IP fragmentation

Giorgio Daniele Luppina
s295445@studenti.polito.it

Abstract

Internet Protocol (IP) fragmentation plays a vital role in facilitating the transmission of data across diverse network environments. This essay explores the concept of IP fragmentation, shedding light on its purpose and mechanics.

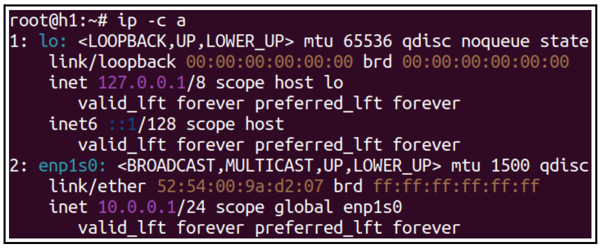
1. Introduction

IP fragmentation serves as the mechanism that allows data packets to traverse networks with varying **Maximum Transmission Unit** (MTU) sizes. Not all networks have the same capacity to transmit data, and it is the role of IP fragmentation to break down large packets into smaller fragments that can successfully traverse networks with lower MTU values. By doing so, IP ensures that data can reach its destination, even when traversing networks with different transmission capabilities.

2. Maximum Size before Fragmentation of ICMP messages

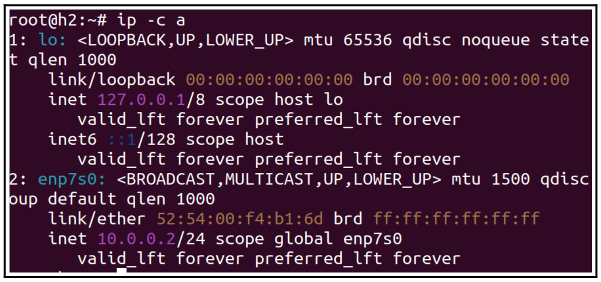
If using an **MTU** of **1500 bytes**, each host can not place more than 1480 byte in the IPv4 payload as data. For the sake of simplicity, we assume to not have any options, and therefore sending only 20-bytes long IPv4 header packets. If the upper layer protocol is **ICMP**, provided that that an ICMP header is 8-bytes long, each host should not include more than **1472 byte** in the ICMP payload. Otherwise, we expect to see fragmentation.

Host H1 has the following configuration:



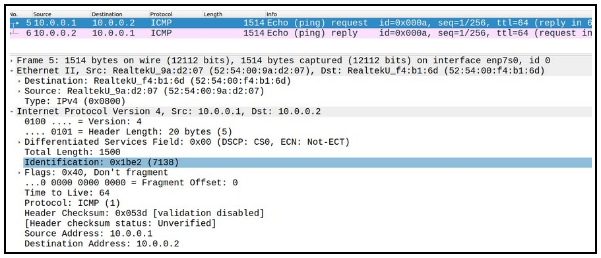
```
root@h1:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
    link/ether 52:54:00:9a:d2:07 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.1/24 scope global enp1s0
       valid_lft forever preferred_lft forever
```

Host H2 has the following configuration:



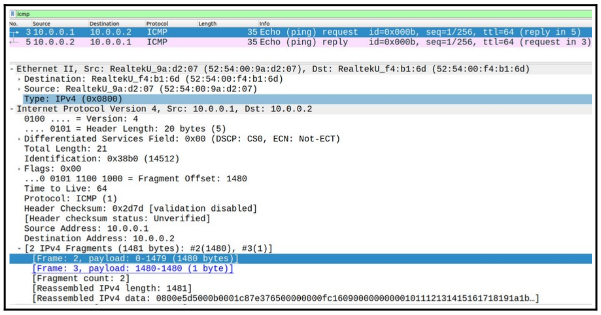
```
root@h2:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
oup default qlen 1000
    link/ether 52:54:00:f4:b1:6d brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.2/24 scope global enp7s0
       valid_lft forever preferred_lft forever
```

Sending an ICMP echo-request with 1472 bytes as payload does not cause fragmentation, as followed depicted.



```
No.  Source      Destination  Protocol  Length  Info
  5 10.0.0.1    10.0.0.2     ICMP        1514 Echo (ping) request  id=0x000a, seq=1/256, ttl=64 (reply in 6
  6 10.0.0.2    10.0.0.1     ICMP        1514 Echo (ping) reply    id=0x000a, seq=1/256, ttl=64 (request in

› Frame 5: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp7s0, id 0
˅ Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Destination: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Source: RealtekU_9a:d2:07 (52:54:00:9a:d2:07)
    Type: IPv4 (0x0800)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0x1be2 (7138)
  › Flags: 0x40, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x053d [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.0.2
```

