Layer 4.
Transport Layer (UDP)

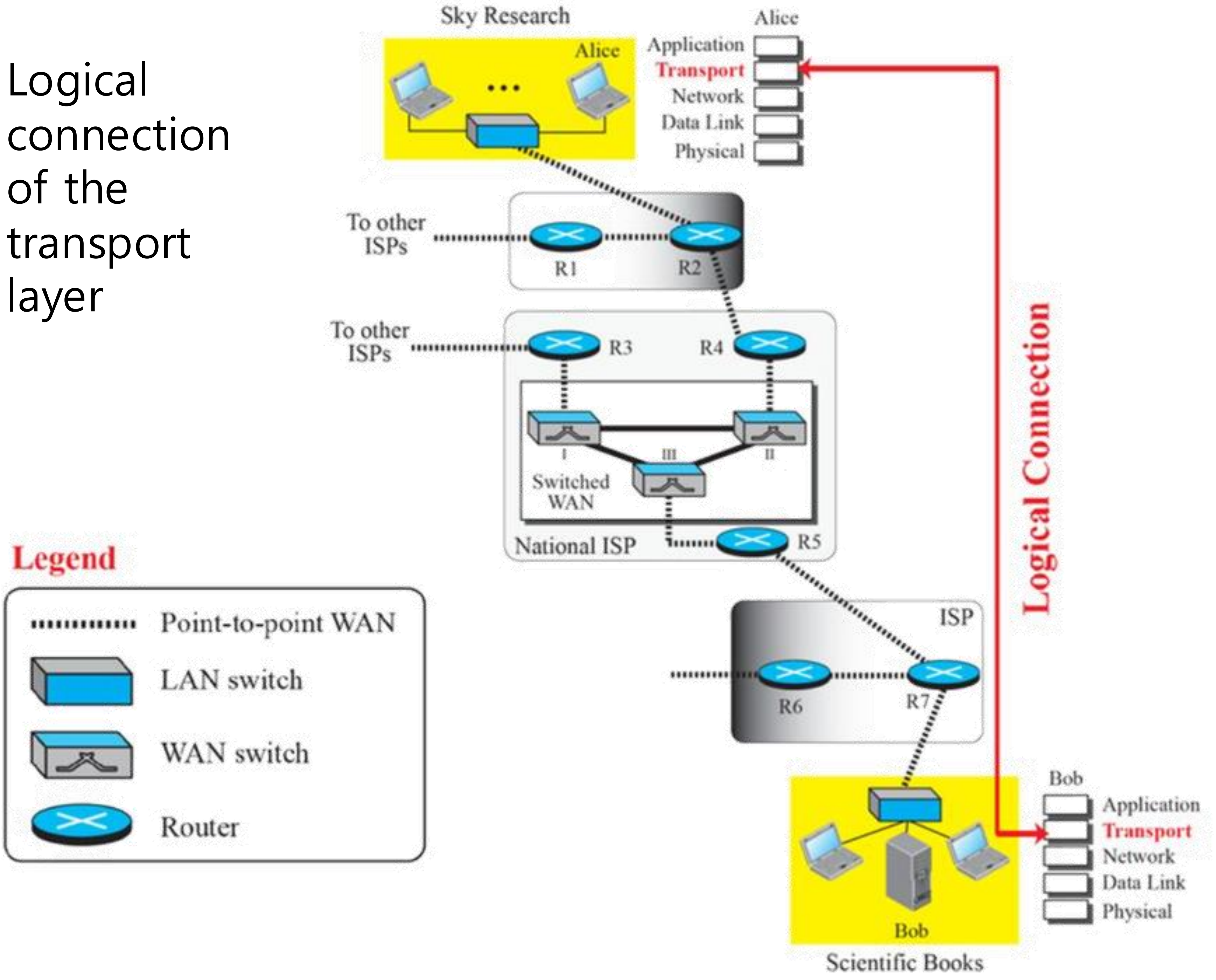
Game Server Programming

Contents

Transport Layer

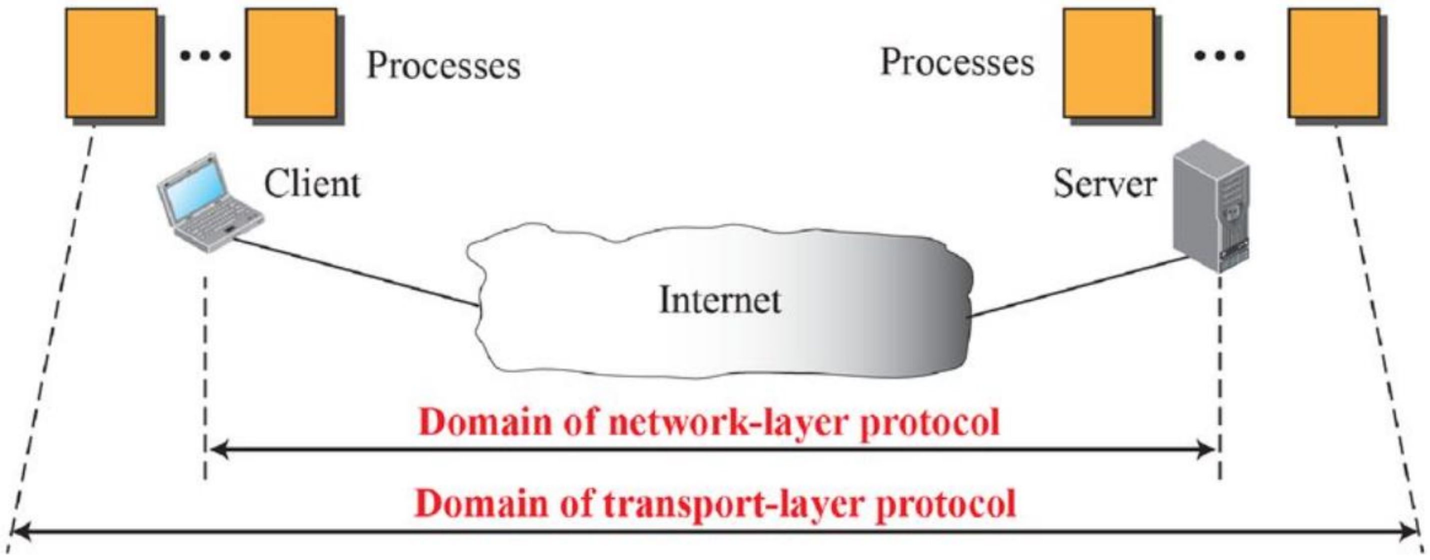
- Located between the network layer and the application layer
- Obligated to provide services to the application layer
- Receives services from the network layer

