Day4_hw2 보고서(UDP통신)

20기 인턴 우정윤

1. 서론

현대 사회에서 네트워크 통신은 일상생활과 산업 전반에 걸쳐 필수적인 요소로 자리 잡고 있다. 스마트폰, 컴퓨터, 사물인터넷(IoT) 기기 등 수많은 장치들이 서로 데이터를 주고받으며, 이러한 데이터 교환을 가능하게 하는 기반 기술이 바로 네트워크 프로토콜이다. 특히, 인터넷에서 널리 사용되는 프로토콜 중 하나인 UDP(User Datagram Protocol)는 연결 설정 과정 없이 빠르게 데이터를 전송할 수 있는 특징으로 인해 다양한 분야에서 활용되고 있다.

UDP는 TCP(Transmission Control Protocol)와 달리 신뢰성보다는 전송 속도를 우선시하기 때문에, 음성 통화(VoIP), 동영상 스트리밍, 온라인 게임 등 실시간성이 중요한 응용 프로그램에서 자주 사용된다. 반면에 패킷 손실이나 순서 보장이 되지 않는다는 한계점도 존재한다. 그럼에도 불구하고 단순하고 가벼운 구조 덕분에, 신속한 데이터 교환이 필요한 환경에서는 여전히 중요한 역할을 수행한다.

본 보고서는 Qt 프레임워크의 QUdpSocket 클래스를 활용하여 UDP 통신 프로그램을 구현하고, 이를 통해 UDP의 특징을 직접 실험하고 분석하는 것을 목적으로 한다. 프로그램은 사용자가 간단한 메시지를 송수신할 수 있는 채팅 형태로 제작되었으며, 구현 과정에서 IP 자동 탐색, 소켓 바인딩, 데이터 송신 및 수신 기능 등을 포함하였다. 또한 실험을 통해 실제 통신 과정을 검증하고자 하였다.

2. 네트워크 통신 개요

2-1. 컴퓨터 네트워크의 기본 구조

네트워크란 여러 장치가 서로 연결되어 정보를 주고받을 수 있는 통신망을 말한다. 모든 네트워크는 노드와 노드를 연결하는 간선, 노드 간 주고받는 메시지로 구성된다. 노드는 정보를 주고 받을 수 있는 장치, 간선은 정보를 주고받을 수 있는 유무선의 통신 매체라고 할 수 있다.

네트워크의 가장자리에 위치한 노드는 네트워크를 통해 흐르는 정보를 최초로 생성 및 송신하고, 최종적으로 수신한다. 이는 서버 컴퓨터가 될 수도 있고, 개인 데스크톱, 노트북, 스마트폰이 될 수 있다. 이러한 가장자리 노드를 네트워크에서는 호스트라고 부른다. 때로는 호스트가 네트워크상에서 특정한 역할을 수행하기도 하는데, 대표적인 역할로는

서버와 클라이언트가 있다.

서버는 어떠한 서비스를 제공하는 호스트다. 여기서 어떠한 서비스는 파일이 될 수도 있고, 웹 페이지가 될 수도 있다. 반면 클라이언트란 서비스를 요청하고 서버의 응답을 제공받는 호스트를 말한다.

2-2. OSI 7계층 모델과 UDP의 위치

OSI 7계층이란 네트워크에서 통신이 일어나는 과정을 7단계로 나눈 것을 말한다.

1계층은 물리계층을 말한다. 전기적, 기계적, 기능적인 특성을 이용해서 통신 케이블로 데이터를 전송하게 된다. 통신 단위는 비트이며 이것은 1과 0으로 나타내어지는, 즉 전기적으로 ON, Off 상태라고 생각하면 된다. 이 계층에서는 단지 데이터를 전달만 할뿐 전송하려는 데이터가 무엇인지, 어떤 에러가 있는지 등에는 신경을 쓰지 않는다. 단지 데이터를 전기적인 신호로 변환해서 주고받는 기능만 수행한다. 대표적인 장비로는 통신 케이블, 리피터, 허브 등이 있다.