As expected, an ICMP echo-request with 1473 bytes as payload cause fragmentation in the sender because of its own NIC MTU, as followed depicted.



```
icmp
No.  Source      Destination  Protocol  Length  Info
  3 10.0.0.1    10.0.0.2     ICMP          35 Echo (ping) request  id=0x000b, seq=1/256, ttl=64 (reply in 5)
  5 10.0.0.2    10.0.0.1     ICMP          35 Echo (ping) reply    id=0x000b, seq=1/256, ttl=64 (request in 3)

˅ Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Destination: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Source: RealtekU_9a:d2:07 (52:54:00:9a:d2:07)
    Type: IPv4 (0x0800)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 21
    Identification: 0x38b0 (14512)
  › Flags: 0x00
    ...0 0101 1100 1000 = Fragment Offset: 1480
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x2d7d [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.0.2
  ˅ [2 IPv4 Fragments (1481 bytes): #2(1480), #3(1)]
      [Frame: 2, payload: 0-1479 (1480 bytes)]
      [Frame: 3, payload: 1480-1480 (1 byte)]
      [Fragment count: 2]
      [Reassembled IPv4 length: 1481]
      [Reassembled IPv4 data: 0800e5d5000b0001c87e376500000000fc160900000000000101112131415161718191a1b…]
```

Overall, given an MTU, if sending ICMP messages, fragmentation occurs whenever the ICMP payload size is greater than the:

MTU – len (ICMP header) – len (IP header)

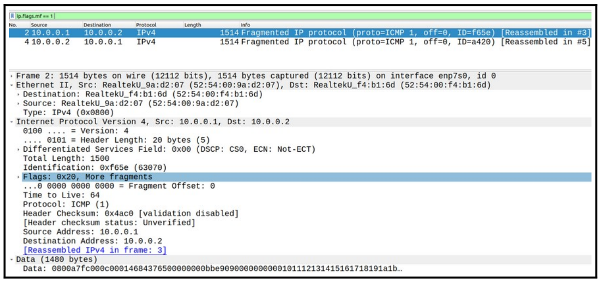
3. IP header fragmentation fields

The IP header contains vital information that guides the process of fragmentation and reassembly. Among its key fields, the **Total Length** (TL) and the **Fragment Offset** (FO) determine the size and positioning of each fragment. Additionally, the **More Fragments** (MF) flag communicates whether a packet has been fragmented or if further fragments are expected.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Version|  IHL  |Type of Service|          Total Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Identification        |Flags|      Fragment Offset    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Time to Live |    Protocol   |         Header Checksum       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       Source Address                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Destination Address                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Options                    |    Padding    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

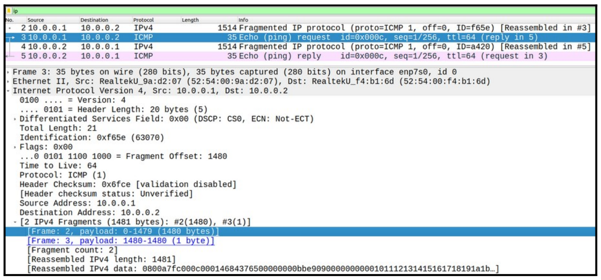
When an IP packet is fragmented, each fragment has the same **Identification**. Conversely, the reassembly process, i.e. the complementary process in the traffic *receiver*, would not able to say which IP packet this fragment belongs to. The More Fragment bit is set to zero, if and if only this is the last fragment, while the rest of fragments have the bit set to one. Finally, the Offset field serves as an index to place the fragment in the proper position in the original packet. In case of the first fragment, this is always equal to zero, while other fragments have something other than zero.

The first ICMP echo-request fragment:



```
ip.flags.mf == 1
No.  Source     Destination  Protocol  Length  Info
  2  10.0.0.1   10.0.0.2     IPv4        1514  Fragmented IP protocol (proto=ICMP 1, off=0, ID=f65e) [Reassembled in #3]
  4  10.0.0.2   10.0.0.1     IPv4        1514  Fragmented IP protocol (proto=ICMP 1, off=0, ID=a420) [Reassembled in #5]

› Frame 2: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp7s0, id 0
⌄ Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Destination: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
  › Source: RealtekU_9a:d2:07 (52:54:00:9a:d2:07)
    Type: IPv4 (0x0800)
⌄ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0xf65e (63070)
  › Flags: 0x20, More fragments
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x4ac0 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.0.2
    [Reassembled IPv4 in frame: 3]
⌄ Data (1480 bytes)
    Data: 0800a7fc000c000146843765000000000bbe9090000000000001011121314151617181 91a1b…
```

