靳栋 2020533190　　罗永健　2020533191

In this course, we embark on a journey of hands-on exploration through the implementation of four distinct projects, each tailored to provide a comprehensive understanding of modern computer network technologies. These projects enable us to delve systematically from the bottom to the top, unraveling the intricacies of each implementation level spanning from the physical layer to the application layer.

In the initial project of this course, our focus is on the transmission of data streams utilizing audio information. This endeavor challenges us to ingeniously employ audio signals as a conduit for data transfer. This practical undertaking not only facilitates a deeper comprehension of the theoretical foundations but also instills a profound appreciation for the nuances associated with each layer of implementation. Through this experiential learning approach, we are not merely acquiring knowledge; we are actively engaging with the principles that underpin modern computer networking. This hands-on methodology empowers us to bridge the gap between theory and application, fostering a holistic and insightful grasp of the subject matter.In Project 1, we first need to familiarize ourselves with the driver tool we are using, adjust its settings to match the computer, and understand its usage. In digital communication, there are several methods to simulate 0 and 1 in audio signals: Frequency Shift

Keying(FSK). This method represents 0 and 1 by switching between two or more different frequencies. For instance, 0 and 1 may be transmitted at two different frequencies, and the receiver detects the frequency of the received signal to identify 0 and 1. Phase Shift Keying(PSK). PSK utilizes the phase of the signal to represent 0 and 1. Different phases correspond to different bits. For example, binary 0 may be represented by a 0-degree phase, while binary 1 may be represented by a 180-degree phase. Amplitude Shift Keying(ASK). In ASK, binary 0 and 1 are represented by the amplitude of the signal. Two different amplitudes can be used to represent 0 and 1. Quadrature Amplitude Modulation(QAM): QAM combines amplitude and phase. QAM signals typically transmit two independent digital signals on orthogonal axes in signal space to achieve higher data transfer rates.We simulate the difference between 0 and 1 using raw phase differences(PCK), enabling data transmission between two different devices. The challenges we face in this project primarily involve the fact that the driver settings that work for normal data transmission in one location may result in errors when moved to another location. Additionally, the experimental results for the same settings can vary significantly at different times. For instance, the code might run smoothly in the dormitory experiment one day but encounter errors the next day. This necessitates us to complete the relevant functionality as early as possible and experiment with our code in as many environments or conditions as possible to find a relatively stable state. In this project, we completed the code writing early and spent a considerable amount of time familiarizing ourselves with the driver and debugging hardware settings. Moreover, since the course recommends

using Java for code writing, and both I and my group members had not worked with this language before, we also spent some time getting acquainted with Java's code structure, syntax, packages, and related information.

In Project 2, unlike Project 1 where we transmit data through sound signals, we need to implement data transmission through audio data lines, i.e., electrical signals. Additionally, we are required to implement more advanced functionalities than Project 1, such as receiving and sending ACK, understanding and implementing CSMA principles to achieve faster and more stable data transmission. We first implemented a buzzing signal, generated by producing continuous audio oscillations to create a buzzing sound signal resembling a beep. This is typically achieved through rapid alternation of high and low voltages. Such a signal is challenging to spontaneously generate and serves as a helpful indicator for later determining whether the received data aligns with what we intend to receive.The main issues we face in this project are similar to those in Project 1, where the same settings may yield different results in different locations or at different times. Initially, this project requires us to implement the functionality of only transmitting data, with specific requirements for transmission speed and accuracy. We divided this task into two parts, with one person implementing the sender and the other implementing the

receiver. State machine transitions were jointly completed during the testing of the code. In this task, the accuracy of our code implementation for data transmission can stabilize at around 95%. However, we also encountered some problems. For example, we were unable to meet the required speed throughout the process. Adjusting parameters such as sampling frequency and the number of bits sent per transmission did not effectively improve transmission efficiency. In the end, we settled on a sampling frequency of 44100Hz, with the optimal scenario being the transmission of 5 bits at a time. However, this setting only resulted in an average score, and we speculate that this situation may be related to the way we implemented the functionality. However, there was no conceivable method available for reference after this project (although it is not crucial for the subsequent projects). Furthermore, this project also requires us to implement CSMA-related content. CSMA stands for "Carrier Sense Multiple Access." It is a multiple access protocol used for shared transmission media.

The basic idea of the CSMA protocol is to listen to the channel (transmission medium) before sending data. If the channel is idle, data transmission begins; if the channel is busy, the device waits for a random amount of time before attempting to listen again. "Carrier Sense" refers to listening to the channel, and "Multiple Access" indicates multiple devices sharing the same transmission medium.

