

Lab 3 IP Fragmentation

Objective

- Understand the concept of MTU
- Understand the process of IP Fragmentation

Requirements

You need to install following tools on your computer beforehand:

- **Wireshark:** This lab uses the Wireshark software tool to capture and examine a packet trace. Refer to previous labs for details.
- **ping:** refer to previous labs for details. **The parameters on Windows and Linux are different.**
- **netsh:** On **Windows OS**, *netsh* is a command-line scripting utility that allows you to display or modify the network configuration of a computer that is currently running. Refer to the course website for details of *netsh*. In this lab, we will use *netsh* to change the MTU of network interface on Windows OS.
- **ifconfig:** On **some Unix-like operating systems**, *ifconfig* is used to configure or view the configuration of a network interface.

Note:

Since there are many differences on the usage of *ping* command and the operations of MTU on Windows and Linux, **this lab assumes Windows operating system.**

For students who use Linux or MacOS, please refer to the course website or Google for the detailed usage of *ping* and *ifconfig*.

Exercise

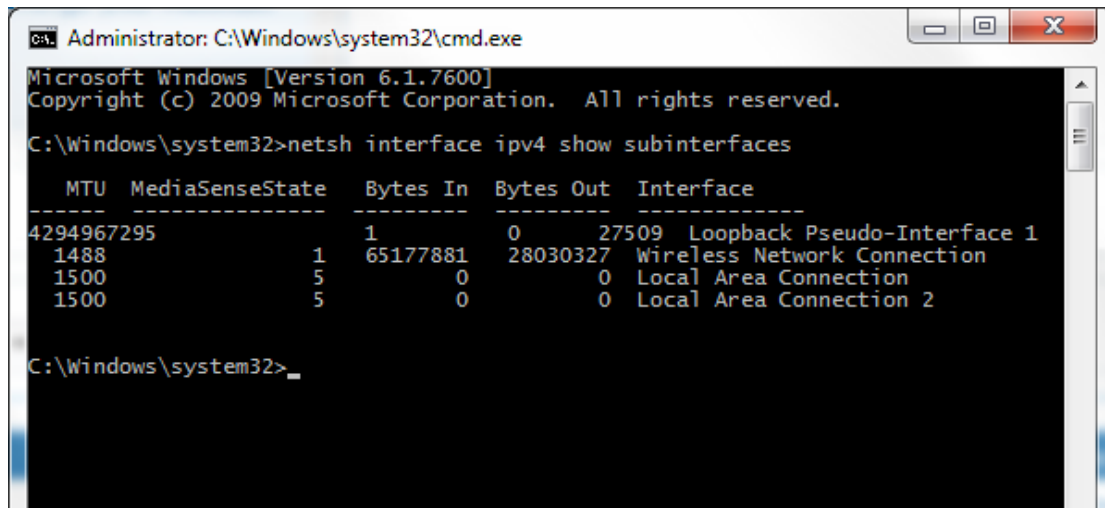
Task 1: Determine the Value of MTU

1. On Windows, execute the following command.

“netsh interface ipv4 show subinterfaces”

Tips:

- You should get a list of all your network adapters installed on your PC. The MTU value is listed on the left.
- The effective MTU size may be smaller than the number displayed here, depending on your network and protocols used. For example, PPP connections (Point-to-Point Protocol) have a default MTU size of 1500 bytes and VPN connections have a default size of 1400.



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>netsh interface ipv4 show subinterfaces

   MTU   MediaSenseState   Bytes In   Bytes Out   Interface
-----
4294967295      1      65177881  28030327  Loopback Pseudo-Interface 1
1488           5           0           0  Wireless Network Connection
1500           5           0           0  Local Area Connection
1500           5           0           0  Local Area Connection 2