The second ICMP echo-request fragment:



```
ip
No.  Source     Destination  Protocol  Length  Info
  2  10.0.0.1   10.0.0.2     IPv4        1514  Fragmented IP protocol (proto=ICMP 1, off=0, ID=f65e) [Reassembled in #3]
  3  10.0.0.1   10.0.0.2     ICMP          35  Echo (ping) request  id=0x000c, seq=1/256, ttl=64 (reply in 5)
  4  10.0.0.2   10.0.0.1     IPv4        1514  Fragmented IP protocol (proto=ICMP 1, off=0, ID=a420) [Reassembled in #5]
  5  10.0.0.2   10.0.0.1     ICMP          35  Echo (ping) reply    id=0x000c, seq=1/256, ttl=64 (request in 3)

› Frame 3: 35 bytes on wire (280 bits), 35 bytes captured (280 bits) on interface enp7s0, id 0
› Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
⌄ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 21
    Identification: 0xf65e (63070)
  › Flags: 0x00
    ...0 0101 1100 1000 = Fragment Offset: 1480
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x6fce [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.0.2
  ⌄ [2 IPv4 Fragments (1481 bytes): #2(1480), #3(1)]
      [Frame: 2, payload: 0-1479 (1480 bytes)]
      [Frame: 3, payload: 1480-1480 (1 byte)]
      [Fragment count: 2]
      [Reassembled IPv4 length: 1481]
      [Reassembled IPv4 data: 0800a7fc000c000146843765000000000bbe909000000000000101112131415161718191a1b…]
```

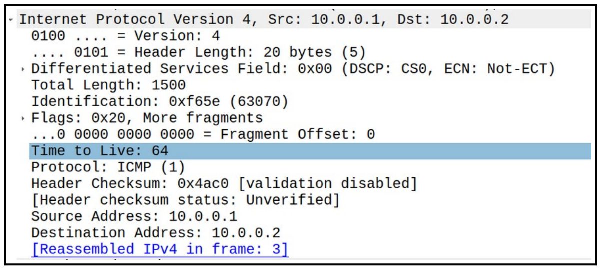
As expected, they share the same Identification, i.e. 0xF65E (63070). The first fragment has More Fragment bit set to one, while Fragment Offset is other than zero. Conversely, the last fragment has Fragment bit set to zero, while the Fragment Offset is other than zero.

3. Reassembly process

The reassembly process occurs in the receiver only. So, if having the two hosts are in a different networks, the router in between will never reassemble the fragments the sender is transferring. At high level, **the overall process relies on a buffer and a timer**; the buffer is used to stage the incoming fragments before sending them all to the upper layer, while the timer is used to avoid the process waiting endlessly for incoming fragments. Since all the fragments share the same Identification, the buffer is also identified by means a key. According to the official RFC, the buffer is key is nothing but the source and destination addresses, the protocol, and the Identification fields computed as follows: `BUFID = src|dst|pto|idf`. As a result, the overall process can be easily translated in pseudo-code:

```
// Compute the BUFID
BUFID = src|dst|pto|idf

// Is this a fragment?
IF OFFSET = 0 AND MF = 0: // No, it is not
      // This a packet as whole
      IF BUFID already exists:
               free the buffer with BUFID as key
      send the packet to the upper-layer
      DONE;

ELSE // Yes it is
      IF BUFID not already exists:
               have a new buffer with BUFID as key
      set timer
      // Use FO * 8 as start-point, use
      // and FO + ((TL - (IHL * 4)) + FO * 8
      // to access to the end-point
      copy the fragment payload into the buffer

      // When receiving a fragment, the receiver
      // should track which fragments have been
      // already received
      set the received fragments bit map (RCVFB)
      // If the last fragment
      IF MF = 0:
         compute the overall length
      // If the first fragment
      IF FO = 0:
         copy the IP header in a separated buffer

      IF overall length > 0 AND all bits in the
        RCVFB are set to 1 from 0 to (TDL + 7) / 8:

               free the buffer with BUFID as key
               send the packet to the upper-layer
               DONE;
give up until the next fragment or time is off
```

