CS305 Programming Assignment 2(2024F@sustech)

This assignment consists of Part 1 and Part 2, with a total score of 110 points. 100 points or above will be regarded as full marks, while the extra scores over 100 will **NOT** be added to other scoring items of CS305 course.

Part 1 - Lab Practices (45 pts)

- Complete 2 practices for each lab in the lab11, lab12, lab13, lab14 experimental slides. **(5*2*4 = 40 pts)**
- Practices on H3C device. **(5 pts)**
    - Log in to the H3C network device through the console port, show the summary information of all interfaces on the device, and distinguish between layer 2 interfaces and layer 3 interfaces. **(2 pts)**
    - Set up H3C devices remote login, allowing users to log in to the device via telnet on a local PC for viewing and configuration work. **(2 pts)**
    - Users can also log in to the device via telnet on non directly connected PCs for viewing and configuration work. **(1 pt)**

Tips

- You are expected to complete the practices in the experimental class.
- If you cannot complete them during the experiment class, please make an appointment with the instructorsor/SAs for a separate check and grading before 5:00 pm on Friday of week 16.

Part 2 - PMTU Detection (65 pts)

Introduction

In computer networks, the **Maximum Transmission Unit (MTU)** is the largest size of a packet that can be sent in a packet- or frame-based network, such as the Internet. MTU is typically measured in bytes and varies between different networks and interfaces. Understanding the MTU along a network path is crucial for optimizing data transmission and avoiding inefficiencies caused by fragmentation.

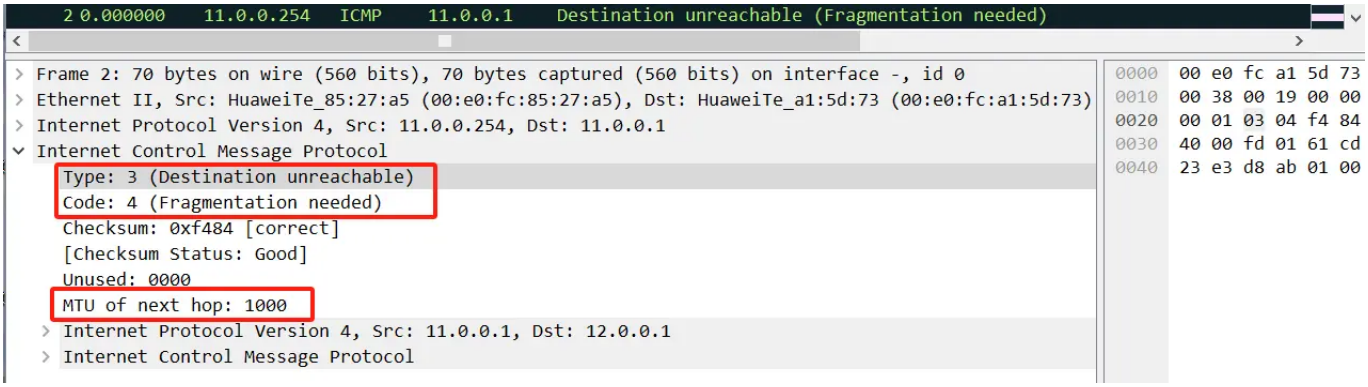
When a packet is larger than the MTU of the network path it traverses, it must be fragmented or dropped. In cases where fragmentation is not allowed, the packet is discarded, and the router sends back an ICMP (Internet Control Message Protocol) error message indicating that the packet is too big or that fragmentation is needed.

Path MTU Discovery (PMTUD) is a method used to determine the MTU along a path between two hosts. PMTUD works differently in IPv4 and IPv6, due to differences in how fragmentation and the "Don't Fragment" flag are handled.

