CS 305 Lab Tutorial
Lab11 IPv4 & ICMPv4

Dept. Computer Science and Engineering
Southern University of Science and Technology

SUSTech
Southern University
of Science and Technology

Thanks to Wei WANG

Topic

- IPv4
  - Best effort, IP address, IP fragment and assemble
- ICMPv4
  - Detect and report
- IPv6
  - The difference between IPv4 and IPv6

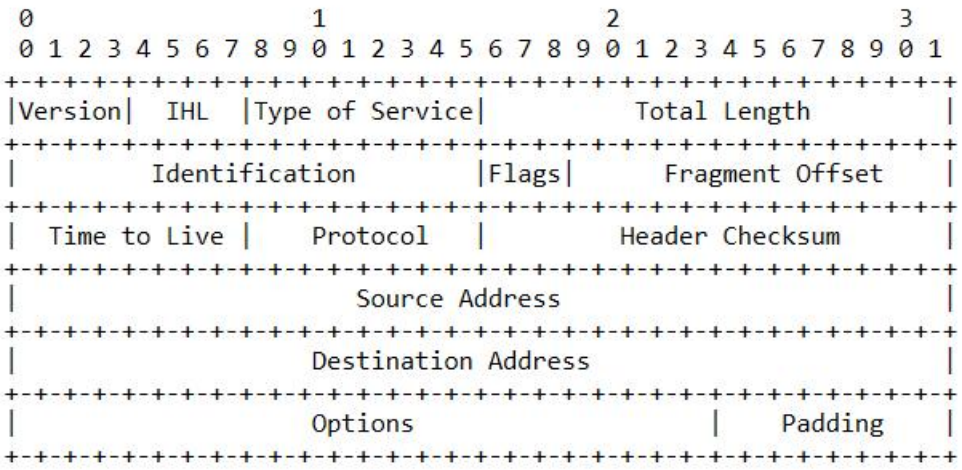
Part A. IPv4

- **Best effort** : NO connection, NO flow control, NO congestion control, NO retransmission...

- The internet protocol implements two basic functions: **addressing** and **fragmentation.**
  - The internet modules use the addresses carried in the internet header to transmit internet datagrams toward their destinations. The selection of a path for transmission is called **routing**.
  - The internet modules use fields in the internet header to **fragment** and **reassemble** internet datagrams when necessary for transmission through "small packet" networks. The model of operation is that an internet module resides in each host engaged in internet communication and in each gateway that interconnects networks.

https://tools.ietf.org/html/rfc791

IPv4 Datagram



Example Internet Datagram Header

- **Type of Service:**

The major choice is a three way tradeoff between low-delay, high-reliability, and high-throughput.

- **Time to Live (TTL):**

an indication of an upper bound on the lifetime of an internet datagram. **It is set by the sender of the datagram and reduced at the points along the route where it is processed.** An IP datagram with zero TTL will be dropped.

- **Header Checksum:**

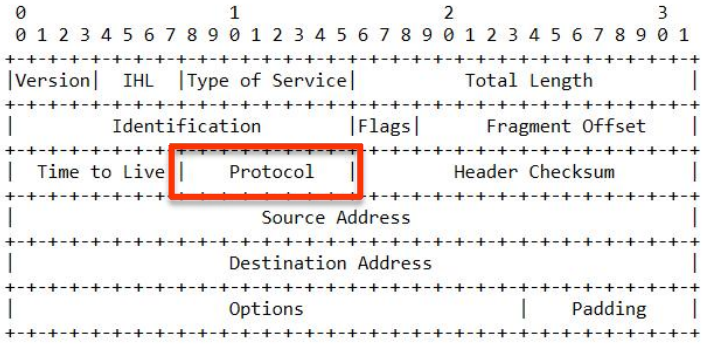
provides a verification that the information used in processing internet datagram has been transmitted correctly. The data may contain errors. If the header checksum fails, the internet datagram is discarded at once by the entity which detects the error.

- **Options:**

provide for control functions needed or useful in some situations but unnecessary for the most common communications. The options include provisions for timestamps, security, and special routing.

https://tools.ietf.org/html/rfc791

Protocol Field



Example Internet Datagram Header

```
Internet Protocol Version 4, Src: 192.168.2.104 (192.168.2.104), Dst: t
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 52
    Identification: 0x05ec (1516)
  > Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x0fda [validat
    [Header checksum status: Unverif
    Source: 192.168.2.104 (192.168.2
    Destination: tg-in-f113.1e100.ne
> Transmission Control Protocol, Src
```

```
Internet Protocol Version 4, Src: tw.net-east.com (116.77.76.254), Dst
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 128
    Identification: 0x311d (12573)
  > Flags: 0x0000
    Time to live: 57
    Protocol: UDP (17)
    Header checksum: 0xcbf4 [va
    [Header checksum status: Un
    Source: tw.net-east.com (11
    Destination: 192.168.2.104
> User Datagram Protocol, Src Pc
> Domain Name System (response)
```

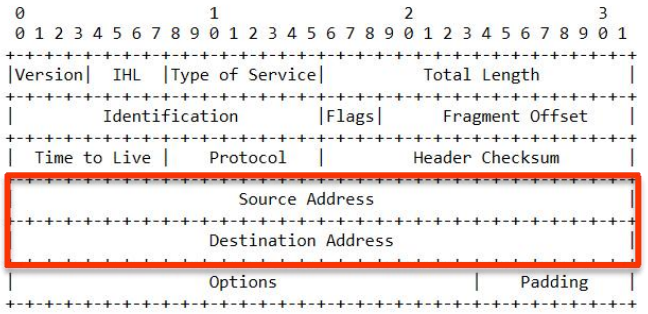
```
Internet Protocol Version 4, Src: 192.168.2.104 (192.168.2.104), Dst
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1020
    Identification: 0x0a9a (2714)
  > Flags: 0x00b9
    Time to live: 6
    Protocol: ICMP (1)
    Header checksum: 0x8493 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.2.104 (192.168.2.104)
    Destination: 116.7.234.3 (116.7.234.3)
  > [2 IPv4 Fragments (2480 bytes): #1(1480), #2(1000)]
v Internet Control Message Protocol
```

SUSTech
Southern University
of Science and Technology

Source and Destination Field

```
> Frame 4: 216 bytes on wire (1728 bits), 216 bytes captured (1728 bits) on interface 0
> Ethernet II, Src: IntelCor_5c:69:58 (90:61:ae:5c:69:58), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
v Internet Protocol Version 4, Src: 192.168.2.104 (192.168.2.104), Dst: 239.255.255.250 (239.255.255.250)
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 202
    Identification: 0x7437 (29751)
  > Flags: 0x0000
    Time to live: 1
    Protocol: UDP (17)
    Header checksum: 0x91e1 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.2.104 (192.168.2.104)
    Destination: 239.255.255.250 (239.255.255.250)
> User Datagram Protocol, Src Port: 58806 (58806), Dst Port: ssdp (1900)
> Simple Service Discovery Protocol
```

```
v Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 328
    Identification: 0xb310 (45840)
  > Flags: 0x0000
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x8695 [validation disabled]
    [Header checksum status: Unverified]
    Source: 0.0.0.0 (0.0.0.0)
    Destination: 255.255.255.255 (255.255.255.255)
```

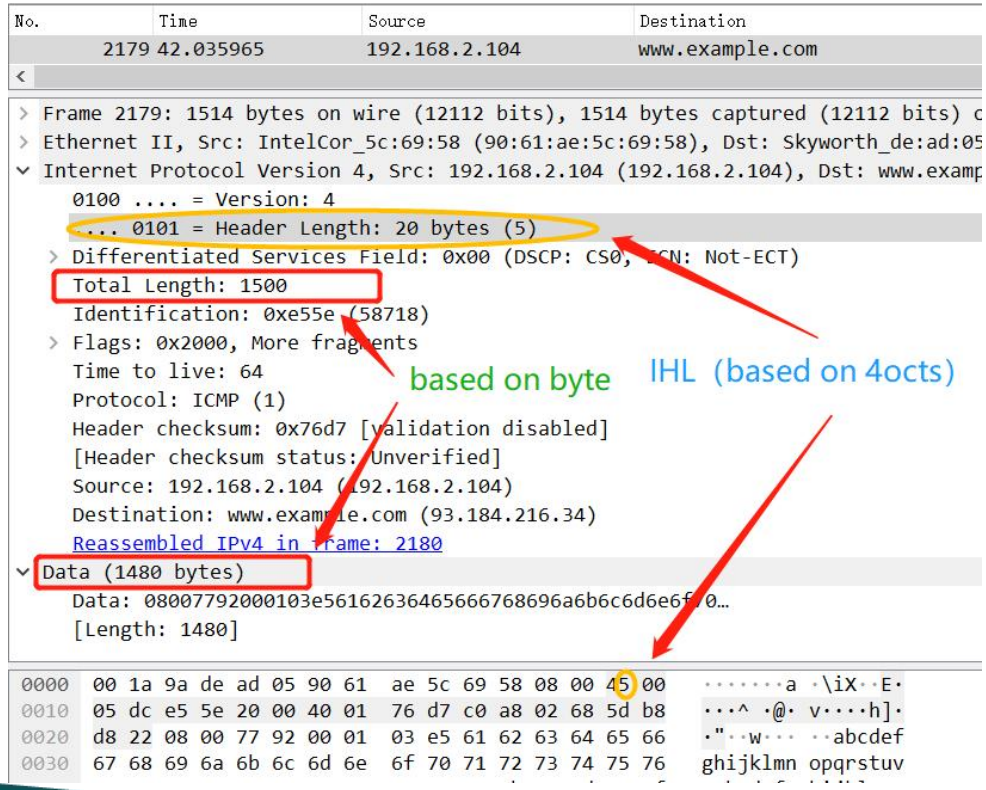


Example Internet Datagram Header

SUSTech
Southern University of Science and Technology

IHL and Total Length

Initial the session with following cmd command: ping www.example.com –l 2000

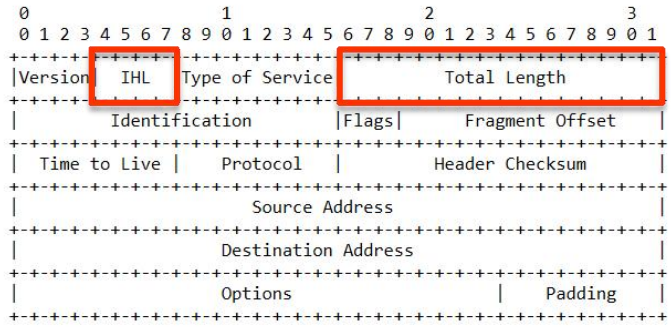


IHL: 4 bits

Internet **H**eader **L**ength is the length of the internet header in 32 bit words, and thus points to the beginning of the data. Note that the minimum value for a correct header is 5.