- Logical connection of the transport layer



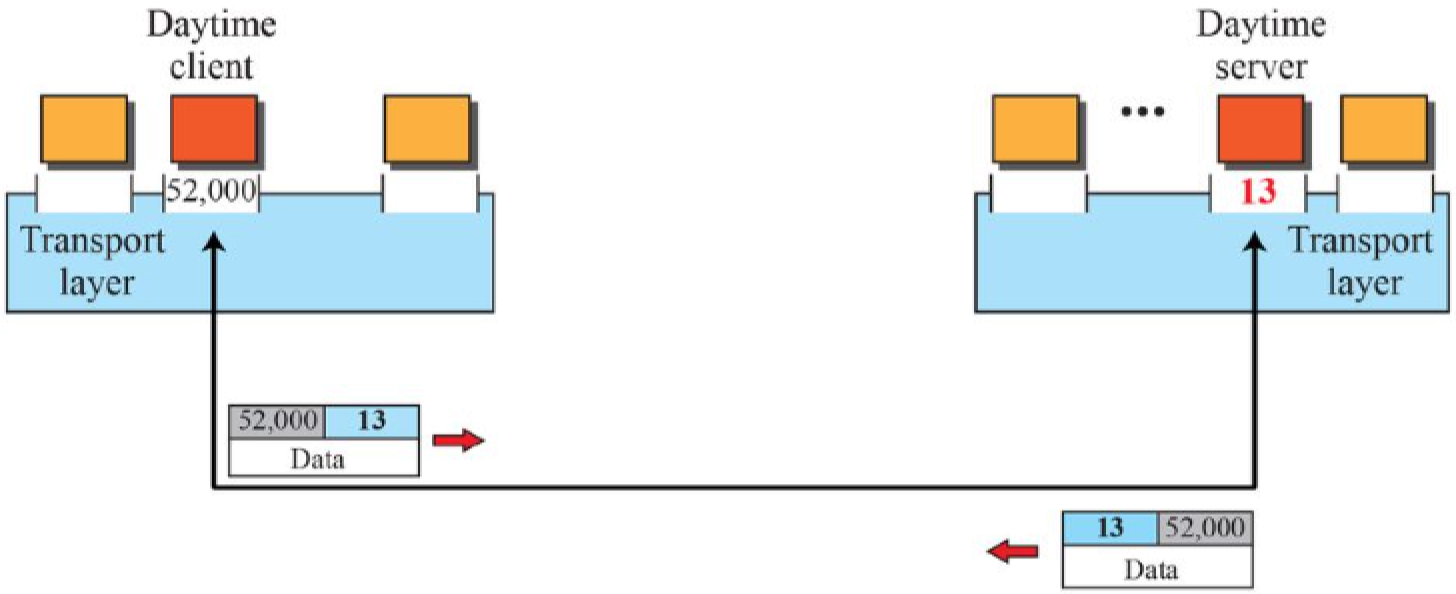
Transport Layer Service

- Provides process-to-process communication
  - ► A process$^{\text{running program}}$ is an application layer entity that uses transport layer services
  - ► Responsible for delivering messages to the appropriate process


Processes
Processes
Client
Server
Internet
Domain of network-layer protocol
Domain of transport-layer protocol

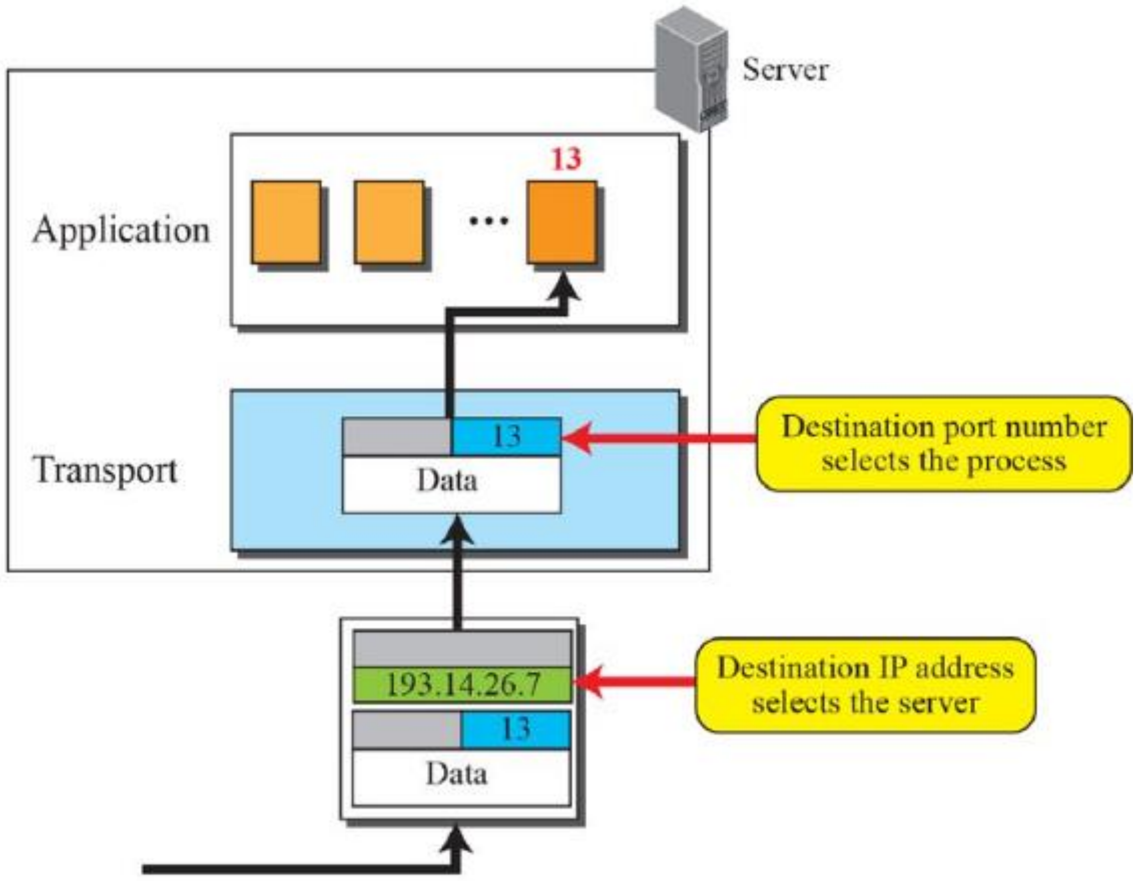
► The Role of Port Numbers in Process Communication