2계층은 데이터 링크계층은 물리계층을 통해 송수신되는 정보의 오류와 흐름을 관리하여 안전한 정보의 전달을 수행할 수 있도록 도와주는 역할을 한다. 따라서 통신에서의 오류도 찾아주고 재전송도 하는 기능을 가지고 있는 것이다. 2계층에서는 맥 주소를 가지고 통신하게 된다. 2계층에서 전송되는 단위를 프레임라고 하며 대표적인 장비로는 브리지, 스위치 등이 있다. 데이터 링크 계층(Data link layer)은 포인트 투 포인트(Point to Point) 간 신뢰성있는 전송을 보장하기 위한 계층으로 CRC 기반의 오류 제어와 흐름 제어가 필요하다. 네트워크 위의 개체들 간 데이터를 전달하고, 물리 계층에서 발생할 수 있는 오류를 찾아 내고, 수정하는 데 필요한 기능적, 절차적 수단을 제공한다. 주소 값은 물리적으로 할당 받는데, 이는 네트워크 카드가 만들어질 때부터 맥 주소(MAC address)가 정해져 있다는 뜻이다. 주소 체계는 계층이 없는 단일 구조이다. 데이터 링크 계층의 가장 잘 알려진 예는 이더넷이다. 이 외에도 HDLC나 ADCCP 같은 포인트 투 포인트(point-to-point) 프로토콜이나 패킷 스위칭 네트워크나 LLC, ALOHA 같은 근거리 네트워크용 프로토콜이 있다. 네트워크 브릿지나 스위치 등이 이 계층에서 동작하며, 직접 이어진 곳에만 연결할 수 있다.

3계층은 네트워크 계층이다. 3계층에서 가장 중요한 기능은 데이터를 목적지까지 가장 안전하고 빠르게 전달하는 기능이다. 여기에 사용되는 프로토콜의 종류도 다양하고, 라우팅하는 기술도 다양하다. 이 계층은 경로를 선택하고 주소를 정하고 경로에 따라 패킷을 전달해주는 것이 3계층의 역할이다. 이 계층은 대표적인 장비는 라우터이며, 요즘은 2계층의 장비 중 스위치라는 장비에 라우팅 기능을 장착한 layer 3 스위치도 있다. 네트워크 계층은 여러 개의 노드를 거칠때마다 경로를 찾아주는 역할을 하는 계층으로 다양한 길이의 데이터를 네트워크들을 통해 전달하고, 그 과정에서 전송 계층이 요구하는 서비스 품질을 제공하기 위한 기능적, 절차적 수단을 제공한다.

네트워크 계층은 라우팅, 흐름 제어, 세그멘테이션, 오류 제어, 인터네트워킹등을 수행한다. 라우터가 이 계층에서 동작하고 이 계층에서 동작하는 스위치가 있다. 데이터를 연결하는 다른 네트워크를 통해 전달함으로써 인터넷이 가능하게 만드는 계층이다. 논리적인 주소 구조, 곧 네트워크 관리자가 직접 주소를 할당하는 구조를 가지며 계층적이다. 서브네트의 최상위 계층으로 경로를 설정하고, 청구 정보를 관리한다. 개방형 시스템들의 사이에서 네트워크 연결을 설정, 유지, 해제하는 기능을 부여하고, 전송 계층 사이에 네트워크 서비스 데이터 유닛을 교환하는 기능을 제공한다.

