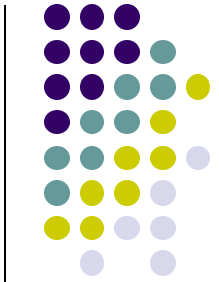
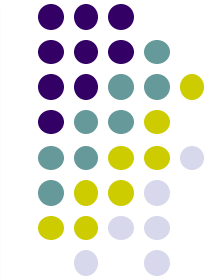


Part 2.6

UDP Principles (Chapter 24)

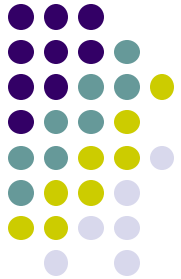
(User Datagram Protocol)

UDP: User Datagram Protocol



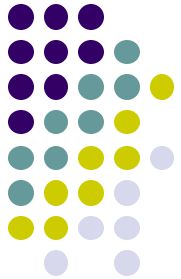
- In TCP/IP protocol suite, using IP to transport datagram (similar to IP datagram).
- Allows a application to send datagram to other application on the remote machine.
- Delivery and duplicate detection are not guaranteed.
- Low overhead: faster than TCP

UDP Characteristics



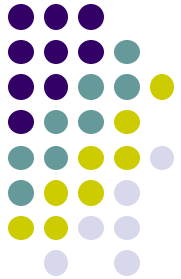
- **End-to-End**: an application sends/receives data to/from another application.
- **Connectionless**: Application does not need to preestablish communication before sending data; application does not need to terminate communication when finished.
- **Message-oriented**: application sends/receives individual messages (UDP datagram), not packets.
- **Best-effort**: same best-effort delivery semantics as IP. I.e. message can be lost, duplicated, and corrupted.
- **Arbitrary interaction**: application communicates with many or one other applications.
- **Operating system independent**: identifying application does not depend on O/S.

Identifying An Application



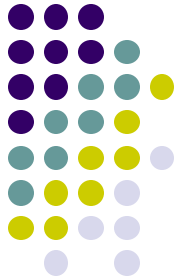
- UDP cannot extend IP address
  - No unused bits
- Cannot use OS-dependent quantity
  - Process ID, Task number, Job name
- Must work on all computer systems
- Technique
  - Each application assigned unique integer
  - Called ***protocol port number***

Protocol Port Number



- UDP uses ***Port Number*** to identify an application as an endpoint.
- UDP messages are delivered to the port specified in the message by the sending application
- In general, a port can be used for any datagram, as long as the sender and the receiver agrees
- In practice, a collection of well-known ports are used for special purposes such as telnet, ftp, and email. E.g. port 7 for Echo application.
- Local operating system provides an interface for processes to specify and access a port.

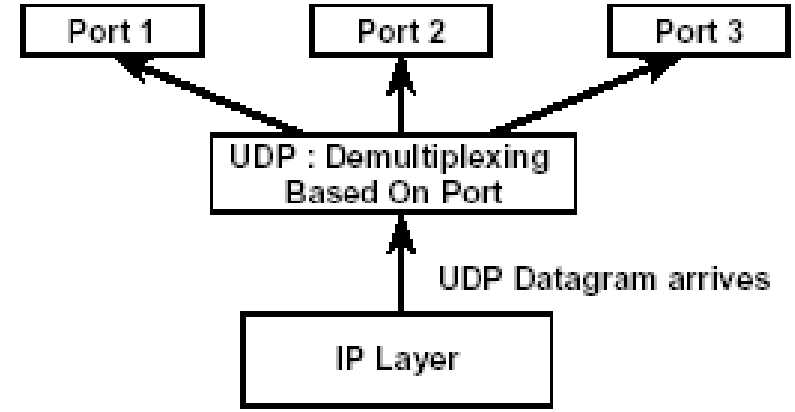
Well-known Port Numbers



- list of UDP ports copied from /etc/services on Solaris 2.5:

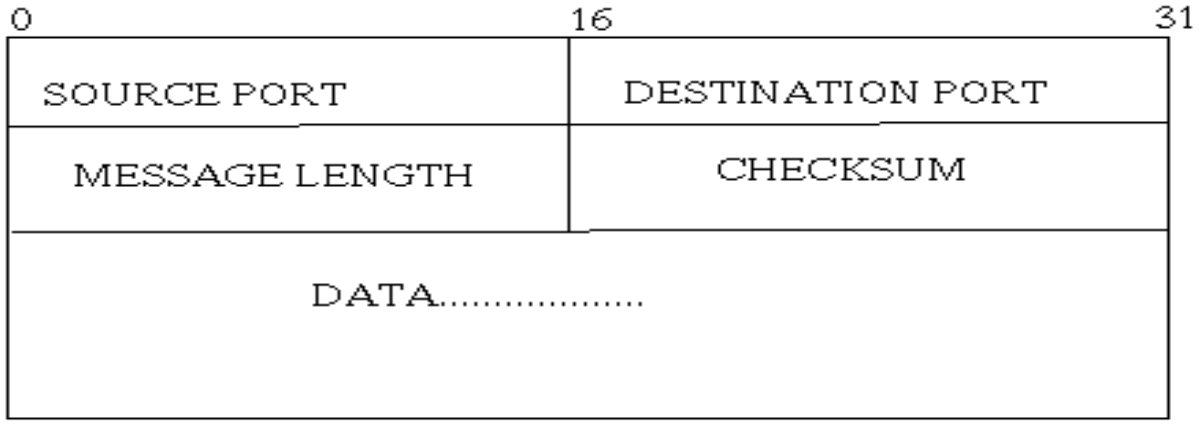
```
echo            7/udp           #echo
discard         9/udp           #discard
daytime        13/udp           #daytime
chargen        19/udp           #character generator
time           37/udp           #timserver
name           42/udp           #host nameserver
domain         53/udp           #domain name server
bootps         67/udp           # BOOTP/DHCP server
bootpc         68/udp           # BOOTP/DHCP client
tftp           69/udp           #trivial file transfer
ntp           123/udp           # Network Time Protocol
```

UDP Multiplexing & Demultiplexing



- Sender: multiplexing of UDP datagrams.
  - UDP datagrams are received from multiple application programs.
  - A single sequence of UDP datagrams is passed to IP layer.
- Receiver: demultiplexing of UDP datagrams.
  - Single sequence of UDP datagrams received from IP layer.
  - UDP datagram received is passed to appropriate application.