- Local host and remote host: IP address
- Process: Port number
- Port number range: Integer between 0 and 65,535
- Well-known port number
- Ephemeral port number



* daytime service (port# 13): A simple protocol that allows a server to communicate the current date and time to a client. Defined in RFC 867, and used primarily for testing and debugging purposes.

► IP addresses vs port numbers

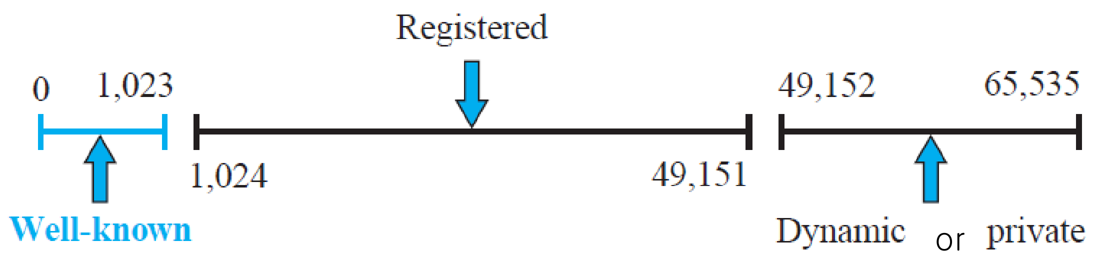

Server
Application
13
...
Transport
13
Data
Destination port number selects the process
193.14.26.7
13
Data
Destination IP address selects the server

Transport Layer Service

▶ ICANN ranges Internet Corporation for Assigned Names and Numbers

- Well-known port: 0 ~ 1,023
  - ✓ Ex> http:80, ssh:22
- Registered port: 1,024 ~ 49,151
  - ✓ Ex> MySQL:3306, RDP:3389
- Dynamic port: 49,152 ~ 65,535

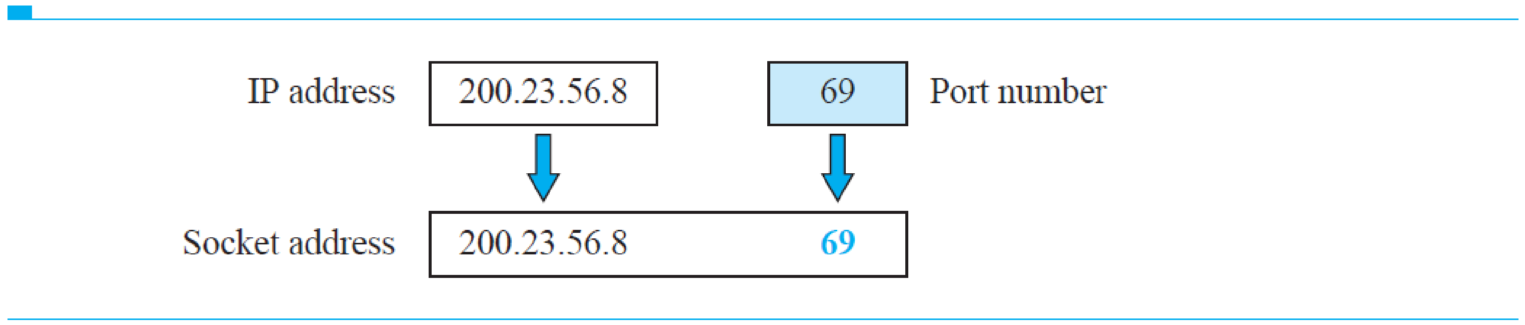
ICANN ranges



*ICANN: A non-profit international organization that coordinates DNS management, IP address allocation, and protocol parameter allocation.