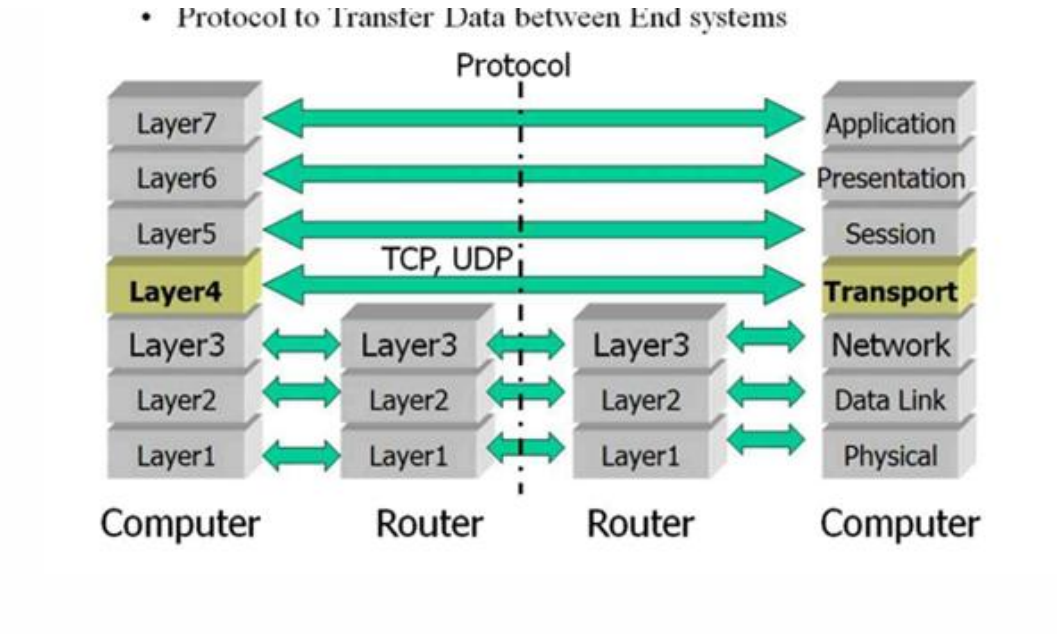
4계층은 전송 계층으로서 통신을 활성화하기 위한 계층이다. 보통 TCP 프로토콜을 이용하며, 포트를 열어서 응용 프로그램들이 전송을 할 수 있게 한다. 만약 데이터가 왔다면 4레층에서 해당 데이터를 하나로 합쳐서 5계층에 넘겨준다. 전송 계층은 양 끝단의 사용자들이 신뢰성있는 데이터를 주고 받을 수 있도록 해 주어, 상위 계층들이 데이터 전달의 유효성이나 효율성을 생각하지 않도록 해준다. 시퀀스 넘버 기반의 오류 제어 방식을 이용한다.

5계층은 세션 계층으로 데이터가 통신하기 위한 논리적인 연결을 말한다. 하지만 4계층에서도 연결을 맺고 종료할 수 있기 때문에 우리가 어느 계층에서 통신이 끊어 졌나 판단하기는 한계가 있다. 그러므로 세션 계층은 4계층과 무관하게 응용 프로그램 관점에서 봐야 한다. 세션 설정, 유지, 종료, 전송 중단시 복구 등의 기능이 있다.

세션 계층은 양 끝단의 응용 프로세스가 통신을 관리하기 위한 방법을 제공한다. 동시 송수신 방식, 반이중 방식, 전이중 방식의 통신과 함께, 체크 포인팅과 유휴, 종료, 다시 시작등의 과정을 수행한다.

6계층은 표현 계층으로 데이터 표현이 상이한 응용 프로세스의 독립성을 제공하고 암호화 한다. 표현 계층은 코드 간의 번역을 담당하여 사용자 시스템에서 데이터의 형식상 차이를 다루는 부담을 응용 계층으로부터 덜어 준다. MIME 인코딩이나 암호화 등의 동작이 이 계층에서 이루어진다.

7계층은 응용 계층이다. 최종 목적지로서 HTTP, FTP, SMTP, POP3, IMAP, TELENET 등과 같은 프로토콜이 있다. 해당 통신 패킷들은 방금 나열된 프로토콜에 의해 모두 처리되며 우리가 사용하는 브라우저나, 메일 프로그램은 프로토콜을 보다 쉽게 사용하게 해주는 응용프로그램이다.



UDP의 위치는 4계층으로 전송 계층 프로토콜 중 하나이다. 같은 전송 계층에 속하는 프로토콜로 TCP가 있으며 둘 다 상위 계층에서 내려온 데이터를 하위 계층으로 전달하는 역할을 한다.