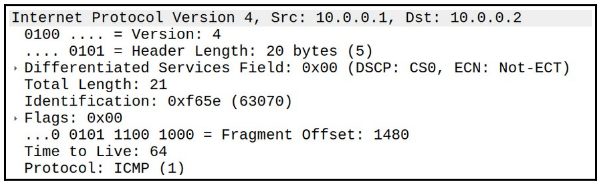
Notice that the Fragment Offset is a 13 bit-long field, while the overall size of an IP packet is measured in 16 bit. As a result, the real Fragment Offset must always be multiplied by a factor of 8 (Wireshark does it on our behalf). The receiver uses the Fragment Offset as an index to get access the proper location in the buffer; this acts as a C-like pointer in a sort of C-like memcpy() function. As a consequence, until the receiver does not have the last fragment on hand, it can not say how many fragments, or rather how much space is missing in the buffer to be completed. Let's say the receiver receives the first fragment, as follows.



```
Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0xf65e (63070)
  Flags: 0x20, More fragments
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 64
  Protocol: ICMP (1)
  Header Checksum: 0x4ac0 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.0.1
  Destination Address: 10.0.0.2
  [Reassembled IPv4 in frame: 3]
```

The IP header tells the receiver this is 1500 bytes-long packet, but because of the More Fragment bit this is not the packet as whole. Until the receiver does not have the last fragment, the receiver believes it has just just 1500 out of the theoretical 65546 bytes, i.e. roughly the 2.2%.

Yet, when receiving the last (second) fragment, the receiver can compute how big was the original packet.



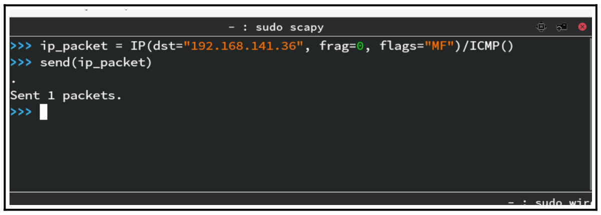
```
Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 21
  Identification: 0xf65e (63070)
  Flags: 0x00
  ...0 0101 1100 1000 = Fragment Offset: 1480
  Time to Live: 64
  Protocol: ICMP (1)
```

If this fragment is 21 bytes-long, therefore the overall size of the original packet was 20 bytes (original IP header size) + 1 byte (last IP fragment payload size) + 1480 bytes (cumulative fragments payload size), i.e. 1501 bytes. If the receiver had received 1500 bytes, it means that it had actually received the 99% of the overall packet size. After receiving the last fragment, the buffer is actually full-filled.

In the end, **if and if only the receiver has the last fragment, it can compute the overall size of the original packet, and therefore exits successfully in case of buffer completed**. However, if the timer goes off, the process exits anyway with an error, that is further translated into ICMP Time Exceeded. In that case, all the fragments are discarded, and the overall packet is considered to be lost.

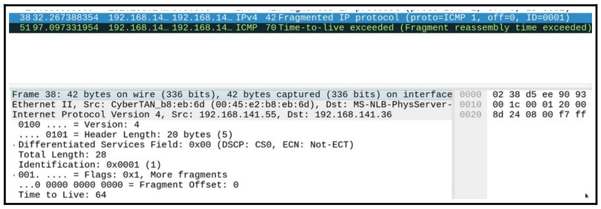
4. IP fragmentation attack

If the attacker never sends the last fragment, the receiver awaits until the timer expires before evicting the buffer from the memory; **if the attacker is fast enough to force the receivers to allocate more buffer than the receiver removes from memory, then the attacker can carry out a DoS attack**. An attacker can forge IP packets of arbitrary size, even just a few bytes, but use the More Fragment field to invite the receiver to create a buffer in which to accommodate subsequent fragments; however, the attacker will never send the other fragments.



```
>>> ip_packet = IP(dst="192.168.141.36", frag=0, flags="MF")/ICMP()
>>> send(ip_packet)
.
Sent 1 packets.
>>> 
```

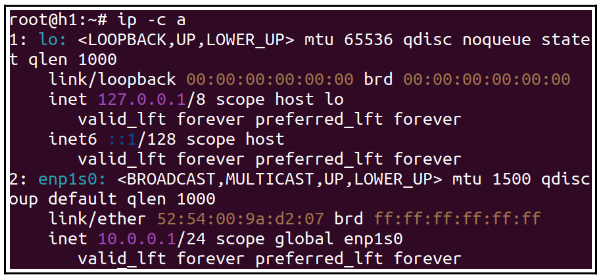
After a while, the receiver tells the sender the buffer has been flushed away, and therefore the theoretical IP packet is lost. In between, the attacker can flood the receiver with multiple fake fragments.



```
38 32.267388354  192.168.14… 192.168.14… IPv4  42 Fragmented IP protocol (proto=ICMP 1, off=0, ID=0001)
51 97.097331954  192.168.14… 192.168.14… ICMP  70 Time-to-live exceeded (Fragment reassembly time exceeded)

Frame 38: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface   0000  02 38 d5 ee 90 93
Ethernet II, Src: CyberTAN_b8:eb:6d (00:45:e2:b8:eb:6d), Dst: MS-NLB-PhysServer-   0010  00 1c 00 01 20 00
Internet Protocol Version 4, Src: 192.168.141.55, Dst: 192.168.141.36             0020  8d 24 08 00 f7 ff
 0100 .... = Version: 4
 .... 0101 = Header Length: 20 bytes (5)
 Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
 Total Length: 28
 Identification: 0x0001 (1)
 001. .... = Flags: 0x1, More fragments
 ...0 0000 0000 0000 = Fragment Offset: 0
 Time to Live: 64
```

