数据报封装在GRE内，以获取1500字节的GRE IPv4数据报。

此GRE IPv4报头设置了DF位(DF = 1)，因为原始IPv4数据报设置了DF位。

然后，此路由器将此数据包转发到隧道目标。



3. 同样，假设隧道源和目标之间有一个路由器，且链路MTU为1400。

由于设置了DF位(DF=1)，此路由器不会对隧道数据包进行分段。

此路由器必须丢弃数据包并向隧道源路由器发送ICMP错误消息，因为它是数据包上的源IPv4地址。

4. 位于隧道源的转发路由器收到此“ICMP”错误消息，并将GRE隧道IPv4 MTU降至1376 (1400 – 24)。

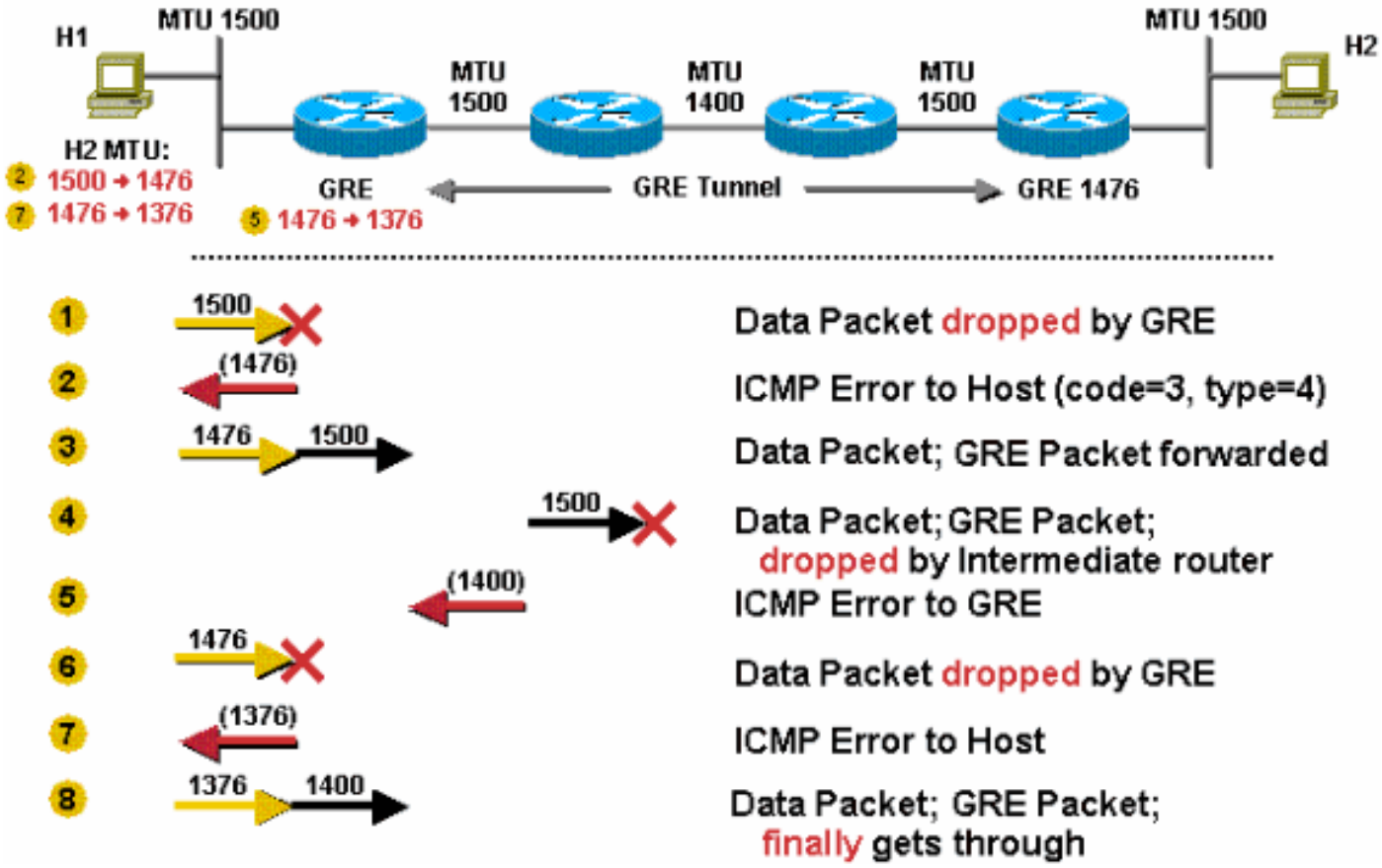
下一次发送主机重新传输1476字节的IPv4数据包中的数据时，该数据包可能太大，然后路由器会向发送方发送“ICMP”错误消息，MTU值为1376。

当发送主机重新传输数据时，它会以1376字节的IPv4数据包的形式发送数据，此数据包会通过GRE隧道发送到接收主机。

示例 5

本示例说明GRE分段。先对GRE进行分段，再对数据包执行PMTUD，当IPv4数据包由GRE封装时，不会复制DF位。

未设置DF位。GRE 隧道接口的 IPv4 MTU 默认比物理接口的 IPv4 MTU 少 24 字节，因此 GRE 接口的 IPv4 MTU 为 1476 字节，如下图所示。