2-3. TCP vs UDP

프로토콜 종류	TCP	UDP
연결 방식	연결형 서비스 (패킷 교환 방식)	비연결형 서비스 (데이터그램 방식)
전송 순서	전송 순서 보장	전송 순서가 바뀔 수 있음
수신 여부 확인	수신 여부를 확인함	수신 여부를 확인하지 않음
통신 방식	1:1 통신	1:1 OR 1:N or N:N 통신
신뢰성	높다	낮다
속도	느리다	빠르다

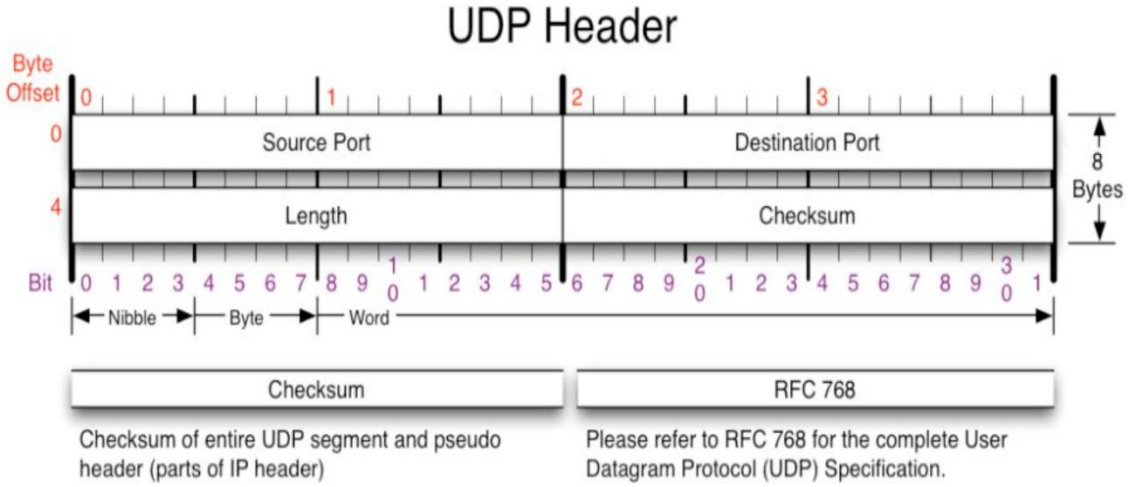
TCP의 특징으로는 연결 지향 방식으로 패킷 교환 방식을 사용한다. 3-WAY HANDSHAKING 과정을 통해 연결을 설정하고 4-WAY HANDSHAKING을 통해 해제한다. 흐름을 제어 및 혼잡 제어를 통해 높은 신뢰성을 보장하지만 UDP보다 속도가 느리다는 단점이 있다. UDP의 특징으로는 비연결형 서비스로 데이터그램 방식을 제공한다. 정보를 주고 받을 때 정보를 보내거나 받는다는 신호절차를 거치지 않는다. 다만 UDP 헤더의 CHECKSUM 필드를 통해 최소한의 오류만 검출하기에 신뢰성이 낮지만 TCP보다 속도가 빠르다. 따

라서 신뢰성보다는 연속성이 중요한 스트리밍 서비스 등에 사용된다.

3. UDP 프로토콜 이론

3-1. UDP 헤더 구조

.



출발지 포트(16Bit)	목적지 포트(16Bit)	길이(16Bit)	오류 검사(16Bit)

전송 단위가 데이터그램인 UDP 헤더의 길이는 그림과 같이 8바이트다. 출발지 포트 번호 항목과 목적지 포트 번호 항목은 각각 16비트 길이를 이룬다. 그 다음 나오는 길이 항목과 오류검사 항목도 각각 16비트 길이를 이룬다. 이에 따라 UDP 헤더의 길이는 총 64비트이다.