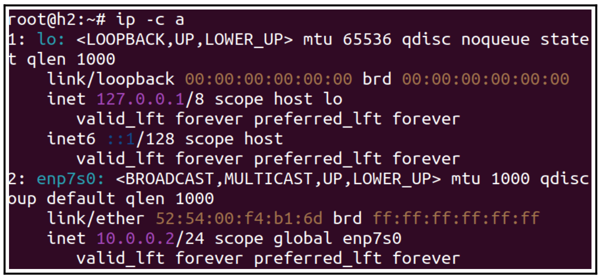
5. Different MTUs on the path

Host H1 has the following configuration, with an MTU of 1500 bytes:



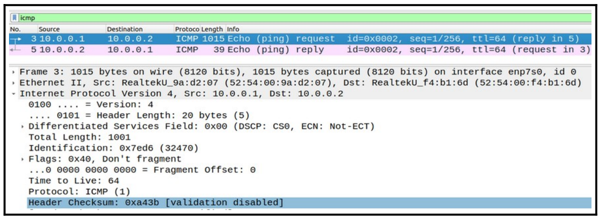
```
root@h1:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
oup default qlen 1000
    link/ether 52:54:00:9a:d2:07 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.1/24 scope global enp1s0
       valid_lft forever preferred_lft forever
```

Host H2 has the following configuration, with an MTU of 1000 bytes:



```
root@h2:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1000 qdisc
oup default qlen 1000
    link/ether 52:54:00:f4:b1:6d brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.2/24 scope global enp7s0
       valid_lft forever preferred_lft forever
```

If the host H1 sends an ICMP packet with more than 972 bytes as payload, it tops the H2's MTU. However, since fragmentation only happens at the sender-side, we expect to see the ICMP packet as whole.

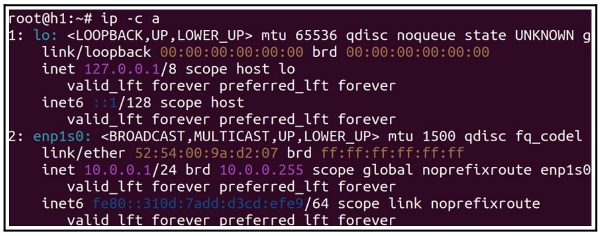


```
icmp
No.  Source     Destination  Protoco Length Info
  3 10.0.0.1   10.0.0.2     ICMP 1015 Echo (ping) request  id=0x0002, seq=1/256, ttl=64 (reply in 5)
  5 10.0.0.2   10.0.0.1     ICMP   39 Echo (ping) reply    id=0x0002, seq=1/256, ttl=64 (request in 3)

› Frame 3: 1015 bytes on wire (8120 bits), 1015 bytes captured (8120 bits) on interface enp7s0, id 0
› Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1001
    Identification: 0x7ed6 (32470)
  › Flags: 0x40, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0xa43b [validation disabled]
```

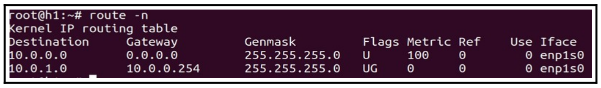
As expected, the IP packet which is delivering the ICMP echo-request is not fragmented at all. The overall size is 1001 bytes, which is 1 byte more than the MTU of H2.

If H1 and H2 are in two different networks, there should be a router in between to let them talk each other. As mentioned before, the fragmentation process happens at sender-side only; however, when forwarding a packet to a router, this one acts as a sender. As a result, an IP packet may be fragmented multiple times along the path, since the routers in between connects either two networks using either different technologies, e.g. Ethernet and Token Ring, or two networks with the same technology but with a different set of devices and appliances, e.g. a traditional Ethernet network with an MTU of 1500 bytes and an Ethernet network with jumbo frames, e.g. in a data-center.