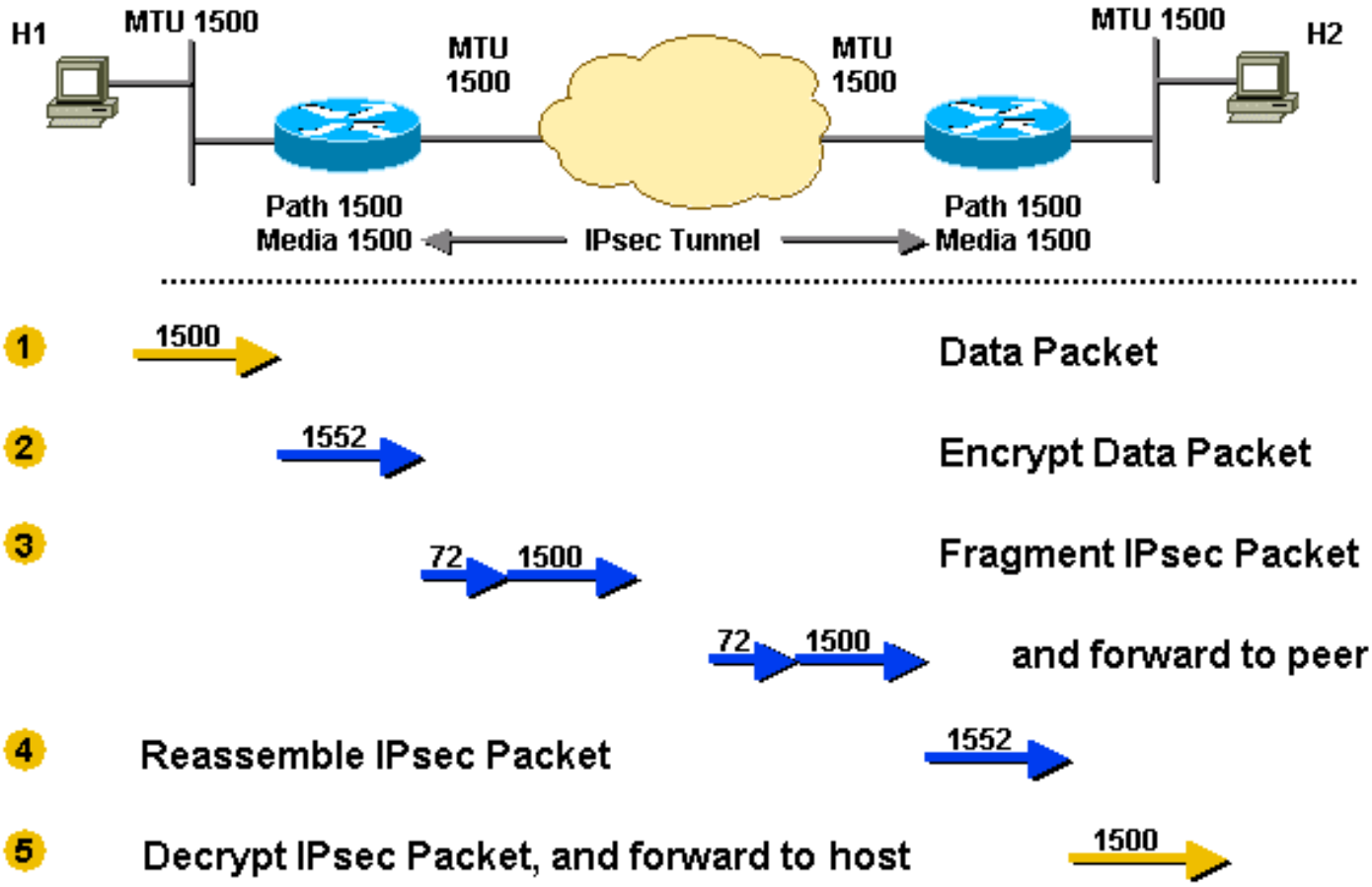
- 发送端发送一个 1500 字节的数据包（20 字节 IPv4 报头 + 1480 字节 TCP 负载）。
- 由于GRE隧道的MTU是1476，因此1500字节的数据包将被分解为两个IPv4分段（1476和44字节），每个分段预计会额外增加24个字节的GRE报头。
- 每个 IPv4 分段增加 24 字节的 GRE 报头。现在，两个分段分别为 1500 字节 (1476 + 24) 和 68 字节 (44 + 24)。
- 包含两个 IPv4 分段的 GRE + IPv4 数据包被转发到 GRE 隧道对等路由器。
- GRE 隧道对等路由器将删除两个数据包中的 GRE 报头。
- 此路由器将两个数据包转发到目标主机。
- 目标主机将 IPv4 分段重组为原始 IPv4 数据报。

示例 6

此示例与示例5类似，但这次设置了DF位。使用 **tunnel path-mtu-discovery**命令将路由器配置为对GRE + IPv4隧道数据包执行PMTUD，并且DF位从原始IPv4报头被复制到GRE IPv4报头。

如果路由器收到面向 GRE + IPv4 数据包的 ICMP 错误，则会降低 GRE 隧道接口的 IPv4 MTU 值。

默认情况下，GRE隧道IPv4 MTU设置为比物理接口MTU小24字节，因此，此处的GRE IPv4 MTU为1476。GRE隧道路径中有一个1400 MTU链路，如图所示。



- 路由器收到 1500 字节的数据包（20 字节 IPv4 报头 + 1480 字节 TCP 负载），然后丢弃该数据包。路由器丢弃数据包是因为该数据包大于 GRE 隧道接口上的 IPv4 MTU (1476)。

- 路由器向发送者发送一条 ICMP 错误，通知发送者下一跳 MTU 为 1476。主机在其路由表中记录此信息，通常作为目的主机路由。

- 当重新发送数据时，发送主机采用 1476 字节作为数据包大小。GRE 路由器添加 24 字节的 GRE 封装，然后发送一个 1500 字节的数据包。

- 该 1500 字节的数据包无法通过 1400 字节的链路，因此中间路由器将丢弃该数据包。

- 中间路由器将向 GRE 路由器发送一个含有下一跳 MTU 为 1400 的 ICMP（类型 = 3，代码 = 4）。GRE 路由器将其降低至 1376 (1400 - 24) 字节，并在 GRE 接口上设置内部 IPv4 MTU 值。仅当使用 **debug tunnel** 命令时，才会显示此更改；它不会在 **show ip interface tunnel**<#> 命令的输出中显示。