Total Length: 16 bits

the length of the datagram, measured in octets, including internet header and data.



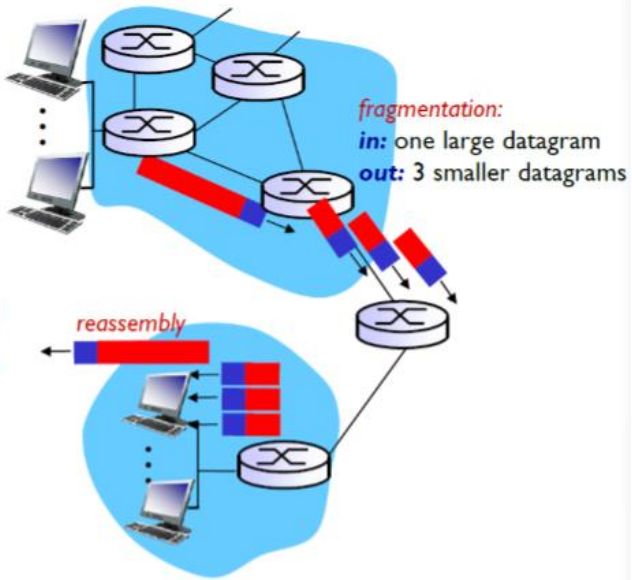
Example Internet Datagram Header

SUSTech
Southern University of Science and Technology

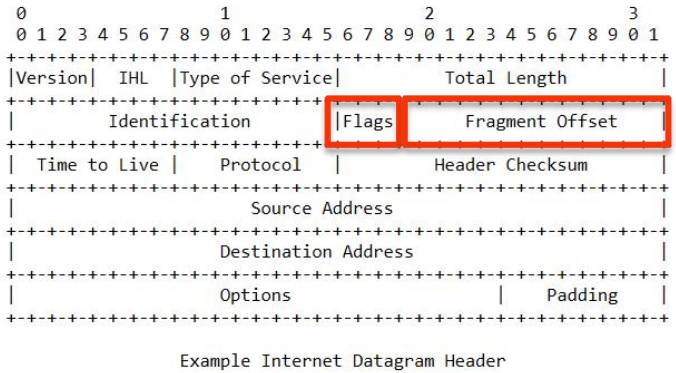
IP Fragmentation and Reassembly

IP fragmentation, reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP Fragment(1)

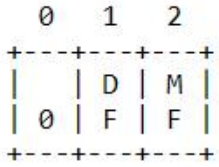


Example Internet Datagram Header

Flags: 3 bits

Various Control Flags.

Bit 0: reserved, must be zero
Bit 1: (DF) 0 = May Fragment, 1 = Don't Fragment.
Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.



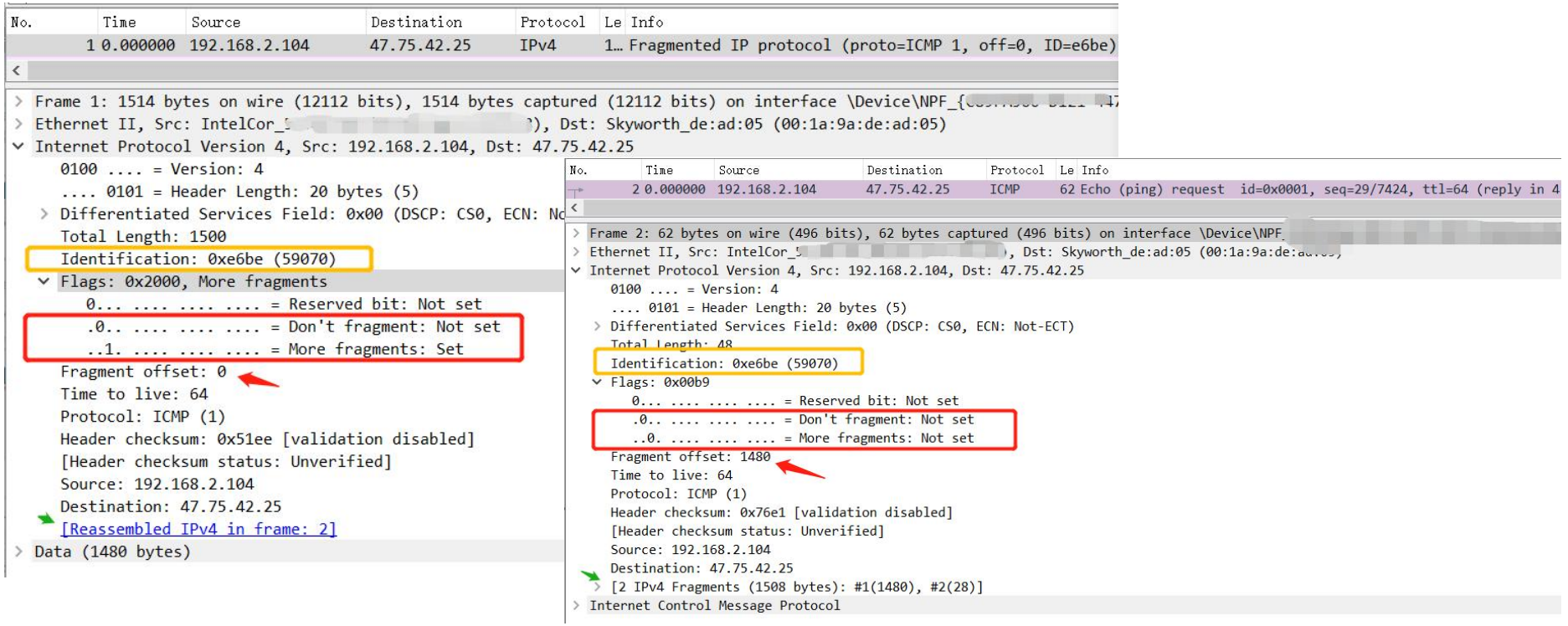

```
Internet Protocol Version 4, Src: 192.168.2.104 (192.168.2.104), Dst: 116.7.234.3 (116.7.234.3)
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1020
  Identification: 0x0a9c (2716)
  Flags: 0x00b9
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 1011 1001 = Fragment offset: 185
```

Fragment Offset: 13 bits
This field indicates where in the datagram this fragment belongs.
The fragment offset is measured **in units of 8 octets** (64 bits). The first fragment has offset zero.

Tips in Wireshark : ip.flags.mf

IP Fragment(2)

Initial the session with following cmd command: ping www.example.cn –l _?_



Identification: An internet header field carrying the identifying value assigned by the sender to aid in assembling the fragments of a datagram.

Tips in Wireshark : ip.id

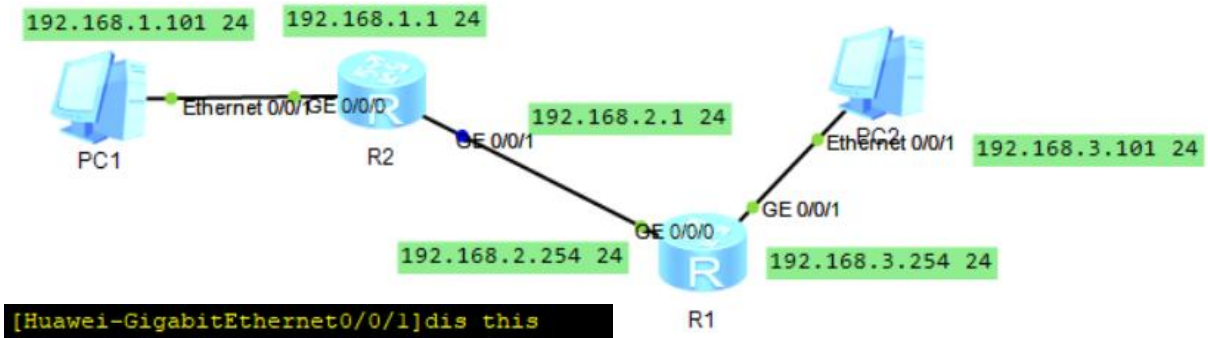
Demo1 - IP fragment(1)

Build the network as the topology shown on the right hand, after the configuration, PC1 could send/receive the pkt to/from PC2.