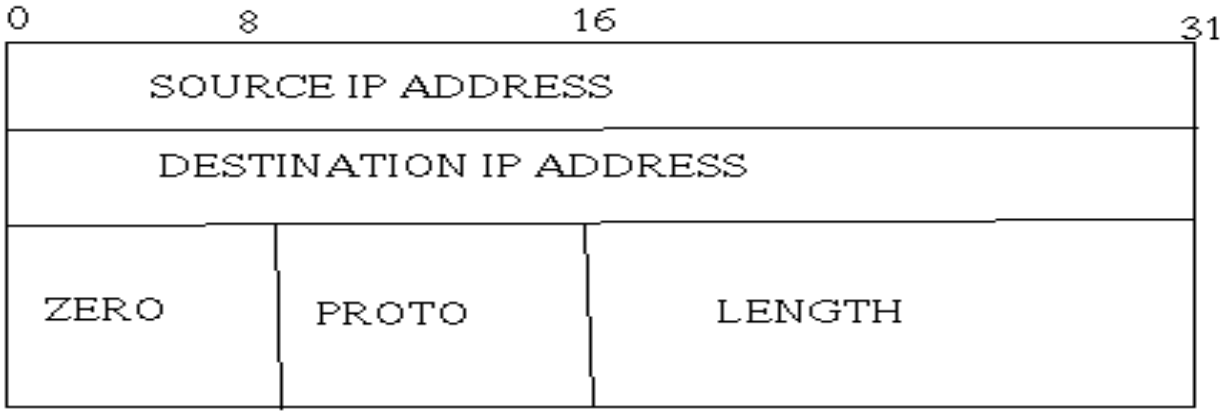
UDP Datagram Format



- Source Port - 16 bit port number
- Destination Port - 16 bit port number
- Length (of UDP header + data) - 16 bit count of octets
- UDP checksum - 16 bit field. if 0, then there is no checksum, else it is a checksum over a pseudo header + UDP data area

Checksum and Pseudo Header

- UDP uses a pseudo-header to verify that the UDP message has arrived at both the correct machine and the correct port.

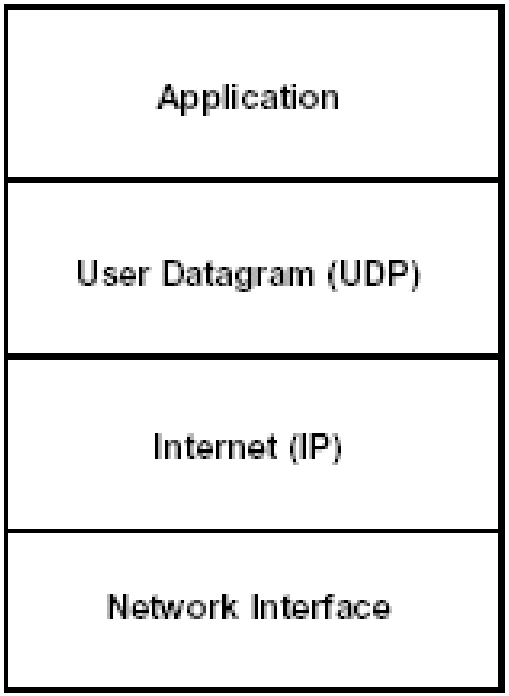


- Proto : IP protocol type code.
- Length : Length of the UDP datagram.

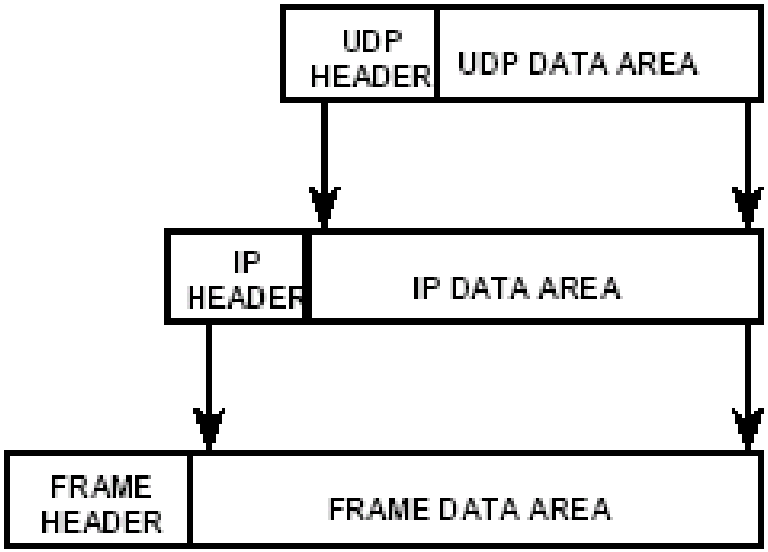
Encapsulation and Layering

- Protocol layering

Conceptual Layering



- UDP encapsulation



- UDP message is encapsulated into an IP datagram.
- IP datagram in turn is encapsulated into a physical frame for actually delivery.