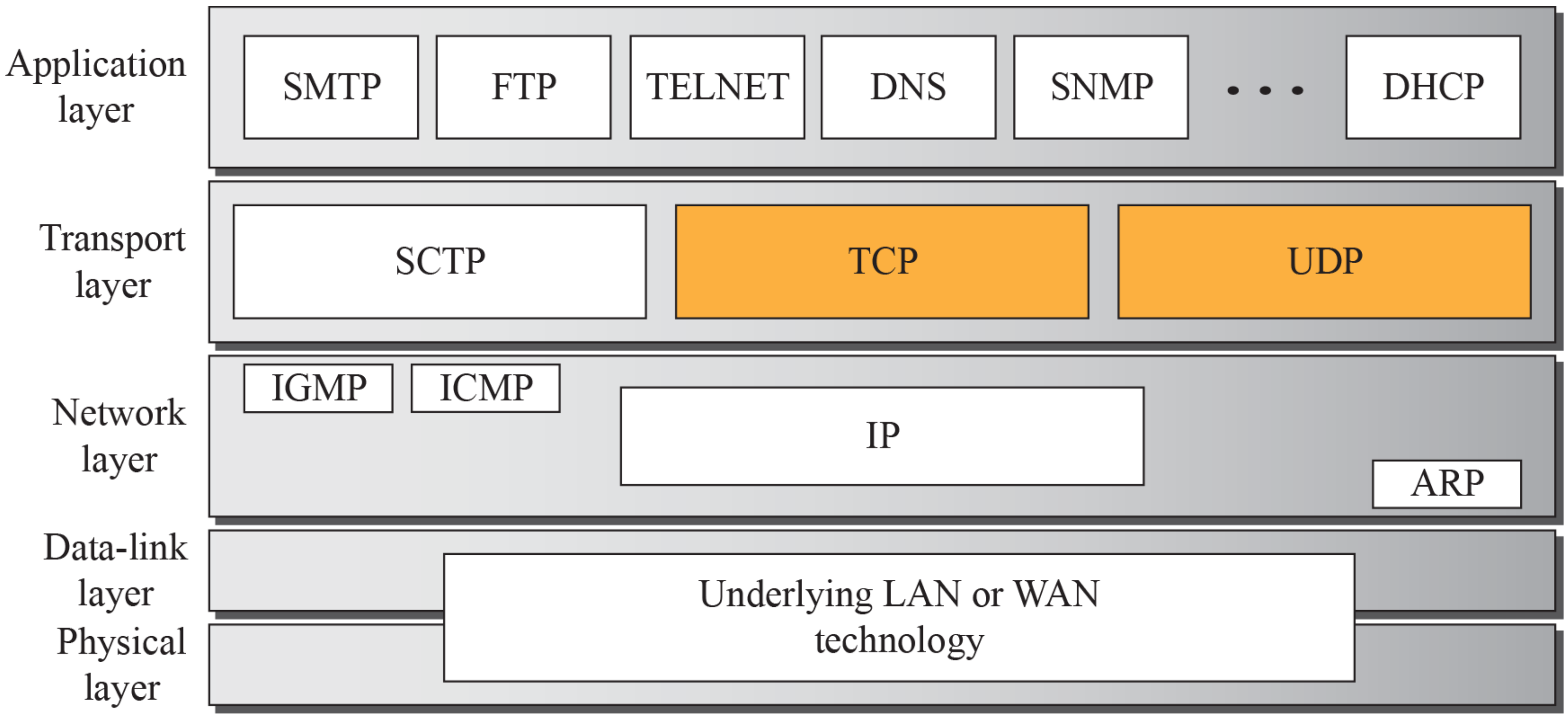
# *3.2.6 Internet Transport-Layer Protocols*

***A network is the interconnection of a set of devices capable of communication. In this definition, a device can be a host such as a large computer, desktop, laptop, workstation, cellular phone, or security system. A device in this definition can also be a connecting device such as a router a switch, a modem that changes the form of data, and so on.***