Tips: in eNSP, router support IP fragment while PC don't.

Set the the MTU of R2's interface GE0/0/1 as **50**.

Invoke "ping" test on R2, and capture the packet ont its interface GE 0/0/1




```
[Huawei-GigabitEthernet0/0/1]dis this
#
interface GigabitEthernet0/0/1
 mtu 50
 ip address 192.168.2.1 255.255.255.0
#
return
```

R2

```
[Huawei]ping 192.168.3.254
  PING 192.168.3.254: 56  data bytes, press CTRL_C to break
    Reply from 192.168.3.254: bytes=56 Sequence=1 ttl=255 time=60 ms
    Reply from 192.168.3.254: bytes=56 Sequence=2 ttl=255 time=30 ms
    Reply from 192.168.3.254: bytes=56 Sequence=3 ttl=255 time=10 ms
    Reply from 192.168.3.254: bytes=56 Sequence=4 ttl=255 time=50 ms
    Reply from 192.168.3.254: bytes=56 Sequence=5 ttl=255 time=40 ms
```

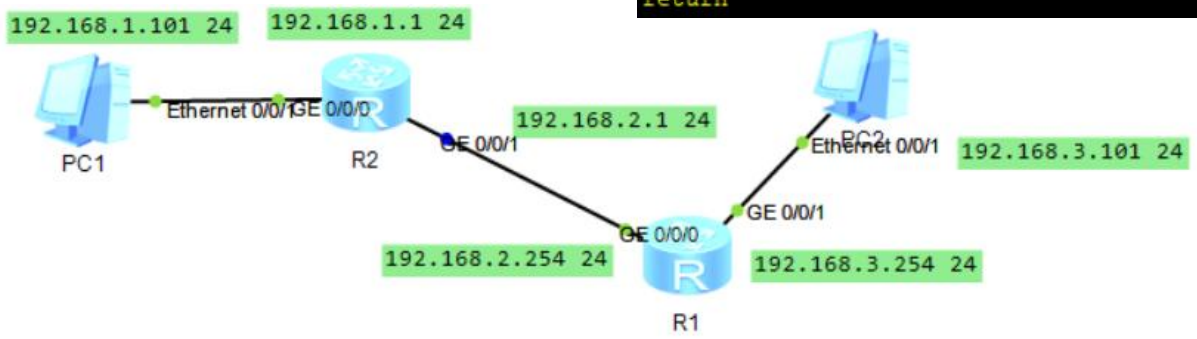
Tips: in eNSP, route support IP fragment while PC don't.

Demo1 - IP fragment(2)

```
[Huawei-GigabitEthernet0/0/1]dis this
#
interface GigabitEthernet0/0/1
 mtu 50
 ip address 192.168.2.1 255.255.255.0
#
return
```

The MTU of R2's interface GE0/0/1 is set as 50.

Invoke "ping" test on R2, and capture the packet ont its interface GE 0/0/1



	Time	Source	Protocol	Destination	Length	Info
1	0.000000	192.168.2.1	IPv4	192.168.3.254	60	Fragmented IP protocol (proto=ICMP 1, off=0, ID=001c) [Reassembled in #3]
2	0.000000	192.168.2.1	IPv4	192.168.3.254	60	Fragmented IP protocol (proto=ICMP 1, off=24, ID=001c) [Reassembled in #3]
3	0.000000	192.168.2.1	ICMP	192.168.3.254	60	Echo (ping) request id=0xd0ab, seq=256/1, ttl=255 (reply in 4)
4	0.015000	192.168.3.254	ICMP	192.168.2.1	98	Echo (ping) reply id=0xd0ab, seq=256/1, ttl=255 (request in 3)

```
[3 IPv4 Fragments (64 bytes): #1(24), #2(24), #3(16)]
   [Frame: 1, payload: 0-23 (24 bytes)]
   [Frame: 2, payload: 24-47 (24 bytes)]
   [Frame: 3, payload: 48-63 (16 bytes)]
   [Fragment count: 3]
   [Reassembled IPv4 length: 64]
   [Reassembled IPv4 data: 080075d9d0ab01007e3808000000
```

Q1: As shown in the packet 1-4, only the packet send by 192.168.2.1 is fraged, the packet it received is not, why?

Q2： How does the Wireshak identify the order of each fragment in an IP packet ?

Demo1 - IP fragment(3)

	Time	Source	Protocol	Destination	Length	Info
1	0.000000	192.168.2.1	IPv4	192.168.3.254	60	Fragmented IP protocol (proto=ICMP 1, off=0, ID=001c) [Reassembled in #3]
2	0.000000	192.168.2.1	IPv4	192.168.3.254	60	Fragmented IP protocol (proto=ICMP 1, off=24, ID=001c) [Reassembled in #3]
3	0.000000	192.168.2.1	ICMP	192.168.3.254	60	Echo (ping) request id=0xd0ab, seq=256/1, ttl=255 (reply in 4)
4	0.015000	192.168.3.254	ICMP	192.168.2.1	98	Echo (ping) reply id=0xd0ab, seq=256/1, ttl=255 (request in 3)

```
[3 IPv4 Fragments (64 bytes): #1(24), #2(24), #3(16)]
    [Frame: 1, payload: 0-23 (24 bytes)]
    [Frame: 2, payload: 24-47 (24 bytes)]
    [Frame: 3, payload: 48-63 (16 bytes)]
    [Fragment count: 3]
    [Reassembled IPv4 length: 64]
    [Reassembled IPv4 data: 080075d9d0ab01007e38080000000
```

```
Internet Protocol Version 4, Src: 192.168.2.1, Dst: 192.168.3.254
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 44
    Identification: 0x001c (28)
  v 001. .... = Flags: 0x1, More fragments
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: ICMP (1)
    Header Checksum: 0x1465 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.2.1
    Destination Address: 192.168.3.254
    [Reassembled IPv4 in frame: 3]
v Data (24 bytes)
    Data: 080075d9d0ab01007e380800000000000000102030
    [Length: 24]
```

```
Internet Protocol Version 4, Src: 192.168.2.1, Dst: 192.168.3.254
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 44
    Identification: 0x001c (28)
  v 001. .... = Flags: 0x1, More fragments
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..1. .... = More fragments: Set
    ...0 0000 0000 0011 = Fragment Offset: 24
    Time to Live: 255
    Protocol: ICMP (1)
    Header Checksum: 0x1462 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.2.1
    Destination Address: 192.168.3.254
    [Reassembled IPv4 in frame: 3]
Data (24 bytes)
    Data: 08090a0b0c0d0e0f101112131415161718191a1b1c1d1
    [Length: 24]
```

```
Internet Protocol Version 4, Src: 192.168.2.1, Dst: 192.168.3.254
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 36
    Identification: 0x001c (28)
  v 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0110 = Fragment Offset: 48
```

Q2: How does the Wireshak identify the order of each fragment in an IP packet ?

A2: Flags in the IP header:

1) More flagments
0: not the last piece
1: the last piece

2) Fragment offset
0: the 1st piece
other number: the order of each fragment in an IP packet

SUSTech
Southern University of Science and Technology

Part B. ICMP

- ICMP is used from gateways to hosts and between hosts to report errors and make routing suggestions.

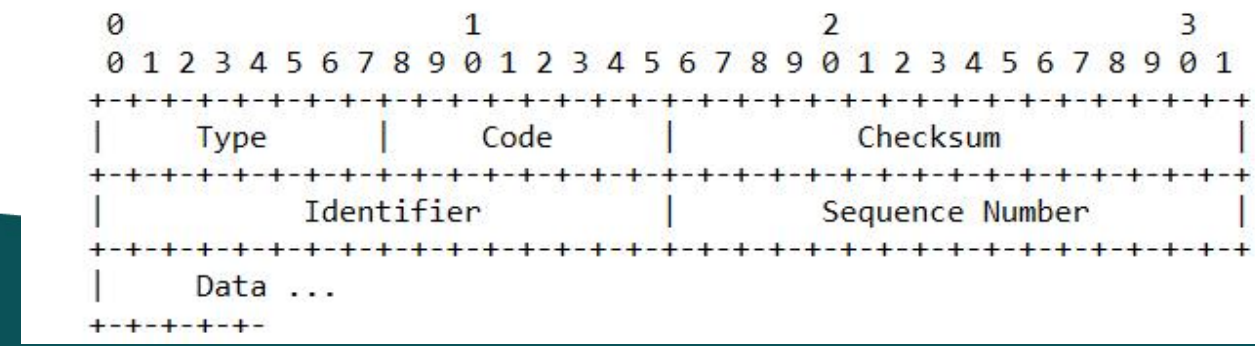
- ICMP and IP :
  - Internet protocol errors may be reported via the ICMP messages
  - ICMP uses the basic support of IP as if it were a higher level protocol, however, ICMP is actually an integral part of IP, and must be implemented by every IP module.

https://tools.ietf.org/html/rfc792

ICMP (Echo and Echo Reply)