Host H1 belongs to the 10.0.0.0/24 network and it has the following configuration, with an MTU of 1500 bytes:

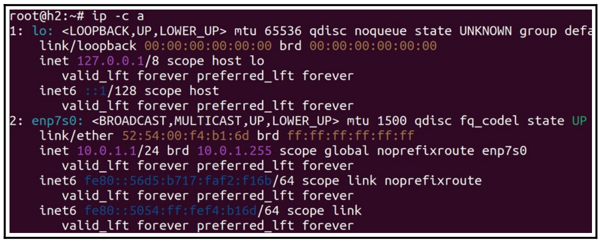


```
root@h1:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN g
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
    link/ether 52:54:00:9a:d2:07 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.1/24 brd 10.0.0.255 scope global noprefixroute enp1s0
       valid_lft forever preferred_lft forever
    inet6 fe80::310d:7add:d3cd:efe9/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

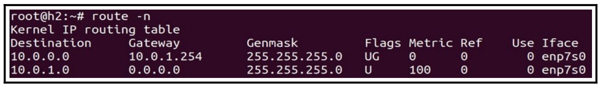


```
root@h1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 enp1s0
10.0.1.0        10.0.0.254      255.255.255.0   UG    0      0        0 enp1s0
```

Host H2 belongs to the 10.0.1.0/24 network and it has the following configuration, with an MTU of 1500 bytes:

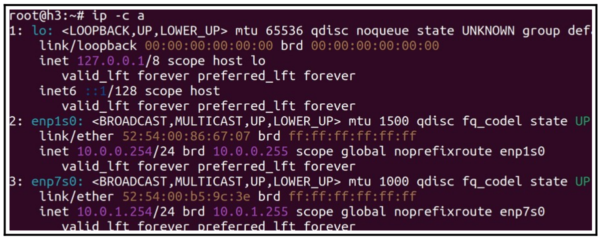


```
root@h2:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defa
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    link/ether 52:54:00:f4:b1:6d brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.1/24 brd 10.0.1.255 scope global noprefixroute enp7s0
       valid_lft forever preferred_lft forever
    inet6 fe80::56d5:b717:faf2:f16b/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fef4:b16d/64 scope link
       valid_lft forever preferred_lft forever
```



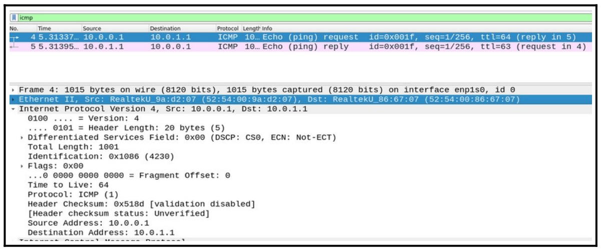
```
root@h2:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        10.0.1.254      255.255.255.0   UG    0      0        0 enp7s0
10.0.1.0        0.0.0.0         255.255.255.0   U     100    0        0 enp7s0
```

H3 is a dual-homed host, and it plays as a router, with an MTU of 1500 bytes for the network 10.0.0.0/24 and an MTU of 1000 bytes for the network 10.0.1.0/24:



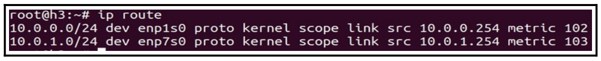
```
root@h3:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    link/ether 52:54:00:86:67:07 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.254/24 brd 10.0.0.255 scope global noprefixroute enp1s0
       valid_lft forever preferred_lft forever
3: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1000 qdisc fq_codel state UP
    link/ether 52:54:00:b5:9c:3e brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.254/24 brd 10.0.1.255 scope global noprefixroute enp7s0
       valid_lft forever preferred_lft forever
```

The router (H3) receives an IP packet of 1001 bytes from the interface *enp1s0* (52:54:00:86:67:07).



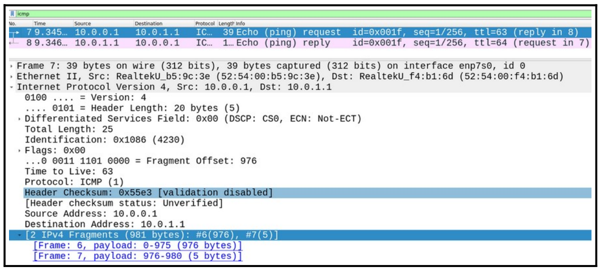
```
icmp
No.  Time        Source     Destination  Protocol Length Info
  4 5.31337…   10.0.0.1   10.0.1.1     ICMP  10… Echo (ping) request  id=0x001f, seq=1/256, ttl=64 (reply in 5)
  5 5.31395…   10.0.1.1   10.0.0.1     ICMP  10… Echo (ping) reply    id=0x001f, seq=1/256, ttl=63 (request in 4)

› Frame 4: 1015 bytes on wire (8120 bits), 1015 bytes captured (8120 bits) on interface enp1s0, id 0
› Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_86:67:07 (52:54:00:86:67:07)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1001
    Identification: 0x1086 (4230)
  › Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x518d [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.1.1
```

According to the routing table, the packet is supposed to exit the router from the interface *enp7s0*; however, since the MTU is 1000 bytes on that link, the router has to break up the packet.