출발지 포트 번호 항목과 목적지 포트 번호 항목에는 각각 출발지 포트 번호 정보와 목적지 포트 번호 정보가 있다. 길이 항목에는 응용 계층에서 생성한 UDP페이로드와 전송 계층에서 생성한 UDP헤더가 더해진 데이터그램 길이 정보가 있고, 오류 검사 항목은 일반적으로 비활성 상태다. 하지만 TCP/IP 소켓을 개발할 경우 오류검사 항목의 활성 상태와 비활성 상태에 따라 데이터그램의 모습이 다르다는 점을 인식해야 한다.

오류 검사 항목이 비활성 상태인 경우 데이터그램은 UDP 페이로드와 UDP 헤더로 이루어지게 되고 오류 검사 항목이 활성 상태인 경우 데이터그램의 모습은 UDP 페이로드 UDP 헤더 가상헤더로 이루어진다. UDP헤더의 길이는 8바이트지만 가상 헤더의 길이는 12바이트다. 가상 헤더의 구조는 다음과 같다.

<table>
<tr><td colspan="3">출발지 IP(32)</td></tr>
<tr><td colspan="3">목적지 IP(32)</td></tr>
<tr><td>제로(8)</td><td>프로토콜 ID(8)</td><td>길이(16)</td></tr>
</table>

가상 헤더는 위의 그림과 같은 구조로서 출발지 IP주소 항목과 목적지 IP주소 항목은 각각 32비트이고 제로 항목과 프

로토콜 ID 항목은 각각 8비트이고 길이 항목은 16비트다. 이에 따라 가상 헤더의 길이는 96비트다.

출발지 IP주소 항목과 목적지 IP 주소 항목에는 각각 출발지 IP 주소 정보와 목적지 IP주소 정보가 있다. 또한 제로 항목은 모두 0이고, 프로토콜 ID 항목에는 UDP 또는 TCP 식별자 정보가 있고, 길이 항목에는 프로토콜 ID 항목에 따른 UDP데이터 그램의 길이 정보 또는 TCP세그먼트 길이 정보가 있다.

3-2. 주요 특징

UDP는 전송 계층의 프로토콜 중 하나로서 비연결형 통신 프로토콜이다. 비연결형 통신이라서 데이터를 전송 할 때 TCP처럼 확인 작업을 일일히 하지 않는다. UDP는 효율성을 중요하게 여기는 프로토콜이라서 스트리밍 방식으로 전송하는 동영상 서비스와 같은 곳에 사용된다. UDP의 주요 특징으로는 첫째 실시간이 있다. UDP 통신은 제약 조건이 거의 없고 TCP에 비해 매우 빨라, 실시간 전송이 필요한 부분에 대해서 많이 사용된다. 두번째, 간단한 트랜젝션이 있다. 같은 전송 계층인 프로토콜 TCP와 비교했을 때 TCP는 SETUP, 종료, ACK로 이루어지는 복잡한 변환이 요구되지만 UDP는

단순하다는 특징이 있다. 셋째, 멀티캐스트 및 브로드캐스트가 가능하다는 특징이 있다. TCP는 전송측과 수신측이 서로 검증이 완료가 되어야 보낼 수 있으나 UDP는 검증을 하지 않는다는 특징이 있다.

3-3. 활용 사례

3-3-1 네트워크 서비스

- DNS(Domain Name System)

도메인 이름을 IP 주소로 변환할 때 사용하며 청/응답이 매우 짧아서 TCP 연결보다 UDP가 효율적이다.

- DHCP(Dynamic Host Configuration Protocol)

네트워크에 접속할 때 자동으로 IP 주소를 할당하는 프로토콜이며 브로드캐스트 통신을 이용하기 때문에 UDP 기반으로 동작한다.

- SNMP(Simple Network Management Protocol)

네트워크 장비 상태 모니터링을 말하며 빠른 전송을 위해 UDP를 사요한다.

3-3-2 실시간 스트리밍

- VoIP(인터넷 전화)

  음성 데이터는 순서보다 실시간성이 중요하기 때문에 UDP가 적합하여 사용되며 실시간 영상 스트리밍도 마찬가지로 프레임이 몇 개 손실되어도 끊김 없이 전송하는 것이 중요하기 때문에 UDP 통신이 사용된다.