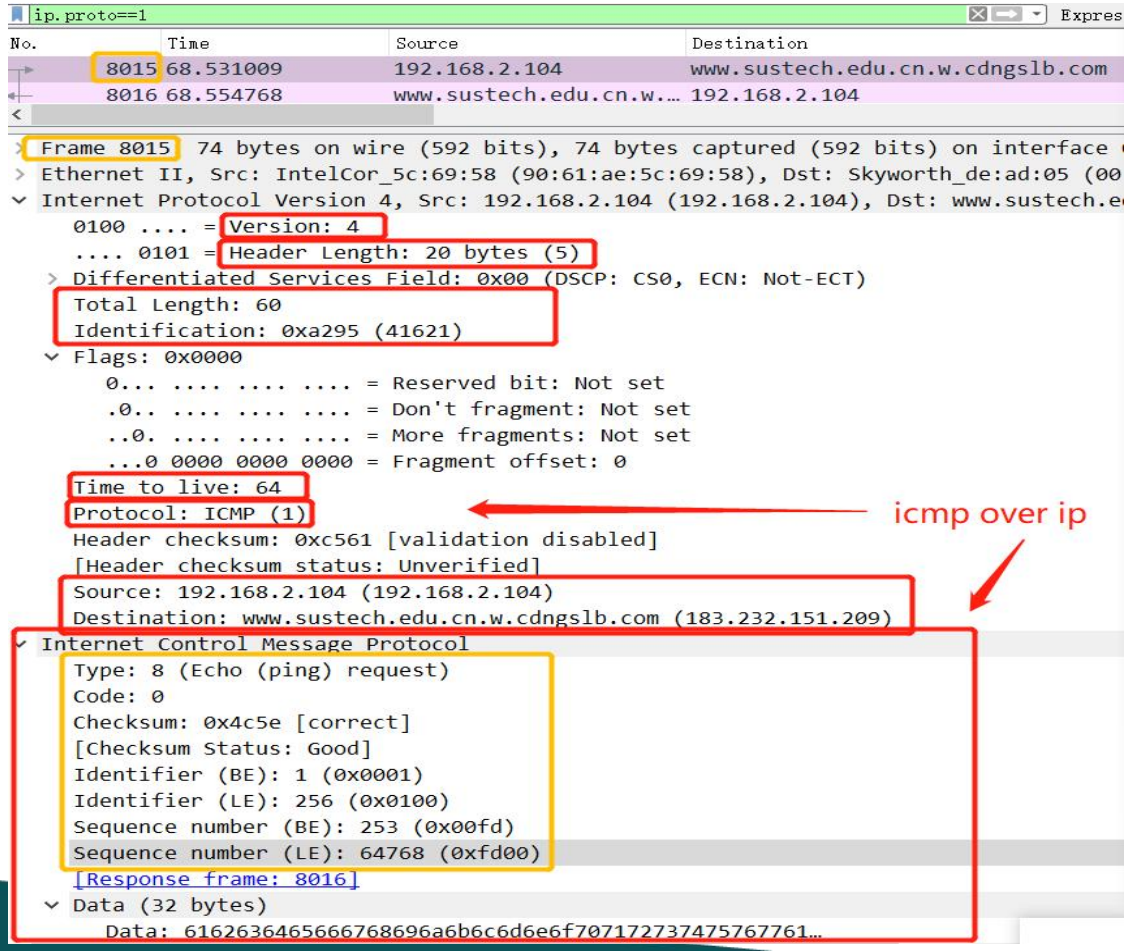
- The data received in the echo message must be returned in the echo reply message.
- Type
  - **8** for **echo request** message;
  - **0** for **echo reply** message.
- Code
  - 0
- The identifier and sequence number may be used by the echo sender to aid in matching the replies with the echo requests. The echoer returns these same values in the echo reply.

Echo or Echo Reply Message



ICMP Echo Request

Initial the session with following cmd command: ping www.sustech.edu.cn



ICMP. type : 8
ICMP. code : 0

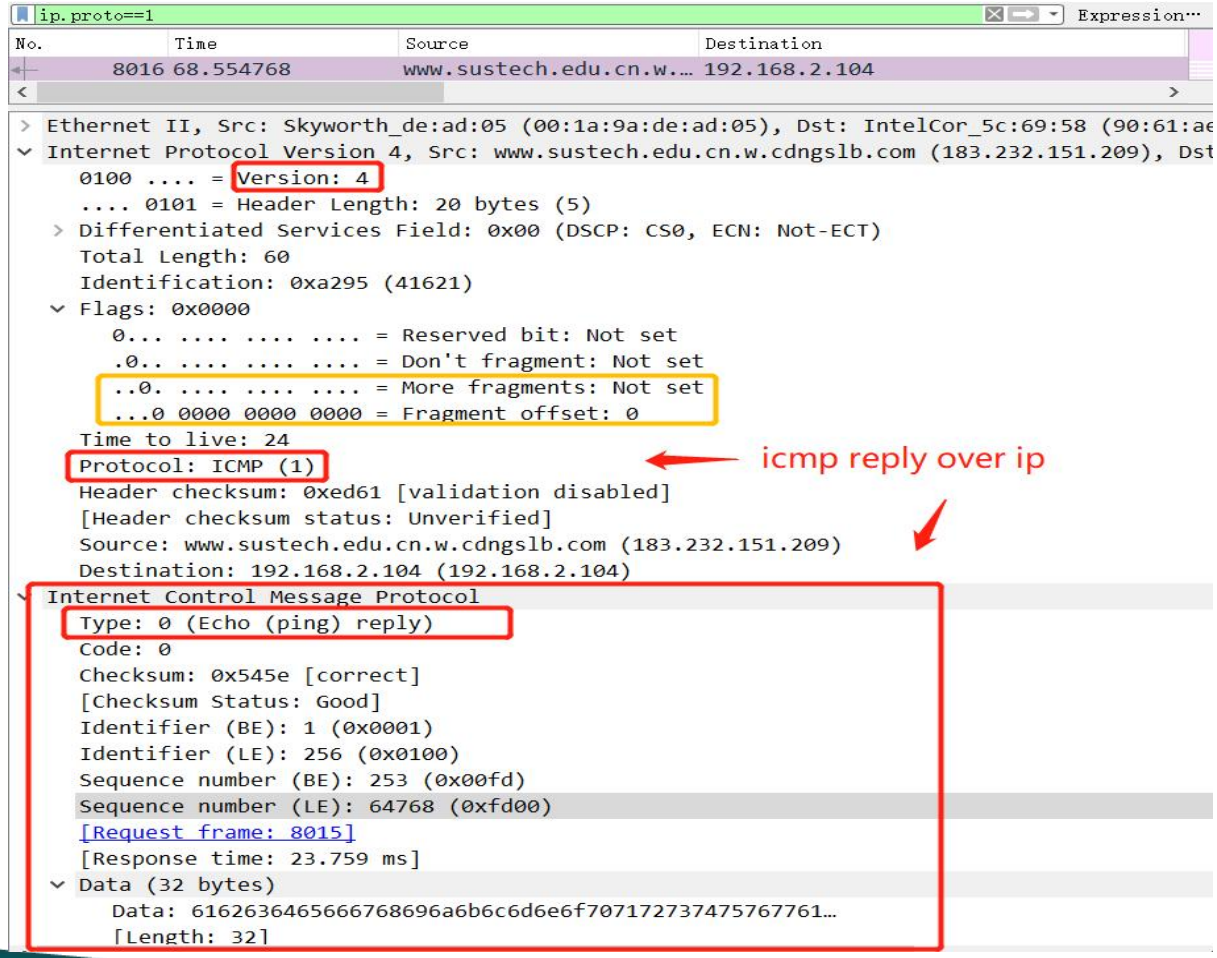
Tips in Wireshark : ip.proto == 1 or ICMP.type

Q1. What's the size of this ICMP Header, ICMP message and this IP packet?

Q2. What's value of Identifier and Sequence number field in this ICMP message?

