CST234
Chapter 2 - Transport Layer

UDP & TCP

User Datagram Protocol

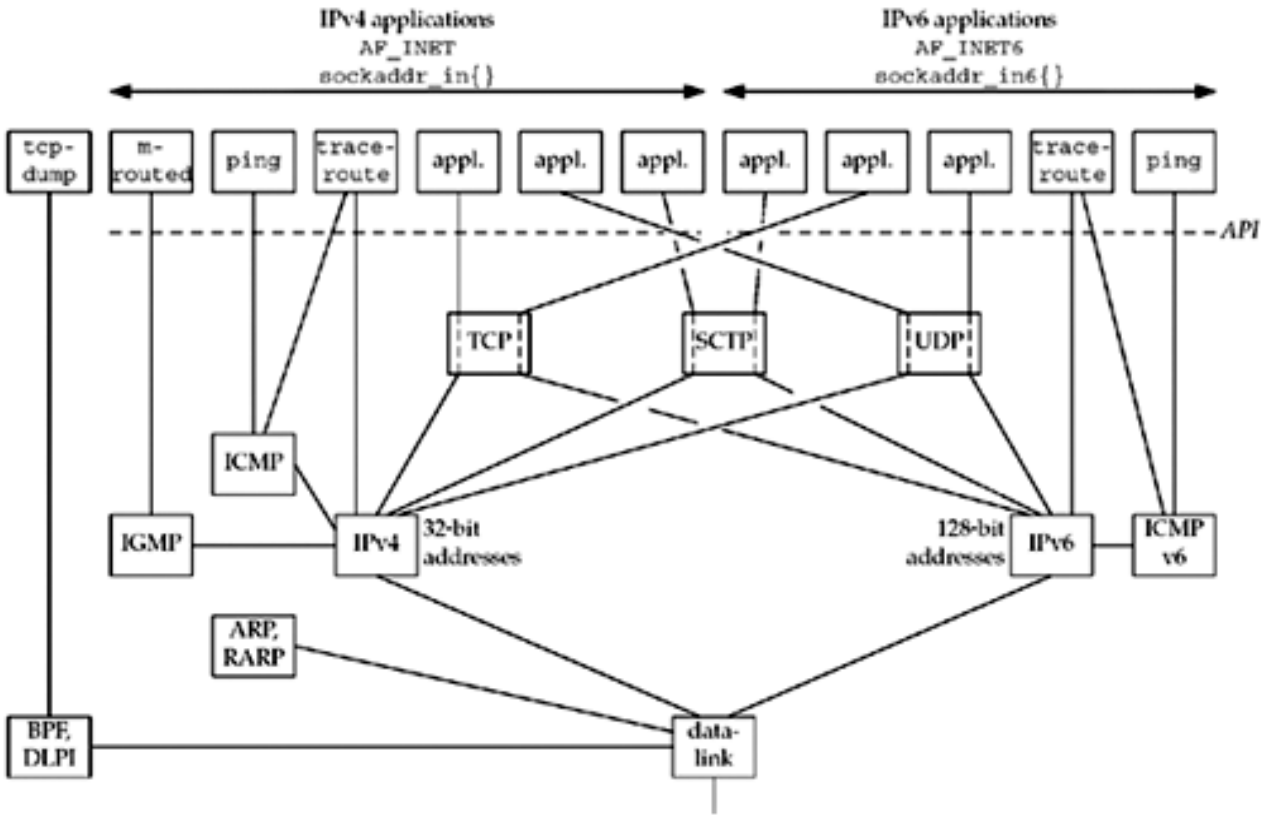
- "connection-less"
  - simple
  - unreliable
  - no error-control
  - no flow-control

Transport Control Protocol

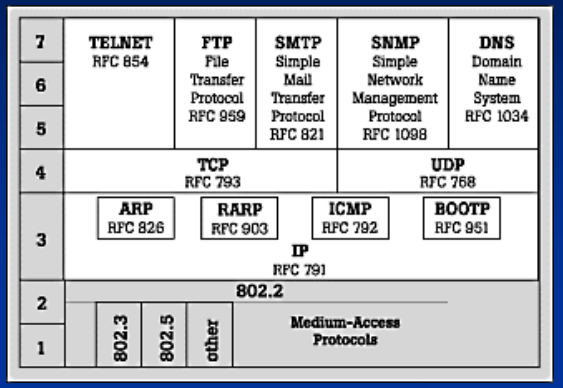
- "connection-oriented"
  - sophisticated
  - reliable
  - robust
  - error-control
  - flow-control
  - full-duplex
- popular for networked applications

TCP/IP Protocols

Figure 2.1. Overview of TCP/IP protocols.