3-3-3 온라인 게임

FPS, MMORPG, MOBA 게임들을 플레어 위치, 공격 정보 등이 빠르게 전달되어야 하며 약간의 패킷 손실보다 속도가 중요하기에 UDP통신을 사용한다.

3-3-4 사물인터넷

- 센서 네트워크 데이터 전송
- 스마트홈 기기 제어

조명 및 다양한 센서값들을 전달한다.

3-3-5 방송 및 미디어

- IPTV

UDP 멀티캐스트로 여러 사용자에게 동시에 송출한다.

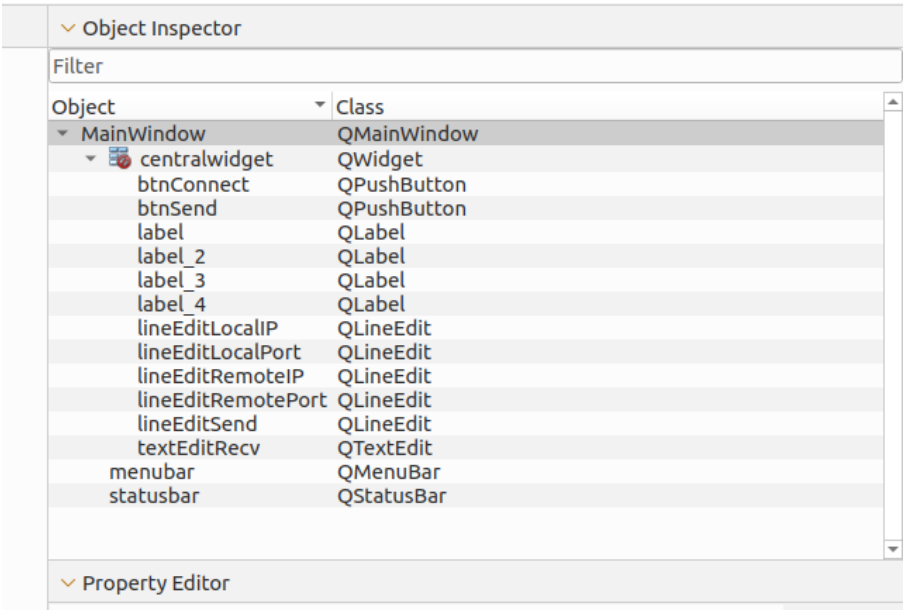
- 라디오 스트리밍

실시간성이 중요하고 약간의 손실이 허용 가능하므로 사용된다.

4. UDP 통신 구현

UDP 통신은 소켓을 생성한 뒤 클라이언트와 서버가 데이터를 직접 주고받는 방식으로 구현된다. 주요 단계는 서버에서 소켓을 생성하고 IP와 포트 번호로 바인딩한 뒤 클라이언트의 데이터를 기다리거나, 클라이언트에서 소켓을 생성하고 서버의 IP와 포트 번호로 데이터를 전송한다.

5. 구현 실험 및 상세



UDP 통신을 구현하기 위해 QT CREATOR을 활용하여 UI를 제작하여 채팅 프로그램을 제작하여 통신을 해보고자 하였다. 우선 수신 설정을 위하여 lineEditLocalIP, lineEditLocalPort을 추가하였으며 송신 대상을 설정하기 위해 lineEditionRemoteIP와 lineEditRemotePort를 사용하였다. 전송할 메시지를 입력할 메시지 창을 구현하기 위해서 lineEditSend를 사용하였으며 송수신 로그를 표시하기 위해 textEditRecv, IP끼리 연결하는 바인드를 실행하기 위한 btnConnect(다시 클릭시 해제하는 기능도 포함되어 있음), 메시지 송신을 위한 btnSend가 UI를 구성하고 있다.

동작 흐름은 다음과 같이 이루어진다.

