The Never-Ending Story of IP Fragmentation

By Ivan Pepelnjak

1. 1. 2008

I first encountered IP fragmentation issues almost 15 years ago, when people started deploying carelessly designed firewalls that blocked all Internet Control Messages Protocol (ICMP) traffic. One would hope that the situation would get better as network designers and operations engineers gained experience, but it's constantly getting worse with the introduction of new encapsulation techniques like PPP-over-Ethernet (PPPoE) used in DSL connections, IPSec-based encryption and IP-over-IP tunnels used to fix IP routing problems or implement topologies that some "service providers" cannot support. In this article, you'll find the reasons behind IP fragmentation, the detailed description of how *Path MTU discovery* works and the various mechanisms you can use on Cisco routers to alleviate the IP fragmentation-related problems.

MTU Basics

The basic fact in networking is that not all networking technologies were created equal. One of the differences between various layer-2 technologies is the maximum payload (commonly called *Maximum Transmission Unit* – MTU) a layer-2 frame can transport. For example, regular Ethernet packets can be up to 1518 bytes long (including the CRC bytes), but they can transport only a 1500-byte payload if you're using the default encapsulation (the payload size is reduced to 1492 bytes if you use SNAP encapsulation). Token Ring and FDDI can transport much larger packets and the larger packet sizes are sometimes used to reduce the overhead in high-speed peer-to-peer data transfer.

Note: Frames longer than 1518 bytes (called jumbo frames) are also allowed in Fast Ethernet and Gigabit Ethernet environments to get the same results.

On the other hand, slow-speed serial links used lower MTU sizes to reduce the serialization delay (transmitting a single 1500-byte IP packet on a 64 kbps link takes almost 200 milliseconds). This behavior is rarely seen today due to widespread deployment of link fragmentation and interleaving (LFI) over PPP links.

Encapsulation and tunneling techniques add their own limitations: for example, if you're using PPP-over-Ethernet, the PPPoE header takes eight bytes from the Ethernet payload, leaving 1492 bytes for the IP packet. Similarly, Generic Route Encapsulation (GRE) uses 24 bytes headers, reducing the MTU on GRE tunnels to 1476 bytes. Obviously, the combination of various encapsulation techniques further reduces the MTU size; the MTU of a GRE tunnel running over an ADSL is 1468 bytes.

Technical details: While the GRE header is only four bytes long (unless you use the GRE key, which extends the GRE header to eight bytes), the IP-in-IP encapsulation requires an extra IP header (20 bytes), resulting in 24-byte overhead. A pure IP-over-IP tunnel configured with ***tunnel mode ipip*** *has a 20-byte overhead.*

IPSec further complicates the MTU calculations, as the size of the IPSec header that is inserted in the IP packet depends on the parameters of the

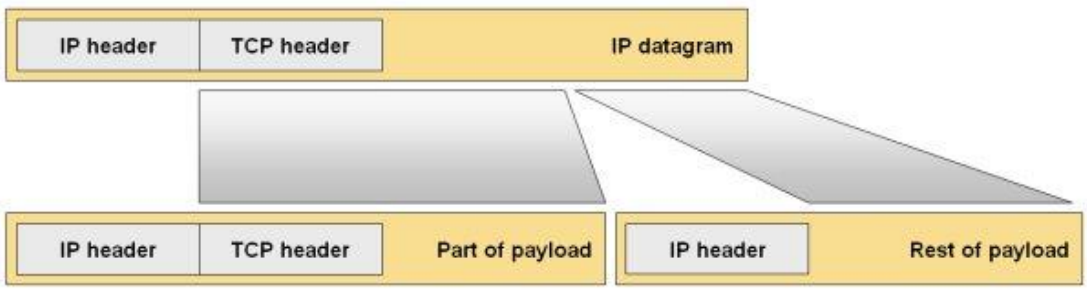
IPSec transform sets (combinations of tunneling mode, encryption, NAT-T usage and variable padding to 8- or 16-byte blocks).