## Figure 3.38: Position of transport-layer protocols in the TCP/IP protocol suite



## Table 3.1: Some well-known ports used with UDP and TCP

| Port | Protocol | UDP | TCP | Description |
|---|---|---|---|---|
| 7 | Echo | √ | | Echoes back a received datagram |
| 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 |
| 20, 21 | FTP | | √ | File Transfer Protocol |
| 23 | TELNET | | √ | Terminal Network |
| 25 | SMTP | | √ | Simple Mail Transfer Protocol |
| 53 | DNS | √ | √ | Domain Name Service |
| 67 | DHCP | √ | √ | Dynamic Host Configuration Protocol |
| 69 | TFTP | √ | | Trivial File Transfer Protocol |
| 80 | HTTP | | √ | Hypertext Transfer Protocol |
| 111 | RPC | √ | √ | Remote Procedure Call |
| 123 | NTP | √ | √ | Network Time Protocol |
| 161, 162 | SNMP | | √ | Simple Network Management Protocol |

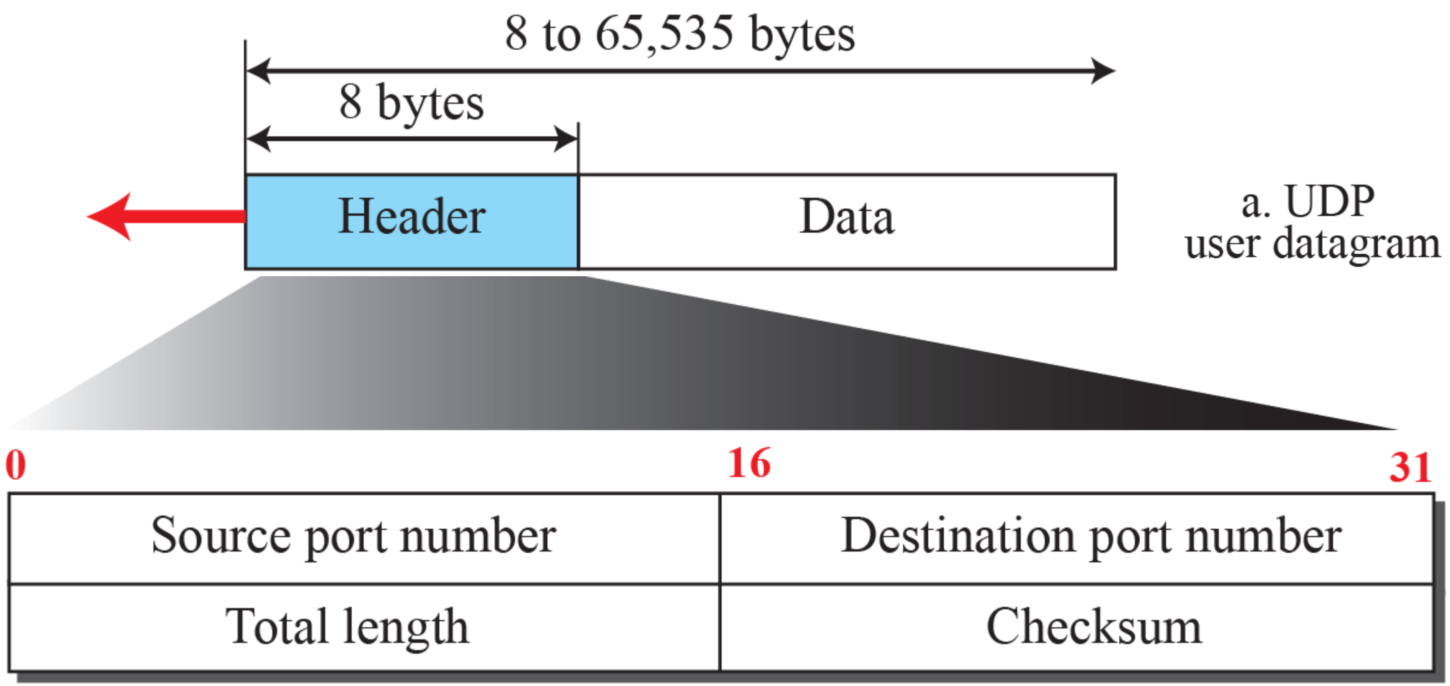
## *3-3 USER DATAGRAM PROTOCOL (UDP)*

***The User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol. It does not add anything to the services of IP except for providing process-to-process instead of host-to-host communication. UDP is a very simple protocol using a minimum of overhead.***

## 3.3.1 User Datagram

***UDP packets, called user datagrams, have a fixed size header of 8 bytes made of four fields, each of 2 bytes (16 bits). Figure shows the format of a user datagram. The first two fields define the source and destination port numbers. The third field defines the total length of the user datagram, header plus data. The 16 bits can define a total length of 0 to 65,535 bytes.***

# Figure 3.39: User datagram packet format



b. Header format

## Example 3.11

***The following is the contents of a 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. What is the length of the data?*

*e. Is the packet directed from a client to a server or vice versa?*

*f. What is the client process?*

# Example 3.11 (continue)

**Solution**

a. The source port number is the first four hexadecimal digits $(CB84)_{16}$ or 52100

b. The destination port number is the second four hexadecimal digits $(000D)_{16}$ or 13.

c. The third four hexadecimal digits $(001C)_{16}$ define the length of the whole UDP packet as 28 bytes.

d. The length of the data is the length of the whole packet minus the length of the header, or $28 - 8 = 20$ bytes.