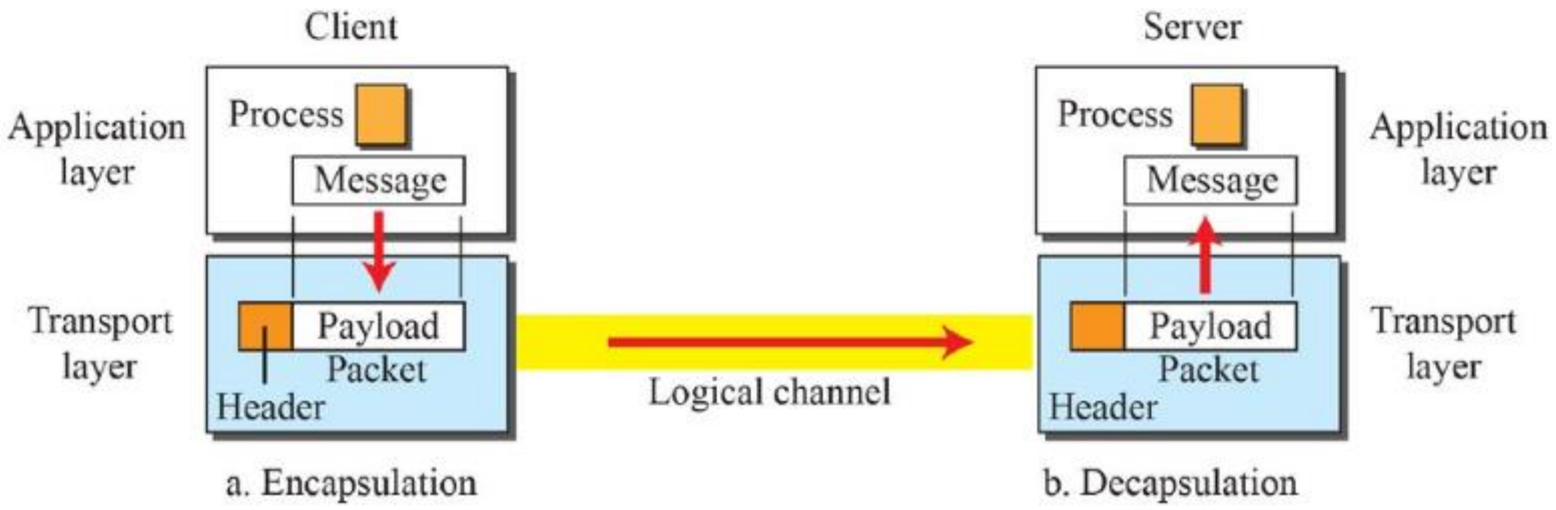
Transport Layer Service

- ▶ Socket address (end point)
  - Addresses required to create a connection at each endpoint