CSMA has several variants, with the most common ones being:

CSMA/CD(Carrier Sense Multiple Access with Coliision Detection): Primarily used in Ethernet local area networks. If two devices attempt to send data simultaneously, a collision may occur, and they stop transmitting and wait for a random time before trying again. This protocol is less common in modern Ethernet networks, which often use switches instead of hubs, reducing the likelihood of collisions.

CSMA/CA(Carrier Sense Multiple Access with Collision Avoidance): Mainly used in wireless local area networks (WLANs). Unlike CSMA/CD, CSMA/CA waits for a random DIFS (Distributed Interframe Space) time after sensing an idle channel to reduce the possibility of collisions. It also employs RTS/CTS (Request to Send/Clear to Send) frames to coordinate data transmission.

These CSMA protocols provide a fair and efficient way for multiple devices to access the same transmission medium. However, collisions may still occur, especially under high loads, impacting performance. In some situations, more advanced multiple access protocols, such as polling or time-based access, may be more suitable.

However, due to time constraints, we were not able to meet the requirements for completing this part of the content.

In Project 3, we need to implement ICMP Echo, routers, NAT, and other functionalities. ICMP is a network control and diagnostic protocol encapsulated within an IP datagram, but it is generally considered as a part of IP. All IP hosts must implement ICMP (RFC 1122), and routers are required to support ICMP. A router is a multi-port node in IP networks, with each port assigned an IP address. Routers are responsible for forwarding IP datagrams

that are not destined for them. To enable certain network functions, routers may need to modify the content of the IP header and the payload. NAT (Network Address Translator, RFC 1631) utilizes TCP/UDP ports to reuse public IPv4 addresses and is prevalent on today's Internet. Due to the approaching graduate entrance exam, two members of our group need to prepare for this exam, and as a result, we did not complete the required content for this project on time. However, this part is the foundation for implementing Project 4, which will be mentioned later.

In Project 4, we need to build on Project 3 and implement DNS functionality, as well as TCP handshake protocol and sliding window. Since we did not complete the tasks of Project 3 earlier, we need to first finish part of Project 3, implementing routers, NAT, and other functionalities. During the implementation of the DNS functionality, we got confused about the meanings of some IP and MAC addresses, resulting in a significant amount of time being consumed. Afterward, we started to write the functionality to ping websites, using a software called Wireshark to analyze packet captures. Wireshark is a network protocol analysis tool designed for capturing and analyzing network packets. It is an open-source software that offers robust network analysis capabilities, suitable for tasks such as troubleshooting, network optimization, and network security.Key features of Wireshark include: Packet Capture: Wireshark can capture and record all packets passing through a computer network interface in real-time. This allows users to view live network

communication and analyze the transmitted data. Packet Analysis: Wireshark provides detailed packet analysis functionality. Users can examine comprehensive information about each packet, including source and destination addresses, protocols, data content, and more. Protocol Support: Wireshark supports numerous network protocols, including common ones such as TCP, UDP, IP, HTTP, DNS, and also more advanced and application-layer protocols. Filtering and Searching: Users can use filters to narrow down displayed packets, focusing on specific protocols or communication streams. Additionally, Wireshark supports powerful search capabilities to quickly locate packets of interest. Statistics and Graphical Analysis: Wireshark offers statistical information, including traffic statistics, protocol distribution, etc. It also supports the generation of graphical analysis reports. Wireshark is a commonly used tool among network administrators, network engineers, security experts, and other professionals in the field of networking. Through Wireshark, users gain deep insights into the details of network communication, swiftly identify and resolve network issues, and perform network traffic analysis and security audits.

Up to this point, we have only reached this step, and we haven't been able to identify the issue with the packet, as we haven't seen the corresponding packet sent out on Wireshark, which is quite strange. It could be due to incorrect filtering conditions or potentially code errors. Nevertheless, time is running out.

This course has proven to be challenging for us, primarily because we are running the code in real-life scenarios, requiring consideration of numerous debugging issues related to software, hardware, etc. The debugging step usually consumes a significant amount of time. On the other hand, we believe that the project arrangement is effective in helping students understand the difficulties and problems faced in information transmission in computer networks, as well as some classical solutions to these issues. It helps students understand the core content of the course from bottom to top. However, the guidance in the course projects is lacking, leading us to spend a considerable amount of time exploring secondary content. We hope that future projects in this course can be improved in this aspect。