SUSTech
Southern University of Science and Technology

ICMP Echo Reply



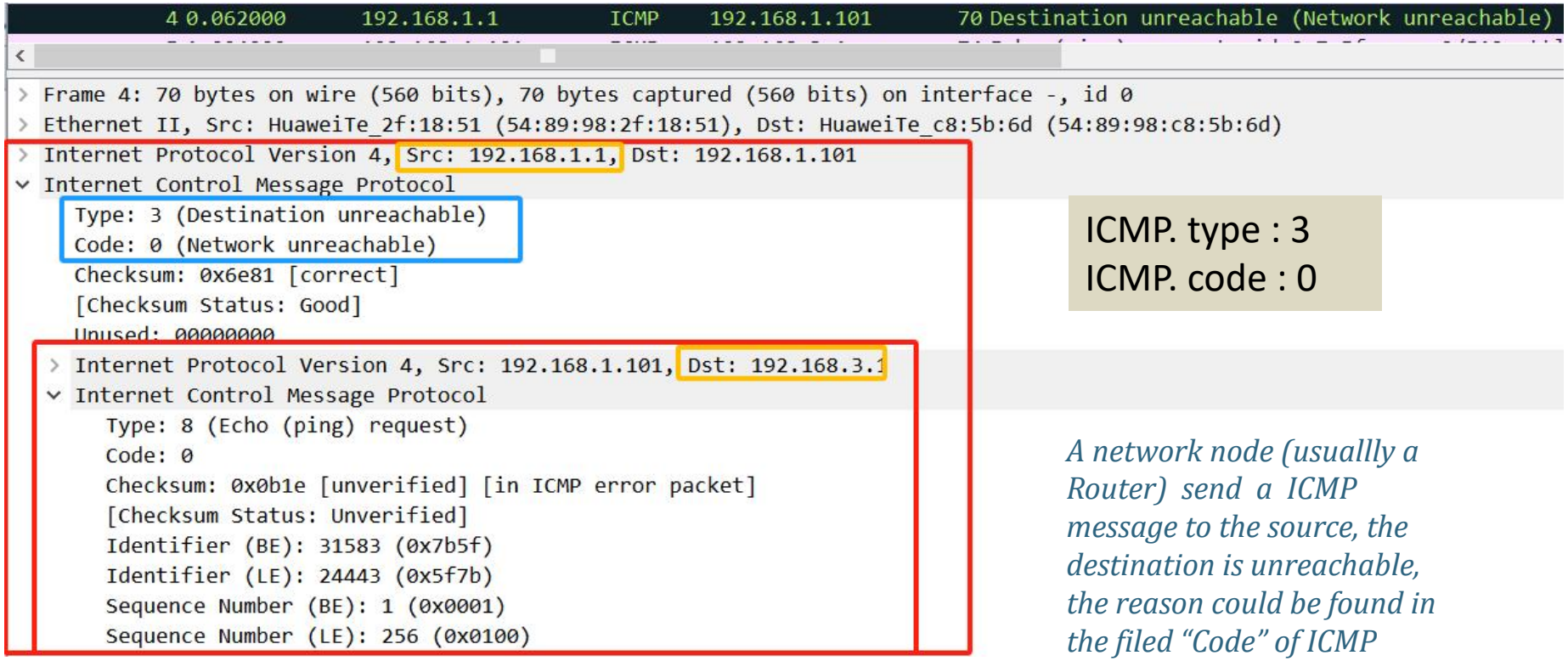
ICMP. type : 0
ICMP. code : 0

Q1. List the differences between the #8015 frame(described on last page) and the #8016 frame(described on this page)

1. soure and destination
2. TTL field of IP Header
3. Total Length field of IP Headeer
4. ICMP Header
…

Tips in Wireshark : ICMP.type

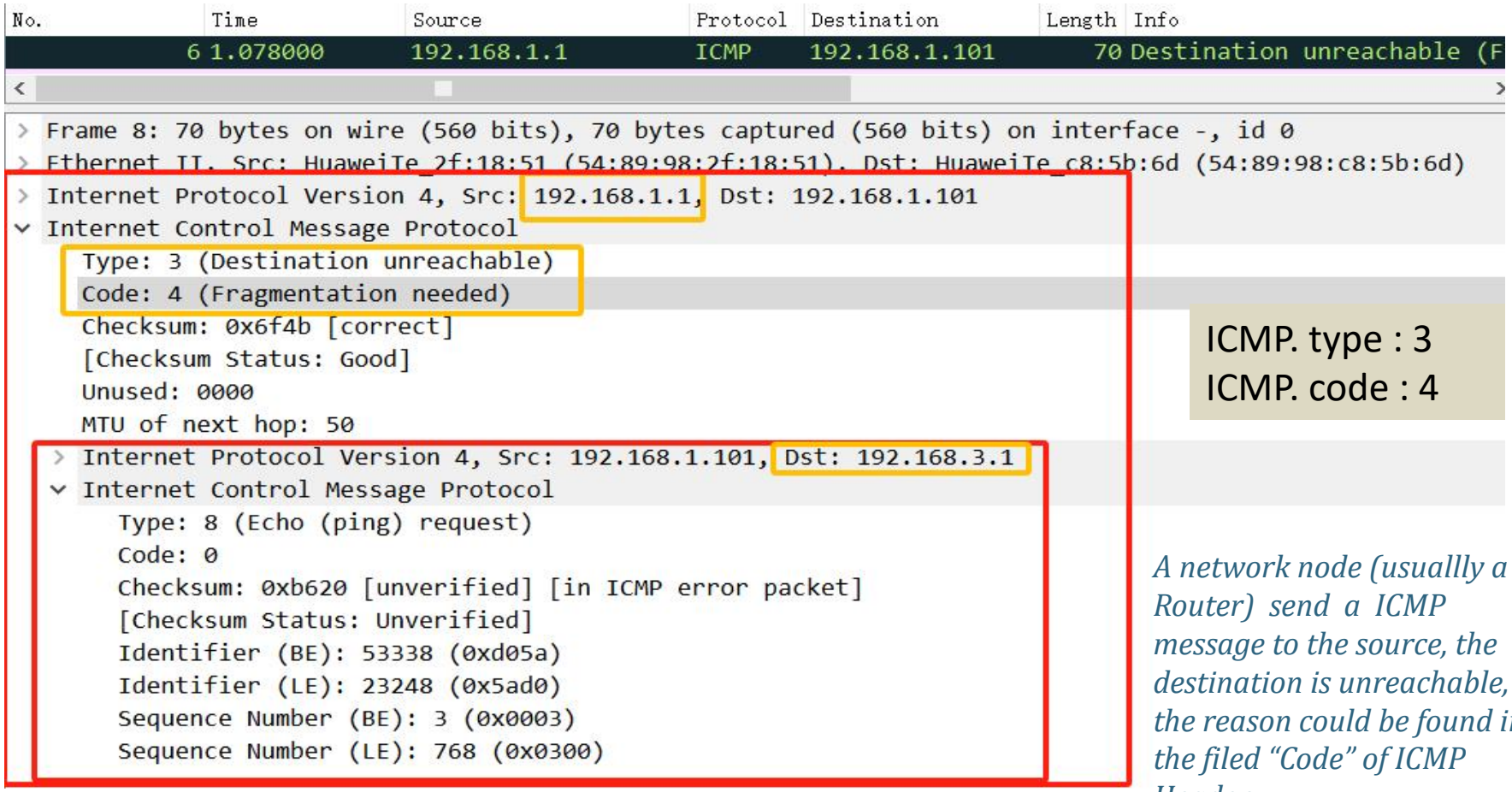
ICMP Destination unreachable(1)



ICMP. type : 3
ICMP. code : 0

A network node (usuallly a Router) send a ICMP message to the source, the destination is unreachable, the reason could be found in the filed "Code" of ICMP Header.

ICMP Destination unreachable(2)



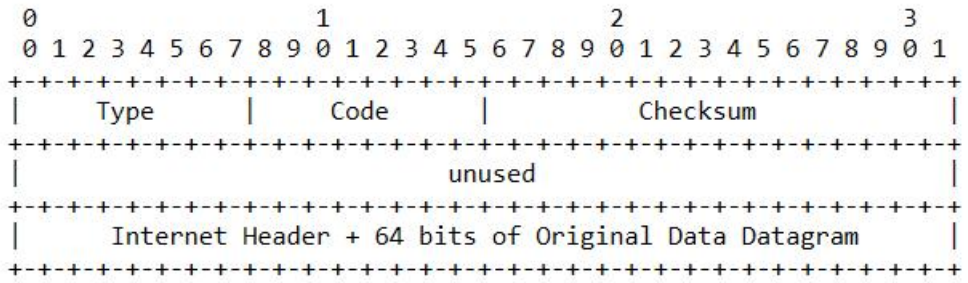
ICMP. type : 3
ICMP. code : 4

A network node (usuallly a Router) send a ICMP message to the source, the destination is unreachable, the reason could be found in the filed "Code" of ICMP Header.

Tips in Wireshark : ICMP.type

ICMP： Time Exceeded(1)

Time Exceeded Message



Type： 11

Code 0 = time to live exceeded in transit;
Code 1 = fragment reassembly time exceeded.

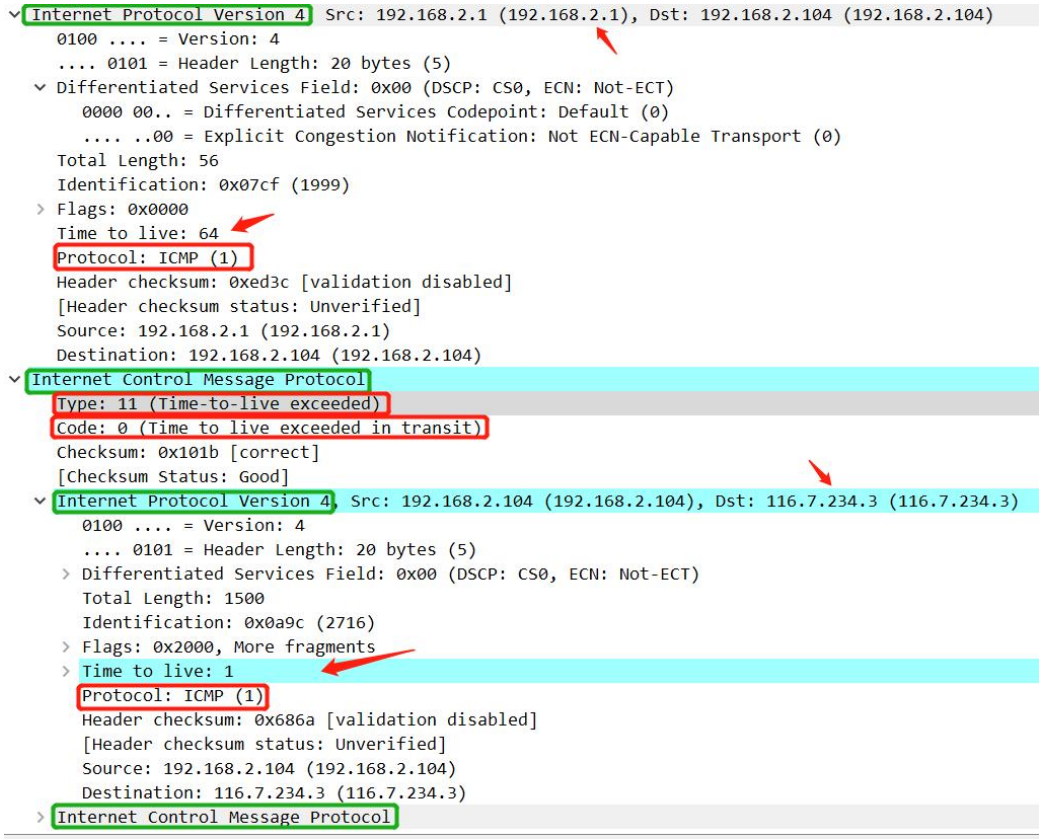
If the gateway processing a datagram finds the time to live field is zero it must discard the datagram. The gateway may also notify the source host via the time exceeded message.