- 下次主机重新发送1476字节的数据包时，GRE路由器会丢弃该数据包，因为它大于GRE隧道接口上当前的IPv4 MTU (1376)。

- GRE路由器向下一跳MTU为1376的发送方发送另一个ICMP（类型= 3，代码= 4），主机使用新值更新其当前信息。

- 主机再次重新发送数据，但现在GRE在一个较小的1376字节数据包中添加24字节的封装，然后继续转发。此时，数据包到达GRE隧道对等体，数据包在该隧道中解封并发送到目的主机。

注意：如果在本案例中，转发路由器上没有配置**tunnel path-mtu-discovery**命令，并且DF位设置在GRE隧道转发的数据包中

，那么主机1向主机2仍成功发送TCP/IPv4数据包，但它们在1400 MTU链路的中途分段。此外，GRE隧道必须在解封装和转发数据包前，对数据包进行重组。

纯 IPsec 隧道模式

IPv4 安全 (IPv4sec) 协议是一种基于标准的方法，用于确保 IPv4 网络上传输的信息保密、完整且真实。

IPv4sec 提供 IPv4 网络层加密。IPv4sec 通过添加至少一个 IPv4 报头（隧道模式）而加长 IPv4 数据包。

添加的报头长度因IPv4sec配置模式而异，但每个数据包不超过–58个字节(封装安全负载(ESP)和ESP身份验证(ESPauth))。

IPv4sec 有两种模式：隧道模式和传送模式。

- 隧道模式是默认模式。在隧道模式下，整个原始 IPv4 数据包都受到保护（加密、身份验证或两者同时执行），并由 IPv4sec 报头和报尾进行封装。然后，新的 IPv4 报信头附加到数据包前面，用于将 IPv4sec 终端（对等体）指定为源端和目标端。隧道模式可用于任何单播 IPv4 流量，但如果将 IPv4sec 用于保护来自 IPv4sec 对等体背后的主机流量，则必须使用隧道模式。例如，隧道模式可与虚拟专用网络 (VPN) 结合使用，其中通过一组 IPv4sec 对等体，一个受保护网络上的主机可向另一个受保护网络上的主机发送数据包。在 VPN 网络中，IPv4sec 隧道通过加密 IPv4sec 对等路由器间的流量来保护主机之间的 IPv4 流量。

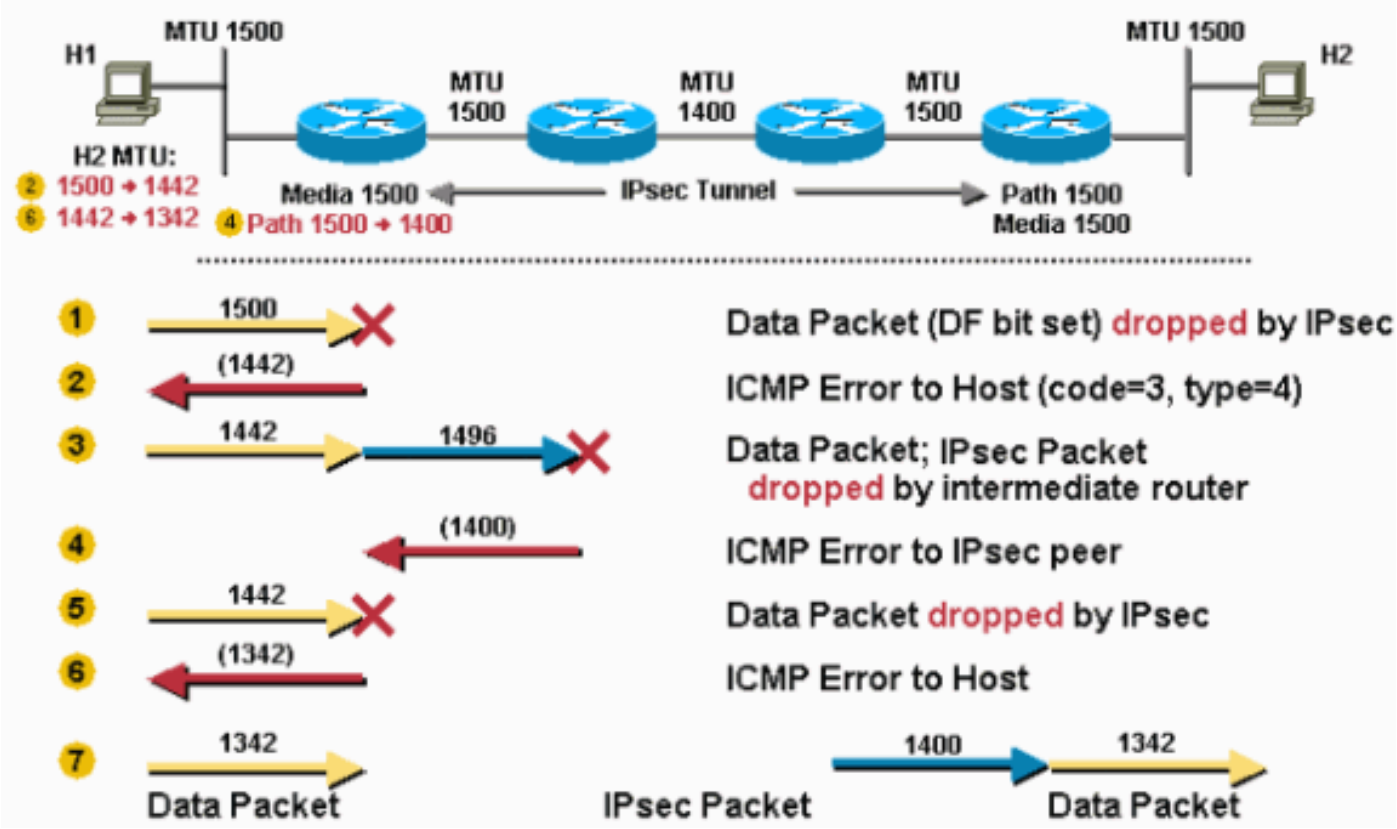
- 使用传输模式(在转换定义上使用子命令 **mode transport**配置)，仅原始IPv4数据包的负载受保护（加密、认证或两者）。负载由 IPv4sec 报头和报尾封装。原始 IPv4 报头保持不变，只不过 IPv4 协议字段更改为 ESP (50)，而原始协议值保存在 IPv4sec 报尾，并在解密数据包时恢复。只有在要保护的 IPv4 流量位于 IPv4sec 对等体之间，并且数据包上的源和目标 IPv4 地址都与 IPv4sec 对等体地址相同时，才使用传送模式。正常情况下，只有在使用另一种隧道协议（例如 GRE）来首先封装 IPv4 数据包，然后使用 IPv4sec 保护 GRE 隧道数据包的情况下，才会使用 IPv4sec 传送模式。