1. 실행 시 UI를 초기화 하고 로컬 IP를 자동으로 표시해준다.

```
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
    , udpSocket(new QUdpSocket(this))
{
    ui->setupUi(this);

    // 내 IP 자동 표시
    ui->lineEditLocalIP->setText(getLocalIP());
    ui->textEditRecv->setReadOnly(true);

    connect(udpSocket, &QUdpSocket::readyRead, this, &MainWindow::udpDataReceived);
}
```

2. 연결 버튼을 클릭하면 bind되어 수신을 준비하게 된다.

```
void MainWindow::on_btnConnect_clicked()
{
    localPort = ui->lineEditLocalPort->text().toUShort();
    remoteAddr = QHostAddress(ui->lineEditRemoteIP->text());
    remotePort = ui->lineEditRemotePort->text().toUShort();

    if (udpSocket->bind(QHostAddress::Any, localPort)) {
        displayMessage("UDP 소켓 바인드 완료 (포트 " + QString::number(localPort) + ")", "SYSTEM");
    } else {
        displayMessage("UDP 소켓 바인드 실패", "SYSTEM");
    }
}
```

3. 보내기 버튼을 클릭하면 UDP송신을 하고 보낸 내역을 기록창에 남긴다.

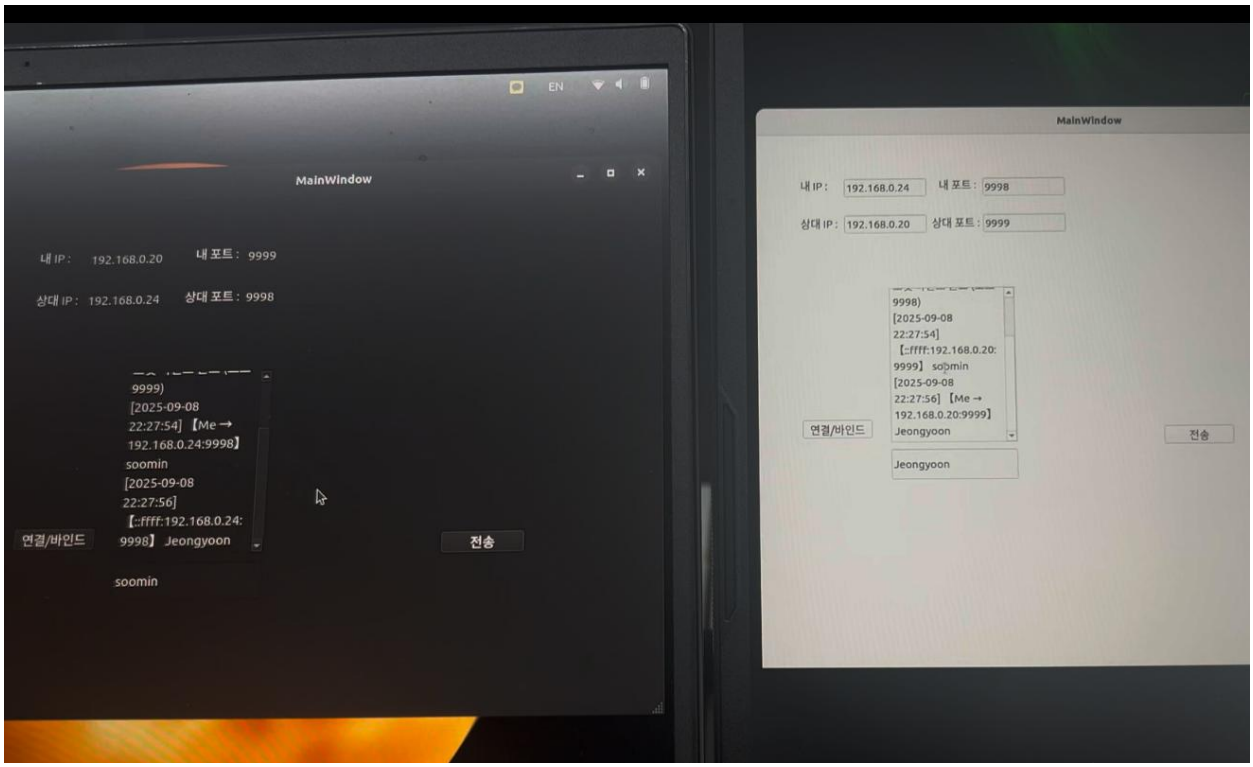
```
void MainWindow::udpDataReceived()
{
    while (udpSocket->hasPendingDatagrams()) {
        QByteArray datagram;
        datagram.resize(udpSocket->pendingDatagramSize());
        QHostAddress sender;
        quint16 senderPort;

        udpSocket->readDatagram(datagram.data(), datagram.size(), &sender, &senderPort);

        QString msg = QString::fromUtf8(datagram);
        displayMessage(msg, sender.toString() + ":" + QString::number(senderPort));
    }
}
```