Note: The original Cisco IOS paradigm where the encrypted packets are routed through the same interface as non-encrypted traffic does not help either, as there is no longer an interface-wide MTU; the MTU size varies based on whether a packet is encrypted or not.

The Drawbacks of IP Fragmentation

The IP protocol stack never had a reliable mechanism by which the end hosts could figure out the maximum payload size to use when communicating with a remote IP host (and it's not present in IPv6 either). The absence of the network-wide MTU mechanism is somewhat understandable, as the IP packets are routed independently of each other and different packets between the same end hosts could take different routes with varying MTU sizes. However, the lack of end-to-end information can quickly result in oversized packets being received by the intermediate routers that have to route them somehow. The IP protocol provides a convenient solution: the IP fragmentation, a mechanism where a single inbound IP datagram is split into two or more outbound IP datagrams. The IP header is copied from the original IP datagram into the fragments, special bits are set in the fragments' IP headers to indicate that they are not complete IP packets, and the payload is spread across the fragments (see Figure 1).

Figure 1: Sample IP fragmentation



Technical Details: The IP fragmentation was particularly bad in the earlier Cisco IOS releases, as the routers had to make copies of the original IP packets to generate the fragments, thus forcing the IP fragmentation into

the process switching path (which is significantly slower than any other switching mechanism). Later IOS releases introduced particle-based fragmentation, which allows IP fragmentation to be performed within Cisco Express Forwarding (CEF).

The IP fragmentation always increases the layer-3 overhead (and thus reduces the actual bandwidth available to user traffic). For example, if the end-host thinks it can use 1500-byte IP packets, but there is a hop in the path with MTU size 1472, each oversized IP packet will be split in two packets, resulting in an additional 20-byte IP header.

Even worse, the application-layer information is missing from the non-first IP fragments, as the TCP or UDP header is not copied into all fragments. This fact has been widely used to break through firewalls using overlapping fragments where the second fragment would rewrite the TCP/UDP header from the first fragment. As a result, some firewalls might be configured to drop IP fragments (resulting in blocked communication between the end-hosts), while others have to consume additional CPU resources to reassemble the fragments and inspect their actual contents. Intrusion Detection/Prevention Systems (IDS/IPS) have to provide similar functionality to effectively detect intrusion signatures.

Last but not least, the IP fragments impose additional burden on the receiving end-system, as it has to reassemble the fragments before they can be delivered to higher protocol layers. The situation is aggravated when the IP traffic is terminated by the router (for example, GRE tunnels or encrypted traffic); if a router performs IP reassembly, the reassembled packets are process switched.

Warning: If you use GRE-over-IPSec design and the GRE packet gets fragmented due to the additional IPSec header, the performance of the router receiving the fragments will fall by a factor of ten to thousand.

Path MTU Discovery

The early IP host implementations were extremely simple: if the destination IP address was directly connected, the interface MTU size was used; otherwise the MTU was fixed at 576 bytes. This algorithm proved

impractical in both low-speed networks due to extra overhead introduced by small packet sizes as well as in high-speed networks due to extra CPU utilization required to process the same amount of data. Furthermore, as the routers started being used to connect LAN segments (for example, in collapsed backbone scenarios), the usage of small packet sizes between LAN segments was bordering on ridiculous.

Technical Details: The overhead of the 40 bytes of IP and TCP headers in a 576-byte packet is 7.5% (40/532), the same overhead in a 1500-byte packet is 2,7%.

The imperfect solution that was proposed almost 20 years ago is still in use today: the mechanism used by vast majority of IP hosts to detect the end-to-end MTU size is the *Path MTU discovery*, defined in RFC 1191. The Path MTU Discovery (PMTUD) relies on the following properties of the IP and ICMP protocols:

- The sending host can indicate that its IP datagrams shall not be fragmented by setting the Don't Fragment (DF) bit in each outgoing datagram.
- The intermediate routers that have to drop oversized IP datagrams (that cannot be fragmented) inform the sending host that the datagram has been dropped with an ICMP Destination Unreachable message with the status code *Fragmentation needed and DF set*.
- An extra field in the ICMP response indicates the maximum MTU the sending router could support on the outgoing link.

