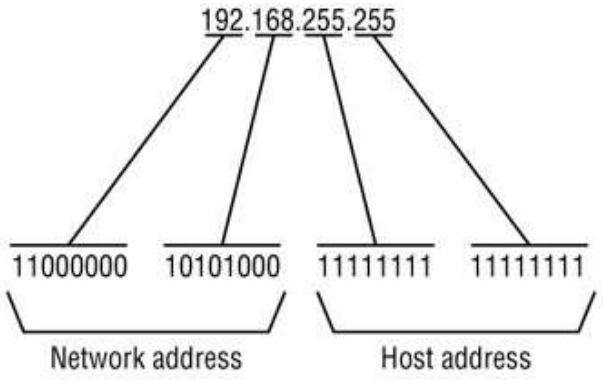
CHƯƠNG 5
LẬP TRÌNH IP MULTICASTING

Thời gian: 3 tiết

BROADCAST

- Dùng để gửi một gói tin đến tất cả các nút trong mạng
- Để thực hiện hình thức quảng bá, địa chỉ đến của gói tin sẽ là địa chỉ quảng bá.
- Có hai loại là:
  - Local BroadCast
  - Global BroadCast



- Ví dụ: Cho mạng lớp B có địa chỉ IP là 192.168.0.0 có SubNet Mask là 255.255.0.0, địa chỉ Local Broadcast là: 192.168.254.255

MULTICAST

- IP Multicast kế thừa
  - Cho phép một ứng dụng gửi gói tin tới một thiết bị trong cả local subnet và mạng khác
  - Cho phép một chương trình kết nối tới nhóm multicast (multicast group) thực hiện các hội nghị trên diện rộng (wide area conference)
- Là công nghệ truyền thông trên nền tảng IP
- Là công nghệ băng thông rộng nhằm làm giảm lưu lượng trong việc phân phối dòng dữ liệu cho nhiều người

MULTICAST

- Khai thác hiệu quả môi trường mạng bằng cách gửi các gói tin
  - Một gói tin có thể chia thành nhiều gói tin gửi đến nhiều người nhận
  - Các nút mạng tái tạo, chuyển tiếp tới người nhận
- IP multicast dùng những địa chỉ IP đặc biệt
- Các dải địa chỉ IP tạo ra các nhóm multicast khác nhau
- Mỗi nhóm multicast bao gồm một nhóm thiết bị đang lắng nghe cùng một địa chỉ IP

BROADCAST VÀ MULTICAST

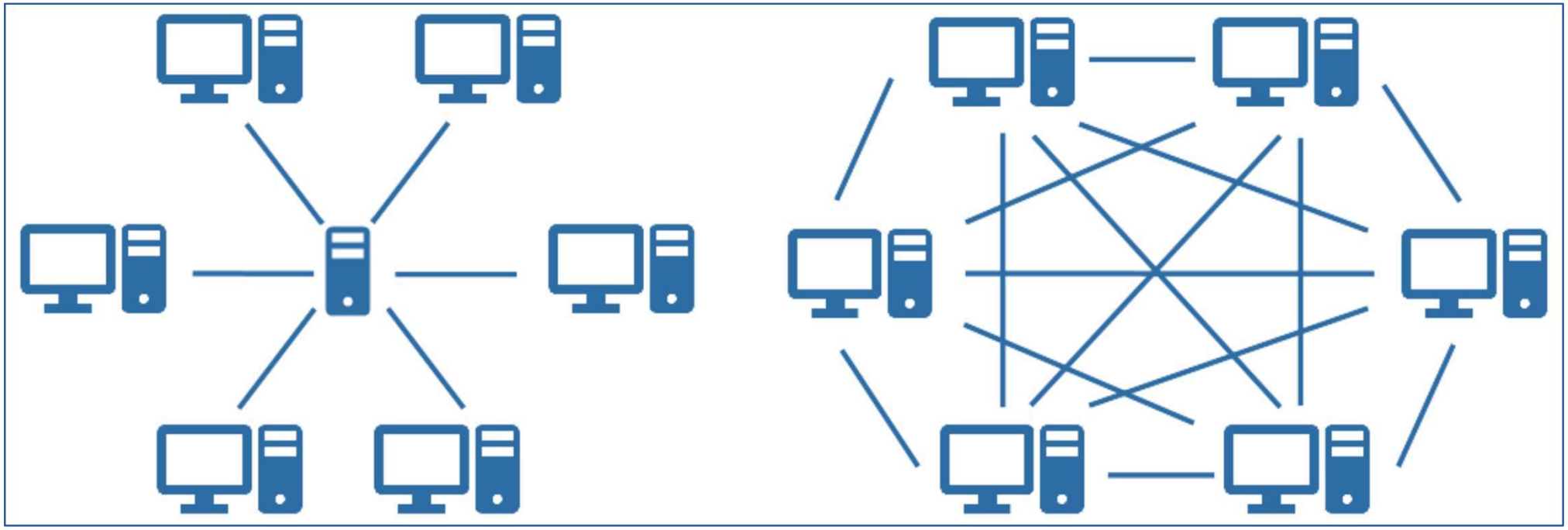
- Vì một gói tin gửi tới đích là địa chỉ nhóm nên mỗi thiết bị phải "lắng nghe địa chỉ" đó để nhận tin
- Không thể gửi một gói tin broadcast với TCP
  - Giao thức TCP yêu cầu 2 thiết bị phải có kết nối tin cậy
- UDP được dùng bởi vì giao thức này có khả năng gửi gói tin mà không cần khởi tạo một kết nối đặc biệt
- Dải địa chỉ 224.0.0.1 đến 239.255.255.255 được gọi là địa chỉ nhóm Multicast