```
root@h3:~# ip route
10.0.0.0/24 dev enp1s0 proto kernel scope link src 10.0.0.254 metric 102
10.0.1.0/24 dev enp7s0 proto kernel scope link src 10.0.1.254 metric 103
```

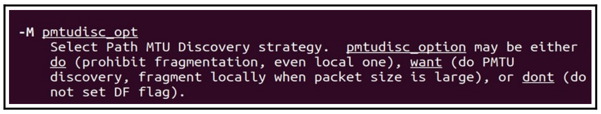
As expected, the out-coming packet is fragmented because of the router output interface MTU.



```
icmp
No.  Time      Source     Destination Protocol Length Info
  7 9.345…    10.0.0.1   10.0.1.1    IC…      39 Echo (ping) request  id=0x001f, seq=1/256, ttl=63 (reply in 8)
  8 9.346…    10.0.1.1   10.0.0.1    IC…      1… Echo (ping) reply    id=0x001f, seq=1/256, ttl=64 (request in 7)

› Frame 7: 39 bytes on wire (312 bits), 39 bytes captured (312 bits) on interface enp7s0, id 0
› Ethernet II, Src: RealtekU_b5:9c:3e (52:54:00:b5:9c:3e), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 25
    Identification: 0x1086 (4230)
  › Flags: 0x00
    ...0 0011 1101 0000 = Fragment Offset: 976
    Time to Live: 63
    Protocol: ICMP (1)
    Header Checksum: 0x55e3 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.1.1
  ˅ [2 IPv4 Fragments (981 bytes): #6(976), #7(5)]
      [Frame: 6, payload: 0-975 (976 bytes)]
      [Frame: 7, payload: 976-980 (5 bytes)]
```

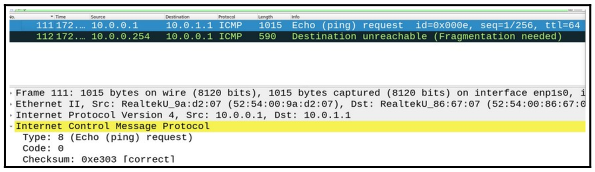
6. Path MTU discovery

IP fragmentation can cause performance and latency issues when fragments are affected by packet loss. As mentioned before, even if the receiver does non have a fragment on hand, it must discard all the other ones, causing a severe loss. As a result, if the sender could know in advance which network interface, i.e. to be traversed, has the lowest MTU, this could generate smaller, minimum MTU-compliant packets between itself and its recipient. The RFC 1191 defines ***Path MTU discovery***, a simple process through which a host can detect a path MTU smaller than its interface MTU. Two components are key to this process: a) the *Don't Fragment (DF)* bit of the IP header; b) the inner code of the ICMP *Destination Unreachable* message, *Fragmentation Needed*. Setting the DF bit in an IP packet avoids a router from performing fragmentation when it encounters an MTU less than the packet size. Instead, the packet is discarded and an ICMP Fragmentation Needed message is sent to the originating host. Essentially, the router is indicating that it needs to fragment the packet but the DF flag won't allow for it.



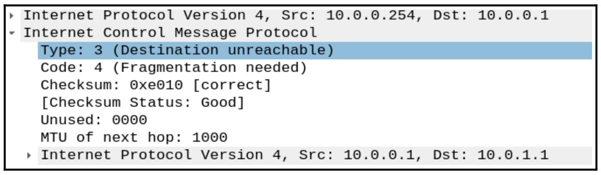
```
-M pmtudisc_opt
    Select Path MTU Discovery strategy.  pmtudisc_option may be either
    do (prohibit fragmentation, even local one), want (do PMTU
    discovery, fragment locally when packet size is large), or dont (do
    not set DF flag).
```

As expected, if telling *ping* command to have **-M do**, the sender will set the DF bit to one: this packet can not be fragmented by anyone, including the sender itself. As a result, the router returns an ICMP Error message, i.e. Destination Unreachable, Fragmentation Needed.



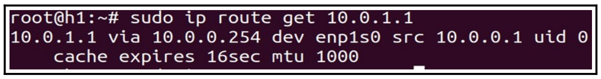
```
No.  Time    Source       Destination Protocol Length Info
 111 172.…   10.0.0.1     10.0.1.1    ICMP     1015   Echo (ping) request  id=0x000e, seq=1/256, ttl=64
 112 172.…   10.0.0.254   10.0.0.1    ICMP     590    Destination unreachable (Fragmentation needed)

› Frame 111: 1015 bytes on wire (8120 bits), 1015 bytes captured (8120 bits) on interface enp1s0, i
› Ethernet II, Src: RealtekU_9a:d2:07 (52:54:00:9a:d2:07), Dst: RealtekU_86:67:07 (52:54:00:86:67:0
› Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.1.1
˅ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xe303 [correct]
```

This router not only communicates that H2 is not reachable but also the MTU which, in fact, denied the delivery of the traffic. In this way, H1 learns the maximally accepted MTU to send traffic to H2.