对于数据包和自己的数据包，IPv4sec 始终会执行 PMTUD。可以使用 IPv4sec 配置命令来修改对 IPv4sec IPv4 数据包的 PMTUD 处理，IPv4sec 可以清除、设置 DF 位，或将 DF 位从数据包 IPv4 报头复制到 IPv4sec IPv4 报头。该功能称为“DF 位覆盖功能”。

注意：完成IPv4sec硬件加密后，请避免在封装后进行分段。硬件加密可提供约50 Mbs的吞吐量（取决于硬件），但如果对IPv4sec数据包进行分段，则会丢失50%至90%的吞吐量。这是因为分段的 IPv4sec 数据包重组时会经过进程交换，然后传送给硬件加密引擎进行解密。上述吞吐量损失会使硬件加密吞吐量降至软件加密的性能水平 (2-10 Mbs)。

示例 7

本场景描述的是 IPv4sec 分段过程。在此方案中，整个路径上的 MTU 为 1500。在此方案中，未设置 DF 位。



- 路由器收到发往主机 2 的 1500 字节的数据包（20 字节 IPv4 报头 + 1480 字节 TCP 负载）。
- 1500 字节的数据包经过 IPv4sec 加密，增加了 52 字节的开销（IPv4sec 报头、报尾和另外的 IPv4 报头）。现在 IPv4sec 需要发送 1552 字节的数据包。由于出站MTU为1500，因此必须对此数据包进行分段。
- IPv4sec 数据包被拆分为两个分段。在分段期间，将为第二个分段添加额外的20字节IPv4报头，从而生成一个1500字节的分段和72字节的IPv4分段。
- IPv4sec 隧道对等路由器接收分段，剥离附加的 IPv4 报头，并将 IPv4 分段合并为原始 IPv4sec 数据包。然后 IPv4sec 解密该数据包。
- 最后，路由器将 1500 字节的原始数据包转发到主机 2。