Protocol Types



- IPv4 – Internet Protocol version 4
- IPv6 – Internet Protocol version 6
- TCP – Transmission Control Protocol
- UDP – User Datagram Protocol
- SCTP – Stream Control Transmission Protocol
- ICMP – Internet Control Message Protocol (router-host error handling)
- IGMP – Internet Group Management Protocol (multicasting)
- ARP – Address Resolution Protocol (IP address → MAC address)
- RARP – Reverse Address Resolution Protocol (MAC address → IP address)
- ICMPv6 – combines features of ICMPv4, IGMP and ARP
- BPF – BSD Packet Filter (access to datalink layer)
- DLPI – Datalink Provider Interface (SVR4) (access to datalink layer)

Simple Daytime Client & Server

```
1 #include    "unp.h"
2 int
3 main(int argc, char **argv)
4 {
5     int     sockfd, n;
6     char    recvline[MAXLINE + 1];
7     struct sockaddr_in servaddr;

8     if (argc != 2)
9         err_quit("usage: a.out <IPaddress>");

10    if ( (sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
11        err_sys("socket error");

12    bzero(&servaddr, sizeof(servaddr));
13    servaddr.sin_family = AF_INET;
14    servaddr.sin_port = htons(13);  /* daytime server */
15    if (inet_pton(AF_INET, argv[1], &servaddr.sin_addr) <= 0)
16        err_quit("inet_pton error for %s", argv[1]);

17    if (connect(sockfd, (SA *) &servaddr, sizeof(servaddr)) < 0)
18        err_sys("connect error");

19    while ( (n = read(sockfd, recvline, MAXLINE)) > 0) {
20        recvline[n] = 0;        /* null terminate */
21        if (fputs(recvline, stdout) == EOF)
22            err_sys("fputs error");
23    }
24    if (n < 0)
25        err_sys("read error");

26    exit(0);
27 }
```

intro/daytimetcpcli.c

Figure 1.5 TCP daytime client.

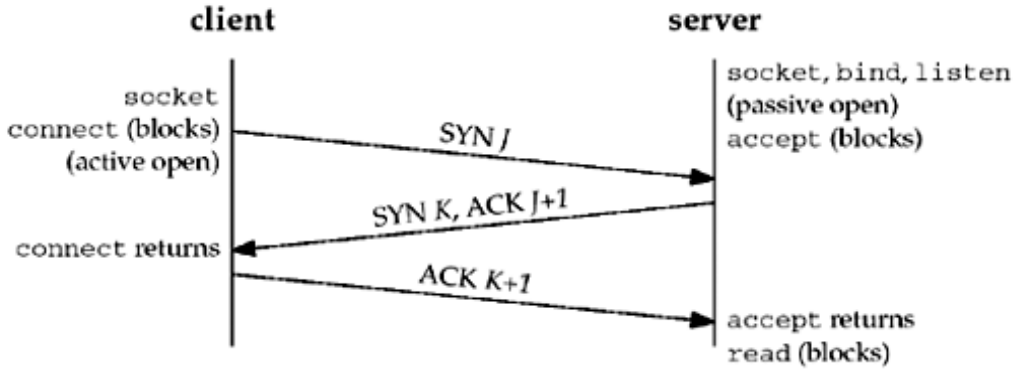
```
1 #include    "unp.h"
2 #include    <time.h>

3 int
4 main(int argc, char **argv)
5 {
6     int     listenfd, connfd;
7     struct sockaddr_in servaddr;
8     char    buff[MAXLINE];
9     time_t  ticks;

10    listenfd = Socket(AF_INET, SOCK_STREAM, 0);

11    bzero(&servaddr, sizeof(servaddr));
12    servaddr.sin_family = AF_INET;
13    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
14    servaddr.sin_port = htons(13);  /* daytime server */

15    Bind(listenfd, (SA *) &servaddr, sizeof(servaddr));

16    Listen(listenfd, LISTENQ);

17    for ( ; ; ) {
18        connfd = Accept(listenfd, (SA *) NULL, NULL);

19        ticks = time(NULL);
20        snprintf(buff, sizeof(buff), "%.24s\r\n", ctime(&ticks));
21        Write(connfd, buff, strlen(buff));

22        Close(connfd);
23    }
24 }
```

intro/daytimetcpsrv.c

Figure 1.9 TCP daytime server.