If a host reassembling a fragmented datagram cannot complete the reassembly due to missing fragments within its time limit it discards the datagram, and it may send a time exceeded message.

Code 0 may be received from a gateway. Code 1 may be received from a host.

ICMP： Time Exceeded(2)

Initial the session with following cmd: tracert / traceroute

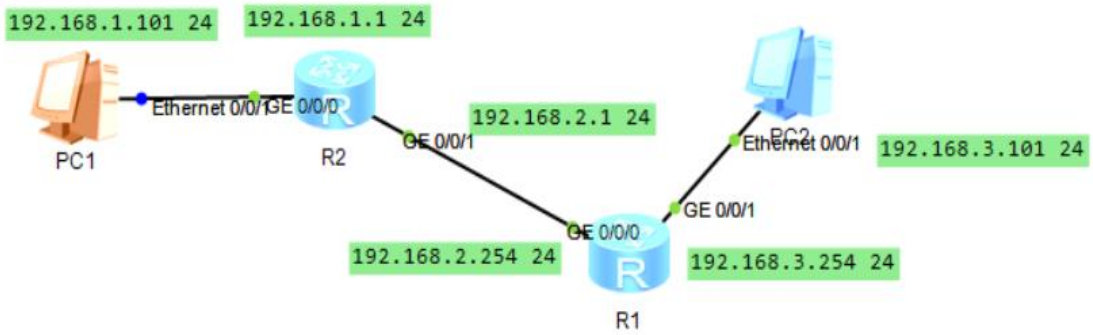


Q1. Is the outside IP's src address same with the inside IP's dest address? Why?

Q2. Is the TTL of outside IP same with which in inside IP? why?

Tips in Wireshark : ICMP.type

Demo2-ICMP Time Exceeded(1)



Build the network as the topology shown on the top.

Do the configuration to make PC1 could send/receive the packet to/from PC2.

Do the capture on the Ethernet 0/0/0 of PC1

Initiate "**tracert**" testing on PC1 to track the route information on the network path from PC1 to PC2.

Answer the following question:

Q1. What's the type of the ICMP message would be received by PC1?

Q2. How does PC1 get the route information on the network path from PC1 to PC2?

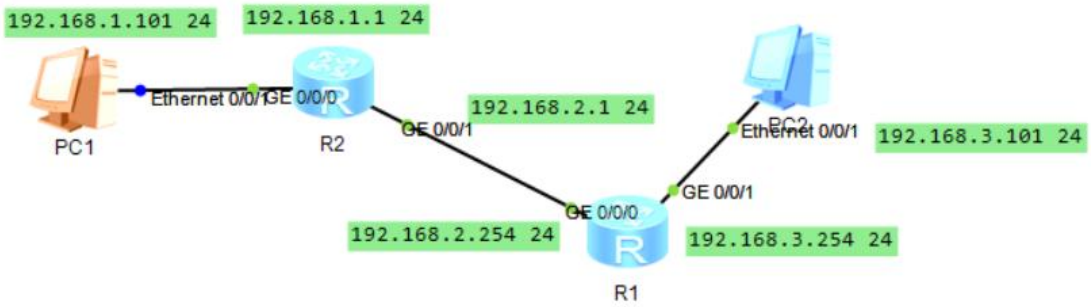
Q3. In this test, Which interfaces on the network path can a PC obtain relevant information from ?

```
PC> tracert 192.168.3.101
1  ?   xxms  xxms  xxms
2  ?   xxms  xxms  xxms
3  ?   xxms  xxms  xxms
```

Demo2- ICMP Time Exceeded(2)

Do the capture on the Ethernet 0/0/0 of PC1

Initiate "**tracert**" testing on PC1 to track the route information on the network path from PC1 to PC2.



PC1

Basic Config | Command | MCPacket | UdpPacket | Cons

```
PC>tracert 192.168.3.101

traceroute to 192.168.3.101, 8 hops max
(ICMP), press Ctrl+C to stop
 1   192.168.1.1    32 ms   15 ms   16 ms
 2   192.168.2.254   47 ms   47 ms   46 ms
 3   192.168.3.101   63 ms   62 ms   79 ms

PC>
```

192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5562, seq=1/256, ttl=1 (no response
192.168.1.1	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5562, seq=2/512, ttl=1 (no response
192.168.1.1	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5562, seq=3/768, ttl=1 (no response
192.168.1.1	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5562, seq=1/256, ttl=2 (no response
192.168.2.254	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5562, seq=2/512, ttl=2 (no response
192.168.2.254	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5662, seq=3/768, ttl=2 (no response
192.168.2.254	ICMP	192.168.1.101	70	Time-to-live exceeded (Time to live exceeded in transit)
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5662, seq=1/256, ttl=3 (reply in 66)
192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) reply id=0x5662, seq=1/256, ttl=126 (request in
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5662, seq=2/512, ttl=3 (reply in 68)
192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) reply id=0x5662, seq=2/512, ttl=126 (request in
192.168.1.101	ICMP	192.168.3.101	106	Echo (ping) request id=0x5662, seq=3/768, ttl=3 (reply in 70)
192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) reply id=0x5662, seq=3/768, ttl=126 (request in

Demo2- ICMP Time Exceeded(3)

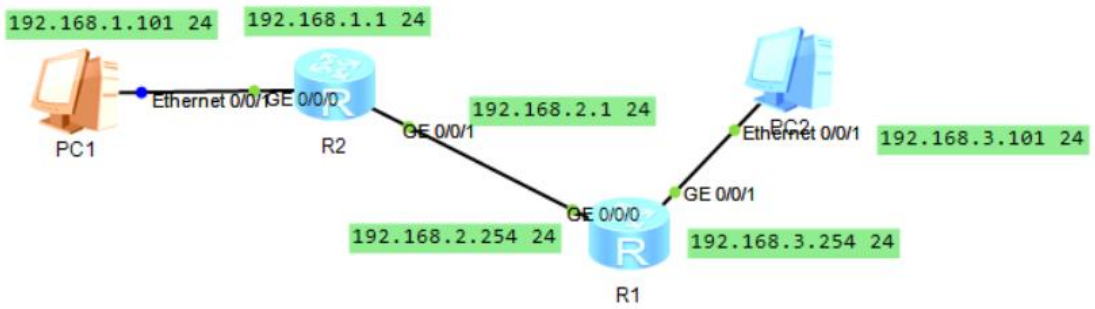
PC1

Basic Config | Command | MCPacket | UdpPacket | Cons

```
PC>tracert 192.168.3.101

traceroute to 192.168.3.101, 8 hops max
(ICMP), press Ctrl+C to stop
 1   192.168.1.1    32 ms   15 ms   16 ms
 2   192.168.2.254    47 ms   47 ms   46 ms
 3   192.168.3.101    63 ms   62 ms   79 ms

PC>
```




```
PC>ipconfig

Link local IPv6 address.....: fe80::5689:98ff:fec8:5b
IPv6 address.....: :: / 128
IPv6 gateway.....: ::
IPv4 address.....: 192.168.1.101
Subnet mask.....: 255.255.255.0
Gateway.....: 192.168.1.1
Physical address.....: 54-89-98-C8-5B-6D
DNS server.....:
```

```
54 729.969000   192.168.1.1   ICMP   192.168.1.101   70 Time-to-live exceede
Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.101
Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xda04 [correct]
    [Checksum Status: Good]
    Unused: 00000000
Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.3.101
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 92
    Identification: 0x6255 (25173)
  > 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
  > Time to Live: 1
    Protocol: ICMP (1)
    Header Checksum: 0x9131 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.101
    Destination Address: 192.168.3.101
Internet Control Message Protocol
```

The destination is not in the same subnet as the source, the packet reaches to the 1st “next hop” : gateway.