示例 8

此示例与示例6类似，不同之处在于，在这种情况下，原始数据包中设置了DF位，并且IPv4sec隧道对等体之间的路径上存在一条链路，其MTU低于其他链路。

本示例说明IPv4sec对等路由器如何执行两种PMTUD角色，如在隧道终端作为PMTUD参与者的路由器部分所述。

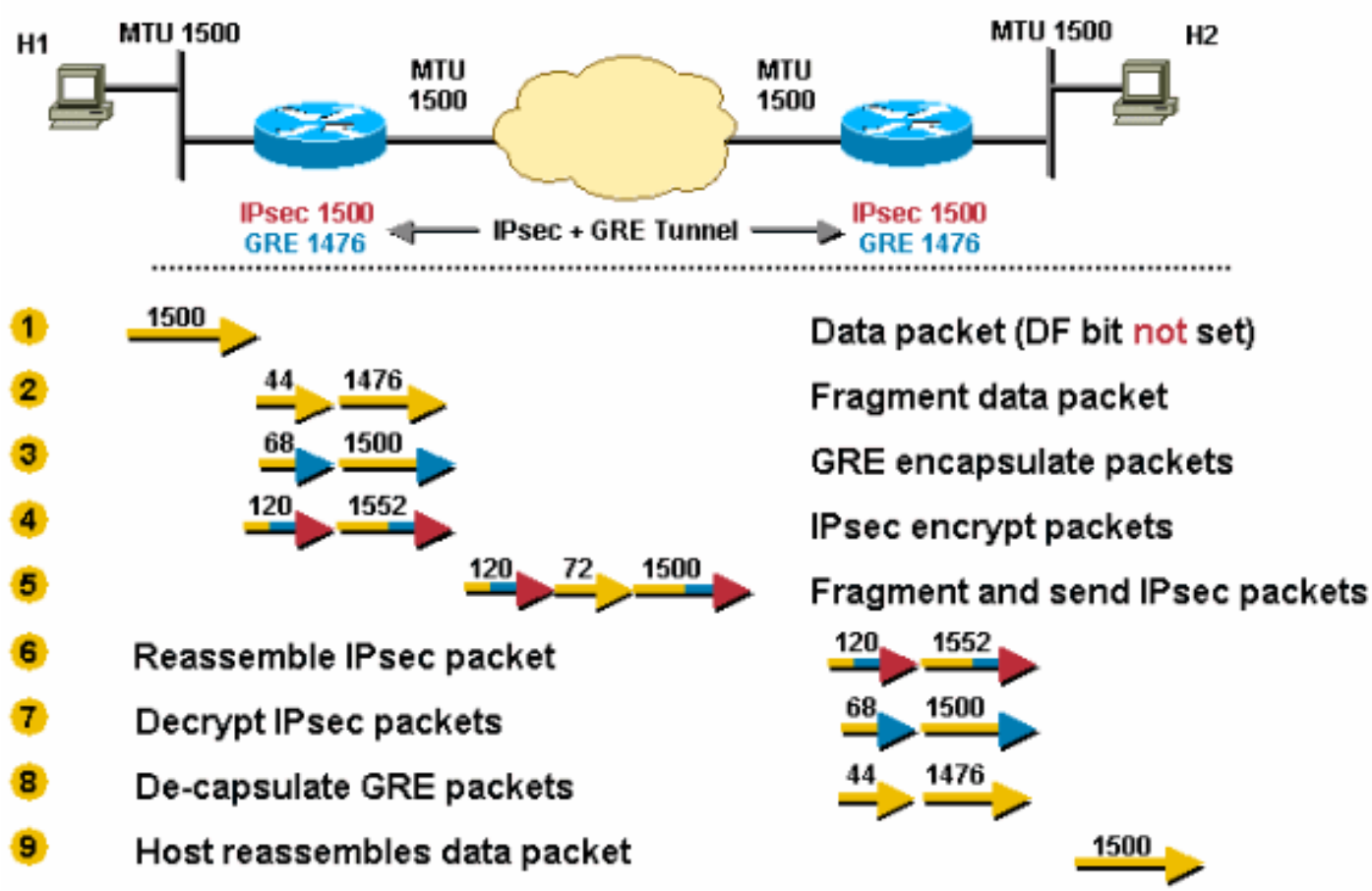
由于需要分段，IPv4sec PMTU变为较低的值。

当IPv4sec加密数据包时，DF位从内部IPv4报头复制到外部IPv4报头。

介质 MTU 和 PMTU 值存储在 IPv4sec 安全关联 (SA) 中。

介质 MTU 基于出站路由器接口的 MTU，而 PMTU 基于 IPv4sec 对等体间路径上看到的最小 MTU。

如图所示，IPv4sec在数据包尝试分段之前会对其进行封装/加密。



- 路由器收到一个1500字节的数据包并将其丢弃，因为添加的IPv4sec开销会使数据包大于PMTU (1500)。

- 路由器向主机 1 发送一条 ICMP 消息，并通知该主机下一跳 MTU 为 1442 (1500 - 58 = 1442)。此58字节是使用IPv4sec ESP和ESPauth时的最大IPv4sec开销。实际IPv4sec开销可能比此值小7个字节。主机 1 通常在其路由表中以目标（主机 2）主机路由的形式记录该信息。

- 主机1将主机2的PMTU降低到1442，因此主机1在将数据重新传输到主机2时发送更小（1442字节）的数据包。路由器接收 1442 字节的数据包，然后 IPv4sec 添加 52 字节的加密开销，由此产生 1496 字节的 IPv4sec 数据包。由于此数据包的报头中设置了 DF 位，因此，采用 1400 字节 MTU 链路的中间路由器将丢弃此数据包。

- 丢弃数据包的中间路由器向 IPv4sec 数据包的发送端（第一个路由器）发送一条 ICMP 消息，告知发送端下一跳 MTU 为 1400 字节。这个值记录在 IPv4sec SA PMTU 中。

- 主机1下次重新传输1442字节的数据包（它未收到确认）时，IPv4sec将丢弃该数据包。路由器会丢弃数据包，因为添加到数据包中的IPv4sec开销会使其大于PMTU (1400)。

- 路由器向主机1发送ICMP消息，告知其下一跳MTU现在为1342。(1400 – 58 = 1342)。主机1再次记录此信息。

- 当主机1再次重新传输数据时，它使用较小的数据包(1342)。此数据包不需要分段，而是通过IPv4sec隧道到达主机2。

GRE 与 IPv4sec 协同工作

使用 IPv4sec 来加密 GRE 隧道时，分段和 PMTUD 的交互会更为复杂。

IPv4sec 和 GRE 以这种方式组合是因为 IPv4sec 不支持 IPv4 组播数据包，这意味着在 IPv4sec VPN 网络上无法运行动态路由协议。

GRE 隧道支持组播，因此可先使用 GRE 隧道加密 GRE IPv4 单播数据包中的动态路由协议组播数据包，然后再使用 IPv4sec 加密单播数据包。

执行此操作时，通常在GRE之上以传输模式部署IPv4sec，因为IPv4sec对等体和GRE隧道终端（路由器）相同，并且传输模式可节省20字节的IPv4sec开销。

有个有趣的案例，其中 IPv4 数据包被拆分为两个分段，并采用 GRE 封装。

在这种情况下，IPv4sec将看到两个独立的GRE + IPv4数据包。通常，在默认配置中，这些数据包中有一个足够大，需要在加密后进行分段。

IPv4sec对等体必须在解密之前重组此数据包。这种发送路由器上的“两次分段”（GRE 前一次，IPv4sec 后一次）会提高时延，并降低吞吐量。

重组是进程交换的，因此每当发生这种情况时，接收路由器上都会发生CPU命中。

如果考虑 GRE 和 IPv4sec 开销而将 GRE 隧道接口的“ip mtu”设置得足够低，则可以避免这种情况（默认情况下，GRE 隧道接口“ip mtu”被设置为实际传出接口 MTU 值 - GRE 开销字节数）。

此表列出了假设传出物理接口的MTU为1500的每个隧道/模式组合的建议MTU值。

隧道组合	需要的特定 MTU	建议的 MTU
GRE + IPv4sec（传送模式）	1440 字节	1400 字节
GRE + IPv4sec（隧道模式）	1420 字节	1400 字节