Transport Layer Service

- Encapsulation and Decapsulation

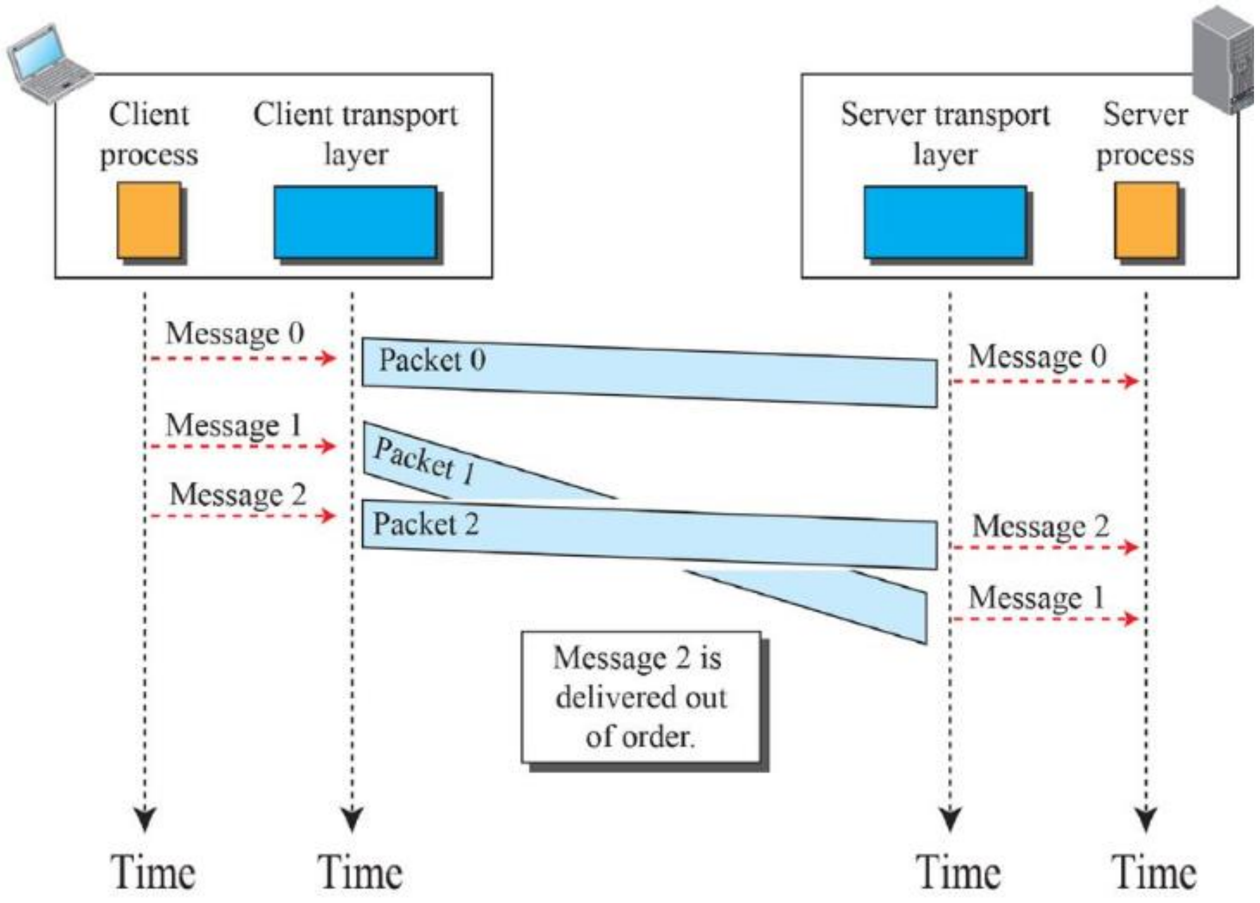


Transport Layer Service

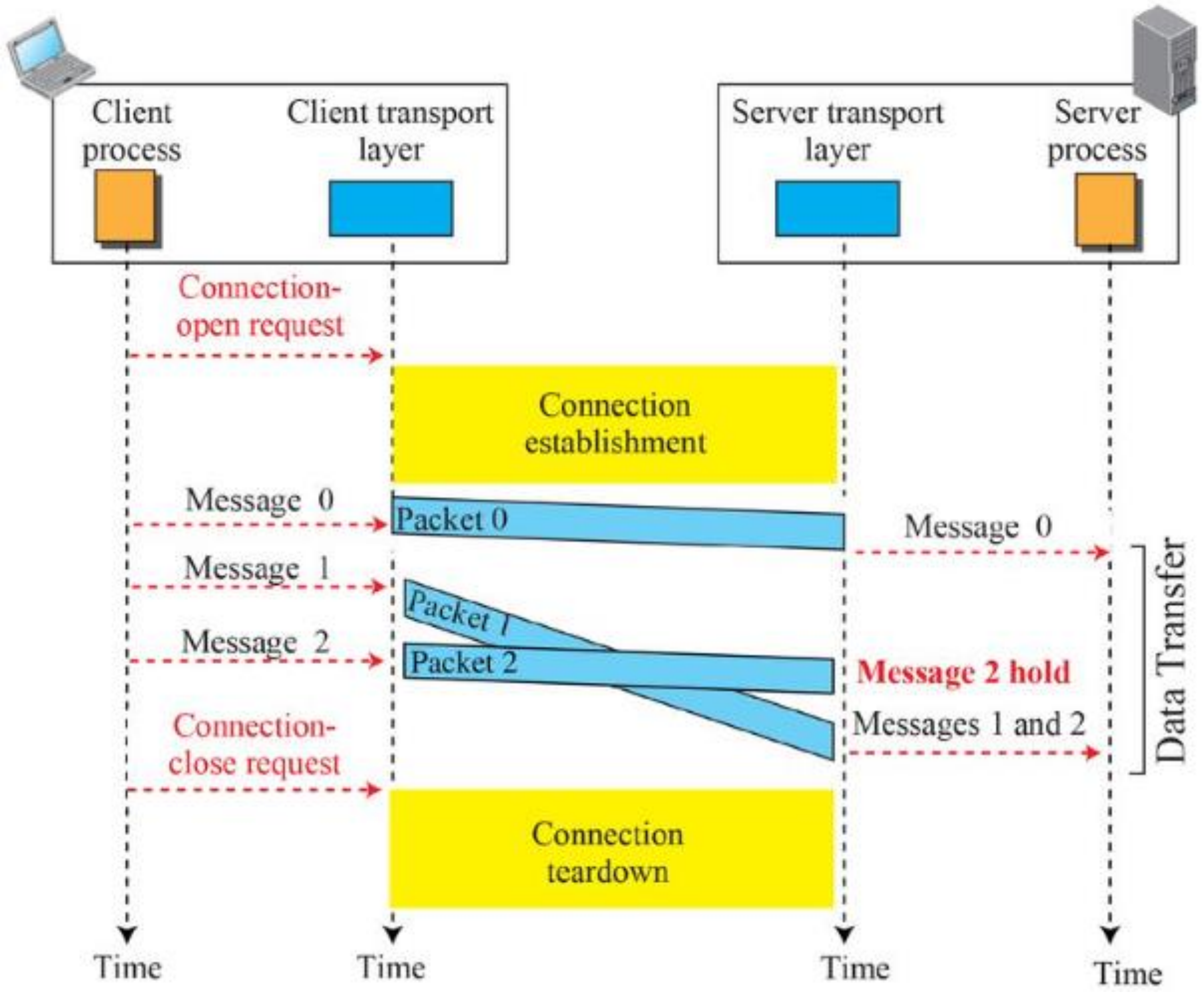
- Connectionless and connection-oriented services
  - ►Connectionless service
    - Messages are divided into small pieces that can be transmitted and sent
    - Each piece is considered an independent unit
    - It’s okay when the data arrives out of order
    - UDP
  - ►Connection-oriented service
    - First, a logical connection is established between the server and the client
    - After data exchange is complete, the connection is released
    - TCP