```
› Internet Protocol Version 4, Src: 10.0.0.254, Dst: 10.0.0.1
˅ Internet Control Message Protocol
    Type: 3 (Destination unreachable)
    Code: 4 (Fragmentation needed)
    Checksum: 0xe010 [correct]
    [Checksum Status: Good]
    Unused: 0000
    MTU of next hop: 1000
  › Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.1.1
```

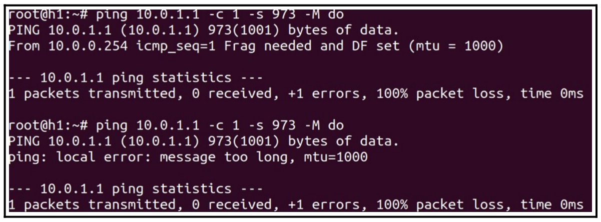
The maximum MTU accepted along the path is therefore cached by H1.



```
root@h1:~# sudo ip route get 10.0.1.1
10.0.1.1 via 10.0.0.254 dev enp1s0 src 10.0.0.1 uid 0
    cache expires 16sec mtu 1000
```

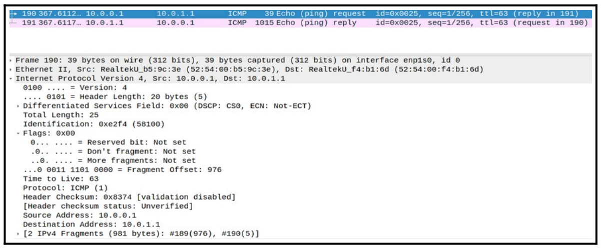
As long as that information is stored in memory, if a running applications generates traffic for H2 which however exceeds the MTU reported by the router, H1 will automatically proceed to fragmentation; conversely, if fragmentation is forbidden explicitly, H1 will not send traffic at all.

As expected, if H1 sends a too much large ICMP message which can not be fragmented (*-M do*), then the router tells H1 there is an MTU of 1000 bytes on the output interface, and therefore the packet can not be delivered successfully. If H1 sends again the same packet, no packet exit the NIC because of a local error, i.e. an over-sized packet which can not be fragmented.



```
root@h1:~# ping 10.0.1.1 -c 1 -s 973 -M do
PING 10.0.1.1 (10.0.1.1) 973(1001) bytes of data.
From 10.0.0.254 icmp_seq=1 Frag needed and DF set (mtu = 1000)

--- 10.0.1.1 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

root@h1:~# ping 10.0.1.1 -c 1 -s 973 -M do
PING 10.0.1.1 (10.0.1.1) 973(1001) bytes of data.
ping: local error: message too long, mtu=1000

--- 10.0.1.1 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

As expected, on the other hand, if the option is *-M dont*, H1 enables fragmentation and because of the previous router hint, H1 manages the fragmentation before placing the packet on the wire.



```
 190 367.6112… 10.0.0.1    10.0.1.1    ICMP    39 Echo (ping) request  id=0x0025, seq=1/256, ttl=63 (reply in 191)
 191 367.6117… 10.0.1.1    10.0.0.1    ICMP  1015 Echo (ping) reply    id=0x0025, seq=1/256, ttl=63 (request in 190)

› Frame 190: 39 bytes on wire (312 bits), 39 bytes captured (312 bits) on interface enp1s0, id 0
› Ethernet II, Src: RealtekU_b5:9c:3e (52:54:00:b5:9c:3e), Dst: RealtekU_f4:b1:6d (52:54:00:f4:b1:6d)
˅ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 25
    Identification: 0xe2f4 (58100)
  ˅ Flags: 0x00
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0011 1101 0000 = Fragment Offset: 976
    Time to Live: 63
    Protocol: ICMP (1)
    Header Checksum: 0x8374 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 10.0.1.1
  › [2 IPv4 Fragments (981 bytes): #189(976), #190(5)]
```

7. References

RFC 791, IP specifications, September 1981,
https://datatracker.ietf.org/doc/html/rfc791

Linux Documentation,
https://docs.kernel.org/networking/ip-sysctl.html