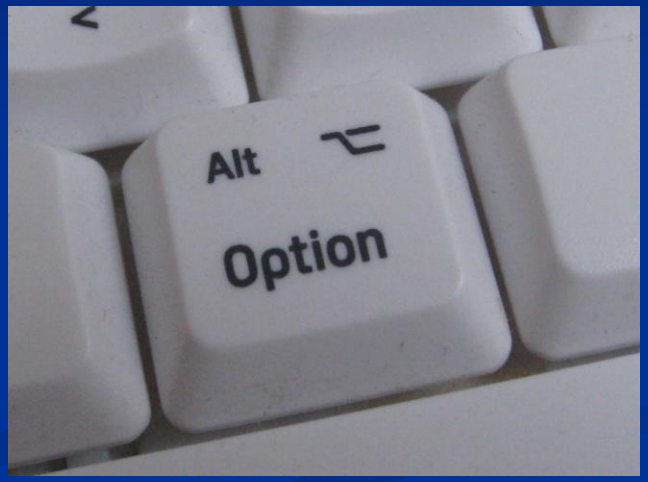
TCP Connection Establishment

Figure 2.2. TCP three-way handshake.



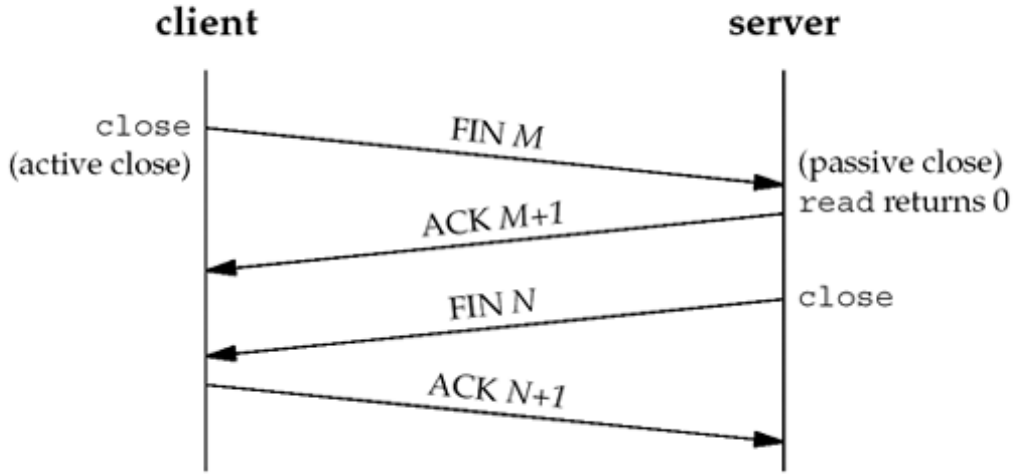
TCP Options

- Options in SYN packet
  - MSS (maximum segment size)
    - maximum data in one segment
    - default = 512 bytes
  - Window Scale
    - sliding window size
    - max = 65,535 bytes
  - Timestamp
    - to prevent data corruption of old, delayed, duplicated segments