IPv4 PMTUD (30 pts)

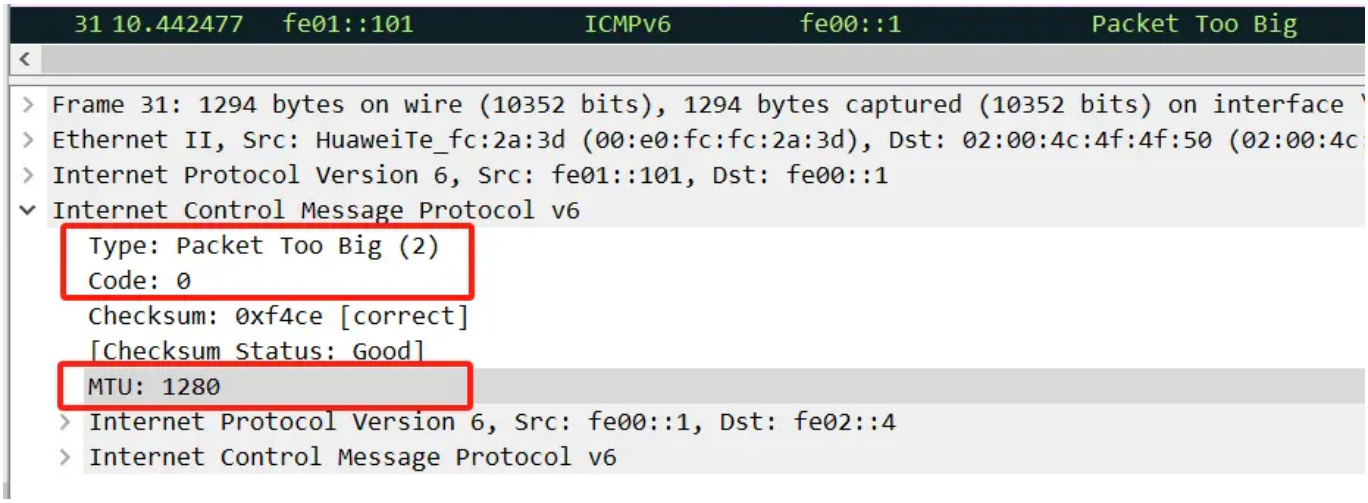
- **IPv4 Fragmentation**: In IPv4, fragmentation can occur at intermediate routers if a packet is larger than the MTU of the outgoing link, unless the "Don't Fragment" (DF) flag is set in the IP header.

- **"Don't Fragment" Flag**: When the DF flag is set, routers are instructed not to fragment the packet. If a router receives a packet with the DF flag set and the packet exceeds the MTU of the next hop, it will drop the packet and send back an ICMP "Destination Unreachable" message with code "Fragmentation Needed" (Type 3, Code 4).
- **ICMP Messages in IPv4**: These ICMP messages indicate that fragmentation is needed. Unlike IPv6, the message type is not "Packet Too Big," but the message includes information that the packet could not be forwarded without fragmentation.



IPv6 PMTUD (30 pts)

- **IPv6 Fragmentation**: In IPv6, fragmentation is handled differently. Intermediate routers do not fragment packets. Instead, fragmentation can only occur at the source node.
- **No "Don't Fragment" Flag**: IPv6 does not have a DF flag in the header because routers are not expected to fragment packets.
- **ICMPv6 "Packet Too Big" Messages**: If an intermediate router receives a packet that is too large for the next hop's MTU, it will drop the packet and send back an ICMPv6 "Packet Too Big" message (Type 2). This message includes the MTU of the next hop, allowing the source node to adjust its packet size accordingly.



By interpreting these ICMP messages, a host can adjust its packet sizes to match the MTU of the path, avoiding fragmentation and optimizing network performance.

In this part, you will use both Python and eNSP to explore and implement the concept of PMTUD. You will:

1. Use eNSP to set up and configure networks with varying MTU values along the path.
2. Write a Python script to probe the MTU along the path using ICMP packets.

Report (5 pts)

In the report, you need to present the following content:

1. Screenshots of two self-test topologies (one for IPv4, one for IPv6)

- Each interface's IP address and MTU settings must be clearly marked on the diagram
- The topology diagram must show the name of each device and each interface

2. Test instructions based on the corresponding topology:

- Specify the parameters used during PMTU testing (The range of testing packet length)
- Include your test results (the detected PMTU value)

Code framework

Here is a code framework for this assignment. You are required to implement the logic inside the `pmtu()` function based on the assignment's requirements.