4. 수신 이벤트가 발생하면 패킷을 읽고 이를 타임스탬프가 포함된 로그로 출력한다.

UI는 수신창을 읽기 전용으로 설정하여 로그 영역의 입력 오류를 방지하였으며 로컬 IP를 자동 탐색 하는 원리는 시스템의 모든 IP주소를 연결하여 IPV4인 첫 주소를 반환한다. 바인드하여 해당 프로세스가 특정 로컬 포트에서 오는 UDP데이터그램을 수신하도록 커널에 등록하면 성공 시점 이후 도착하는 어떤 송신자든 대상 UDP 패킷은 비동기로 전달된다.

수신의 경우 대기 패킷 존재를 확인하게 되면 pendingDatagramSize()로 데이터그램의 정확한 크기를 알아내고 이를 해당 크기에 맞게 배열을 할당한다. 그리고 이를 동일한 포맷으로 누적하여서 타임스탬프를 포함해 출력한다.

6. 실행 결과



다음 사진은 실행 결과이다. 우선 내 IP는 자동으로 불러와 지며 로컬 포트를 지정해준다. 그 다음 상대의 IP와 포트를 타겟으로 각각 설정해 준 다음 연결 버튼을 클릭하면 바인드 된다. 바인드 된 후 채팅을 발신하면 발신 내역 및 시간이 채팅창에 뜨게 되고 수신 시에도 마찬가지로 수신 내역 및 시간이 채팅창에 뜨게 되며 채팅이 정상적으로 수행되는 것을 볼 수 있다.

7. 결론

이번 실험을 통해 컴퓨터 네트워크에서 UDP(User Datagram Protocol)의 동작 원리와 특징을 확인할 수 있었다. UDP는 OSI 7계층 구조 중 전송 계층(4계층)에 위치하며, TCP와 달리 연결 설정 없이 데이터를 전송하는 비연결형 프로토콜임을 알 수 있었다.

또한 신뢰성보다는 속도를 중시하기 때문에, 패킷 손실이나 순서 보장이 필요 없는 상황에서 효율적으로 사용될 수 있음을 확인하였다.

대표적으로 DNS, DHCP, 스트리밍 서비스, 온라인 게임 등과 같이 실시간성이 중요한 분야에서 UDP가 널리 활용되고 있음을 보고서를 통해 학습하였다.

실험에서는 UDP 소켓을 이용하여 송신 프로그램과 수신 프로그램 간의 데이터 전송을 확인하였다. 데이터는 연결 과정 없이 즉시 전송되었으며, 응답 지연이 매우 짧음을 확인하였다. 전송 중 일부 패킷이 손실되거나 순서가 뒤바뀌는 경우가 있었는데, 이는 UDP의 비신뢰성 특성을 잘 보여준다. 그러나 짧은 지연과 빠른 처리 속도로 인해 실시간 통신이 가능하다는 장점을 확인할 수 있었다.

따라서 본 실험을 통해 UDP는 빠른 전송 속도와 간단한 구조를 제공하지만, 신뢰성을 보장하지 않기 때문에 응용 계층에서 별도의 보완이 필요하다는 점을 결론으로 도출할 수 있다.