MỘT SỐ ĐỊA CHỈ MULTICAST

Địa chỉ	Chức năng
224.0.0.0	Địa chỉ cơ sở
224.0.0.1	Tất cả các hệ thống trên mạng con này
224.0.0.2	Tất cả các Router trên mạng con này
224.0.0.5	Các DR trong OSPF
224.0.1.9	Nhóm địa chỉ RIPv2
224.0.1.24	Nhóm địa chỉ WINS server

CÁC KỸ THUẬT MULTICAST

- Peer-to-peer: Tất cả các client có thể gửi thông điệp đến tất cả client khác trong nhóm
- Central sever: Server gửi thông điệp tới nhóm client



Central Sever **Peer-to-peer**

MULTICAST - PEER – TO-PEER

- Tất cả các client trong nhóm multicast đều có quyền ngang nhau
- Bất kỳ client đều có khả năng trao đổi thông điệp với client khác trong nhóm
- Hệ thống IP hỗ trợ nhóm multicast theo peer-to-peer cho phép mọi thiết bị trên mạng gửi và nhận gói tin có đích là địa chỉ nhóm multicast
- Một số yếu tố mã hoá để ngăn chặn client nặc danh nhận dữ liệu trong nhóm nhưng vẫn không có cách để nhận xác thực từ client về dữ liệu

MULTICAST - CENTRAL SERVER

- Hệ thống dùng một thiết bị trên mạng để điều khiển toàn bộ hoạt động của nhóm multicast, gọi là Central Server
- Một client muốn kết nối vào nhóm phải xin quyền từ central server
  - Nếu central server từ chối cho client truy cập nhóm thì không một gói tin nào được chuyển tiếp tới client bị cấm
- Lưu ý: Không được hỗ trợ bởi IP. Hiện nay chỉ có mạng ATM có hỗ trợ nhóm multicast

MULTICAST - SOCKET

- Socket option có thể được sử dụng để
  - Thêm một Socket vào nhóm Multicast
  - Loại một Socket khỏi nhóm Multicast
- SetSocketOption(SocketOptionLevel,SocketOptionName, optionValue)
  - SocketOptionName
    - AddMembership
    - DropMembership

MULTICAST - SOCKET (2)

- optionValue là một đối tượng của lớp MulticastOption
  - MulticastOption(IPAddress)
  - MulticastOption(IPAddress,IPAddress)
- Ví dụ thêm một Socket vào nhóm Multicast 224.100.0.1

```
sock.SetSocketOption(SocketOptionLevel.IP,
                     SocketOptionName.AddMembership,
                     new MulticastOption(IPAddress.Parse("224.100.0.1")));
```

MULTICAST - UDPCLIENT

- Multicast với lớp UdpClient
  - JoinMulticastGroup()
  - DropMulticastGroup()
- JoinMulticastGroup() là phương thức overload
  - JoinMulticastGroup(IPAddress)
  - JoinMulticastGroup(IPAddress, int)

CHƯƠNG TRÌNH MINH HỌA

DEMO 1-BROADCAST-SOCKET-SENDING

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class BadBroadcast
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Broadcast, 9050);
        byte[] data = Encoding.ASCII.GetBytes("This is a test message");
        sock.SendTo(data, iep);
        sock.Close();
    }
}
```

Chương trình không tốt ? Vì theo mặc định, Socket không được phép gửi tin nhắn quảng bá

DEMO 1 (2)

- Dùng kỹ thuật BroadCasting để gửi dữ liệu đến nhiều máy trong mạng cục bộ
- Đối với một ứng dụng C # để gửi các gói dữ liệu Broadcast, sử dụng phương thức SetSocketOption() của lớp Socket :

```
Socket sock = new
Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

sock.SetSocketOption(SocketOptionLevel.Socket,
SocketOptionName.Broadcast, 1);
```

DEMO 2-BROADCAST-SOCKET-SENDING