Transport Layer Service

- Connectionless service

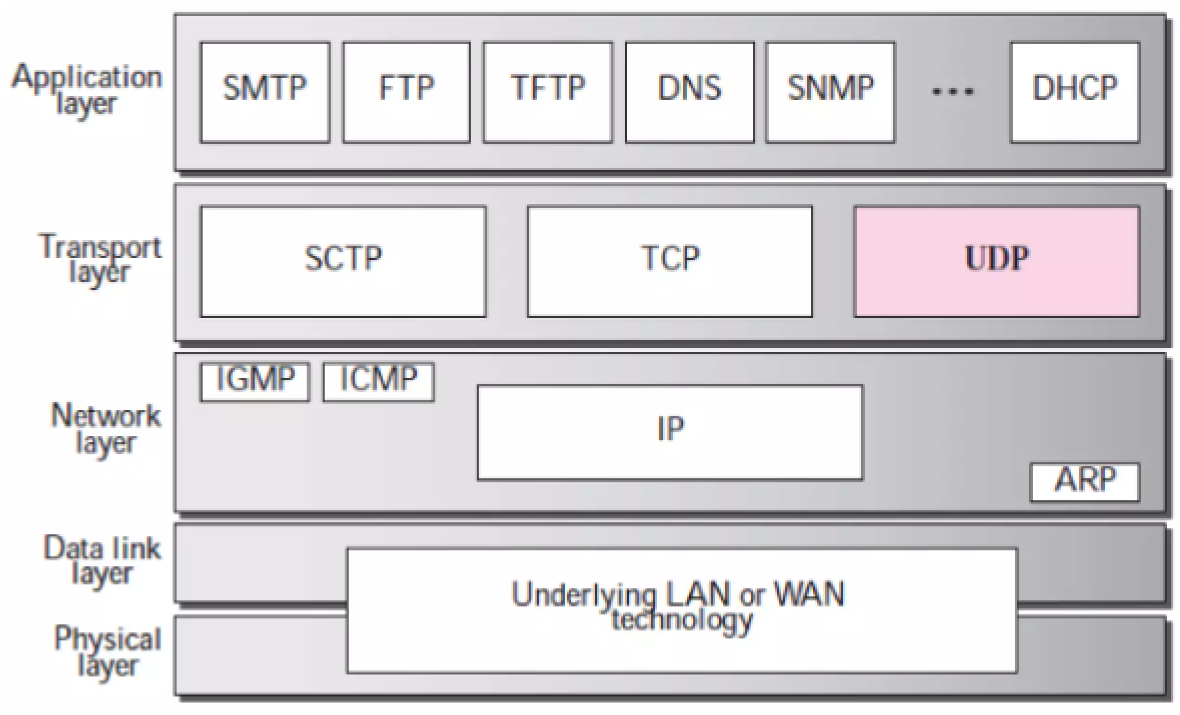


Connection-oriented servcie



UDP overview

- Position of UDP in the TCP/IP protocol suite

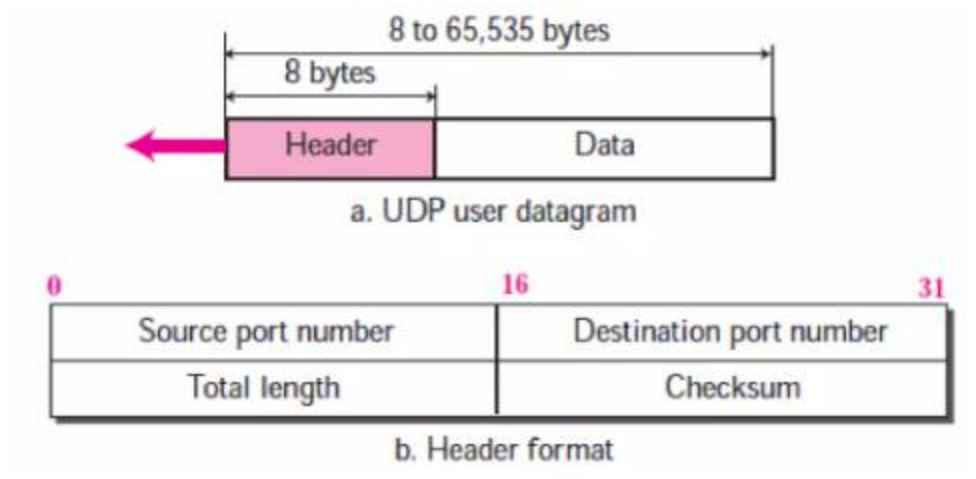


UDP overview

- The UDP transport protocol's mission
  - ► Create process-to-process communication: using port numbers
  - ► Perform minimal error control mechanisms
  - ► Receive data units from processes and provide unreliable delivery
  - ► A connectionless, unreliable transport protocol
  - ► A simple protocol that uses minimal overhead

User datagram format

- 8-byte fixed-size header
- Source port number
  - Port number used by a process running on the source host
- Destination port number
  - Port number used by a process running on the destination host
- Length: Total length of header plus data
- Checksum: Used for error detection



a. UDP user datagram

b. Header format

User Datagram Protocol

► Example 1

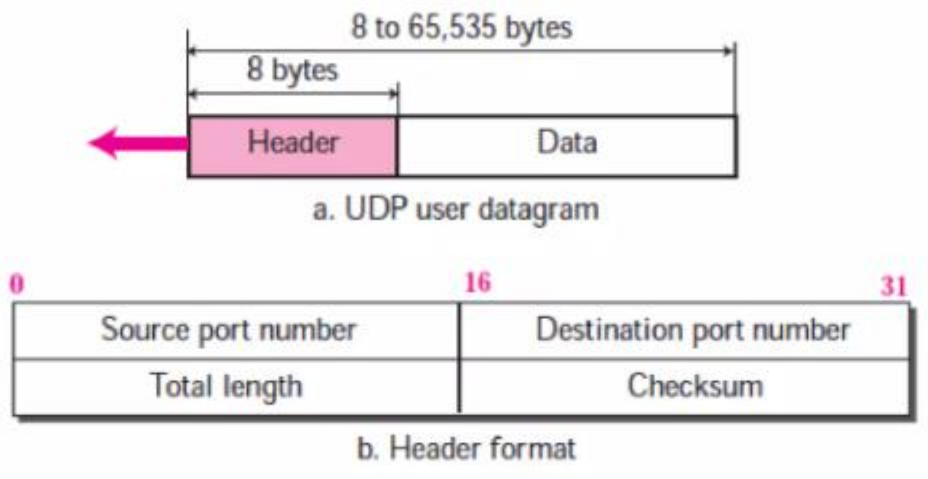
- Here is a dump of the UDP header in hexadecimal format.

```
CB84000D001C001C
```

a. What is the source port number?
b. What is the destination port number?
c. What is the total length of the user datagram?
d. How long is the data?
e. Is the data being sent from client to server or vice versa?
f. What is the client process?

UDP

CB84000D001C001C



a. The source port number is the first four hexadecimal digits ($CB84_{16}$), which is 52,100 in decimal.

b. The destination port number is the second four hexadecimal digits ($000D_{16}$), which is 13 in decimal.