Notice: package 'subprocess' and 'os' in python are NOT allowed to be used in the code of this assignment!

```
import argparse


def pmtu(dest_addr: str, use_ipv6: bool = False, src_addr: str = None) -> int:
    """
    Discover the Path MTU (PMTU) to the specified destination address.

    Args:
        dest_addr (str): The destination IP address to probe.
        use_ipv6 (bool): Flag indicating whether to use IPv6 or IPv4.
                         Your code should first check whether the address
                         is valid.  Given an IPv4 address together with
                         `use_ipv6` set to True is NOT valid.
                         In this case, the function should raise a RuntimeError.
        src_addr (str): The source IP address to use for sending packets.

    Returns:
        An integer value indicating the PMTU detection result.

    Raises:
        RuntimeError: If the address is invalid.
    """
    # TODO: Implement the logic for PMTUD.
    # Using libraries such as "scapy" is allowed.
    pass


def main():
```

```
    parser = argparse.ArgumentParser(description='Discover the Path MTU along a
network path.')
    parser.add_argument('destination', type=str, help='The destination IP address
to probe.')
    parser.add_argument('--ipv6', action='store_true', help='Use IPv6 instead of
IPv4.')
    parser.add_argument('--source', type=str, help='Optional source IP address to
use for probing.')

    args = parser.parse_args()

    try:
        # Call the PMTU discovery function
        mtu = pmtu(dest_addr=args.destination, use_ipv6=args.ipv6,
src_addr=args.source)
        print(f"The Path MTU to {args.destination} is {mtu} bytes.")
    except RuntimeError as e:
        print(f"Error: {e}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")


if __name__ == '__main__':
    main()
```

Submission

When submitting, you need to submit a .pdf report named `sid.pdf` and a python file containing all your code named `sid.py`. Package them into a .zip archive named `sid.zip` and submit it to blackboard site. `sid` is your student ID.

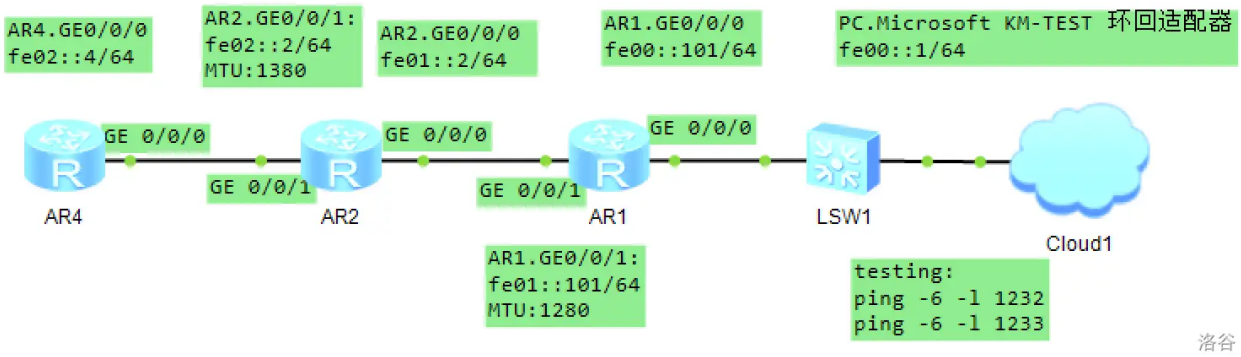
Your archive file tree should be like this:

```
sid.zip/
├── sid.pdf
└── sid.py
```

In your `sid.py`, there **MUST** be a function named as `pmtu()` which only returns one integer stands for your PMTU detection result. Otherwise, your score might be **0**.

Notice

1. This assignment will be graded using scripts. The following eNSP network topology is a example where your code will be tested in:



The testcases include IPv4 network and IPv4/IPv6 dual-stack network. The network topology and MTUs for each router may also be different. When testing, for ipv4, the possible PMTU value is `50-1500`; for ipv6, the possible range is `1280-1500`. To make sure your code is robust enough, you are highly recommended to build sufficiently complex network topos, handle larger detection range and test by yourself.

2. To facilitate script-based grading, please make sure to submit and name your files according to the specified format. Failure to do so may result in serious grading errors.

3. The delayed DDL for the part2 is before 5:00 pm on Friday of week 15. Late submission is NOT allowed for part2.

FAQ

Q: How do I connect the virtual network in eNSP and the real networks?