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

f. The client process is the Daytime .

## *3.3.2 UDP Services*

***Earlier we discussed the general services provided by a transport-layer protocol. In this section, we discuss what portions of those general services are provided by UDP.***

- ***Process-to-Process Communication***
- ***Connectionless Services***
  - ***Each datagram is independent***
  - ***Not numbered***
  - ***No Connection establishment and Connection Release***
- ***Flow Control- NA, Process using UDP should provide***
- ***Error Control-Checksum***

# Checksum

UDP checksum includes three sections:

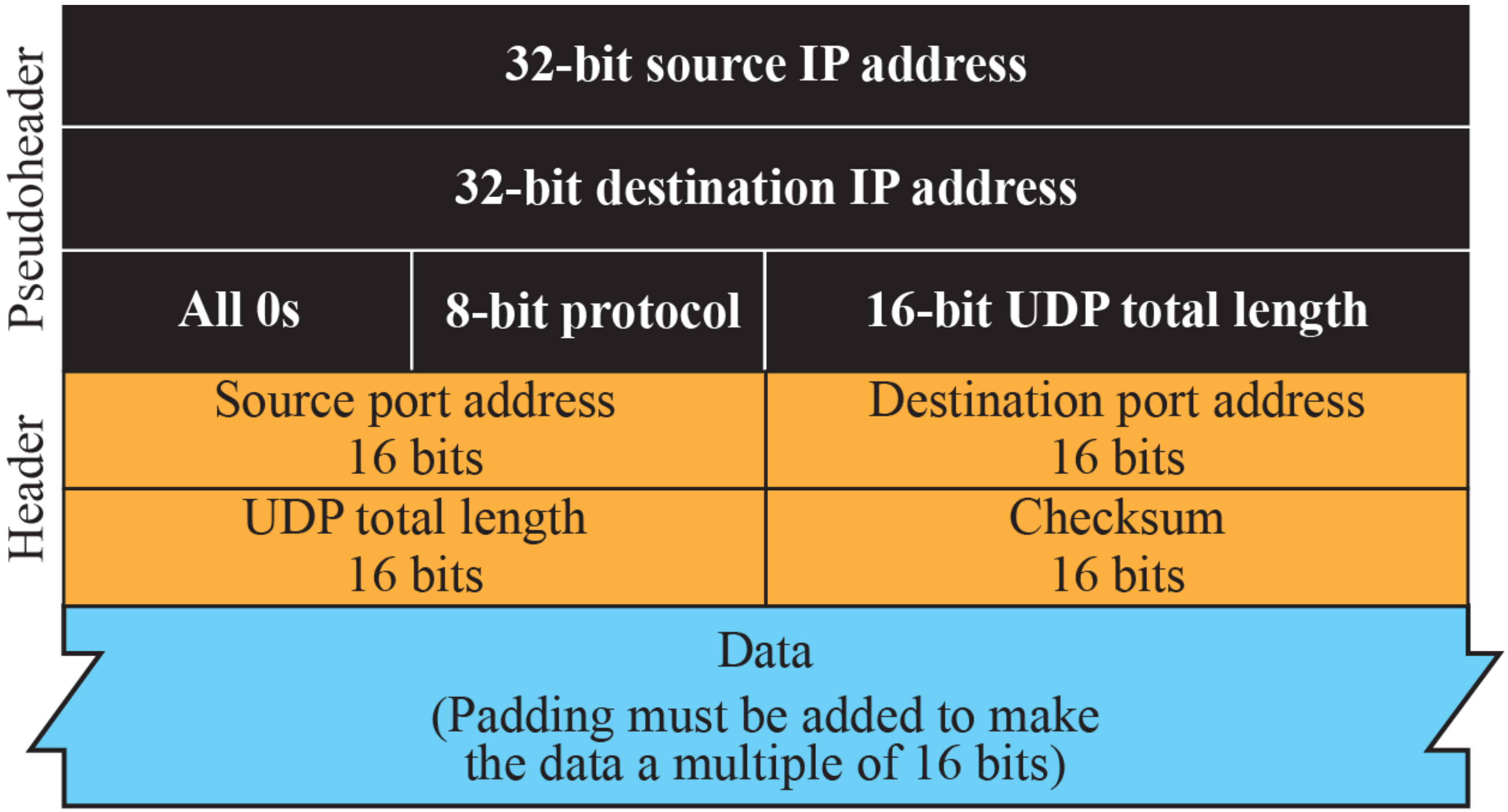
- Pseudoheader
- UDP Header
- Data coming from Application layer.