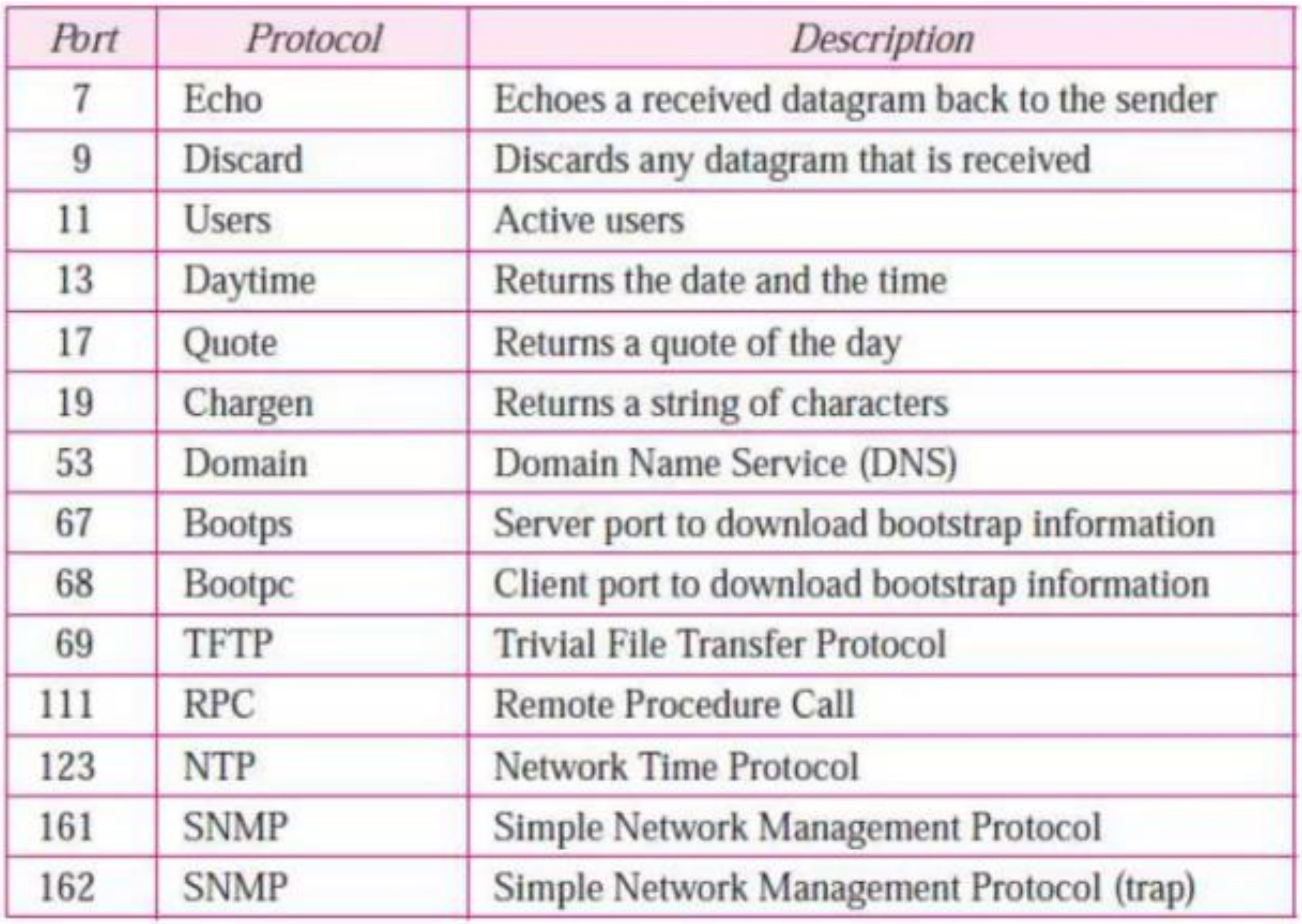
c. The third four hexadecimal digits ($001C_{16}$) represent the total length of the UDP packet, which is 28 bytes.

d. The length of the data is the total length of the packet minus the length of the header, which is 28 - 8 = 20 bytes.

e. Since the destination port number is 13 (a well-known port), the packet was sent from the client to the server.

f. The client process is Daytime.

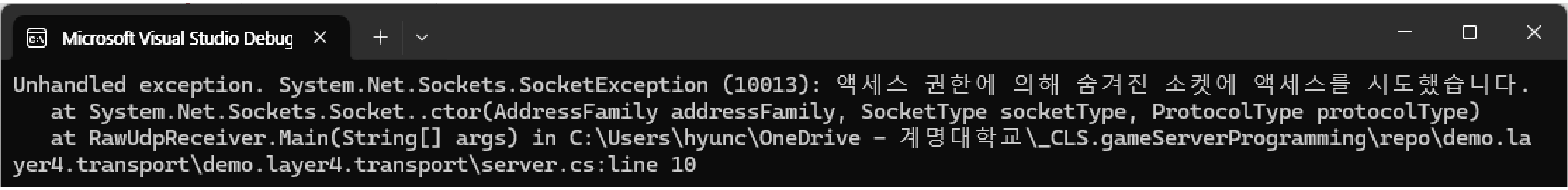
Well-Known Ports used with UDP



Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Domain	Domain Name Service (DNS)
67	Bootps	Server port to download bootstrap information
68	Bootpc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

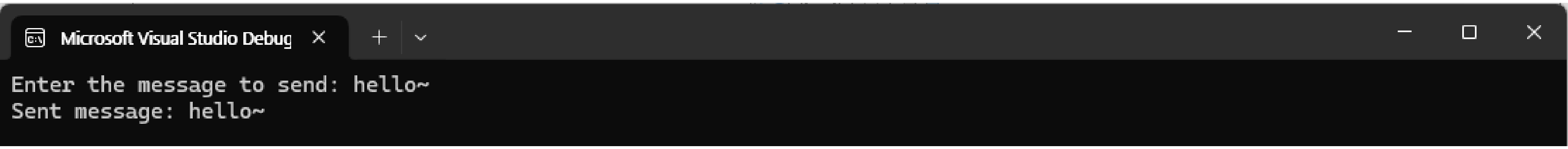
Demo. Check UDP packet contents

- Since the UDP header is usually handled automatically by the IP stack, classes such as UdpClient used in high-level network programming do not directly expose the UDP header to the programmer.
- Therefore, in high-level socket programming, you cannot see the UDP header and IP header.
- However, in low-level network programming, you can check the entire packet including the UDP and IP headers.
  - Raw Socket: You can control all packets at the IP level.
  - To use a raw socket in C#, use the Socket class to create a socket with SocketType.Raw.
  - Operating system privileges may be required (administrator privileges).