注意：建议将MTU值设置为1400字节，因为该值涵盖了最常见的GRE + IPv4sec模式组合。并且，额外允许 20 或 40 字节的开销不会产生显著的负面影响。记住并设置一个值相对较容易，并且该值几乎涵盖了所有情况。

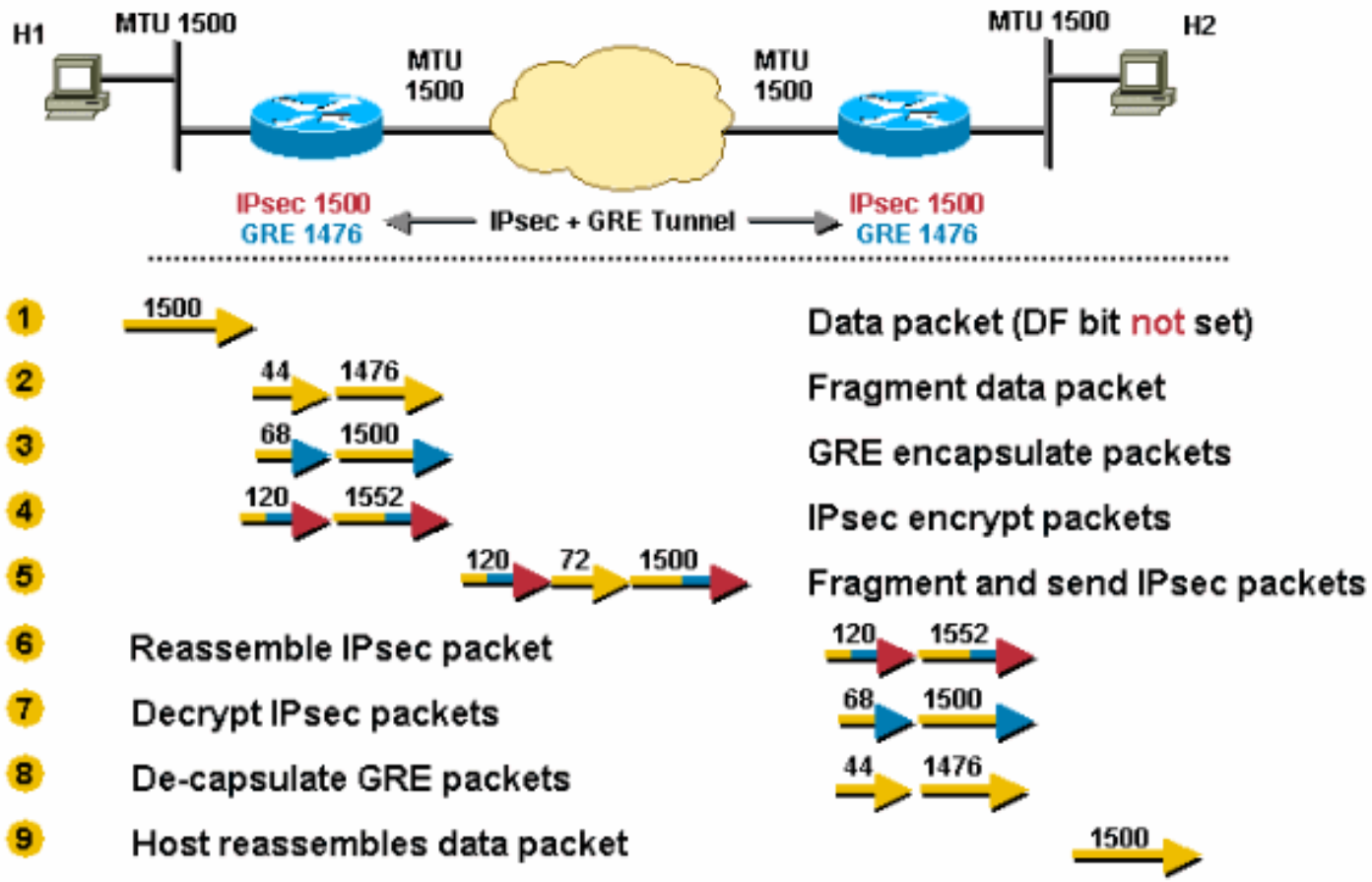
示例 9

IPv4sec 部署在 GRE 之上。传出物理 MTU 为 1500，IPv4sec PMTU 为 1500，GRE IPv4 MTU 为 1476 (1500 - 24 = 1476)。

因此，TCP/IPv4数据包分片两次，一次在GRE之前，一次在IPv4sec之后。

数据包在GRE封装之前分段，其中一个GRE数据包在IPv4sec加密之后再次分段。

在本场景下，如果在 GRE 隧道上配置“ip mtu 1440”（IPv4sec 传送模式）或“ip mtu 1420”（IPv4sec 隧道模式），则可以消除两次分段的可能性。



- 路由器收到一个 1500 字节的数据报。

- 在封装之前，GRE 将 1500 字节的数据包拆分为两个分段，一个 1476 字节 (1500 - 24 = 1476)，另一个 44 字节（24 字节数据 + 20 字节 IPv4 报头）。

- GRE 封装 IPv4 分段，该过程将导致每个数据包增加 24 字节。因而将产生两个 GRE + IPv4sec 字段，一个 1500 字节 (1476 + 24 = 1500)，另一个 68 字节 (44 + 24 = 68)。

- IPv4sec对这两个数据包进行加密，每个数据包会增加52字节（IPv4sec隧道模式）的封装开销，从而产生1552字节和120字节的数据包。

- 由于 1552 字节的 IPv4sec 数据包大于出站 MTU (1500)，因此，路由器会将其分段。1552 字节的数据包被拆分为 1500 字节的数据包和 72 字节的数据包（52 字节负载加上为第二个分段附加的 20 字节 IPv4 报头）。三个数据包（1500 字节、72 字节和 120 字节）被转发到 IPv4sec + GRE 对等设备。