C:\Windows\system32>
```

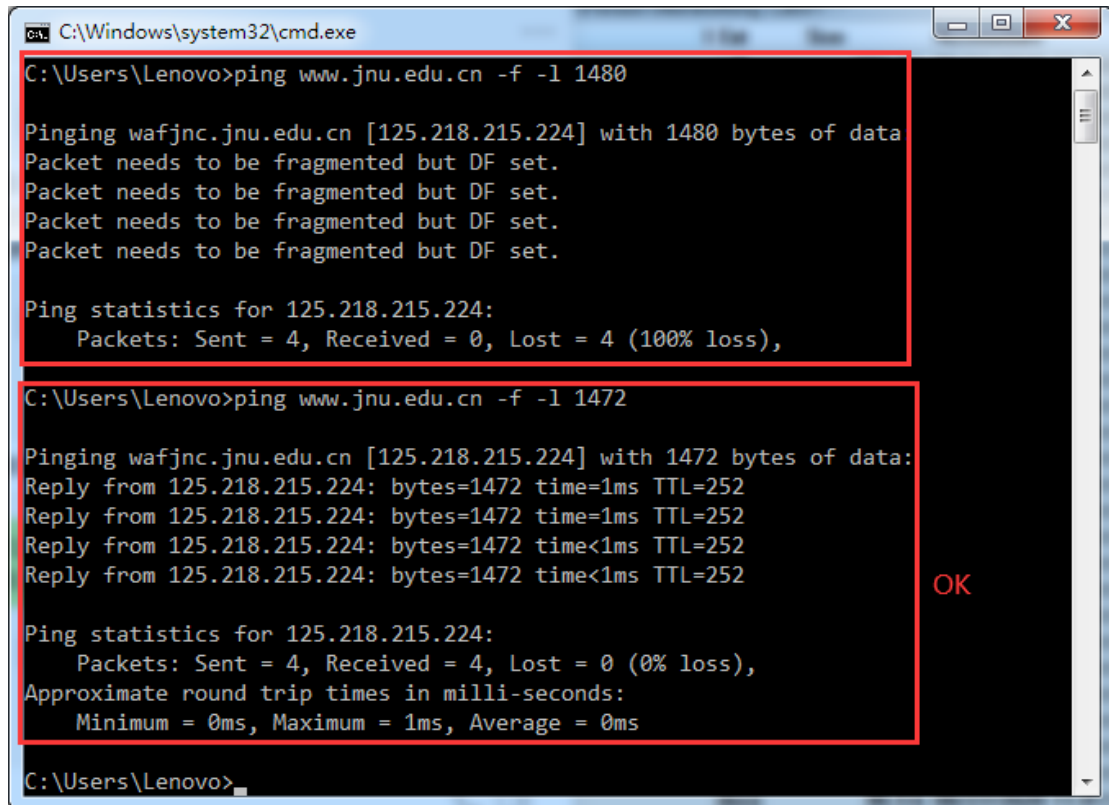
Figure 1: Check MTU on Windows

2. To check the effectiveness of the MTU, say 1500 Bytes, type the following command to ping with an MTU size:

“ping www.jnu.edu.cn -f -l 1472”

Tips:

- The “-f” marks packets that *should not* be fragmented in the *ping*, i.e., DF=1.
 - The “-l ”(lower case L) sets the size of ICMP data, i.e., **payload of ICMP message**. IP header has 20 bytes without options, and ICMP message header has 8 bytes. Thus, the effective ICMP payload size here is $1500 - 28 = 1472$ Bytes.
 - If your network has both IPv4 and IPv6 enabled, you can use “-4” to instruct *ping* to use IPv4.
 - You can choose other web servers for *ping* command in this lab.
3. If you get successful replies, your current MTU is fine for your connection. If you receive error messages like in the following figure, your packets are to be fragmented but DF=1. Keep trying to *ping* with smaller size until you get 4 successful replies.



```
C:\Windows\system32\cmd.exe
C:\Users\Lenovo>ping www.jnu.edu.cn -f -l 1480

Pinging wafjnc.jnu.edu.cn [125.218.215.224] with 1480 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 125.218.215.224:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\Lenovo>ping www.jnu.edu.cn -f -l 1472

Pinging wafjnc.jnu.edu.cn [125.218.215.224] with 1472 bytes of data:
Reply from 125.218.215.224: bytes=1472 time=1ms TTL=252
Reply from 125.218.215.224: bytes=1472 time=1ms TTL=252
Reply from 125.218.215.224: bytes=1472 time<1ms TTL=252
Reply from 125.218.215.224: bytes=1472 time<1ms TTL=252

Ping statistics for 125.218.215.224:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\Lenovo>
```

Figure 2: Determine appropriate ping message size

Task 2: Modify the Value of MTU

1. On Windows, execute the following command.

```
netsh interface ipv4 set subinterface "Local Area Connection" mtu=1490
store=persistent
```

Tips:

- Change the interface name to whatever you're using, e.g., "Local Area Connection 2" or "本地连接 1").
- You need to add 28 bytes on to the value you were using in your *pings*.

```
Administrator: Command Prompt
C:\WINDOWS\system32>netsh interface ipv4 show subinterfaces
Original MTU size
-----
MTU  MediaSenseState  Bytes In  Bytes Out  Interface
-----
4294967295 1 11983933171 0 252206 Loopback Pseudo-Interface 1
1500 1 0 436667513 Ethernet
1500 1 0 43825 VMware Network Adapter VMnet1
1500 1 0 43729 VMware Network Adapter VMnet8
1500 1 0 29115 vEthernet (WSL)

C:\WINDOWS\system32>netsh interface ipv4 set subinterface "Ethernet" mtu=1518 store=persistent
Ok.

