

Elearning 2

UDP (User Datagram Protocol) là giao thức tầng giao vận không kết nối, không đảm bảo tin cậy và không kiểm soát luồng. Do đó, việc **tối ưu hóa truyền thông UDP** là cần thiết trong các ứng dụng đòi hỏi độ tin cậy cao (ví dụ: truyền video, game online, IoT,...).

Mục tiêu của demo này là **mô phỏng các kỹ thuật tối ưu hóa UDP** nhằm giảm mất mát dữ liệu và tăng độ tin cậy của quá trình truyền.

Các kỹ thuật được sử dụng

1. **Đánh số thứ tự gói tin (Sequence Number)**

→ Giúp xác định và phân biệt từng gói tin, hỗ trợ kiểm tra ACK chính xác.

2. **Cơ chế xác nhận (ACK)**

→ Server gửi “ACK|seq” để xác nhận gói đã nhận, tăng độ tin cậy cho UDP.

3. **Gửi lại khi mất ACK (Retransmission / Stop-and-Wait ARQ)**

→ Nếu Client không nhận được ACK sau thời gian chờ, tự động gửi lại gói.

4. **Giả lập mất gói (Packet Loss Simulation)**

→ Server bỏ ngẫu nhiên 20% gói để mô phỏng môi trường mạng không ổn định.

5. **Timeout và Retry**

→ Giới hạn thời gian chờ (1000ms) và số lần gửi lại (3 lần) để tránh treo chương trình.

6. **Hiển thị trạng thái bằng màu sắc (Color Logging)**

→ Giúp dễ nhận biết log: gửi, nhận, mất gói, ACK, thất bại.

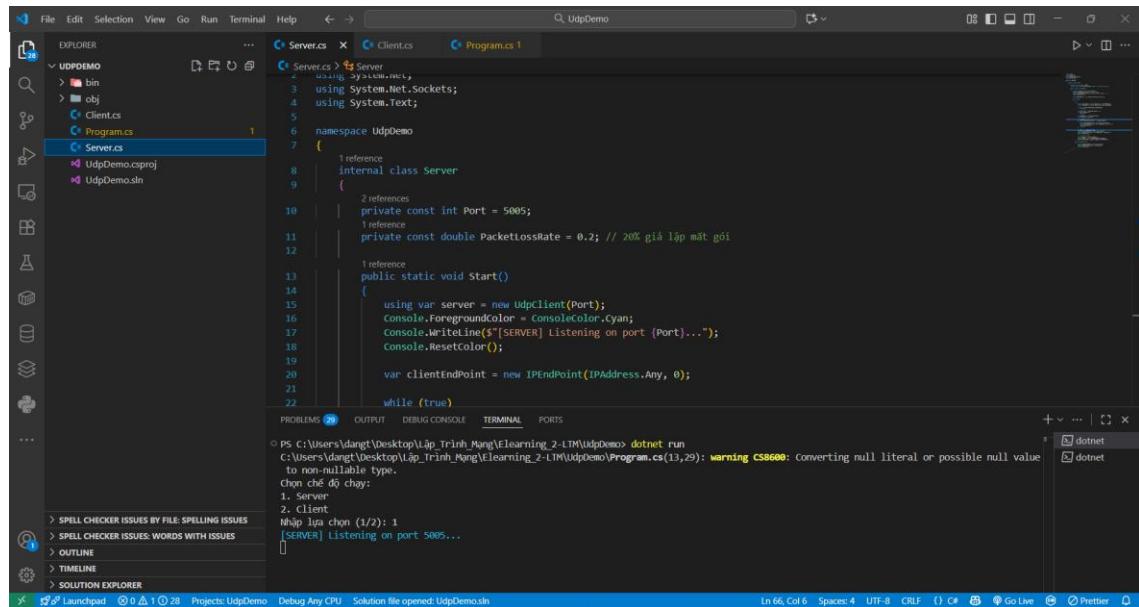
7. Chèn độ trễ (Thread.Sleep)

→ Giảm tải CPU và giúp hiển thị log rõ ràng hơn.