Demo2- ICMP Time Exceeded(4)

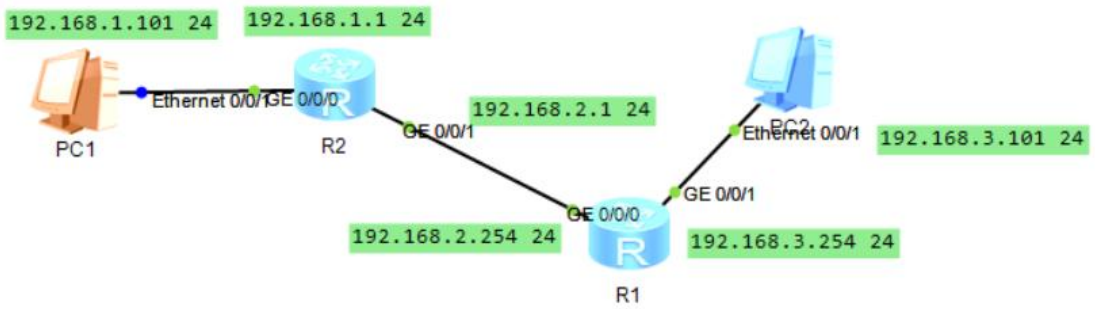
PC1

Basic Config | Command | MCPacket | UdpPacket | Cons

```
PC>tracert 192.168.3.101

traceroute to 192.168.3.101, 8 hops max
(ICMP), press Ctrl+C to stop
 1   192.168.1.1    32 ms   15 ms   16 ms
 2   192.168.2.254    47 ms   47 ms   46 ms
 3   192.168.3.101    63 ms   62 ms   79 ms

PC>
```



R2

```
<Huawei>dis ip routing-table
Route Flags: R - relay, D - download to fib
------------------------------------------------------------------------------
Routing Tables: Public
         Destinations : 7        Routes : 7

Destination/Mask    Proto   Pre  Cost      Flags NextHop         Interface

        127.0.0.0/8   Direct  0    0           D   127.0.0.1       InLoopBack0
        127.0.0.1/32  Direct  0    0           D   127.0.0.1       InLoopBack0
    192.168.1.0/24  Direct  0    0           D   192.168.1.1     GigabitEthernet
0/0/0
    192.168.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet
0/0/0
    192.168.2.0/24  Direct  0    0           D   192.168.2.1     GigabitEthernet
0/0/1
    192.168.2.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet
0/0/1
    192.168.3.0/24  Static  60   0          RD   192.168.2.254   GigabitEthernet
0/0/1
```

The packet reaches to the 2nd “next hop”: 192.168.2.254

```
60 730.047000   192.168.2.254   ICMP   192.168.1.101   70 Time-to-live exceeded (Time to live exceeded in transit)
Internet Protocol Version 4, Src: 192.168.2.254, Dst: 192.168.1.101
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x0015 (21)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: ICMP (1)
    Header Checksum: 0x363c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.2.254
    Destination Address: 192.168.1.101
Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xda04 [correct]
    [Checksum Status: Good]
    Unused: 00000000
  Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.3.101
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 92
      Identification: 0x6258 (25176)
    > 010. .... = Flags: 0x2, Don't fragment
      ...0 0000 0000 0000 = Fragment Offset: 0
    > Time to Live: 1
      Protocol: ICMP (1)
      Header Checksum: 0x912e [validation disabled]
      [Header checksum status: Unverified]
      Source Address: 192.168.1.101
```

SUSTech
Southern University of Science and Technology

Part C. IPv6(1)

- **IPv6** is a new version of the Internet Protocol, designed as the successor to IPv4. The changes from IPv4 to IPv6 fall primarily into the following categories:
- **Expanded Addressing Capabilities**: IPv6 increases the IP address size from **32** bits to **128** bits, to support more levels of addressing hierarchy, a much greater number of addressable nodes, and simpler auto-configuration of addresses. The scalability of multicast routing is improved by adding a "scope" field to multicast addresses. And a new type of address called an "anycast address" is defined, used to send a packet to any one of a group of nodes.
- **Header Format Simplification**: Some IPv4 header fields have been dropped or made optional, to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.
- **Improved Support for Extensions and Options**: Changes in the way IP header options are encoded allows for more efficient forwarding, less stringent limits on the length of options, and greater flexibility for introducing new options in the future.
- **Flow Labeling Capability**: A new capability is added to enable the labeling of packets belonging to particular traffic "flows" for which the sender requests special handling, such as non-default quality of service or "real-time" service.
- **Authentication and Privacy Capabilities**: Extensions to support authentication, data integrity, and (optional) data confidentiality are specified for IPv6.

https://tools.ietf.org/html/rfc2460

IPv6(2)

- **Version**

  4-bit Internet Protocol version number = 6.

- **Traffic Class**

  8-bit traffic class field.

- **Flow Label**

  20-bit flow label.

- **Payload Length**

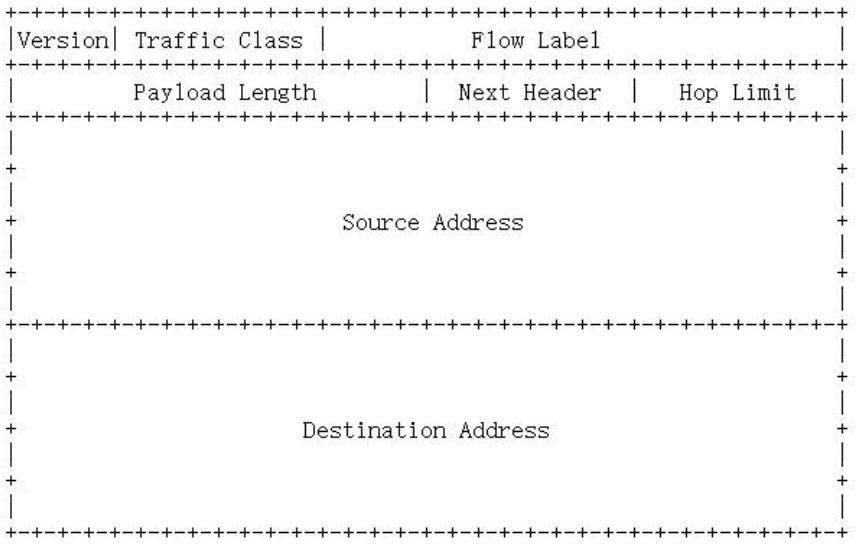
  16-bit unsigned integer. **Length of the IPv6 payload**, i.e., the rest of the packet following this IPv6 header, in octets. (Note that any extension headers present are considered part of the payload, i.e., included in the length count.)

- **Next Header**

  8-bit selector. Identifies the type of header immediately following the IPv6 header.

- **Hop Limit**

  8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.



- **Source Address**

  128-bit address of the originator of the packet

- **Destination Address**

  128-bit address of the intended recipient of the packet. (possibly not the ultimate recipient, if a Routing header is present)

IPv6(3)

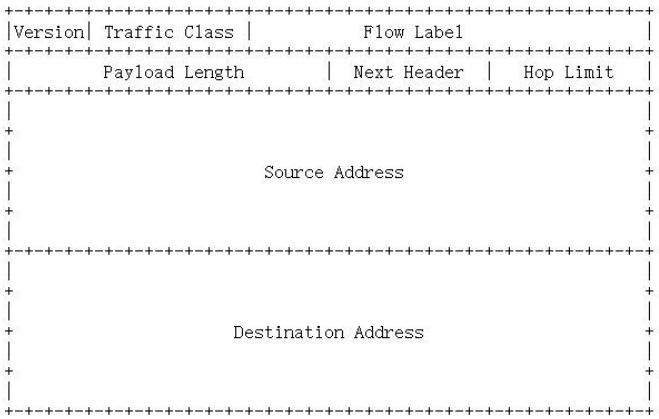
icmpv6

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	::1	::1	ICMPv6	84	Echo (ping) request id=0x0001, seq=

```
> Null/Loopback
v Internet Protocol Version 6, Src: ::1, Dst: ::1
    0110 .... = Version: 6
  > .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
    .... .... .... 0000 0000 0000 0000 0000 = Flow Label: 0x00000
    Payload Length: 40
    Next Header: ICMPv6 (58)
    Hop Limit: 64
    Source: ::1
    Destination: ::1
v Internet Control Message Protocol v6
    Type: Echo (ping) request (128)
    Code: 0
    Checksum: 0xd4a6 [correct]
    [Checksum Status: Good]
    Identifier: 0x0001
    Sequence: 80
    [Response In: 2]
  v Data (32 bytes)
      Data: 6162636465666768696a6b6c6d6e6f707172737475767761…
      [Length: 32]
```

```
0000  18 00 00 00 60 00 00 00  00 28 3a 40 00 00 00 00   ····`··· ·(:@····
0010  00 00 00 00 00 00 00 00  00 00 00 01 00 00 00 00   ···· ···· ····
0020  00 00 00 00 00 00 00 00  00 00 00 01 80 00 d4 a6   ···· ···· ····
```

Internet Protocol Version 6 (ipv6), 40 byte(s) 分组: 161



using 'ping -6 localhost' to invoke an ICMPv6 transaction.