```
class Broadcast
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
        ProtocolType.Udp);
        IPEndPoint iep1 = new IPEndPoint(IPAddress.Broadcast, 9050);
        IPEndPoint iep2 = new IPEndPoint(IPAddress.Parse("192.168.1.255"), 9050);
        string hostname = Dns.GetHostName();
        byte[] data = Encoding.ASCII.GetBytes(hostname);
        sock.SetSocketOption(SocketOptionLevel.Socket,
        SocketOptionName.Broadcast, 1);
        sock.SendTo(data, iep1);
        sock.SendTo(data, iep2);
        sock.Close();
    }
}
```

Dùng kỹ thuật BroadCasting để gửi dữ liệu đến nhiều máy trong mạng cục bộ

DEMO 3-BROADCAST-SOCKET-RECEIVING

```
class RecvBroadcast
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
        ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        sock.Bind(iep); EndPoint ep = (EndPoint)iep;
        Console.WriteLine("Ready to receive…"); byte[] data = new byte[1024];
        int recv = sock.ReceiveFrom(data, ref ep);
        string stringData = Encoding.ASCII.GetString(data, 0, recv);
        Console.WriteLine("received: {0} from: {1}", stringData, ep.ToString());
        data = new byte[1024]; recv = sock.ReceiveFrom(data, ref ep);
        stringData = Encoding.ASCII.GetString(data, 0, recv);
        Console.WriteLine("received: {0} from: {1}", stringData, ep.ToString());
        sock.Close();
    }
}
```

DEMO 4-MULTICAST-SOCKET-SENDING

```
class MultiSend
{
    public static void Main()
    {
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
        byte[] data = Encoding.ASCII.GetBytes("This is a test message");
        server.SendTo(data, iep);
        server.Close();
    }
}
```

DEMO 5-MULTICAST-SOCKET-RECEIVING

```
class MultiRecv
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
        ProtocolType.Udp);
        Console.WriteLine("Ready to receive…");
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        EndPoint ep = (EndPoint)iep;
        sock.Bind(iep);
        sock.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.AddMembership,
        new MulticastOption(IPAddress.Parse("224.100.0.1")));
        byte[] data = new byte[1024];
        int recv = sock.ReceiveFrom(data, ref ep);
        string stringData = Encoding.ASCII.GetString(data, 0, recv);
        Console.WriteLine("received: {0} from: {1}", stringData, ep.ToString());
        sock.Close();
    }
}
```

DEMO 6-MULTICAST TTL-TIME TO LIVE