Sài 2 terminal

Dùng câu lệnh dotnet run

1. Sever đợi phản hồi từ client



The screenshot shows the Visual Studio IDE interface with the following details:

- File Explorer:** Shows the project structure for "UDPDemo" with files like bin, obj, Client.cs, Program.cs, and Server.cs.
- Code Editor:** Displays the content of Server.cs:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace UdpDemo
{
    internal class Server
    {
        private const int Port = 5005;
        private const double PacketLossRate = 0.2; // 20% giả lây mất gói

        public static void Start()
        {
            using var server = new UdpClient(Port);
            Console.ForegroundColor = ConsoleColor.Cyan;
            Console.WriteLine($"[SERVER] Listening on port {Port}...");
            Console.ResetColor();

            var clientEndPoint = new IPEndPoint(IPAddress.Any, 0);
            while (true)
            {
                
```
- Terminal:** Shows the command "dotnet run" being run in the terminal, resulting in the output "[SERVER] Listening on port 5005..." and a warning message about CS8600.
- Bottom Status Bar:** Shows the current file is "Program.cs", the line number is 66, column 6, and the character count is 4.

2. Client gửi yêu cầu về sever

The screenshot shows the Visual Studio IDE interface with the following details:

- File Explorer:** Shows the project structure for "UDPDemo" with files: bin, obj, Client.cs, Program.cs, Server.cs, UdpDemo.csproj, and UdpDemo.sln.
- Code Editor:** Displays the content of Server.cs. The code defines a class Server with a static method Start() that creates a UdpClient on port 5005 and begins receiving messages.
- Terminal:** Shows the command "dotnet run" being executed in the terminal window.
- Output:** Displays the application's log output, showing the server sending "Xin chào" (Hello) and receiving ACKs from clients.
- StatusBar:** Shows the path "C:\Users\dangt\Desktop\Lập Trình Mạng\Elearning_2-LTM\UdpDemo", the solution name "UdpDemo", and other status indicators like "In 66 Col 6" and "UTF-8".

3. sever nhận được yêu cầu từ client

The screenshot shows the Visual Studio IDE interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Toolbar:** Standard icons for file operations like Open, Save, Print, etc.
- EXPLORER:** Shows the project structure for "UDPDemo".
- CODE EDITOR:** Displays the content of "Server.cs".
- TERMINAL:** Shows the command line output of "dotnet run".
- OUTPUT:** Shows build logs.
- DEBUG CONSOLE:** Shows the application's log output.
- PROBLEMS:** Shows spelling checker issues.
- SOLUTION EXPLORER:** Shows the project files.
- STATUS BAR:** In 66 Col 6 Spacing 4 UTF-8 CRLF ⌂ Online ⌂ Preview ⌂

```
Server.cs
1  // Using statements
2  using System;
3  using System.Net.Sockets;
4  using System.Text;
5
6  namespace Udpdemo
7  {
8      // Internal class Server
9      internal class Server
10     {
11         // References
12         private const int Port = 5005;
13         private const double PacketLossRate = 0.2; // 20% giả lặp mất gói
14
15         // Methods
16         public static void Start()
17         {
18             using var server = new UdpClient(Port);
19             Console.ForegroundColor = ConsoleColor.Cyan;
20             Console.WriteLine("RECV'D (listening on port {0})", Port);
21
22             // Read loop
23             while (true)
24             {
25                 var message = server.Receive();
26                 var data = Encoding.UTF8.GetString(message.Buffer);
27
28                 if (data == "STOP")
29                 {
30                     break;
31                 }
32
33                 // Process message
34                 var response = "ACK";
35                 var packet = Encoding.UTF8.GetBytes(response);
36                 server.Send(packet, packet.Length);
37             }
38         }
39     }
40 }
```

Terminal Output:

```
PS C:\Users\duong\Desktop\Lập Trình Mạng\Elearning_2-LTM\UdpDemo> dotnet run
C:\Users\duong\Desktop\Lập Trình Mạng\Elearning_2-LTM\UdpDemo>Program.cs(13,29): warning CS8600: Converting null literal or possible null value to non-nullable type.
Chon ché đc chạy:
1. Server
2. Client
Nhập lựa chọn (1/2): 1
[SERVER] Listening on port 5005...
[LOSS] Gói 0 bị bỏ qua.
[RECV] Gói 0: Xin chào
[RECV] Gói 1: Tối ưu UDP
[RECV] Gói 2: Mật gói mờ phỏng
[RECV] Gói 3: ACK/NACK mìn họa
[RECV] Gói 4: Kết thúc truyền
```