- 接收路由器重组两个 IPv4sec 分段（1500 字节和 72 字节），以便获取原始 1552 字节的 IPv4sec + GRE 数据包。对于 120 字节的 IPv4sec + GRE 数据包无需任何操作。

- IPv4sec 对 1552 字节和 120 字节的 IPv4sec + GRE 数据包进行解密，以便获取 1500 字节和 68 字节的 GRE 数据包。

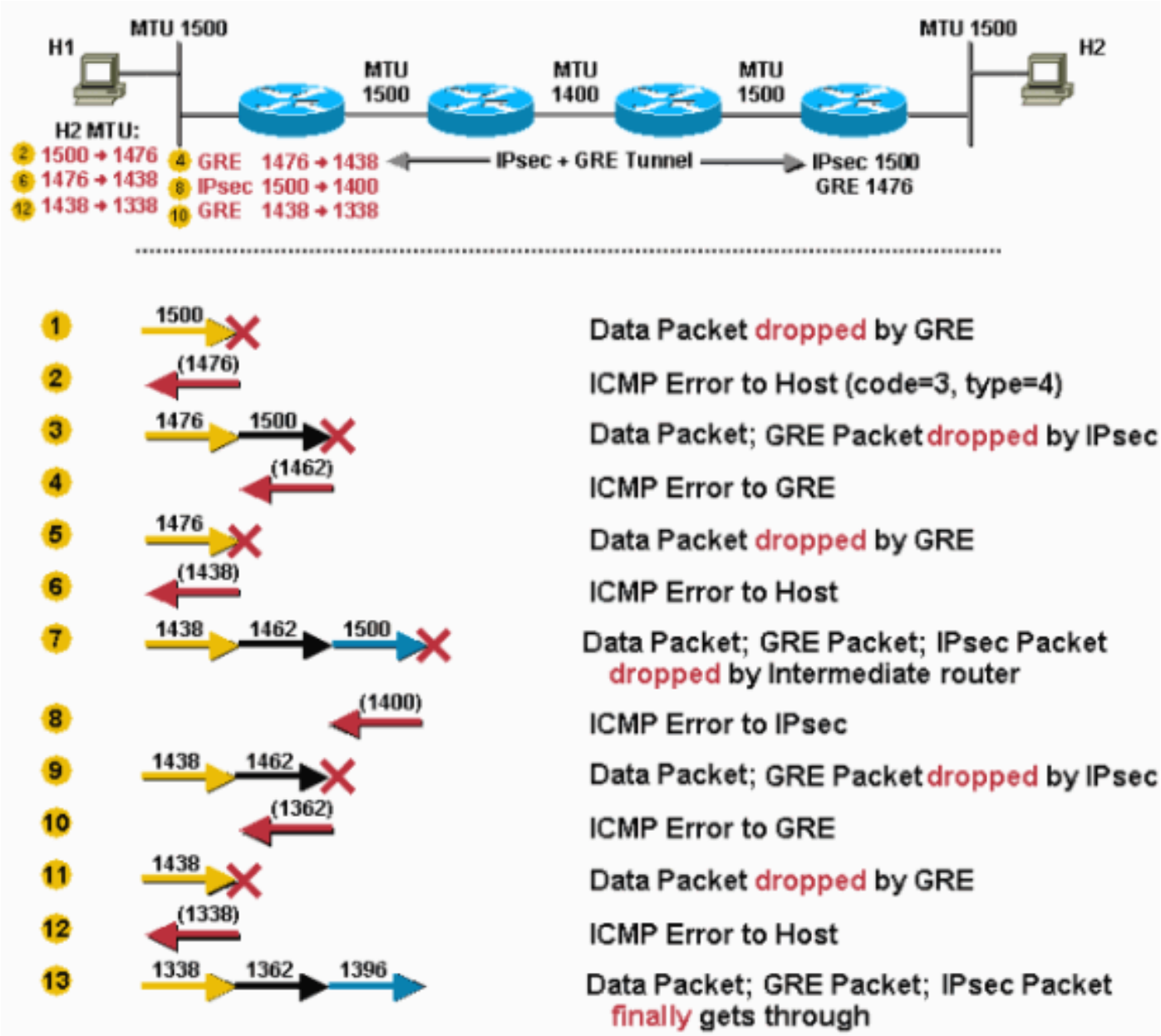
- GRE 解封装 1500 字节和 68 字节的 GRE 数据包，以便获取 1476 字节和 44 字节的 IPv4 数据包分段。然后这些 IPv4 数据包分段被转发到目标主机。

- 主机 2 重组这些 IPv4 分段，以便获取原始 1500 字节的 IPv4 数据报。

方案 10 与方案 8 相似，不同之处在于方案 10 的隧道路径中存在更小 MTU 的链路。对于从主机 1 发往主机 2 的第一个数据包而言，这是最糟糕的场景。在完成本方案的最后一步后，主机1为主机2设置正确的PMTU，并都适合主机1和主机2之间的TCP连接。主机1与其他主机（可通过IPv4sec + GRE隧道访问）之间的TCP流只需经过场景10的最后三个步骤。

在本场景中，GRE隧道上配置了 **tunnel path-mtu-discovery** 命令，并且在来自主机1的TCP/IPv4数据包上设置了DF位。

示例 10



- 路由器收到一个 1500 字节的数据包。由于设置了 DF 位，并且在增加 GRE 开销（24 字节）之后数据包大小超过出站接口“ip mtu”，因此 GRE 无法对该数据包进行分段或转发，并丢弃此数据包。