IPv6 Address

- Text Representation of Addresses
  - The preferred form is x:x:x:x:x:x:x:x, where the 'x's are the hexadecimal values of the eight 16-bit pieces of the address
  - In order to make writing addresses containing zero bits easier a special syntax is available to compress the zeros. The use of “::” indicates multiple groups of 16-bits of zeros. The "::" can only appear once in an address.
- Address Type Representation
  - The address 0:0:0:0:0:0:0:0 is called the unspecified address.
  - The unicast address 0:0:0:0:0:0:0:1 is called the loopback address.
  - Link-Local Unicast Addresses are designed to be used for addressing on a single link for purposes such as auto-address configuration, neighbor discovery, or when no routers are present.

```
IPv6 地址 . . . . . . . . . . . . : 2409:8a55:3050:f6b0:48cf:2c49:a3fe:6381
临时 IPv6 地址. . . . . . . . . . : 2409:8a55:3050:f6b0:6cf9:d6fc:544:f4c7
本地链接 IPv6 地址. . . . . . . . : fe80::48cf:2c49:a3fe:6381%17
```

SUSTech
Southern University of Science and Technology

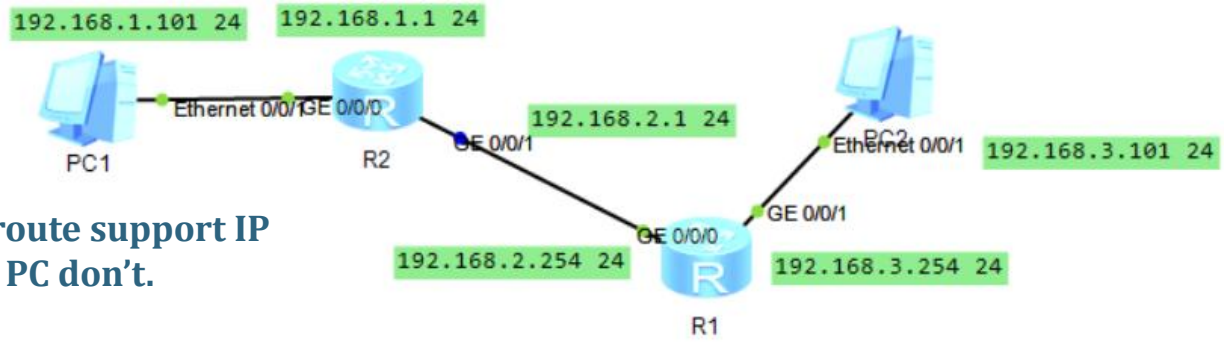
https://tools.ietf.org/html/rfc2373

Practise 11.1

1. Initiates an ICMP session to test if www.example.com is reachable (setting the packet size to 2023B), and capture the packets.
   - How to initiates an ICMP Echo request with 2023B length?
   - Is there any fragmentation on the IP packets, how to find them?
   - How many fragments are the 2023-Byte-length IP packet divided into?

   - How to identify the ICMP Echo request and Echo reply?
   - For the ICMP Echo request, which fragment is the first one, which is the last ? How to identify them?
   - What's the length of each IP fragment? Is the sum of each fragment's length equal to the original IP packet ?

Practise 11.2



Tips: in eNSP, route support IP fragment while PC don't.

Build the network as the topology shown on the top.

- 1st, do the configuration to make PC1 could send/receive packet to/from from PC2.
- 2nd, set the the MTU of R2's interface GE0/0/1 as **50**.
- 3rd, do the capture and the "ping" test, answer the following questions:

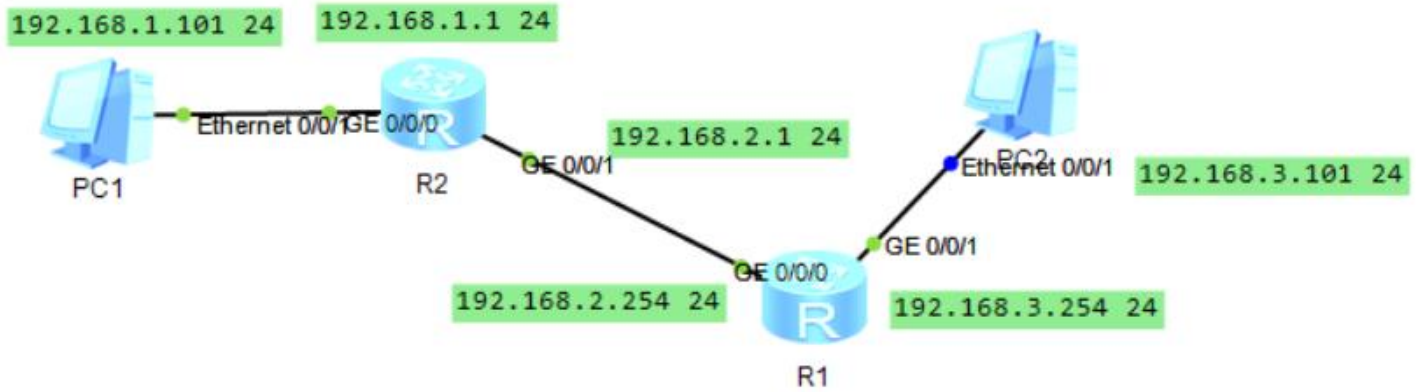
Invoke "ping" test on PC1 to test if PC2 is reachable:

- Could PC1 received the ICMP echo reply message from PC2?
- What's the value of "Don't fragment" bit in IP Header of the packet while it is sent by PC1? What' the value of "Total Length" field in IP Header of the packet while it is sent by PC1?
- if R2 received the IP packet with "Don't fragment" bit is set as 1, and the MTU of its interface GE0/0/1 is set as 50, what would R2 do for the IP packet?

Invoke ping test on PC2 to test if PC1 is reachable

- Could PC1 receive the ICMP echo request message from PC2?
- Could PC2 receive the ICMP echo reply message from PC1?
- Expain the reason.

Practise 11.3(1)



Build the network as the topology shown on the top, do the configuration to make PC1 could send/receive packet to/from PC2.

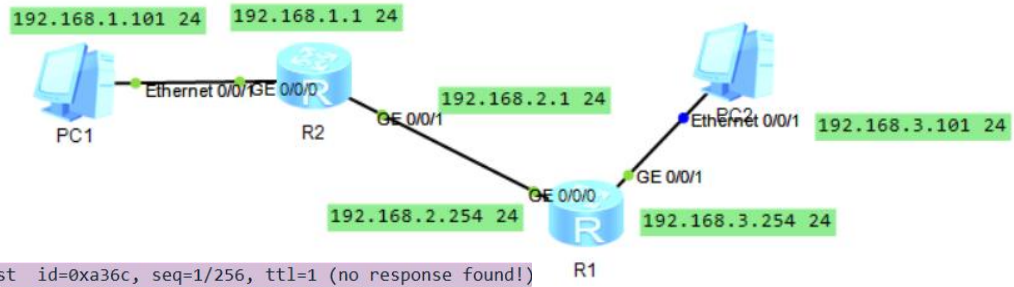
- Use "tracert" to trace the route information from PC2 to PC1, and capture the packets while tracing.
- Please fill in the values of A, B, and C in the table below based on your test, and answer the following questions:

PC> tracert 192.168.1.101

1 A xxms xxms xxms
2 B xxms xxms xxms
3 C xxms xxms xxms
?

Q1. What's the relationship between the three addresses "A ", "B", and "C" and the following items ?
1) the gateway of PC1
2) the gateway of PC2
3) the next-hop of routing entry on R1 to subnet(192.168.1.0 24)
4) the next-hop of routing entry on R2 to subnet(192.168.3.0 24)

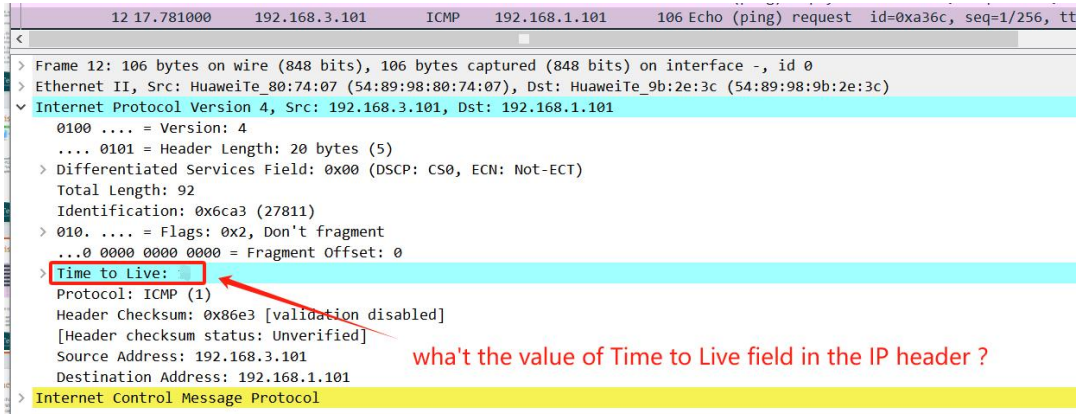
Practise 11.3(2)



12 17.781000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa36c, seq=1/256, ttl=1 (no response found!)
13 17.797000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
14 17.813000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa36c, seq=2/512, ttl=1 (no response found!)
15 17.813000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
16 17.828000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=3/768, ttl=1 (no response found!)
17 17.844000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
18 17.859000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=1/256, ttl=2 (no response found!)
19 17.891000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
20 17.891000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=2/512, ttl=2 (no response found!)
21 17.938000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
22 17.938000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=3/768, ttl=2 (no response found!)
23 17.984000	[redacted]	ICMP	192.168.3.101	70	Time-to-live exceeded (Time to live exceeded in transit)
24 17.984000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=1/256, ttl=3 (reply in 25)
25 18.047000	[redacted]	ICMP	192.168.3.101	106	Echo (ping) reply id=0xa46c, seq=1/256, ttl=126 (request in 24)
26 18.047000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=2/512, ttl=3 (reply in 27)
27 18.109000	[redacted]	ICMP	192.168.3.101	106	Echo (ping) reply id=0xa46c, seq=2/512, ttl=126 (request in 26)
28 18.125000	192.168.3.101	ICMP	192.168.1.101	106	Echo (ping) request id=0xa46c, seq=3/768, ttl=3 (reply in 29)
29 18.172000	[redacted]	ICMP	192.168.3.101	106	Echo (ping) reply id=0xa46c, seq=3/768, ttl=126 (request in 28)

Q2. What is the value of the “TTL” field in the IP header of the following numbered IP packet： 12, 14, 16, 18, 20, 22, 24, 26, 28?

Tips: here #12 is the first ICMP echo request message sent by PC2 during the “tracert” test.



SUSTech
Southern University of Science and Technology