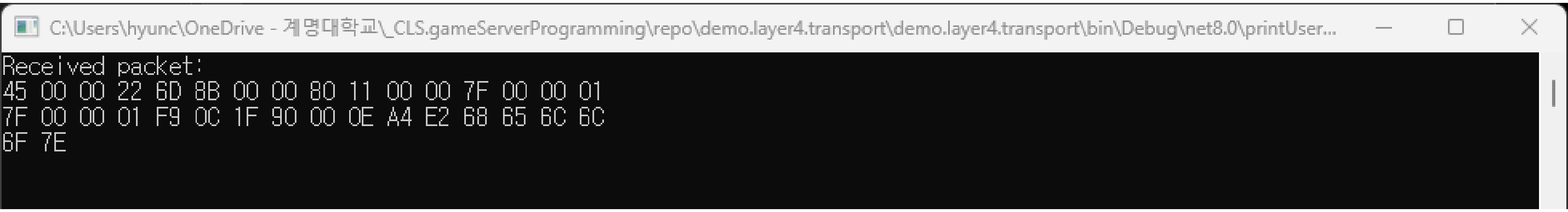
Microsoft Visual Studio Debug

```
Unhandled exception. System.Net.Sockets.SocketException (10013): 액세스 권한에 의해 숨겨진 소켓에 액세스를 시도했습니다.
   at System.Net.Sockets.Socket..ctor(AddressFamily addressFamily, SocketType socketType, ProtocolType protocolType)
   at RawUdpReceiver.Main(String[] args) in C:\Users\hyunc\OneDrive - 계명대학교\_CLS.gameServerProgramming\repo\demo.la
yer4.transport\demo.layer4.transport\server.cs:line 10
```



Microsoft Visual Studio Debug

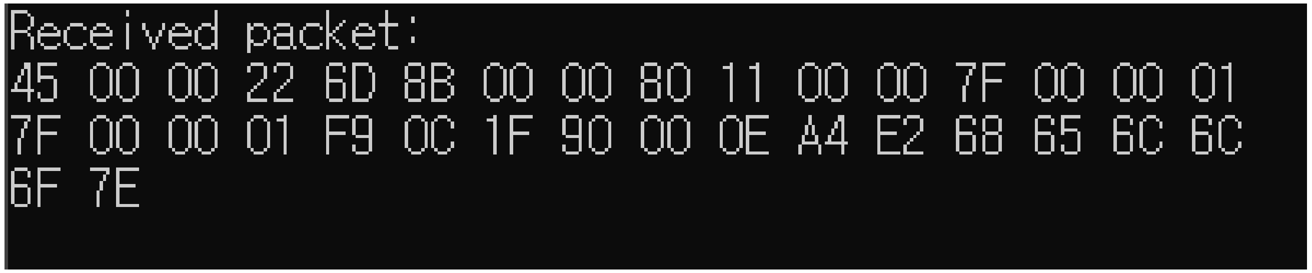
```
Enter the message to send: hello~
Sent message: hello~
```



C:\Users\hyunc\OneDrive - 계명대학교_CLS.gameServerProgramming\repo\demo.layer4.transport\demo.layer4.transport\bin\Debug\net8.0\printUser...

```
Received packet:
45 00 00 22 6D 8B 00 00 80 11 00 00 7F 00 00 01
7F 00 00 01 F9 0C 1F 90 00 0E A4 E2 68 65 6C 6C
6F 7E
```

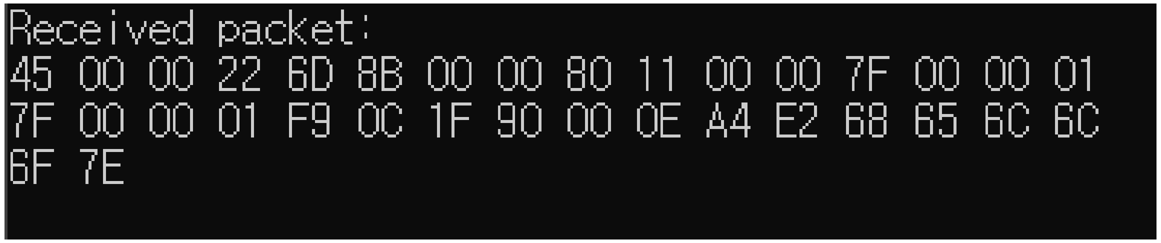
IP header



```
Received packet:
45 00 00 22 6D 8B 00 00 80 11 00 00 7F 00 00 01
7F 00 00 01 F9 0C 1F 90 00 0E A4 E2 68 65 6C 6C
6F 7E
```

- (45):4: IPv4 , Header Length (5 * 4 = 20 bytes)
- Service type (00): General
- Total Length (00 22): 34Bytes (IP Header + UDP Header + Data)
- ID (6D 8B): Identification (unique identification value of the transmitted packet)
- Flags and Offset (00 00): Unsegmented packet (flag 0)
- TTL (Time to Live) (80): 128
- Protocol (11): UDP (0x11 = 17, indicating UDP)
- Header Checksum (00 00)
- Source IP (7F 00 00 01): 127.0.0.1(localhost)
- Destination IP (7F 00 00 01): 127.0.0.1 (localhost)

UDP header



```
Received packet:
45 00 00 22 6D 8B 00 00 80 11 00 00 7F 00 00 01
7F 00 00 01 F9 0C 1F 90 00 0E A4 E2 68 65 6C 6C
6F 7E
```

- Source port (F9 0C): Hexadecimal = 63756
- Port Destination port (1F 90): Hexadecimal = 8080
- Port UDP length (00 0E): Hexadecimal 000E = 14 bytes (UDP header + data length)
- Checksum (A4 E2): Checksum value

- Data field
  - 68 65 6C 6C 6F 7E → "hello~"

UDP Application

- Typical applications
  - ► Suitable for processes that require simple request-response communication and do not require flow and error control
  - ► Suitable for processes that have internal flow and error control (e.g. TFTP)
  - ► Transport protocol suitable for multicasting
  - ► Used for management processes such as SNMP Simple Network Management Protocol
  - ► Used for route update protocols such as RIP Requesting Information Protocol
  - ► Used for real-time applications

UDP Application

- Suppose you are downloading a very large document file over the Internet.
  - ► You need to use a transport layer that provides reliable service.
  - ► That is, you do not want parts of the file to be lost or corrupted when you open it.
  - ► The delay between the delivery of parts of the file is not a significant issue.
  - ► You simply wait for the entire file to be constructed before you can view it.
  - ► In this case, UDP is not a suitable transport layer.

EoF