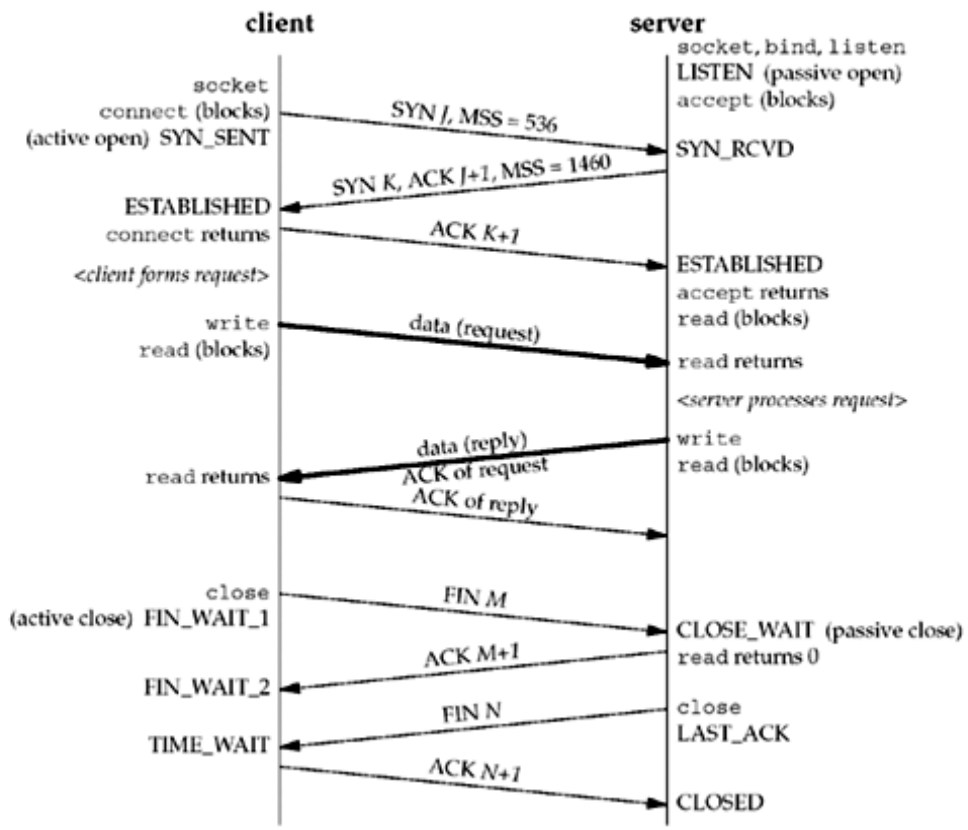
TCP Connection Termination

Figure 2.3. Packets exchanged when a TCP connection is closed.



TCP Packet Exchange

Figure 2.5. Packet exchange for TCP connection.

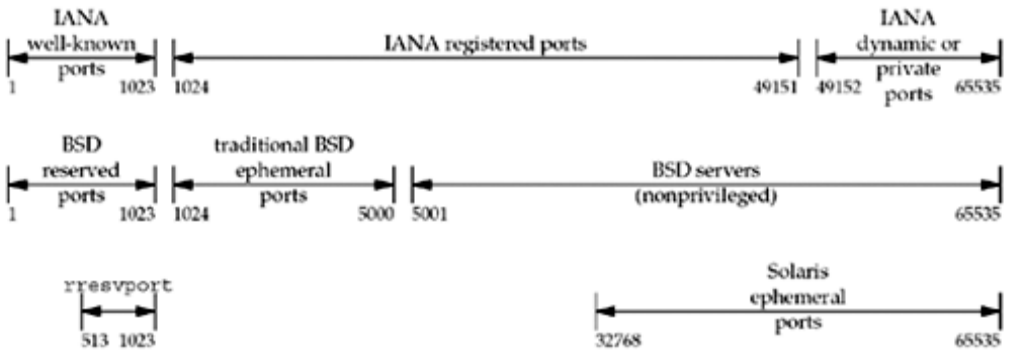


Port Numbers

- 16-bit
- 3 types
  - well-known ports — 0 - 1023
  - registered ports — 1024 - 49151
  - dynamic/private/ephemeral ports — 49152 - 65535

Internet Assigned Numbers Authority

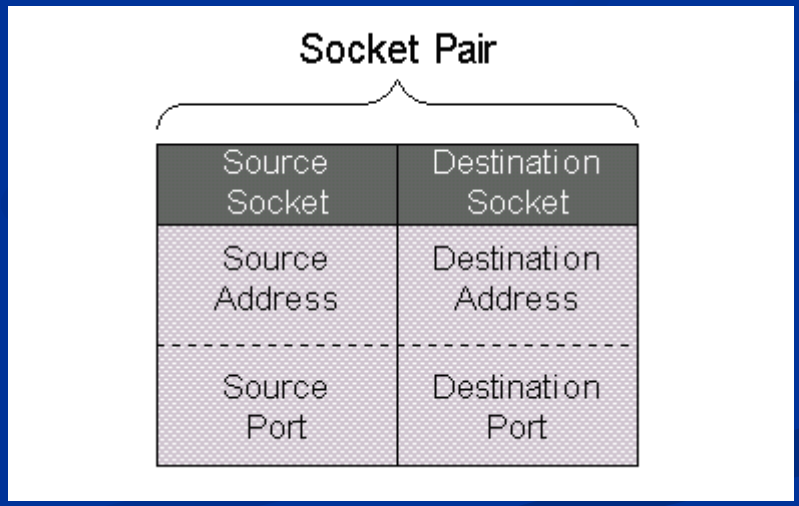
Figure 2.10. Allocation of port numbers.