```
class NewMultiSend
{
    public static void Main()
    {
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9051);
        IPEndPoint iep2 = new IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
        server.Bind(iep);
        byte[] data = Encoding.ASCII.GetBytes("This is a test message");
        server.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.AddMembership,
        new MulticastOption(IPAddress.Parse("224.100.0.1")));
        server.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.MulticastTimeToLive, 50);
        server.SendTo(data, iep2);
        server.Close();
    }
}
```

DEMO 7-MULTICAST-UDPCLIENT-SENDING

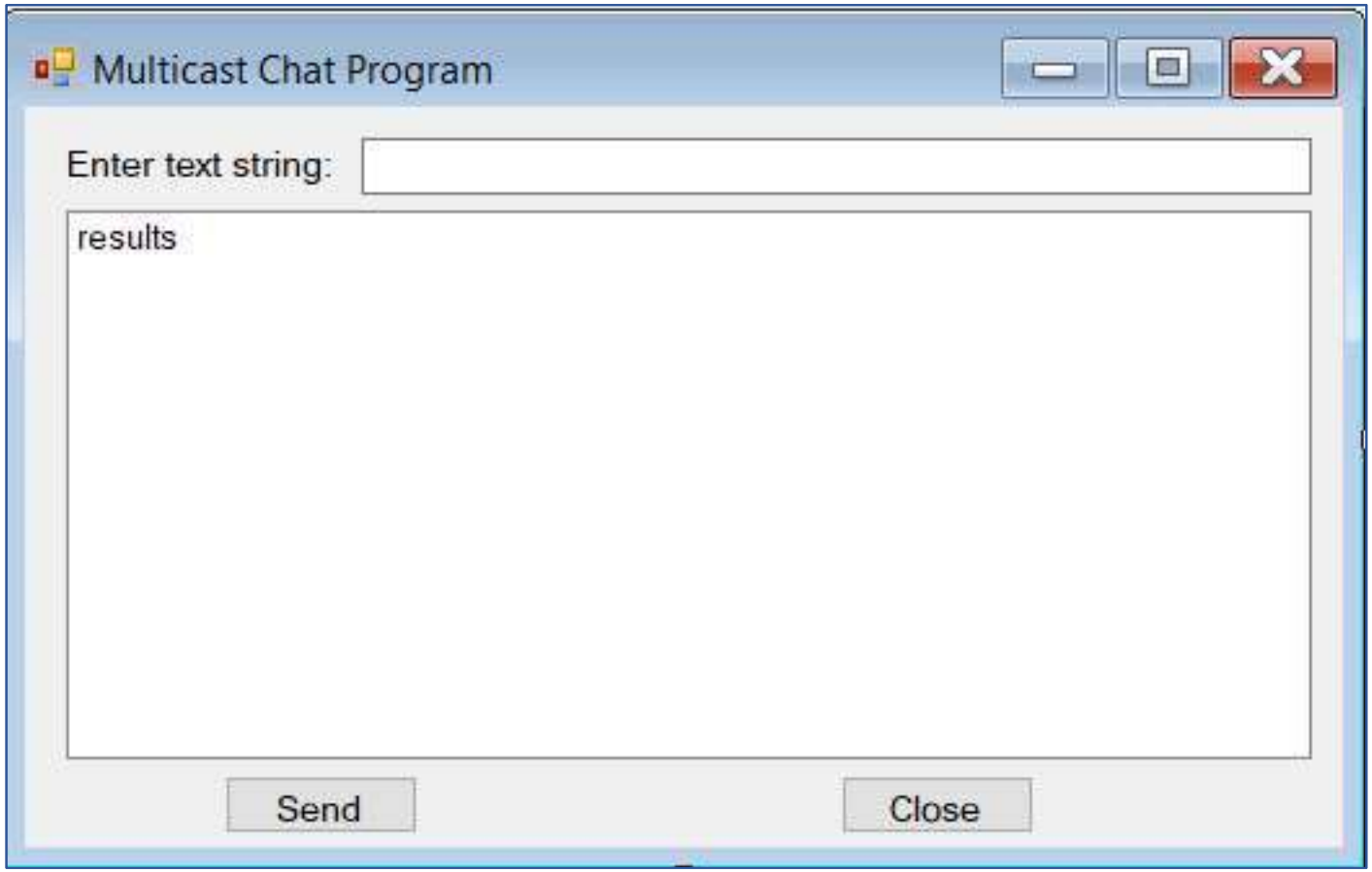
```
class UdpClientMultiSend
{
    public static void Main()
    {
        UdpClient sock = new UdpClient();
        IPEndPoint iep = new IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
        byte[] data = Encoding.ASCII.GetBytes("This is a test message");
        sock.Send(data, data.Length, iep);
        sock.Close();
    }
}
```

DEMO 8-MULTICAST-UDPCLIENT-RECEIVING

```
class UdpClientMultiRecv
{
    public static void Main()
    {
        UdpClient sock = new UdpClient(9050);
        Console.WriteLine("Ready to receive…");
        sock.JoinMulticastGroup(IPAddress.Parse("224.100.0.1"), 50);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);
        byte[] data = sock.Receive(ref iep);
        string stringData = Encoding.ASCII.GetString(data, 0, data.Length);
        Console.WriteLine("received: {0} from: {1}", stringData, iep.ToString());
        sock.Close();
    }
}
```

BÀI TẬP ÁP DỤNG

Tạo ứng dụng gửi/nhận tin nhắn quảng bá trong mạng cục bộ



BÀI TẬP ÁP DỤNG (2)

```
public partial class MulticastChat : Form
{
    Socket sock;
    Thread receiver;
    IPEndPoint multiep = new IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
    public MulticastChat()
    {
        InitializeComponent();
        sock = new Socket(AddressFamily.InterNetwork,
          SocketType.Dgram,ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        sock.Bind(iep);
        sock.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.AddMembership,new MulticastOption(IPAddress.Parse
          ("224.100.0.1")));
        receiver = new Thread(new ThreadStart(packetReceive));
        receiver.IsBackground = true;
        receiver.Start();
    }
```

BÀI TẬP ÁP DỤNG (3)

```
void packetReceive()
{
    EndPoint ep = (EndPoint)multiep;
    byte[] data = new byte[1024];
    string stringData;
    int recv;
    while (true)
    {
        recv = sock.ReceiveFrom(data, ref ep);
        stringData = Encoding.ASCII.GetString(data, 0, recv);
        results.Items.Add("from " + ep.ToString() + ": " + stringData);
    }
}
```

BÀI TẬP ÁP DỤNG (3)

```
        private void sendit_Click(object sender, EventArgs e)
        {
            byte[] message = Encoding.ASCII.GetBytes(newText.Text);
            newText.Clear();
            sock.SendTo(message, SocketFlags.None, multiep);
        }
        private void closeit_Click(object sender, EventArgs e)
        {
            receiver.Abort();
            sock.Close();
            Close();
        }
    }
}
```