A: Please refer to the document titled "Implementing Bridging Between Routers and Network Cloud (Real PC Network Card) in eNSP"/ "eNSP 中实现路由器与网络云（真实PC 网卡）的桥接及IPv6 通信" for detailed instructions.

Q: eNSP reports the error: "Failed to start device AR1. Error code 40."

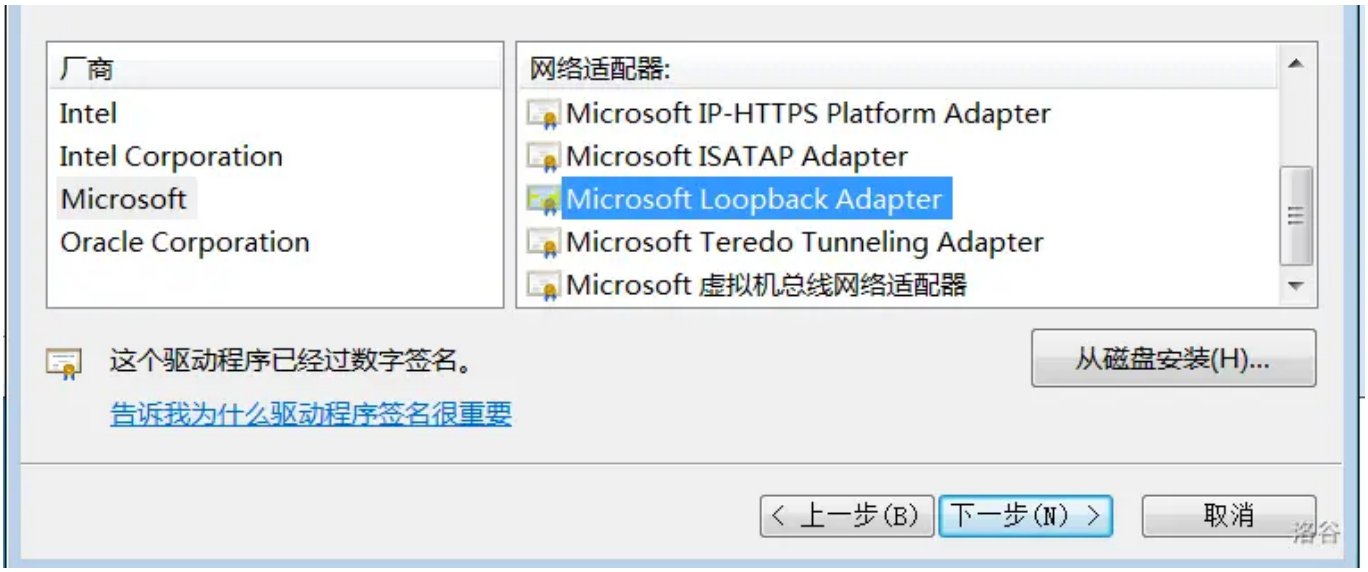
A: This issue is a common problem in eNSP. To resolve it:

1. Follow the steps in the official eNSP FAQ document to troubleshoot the issue.
2. If the problem persists, we recommend using a virtual machine with Windows 7 installed as a workaround. FYI:

[1] https://blog.csdn.net/weixin_43872190/article/details/119853889

[2] https://www.wikihow.com/Install-Windows-7-on-a-VMware-Workstation

In Win7, you might not be able to find `Microsoft KM-TEST 环回适配器`, as indicated in the document titled "eNSP 中实现路由器与网络云（真实PC 网卡）的桥接及IPv6 通信" mentioned above. In this case, use `Microsoft Loopback Adapter` instead.


厂商
Intel
Intel Corporation
Microsoft
Oracle Corporation
网络适配器:
Microsoft IP-HTTPS Platform Adapter
Microsoft ISATAP Adapter
Microsoft Loopback Adapter
Microsoft Teredo Tunneling Adapter
Microsoft 虚拟机总线网络适配器
这个驱动程序已经过数字签名。
从磁盘安装(H)...
告诉我为什么驱动程序签名很重要
< 上一步(B)
下一步(N) >
取消