C:\WINDOWS\system32>netsh interface ipv4 show subinterfaces
New MTU size
-----
MTU  MediaSenseState  Bytes In  Bytes Out  Interface
-----
4294967295 1 11983962548 0 252206 Loopback Pseudo-Interface 1
1518 1 11983962548 436669956 Ethernet
1500 1 0 44762 VMware Network Adapter VMnet1
1500 1 0 44666 VMware Network Adapter VMnet8
1500 1 0 30052 vEthernet (WSL)

C:\WINDOWS\system32>
```

Figure 3: Modifying MTU on Windows

2. Try Step 2 and Step 3 in Task 1 to verify the value of new MTU.

3. Remember to RESTORE the MTU value after the lab! /

实验结束后，务必要恢复 MTU 值!!!

Task 3: Capture a Trace of IP Fragmentation

1. Suppose the MTU is 1500. Launch Wireshark and start a capture with a filter of “*ip.proto==I*”.
2. Execute the following command:

“*ping www.jnu.edu.cn -l 3000 -n 1*”

After the *ping* command is complete, return to Wireshark and stop the trace. You should now have a short trace similar to that shown in the figure below.

Tips:

- The “-n” specifies the number of ICMP Echo Request messages sent. The default is 4. “-n 1” will generate one ICMP Echo Request.
- By default, Wireshark will try to reassemble fragments. You can disable it in preferences setting of IPv4 (Edit→Preferences→Protocols→IPv4→Uncheck “Reassemble fragmented IPv4 datagrams”), or in the right-click menu of an IP packets from captured result (See the following Figure 4).

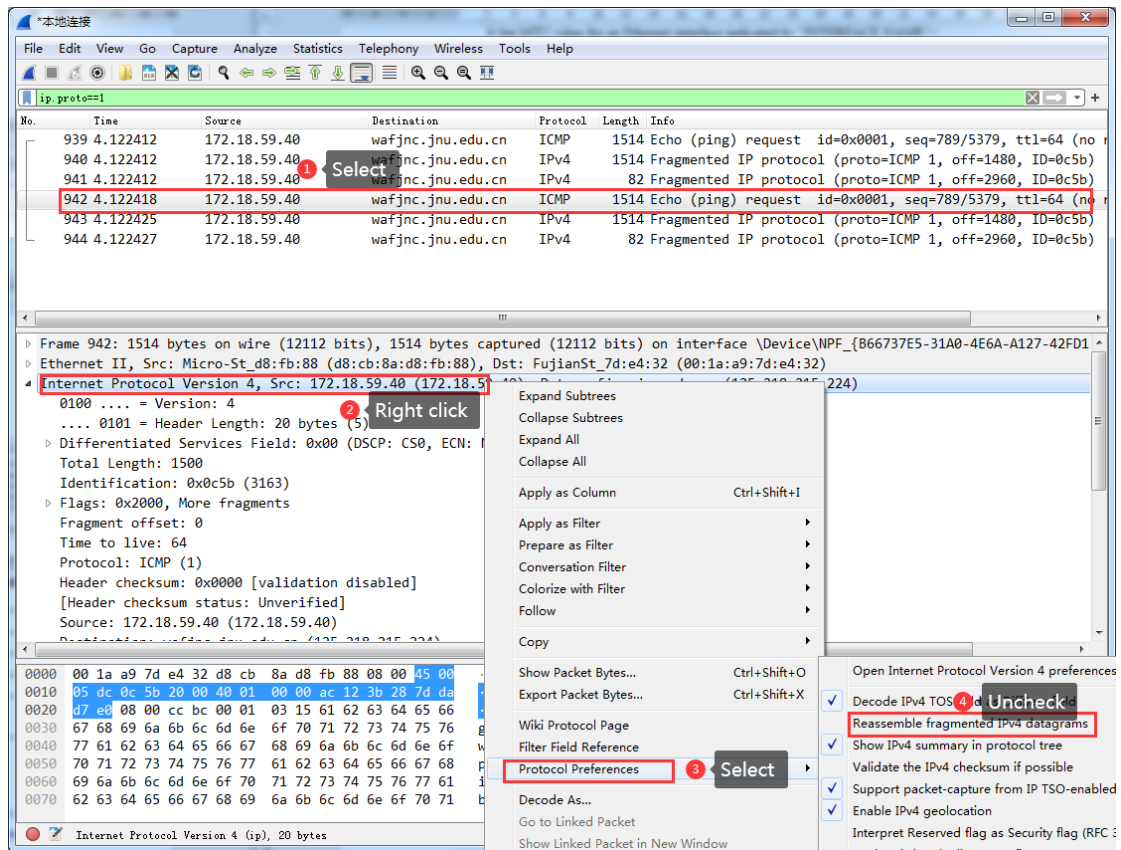


Figure 4: Capture results

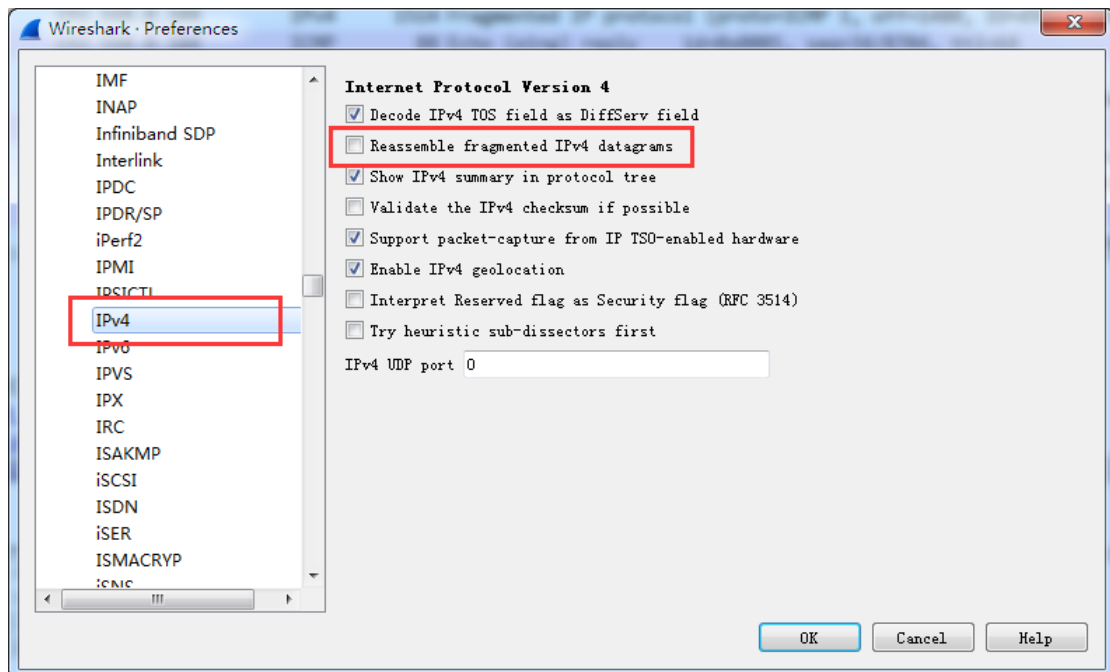


Figure 5: Disable reassemble fragmented IPv4 datagrams

- Locate all fragments of the ICMP Echo Request, and fill the following table:

Fields	Fragmentation 1	Fragmentation 2	Fragmentation 3
"Identification"			
"MF"			
"DF"			
"Fragment Offset" (decimal value)			
Size of transmitted data (IP payload)			

Task 4: IP fragmentation & reassembly (Optional, but strongly suggested!!!)