An IP host using PMTUD performs the following steps:

- Whenever the first PMTUD-aware session with a new destination host is started, the MTU of the outgoing interface is assumed to be the MTU of the overall path.
- All outgoing IP datagrams are sent with the DF bit set.
- Whenever the layer-4 session happens to send an oversized datagram, a router in the path will drop the packet, report that the local egress MTU was exceeded and suggest the new MTU size in the ICMP reply.

- The MTU size reported by an intermediate router is cached as the new MTU for the destination host and all future outgoing datagrams will not exceed that MTU.
- The TCP stack in the originating host or a PMTUD-aware UDP application has to retransmit the data in smaller datagrams.

Note: As you can see from the description, the PMTUD is in most implementations not an active process by which an IP host would measure the end-to-end MTU. The computed end-to-end MTU is a by-product of sending large datagrams and might not be correct if all the sessions between a pair of end-hosts send only small datagrams. There were, however, IP implementations that continuously pinged the remote host to measure path MTU.

The IP hosts (as well as most of the routers) don't have the detailed visibility into the structure of the global Internet and the subnet masks associated with the destination IP hosts, therefore the PMTUD has to be performed on a per-destination-host basis. Since the end-to-end paths through the network might change with time, the hosts eventually age out the computed end-to-end MTU values (the timeout recommended in the RFC 1191 is ten minutes), resulting in renewed PMTUD process. Obviously, the decrease in the path MTU due to a routing table change is discovered as soon as the first datagram exceeds the new path MTU.

The effects of the computed path MTU on host applications depends on the transport protocol they are using. TCP sessions use PMTUD transparently, as the TCP protocol models a continuous stream that is not sensitive to packet boundaries. PMTUD is usually built into the TCP stack and can be disabled only on per-host basis in Windows XP or on a socket-by-socket basis in Linux (by setting the IP_MTU_DISCOVER option). The end-to-end MTU for TCP sessions can also be influenced with the TCP Maximum Segment Size (MSS) option that is negotiated between the endpoints of the TCP session; outbound datagrams use the smaller of the MSS and the path MTU values.

UDP-based applications control their own datagram size and can only be assisted by the operating system:

- If an UDP application sends datagrams without the DF bit set, they are propagated as required and fragmented within the network as needed.
- An UDP application can decide to rely on PMTUD by indicating that its outgoing datagrams shall have the DF bit set. This is usually done with a *setsockopt* call setting the IP_DONTFRAGMENT option (Windows XP) or IP_MTU_DISCOVER option (Linux).
- As soon as the UDP application has indicated it wants to use the PMTUD, all the outgoing datagrams are sent with the DF bit set, potentially resulting in dropped UDP packets.

Technical details: If the UDP application tries to exceed the locally computed path MTU, the outgoing message will be rejected immediately. If an intermediate router drops the UDP packet and reports the local MTU with ICMP Destination Unreachable message, no indication is sent to the UDP application (even though the local path MTU is updated); it has to perform its own retransmission.

Network Implications

The PMTUD is enabled by default in almost all modern TCP/IP implementations; it's thus mandatory that your packet filters and firewalls don't block the PMTUD-related ICMP messages. For example, you should include the two lines from Listing 1 into your IP access-lists (the second line blocks ICMP fragments that could be used to change the meaning of the ICMP response):

Listing 1: Permitting PMTUD-related ICMP packets in an extended ip access-list

```
permit icmp any any packet-too-big

deny icmp any any fragments
```

Likewise, you should enable ICMP inspection (available from IOS release 12.3) with the **ip inspect name** *inspection-name* **icmp** if you use IOS firewall feature set.

Sometimes, the PMTUD will be broken due to circumstances way beyond your control; for example, the path toward the destination host might

include a non-compliant router or the firewall at the other end (protecting the destination server) blocks ICMP messages. In such cases, you can usually get at least the TCP applications working by lowering the TCP MSS value on the router with the **ip tcp adjust-mss** *maximum-size* configuration command. The *maximum-size* parameter specifies the maximum TCP payload and has to be at least 40 bytes smaller than the end-to-end MTU. The Cisco IOS documentation is not very specific on where you should apply this command. As it turns out, you can apply it on inbound or outbound interface; MSS value in TCP packets with SYN bit set is modified in the ingress as well as in the egress part of the packet switching path, resulting in the minimum of the two values if the **ip tcp adjust-mss** is specified on both interfaces.