- 路由器向主机 1 发送 ICMP 消息，以便该主机知晓下一跳 MTU 为 1476 (1500 - 24 = 1476)。

- 主机 1 针对主机 2 将其 PMTU 更改为 1476，并在重新传输数据包时发送更小大小。GRE 封装数据包，并将 1500 字节的数据包传递给 IPv4sec。由于 GRE 从内部 IPv4 报头中复制了 DF 位（已设置），并且加上 IPv4sec 开销（最大 38 字节）后，数据包因太大而无法传出物理接口，因此 IPv4sec 丢弃该数据包。

- IPv4sec向GRE发送ICMP消息，指明下一跳MTU为1462字节（因为加密和IPv4开销最多添加38字节）。GRE在隧道接口上将值1438 (1462 - 24)记录为"ip mtu"。

- **注意**：值的更改在内部存储，无法在 **show ip interface tunnel**<#>命令输出中看到。仅当改用 **debug tunnel** 命令时，才会显示此更改。

- 主机 1 下次重新传输 1476 字节的数据包时，GRE 将丢弃此数据包。
- 路由器向主机 1 发送 ICMP 消息，指明下一跳 MTU 为 1438。
- 主机 1 针对主机 2 减小其 PMTU，并重新传输 1438 字节的数据包。这次 GRE 接受该数据包，对其进行封装，并将其传递给 IPv4sec 进行加密。
- IPv4sec 数据包被转发到中间路由器并被丢弃，因为该路由器出站接口 MTU 为 1400。
- 中间路由器向 IPv4sec 发送 ICMP 消息，指明下一跳 MTU 为 1400。IPv4sec 将该值记录在关联 IPv4sec SA 的 PMTU 值中。
- 当主机 1 重新传输 1438 字节的数据包时，GRE 封装该数据包，然后将其传递给 IPv4sec。IPv4sec 丢弃该数据包，因为其已将自己的 PMTU 改为 1400。
- IPv4sec 向 GRE 发送 ICMP 错误消息，指明下一跳 MTU 为 1362，并且 GRE 在内部记录值 1338。
- 当主机1重新传输原始信息包时(因为没有收到确认)，GRE将丢弃它。
- 路由器向主机 1 发送 ICMP 消息，指明下一跳 MTU 为 1338（1362 - 24 字节）。主机 1 针对主机 2 将其 PMTU 减小至 1338。
- 主机1转发1338字节信息包，同时它可以最终到达主机2。

其他建议

如果在同一台路由器上配置了GRE和IPv4sec，则在隧道接口上配置 **tunnel path-mtu-discovery** 命令可以帮助GRE和IPv4sec交互。

如果未配置 **tunnel path-mtu-discovery** 命令，则会始终在GRE IPv4报头中清除DF位。

这样便可对 GRE IPv4 数据包进行分段，即使已封装的数据 IPv4 报头设置了 DF 位（通常不允许对数据包进行分段）也是如此。

如果在GRE隧道接口上配置了 **tunnel path-mtu-discovery** 命令：

- GRE将DF位从数据IPv4报头复制到GRE IPv4报头。
- 如果在GRE IPv4报头中设置了DF位，并且数据包在物理传出接口上的IPv4 MTU经过IPv4sec加密后“过大”，则IPv4sec将丢弃数据包并通知GRE隧道减小其IPv4 MTU大小。
- IPv4sec对其自己的数据包执行PMTUD，如果IPv4sec PMTU更改（如果减少），则IPv4sec不会立即通知GRE，但当另一个更大的数据包通过时，则会执行第2步中的过程。

- GRE IPv4 MTU现在更小，因此它会丢弃任何设置了DF位且现在过大的数据IPv4数据包，并向发送主机发送ICMP消息。

tunnel path-mtu-discovery

命令有助于GRE接口动态设置其IPv4 MTU，而非使用 **ip mtu** 命令静态设置。实际上，建议同时使用这两个命令。

ip mtu 命令用于为GRE和IPv4sec开销提供空间（相对于本地物理传出接口的IPv4 MTU）。

如果IPv4sec对等体之间的路径上存在IPv4 MTU更小的链路，则 **tunnel path-mtu-discovery** 命令允许GRE隧道IPv4 MTU进一步减小。

如果在已配置 GRE + IPv4sec 隧道的网络中有 PMTUD 相关的问题，可以参考以下解决方案。

下面列表从最可取的解决方案开始列出。

- 修复 PMTUD 无法正常工作的问题，这通常是由阻止 ICMP 的路由器或防火墙造成的。
- 在隧道接口上使用 **ip tcp adjust-mss** 命令，以使路由器减小TCP SYN数据包中的TCP MSS值。这有助于两个终端主机（TCP发送方和接收方）使用足够小的数据包，以便不需要PMTUD。
- 在路由器的入口接口上使用策略路由并配置路由映射，从而在 DF 位传入 GRE 隧道接口之前，清除数据 IPv4 报头中的 DF 位。这允许在GRE封装之前对数据IPv4数据包进行分段。
- 在 GRE 隧道接口上增大“ip mtu”，使其等于出站接口 MTU。这允许数据IPv4数据包进行GRE封装，而无需首先对其进行分段。然后，对GRE数据包进行IPv4sec加密，然后进行分段以传出物理出站接口。在此种情况下，您不必在GRE隧道接口上配置 **tunnel path-mtu-discovery** 命令。由于 IPv4sec 对等体上的 IPv4 数据包重组在进程交换模式下进行，因此这可以极大减少吞吐量。

相关信息

- **IP 路由 支持**页
- **IPSec（IP 安全**协议）支持页
- **RFC 1191 路径 MTU** 发现
- **RFC 1063 IP MTU** 发现选项
- **RFC 791 Internet** 协议
- **RFC 793** 传输控制协议
- **RFC 879 TCP 最大**数据段大小和相关主题
- **RFC 1701 通用路由封装 (GRE)**
- **RFC 1241 Internet 封装**协议方案
- **RFC 2003 IP**

中的 IP 封装

- **技术支持和文档 - Cisco Systems**

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。