We also encourage two students collaborate with each other, using two remote computers, to investigate the operations of IP fragmentation and reassembly. For example, you can consider the following steps:

- Host A *ping* Host B to ensure the reachability of *ping* between A and B.

Tips:

- The *ping* maybe blocked between Host A and B when they are connected to different networks. This is because some networks or computers may use strict network configurations.
- In this case, I suggest you to **use computers in the lab**.

- Both Host A and B start Wireshark to capture the IP packets exchanged between A and B.

Tips:

- You can setup a filter using the IP address of computer on the other end. For example, on Host A: *ip.addr==IP_B*; on Host B: *ip.addr==IP_A*.
- You can also use filter of *ip.dst* or *ip.src* to filter out packet on only one communication direction.

- Host A *ping* Host B using a large ICMP message, using the similar operations as **Task 3**.
- Investigate the captured trace on both Host A and Host B.
- Describe the process of fragmentation and reassembly for an IP packet sent from Host A to Host B.

Questions:

- Most often IP packets in normal operation are not fragmented. But the receiver must have a way to be sure. How does the receiver computer to determine whether a packet is fragmented or not?
- If an IP packet is fragmented, how does the receiver identify all fragments?
- What is the meaning of filter "*ip.proto==1*"? Can you use filter "*icmp*" to obtain all fragments in Task 3 of this lab? What is the difference between the display results of these two filters (i.e., "*ip.proto==1*" and "*icmp*") in Wireshark? Explain

the reason.

4. What are the advantages for large MTU and small MTU respectively?
Tips: see [wiki](#) for more details about MTU.