Fixing broken UDP applications is way harder; some of them might set the DF bit but remain oblivious of its implications (sometimes happily assuming 1500-byte end-to-end MTU). If you cannot increase the MTU size, the only reasonable solution is to clear the DF bit on the first router with the policy-based routing. For example, the configuration in Listing 2 clears the DF bit on all UDP traffic entering the router through the Fast Ethernet interface.

Listing 2: Clear the don't fragment bit for UDP traffic

```
ip access-list extended BrokenUDP
 remark The UDP filter should be more specific
 permit udp any any
!
route-map ClearDF permit 10
 match ip address BrokenUDP
 set ip df 0
!
interface FastEthernet0/0
 ip address 10.0.0.1 255.255.255.240
ip policy route-map ClearDF
```

IP Fragmentation and Tunnels

The impact of IP fragmentation can be devastating if you use high-speed GRE tunnels or IPSec encryption in your network. By default, Cisco IOS assumes a 1500-byte end-to-end MTU between the tunnel endpoints, resulting in 1476 byte IP MTU on the tunnel interface. The GRE packets generated by the router are usually sent without the DF bit and can be fragmented if an intermediate hop between the tunnel endpoints does not support 1500-byte MTU (for example, a PPPoE DSL connection).

Note: Another common problem is the combination of GRE tunnels with IPSec encryption, particularly if the two operations are not performed by the same device.

The GRE or IPSec fragments have to be reassembled on the tunnel tail-end router or IPSec peer, resulting in process switching of all tunneled or encrypted traffic. The process switching is several times slower than CEF switching on software-only platforms, resulting in unexpectedly high CPU load on the tail-end router. The situation is worse on platforms that rely on hardware-based or hardware-assisted packet switching; the switching bandwidth of a high-end router can drop by a factor of hundred or more.

You can solve the GRE-related problems by manually lowering the **ip mtu** on the tunnel interface (ideally in combination with the **ip tcp adjust-mss** configuration command), or you could enable PMTUD for GRE tunnels with the **tunnel path-mtu-discovery** interface configuration command. When you enable the PMTUD on a GRE tunnel, the GRE packets are sent with the DF bit set and the router responds to the incoming ICMP destination unreachable messages with the reduction of the tunnel MTU size. The decreased MTU can only be inspected with the **show interface** command (Listing 3).

Listing 3: Tunnel Path MTU Discovery display

```
Rtr#show interface tunnel 0 | include protocol|Path

Tunnel0 is up, line protocol is up

Tunnel protocol/transport GRE/IP

Path MTU Discovery, ager 10 mins, min MTU 92, MTU 776, expires 00:01:57
```

The Cisco IOS implementation of the tunnel PMTUD is suboptimal (to be polite):

- DF bit is copied from the source IP packet into the GRE envelope. If the source IP packet doesn't have the DF bit set, it won't be set in the outgoing GRE packet, potentially resulting in fragmentation of the GRE packet and expensive reassembly on the tail-end router.
- The tunnel PMTUD process is thus triggered only by incoming packets with DF bit set.
- After the end-to-end tunnel MTU has been computed, it's only applied to the incoming packets with the DF bit set.
- Even if the router knows the correct end-to-end MTU, the incoming packets without the DF bit *are not fragmented*, resulting in GRE packet fragmentation further down the path.

  Note: Regardless of the tunnel PMTUD algorithm, the router fragments or rejects the tunneled packets if their size exceeds the IP MTU size configured on the tunnel interface with the ***ip mtu*** *configuration command.*

Summary

After 20 years of struggles, the IP fragmentation remains one of the challenges in IP network deployment, particularly if you have to implement extra layers in the protocol stack (like PPP over Ethernet) or if you use any IP-over-IP encapsulation or IP encryption techniques. The generic solution to the IP fragmentation issues should be the *Path MTU Discovery* that was issued as an RFC in November 1990 and remains a draft standard ever since. However, misconfigured firewalls still prevent us from using this solution reliably almost 20 years after it was designed.