Pseudoheader is the part of the header of IP packet in which UDP datagram is to be encapsulated with 0 s.

# 3.3.2 (continued)

- ***Checksum***
  - ***Optional Inclusion of Checksum***
- ***Congestion Control***
- ***Encapsulation and Decapsulation***
- ***Queuing***
- ***Multiplexing and Demultiplexing***
- ***Comparison : UDP and Simple Protocol***

## *Figure 3.40: Pseudoheader for checksum calculation*



## Example 3.12

***What value is sent for the checksum in one of the following hypothetical situations?***

***a. The sender decides not to include the checksum.***

***b. The sender decides to include the checksum, but the value of the sum is all 1s.***

***c. The sender decides to include the checksum, but the value of the sum is all 0s.***

***Solution***

***a. The value sent for the checksum field is all 0s to show that the checksum is not calculated.***

***b. When the sender complements the sum, the result is all 0s; the sender complements the result again before sending. The value sent for the checksum is all 1s. The second complement operation is needed to avoid confusion with the case in part a.***

***c. This situation never happens because it implies that the value of every term included in the calculation of the sum is all 0s, which is impossible; some fields in the pseudoheader have nonzero values.***

## 3.3.3 UDP Applications

***Although UDP meets almost none of the criteria we mentioned earlier for a reliable transport-layer protocol, UDP is preferable for some applications. The reason is that some services may have some side effects that are either unacceptable or not preferable. An application designer sometimes needs to compromise to get the optimum.***

***In this section, we first discuss some features of UDP that may need to be considered when one designs an application program and then show some typical applications.***

# 3.3.3 (continued)

- **UDP Features**
  - *Connectionless Service*
  - *Lack of Error Control*
  - *Lack of Congestion Control*
- **Typical Applications**

## Example 3.13

***A client-server application such as DNS uses the services of UDP because a client needs to send a short request to a server and to receive a quick response from it. The request and response can each fit in one user datagram. Since only one message is exchanged in each direction, the connectionless feature is not an issue; the client or server does not worry that messages are delivered out of order.***

## *Example 3.14*

***A client-server application such as SMTP , which is used in electronic mail, cannot use the services of UDP because a user can send a long e-mail message, which may include multimedia (images, audio, or video). If the application uses UDP and the message does not fit in one single user datagram, the message must be split by the application into different user datagrams. Here the connectionless service may create problems. The user datagrams may arrive and be delivered to the receiver application out of order. The receiver application may not be able to reorder the pieces. This means the connectionless service has a disadvantage for an application program that sends long messages.***

## *Example 3.15*

***Assume we are downloading a very large text file from the Internet. We definitely need to use a transport layer that provides reliable service. We don't want part of the file to be missing or corrupted when we open the file. The delay created between the deliveries of the parts is not an overriding concern for us; we wait until the whole file is composed before looking at it. In this case, UDP is not a suitable transport layer.***

## Example 3.16

*Assume we are using a real-time interactive application, such as Skype. Audio and video are divided into frames and sent one after another. If the transport layer is supposed to resend a corrupted or lost frame, the synchronizing of the whole transmission may be lost. The viewer suddenly sees a blank screen and needs to wait until the second transmission arrives. This is not tolerable. However, if each small part of the screen is sent using one single user datagram, the receiving UDP can easily ignore the corrupted or lost packet and deliver the rest to the application program. That part of the screen is blank for a very short period of time, which most viewers do not even notice.*

# Typical Applications

- Process that requires simple request-response communication-FTP
- Process with internal flow and error control-TFTP.
- Multicasting
- Management process-SNMP.
- Route Updating Protocols-RIP.
- Interactive Real Time Applications.