If you cannot get the PMTUD to work reliably in your network, you can fix the TCP sessions by manually setting the TCP Maximum Segment Size on the intermediate routers with the **ip tcp adjust-mss** interface configuration command. Broken UDP applications that pretend to use the PMTUD but ignore its results can be fixed with policy-based routing that clears the DF bit in UDP packets.

The worst impact of IP fragmentation is in the router-to-router communication (GRE tunnels or IPSec encryption). If a router-to-router IP packet is fragmented somewhere in the path, the receiving router has to reassemble the original packet, resulting in significantly reduced switching performance. In these cases, it's best to enable the router's support for PMTUD with the **tunnel path-mtu-discovery** interface configuration command (assuming the end hosts support PMTUD as well). Worst case, you can still lower the tunnel MTU size as well as TCP MSS value, resulting in slightly higher switching overhead but ensuring that the GRE or IPSec packets will not be fragmented.

NIL – More Than Just a Training Company

NIL Learning delivers the leading-edge Cisco training to IT professionals and companies around the globe. Through field-proven experts — each both active engineer and instructor — NIL Learning enhances the standard learning curriculum with real-life experience and helps clients to maximize their training investment.

NIL Learning is part of NIL, a leading global IT solutions provider. Since 1992, NIL has been at the forefront of advanced contributors to strategic partner Cisco's technologies, learning curriculum and value-added solutions deployed to clients around the globe. Today, NIL has earned the highest certifications offered by Cisco, VMware, EMC, HP, IBM, Microsoft, F5, Jive, MobileIron, RSA, VCE and others. Their portfolio of solutions consists of managed services, professional services and learning services.

NIL is headquartered in Slovenia, with regional offices in Croatia, Serbia, Saudi Arabia, the U.S., Turkey, South Africa, Morocco, Nigeria, Kenya and Botswana.

Why learn at NIL LEARNING?

- All NIL LEARNING instructors are **field-proven experts** - each both active engineer and instructor.

- **75% of NIL LEARNING engineers hold CCSI** certifica- tions, and **18 have already achieved the respected CCIE rank.**
- NIL LEARNING **enhances the standard learning curriculum** with real-life experience and helps clients to maximize their training investment.
- NIL has been a Cisco Training Partner for many years; it became a **Cisco Learning Partner in 1993**, and has been a **Cisco Gold Partner since 1995.**
- NIL was awarded the **Cisco Most Business Relevant Learning Partner in MEA in 2010 and the most innova- tive learning partner in MEA.**
- NIL received the **Innovation Award for its Technology Led Training and its extensive contribution to Cisco learning solutions** at the Cisco EMEAR Learning Partner Summit in 2012.
- NIL received the **Innovation Award for its Technology Led Training and Advanced Engineer Program** at the Cisco Global Learning Partner Summit in 2013.
- NIL LEARNING runs a **centralized training schedule across the whole EMEAR region.**

More Info

Slovenia

T: +386 1 4746 500

E: sales-support@nil.com

Botswana

T: +267 318 1684

E: training@it-iq.bw

Croatia

T: +385 (0)51 583 255

E: info-nilcroatia@nil.com

Kenya

T: +27 (0)11 575 4637

E: mea_sales@nil.com

Morocco

T: +212(0) 660 808 394

E: info-nilmorocco@nil.com

Nigeria

T: +27 (0)11 575 4637

E: mea_sales@nil.com

Saudi Arabia

T: +966 1 465 4641

E: info.nilme@nil.com

Serbia

T: +381 11 2282 818

E: info-nilserbia@nil.co.rs

South Africa

T: +27 (0)11 575 4637

E: mea_sales@nil.com

Turkey

T: +902 123 81 8639

E: info-nilturkey@nil.com

USA

T: +1 612 886 3900

E: info-nilusa@nil.com

www.learning.nil.com