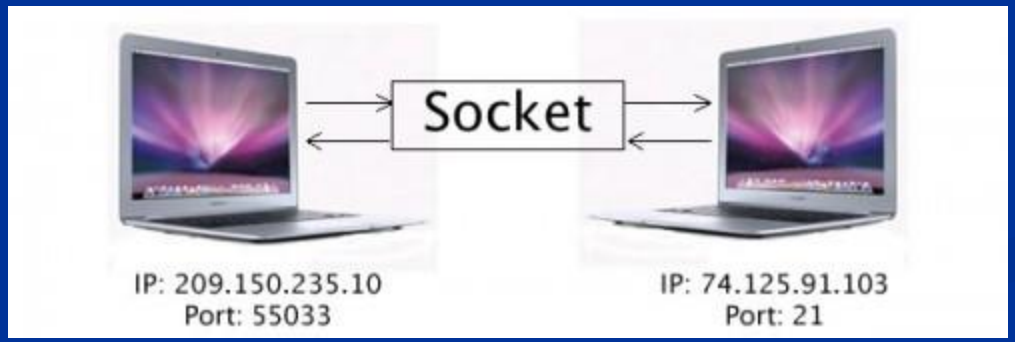


Some Well Know Port Numbers	Services
21	FTP
22	SSH
23	Telnet
25	SMTP
53	DNS
80	HTTP
110	POP3
143	IMAP

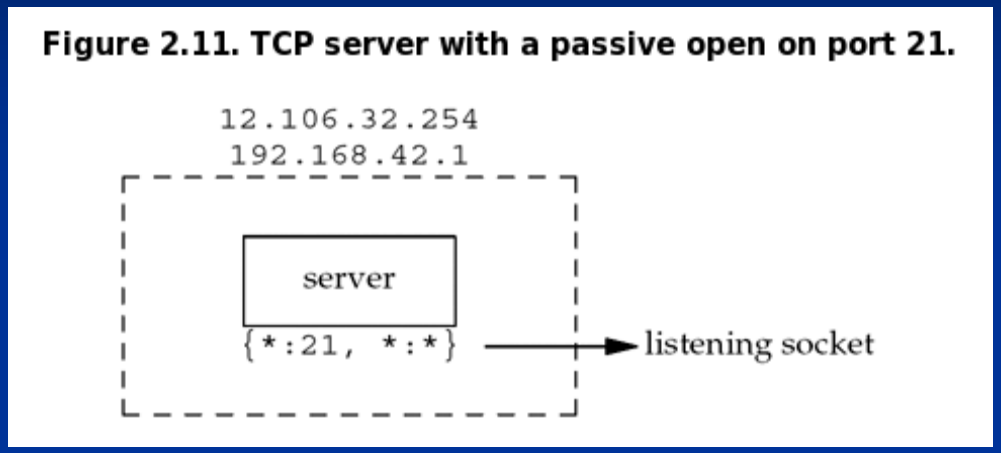
Socket Pair

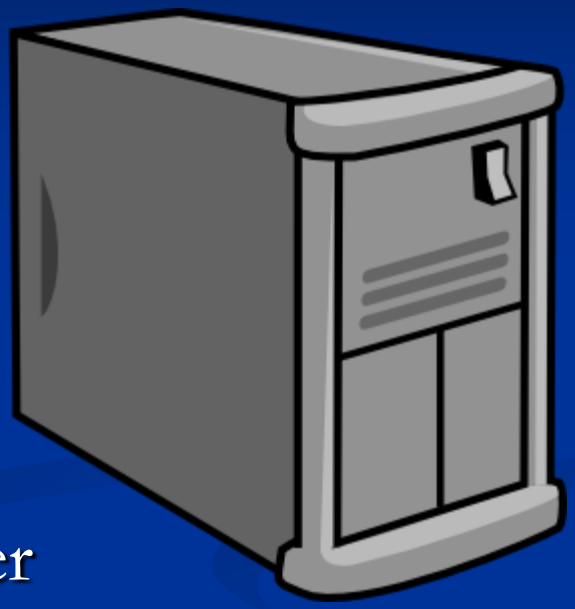
- Two sockets located on either end of the TCP connection
- Socket = IP address + port number
- Uniquely identifies every TCP connection
  - local IP address
  - local port number
  - foreign IP address
  - foreign port number





TCP Server





- { * : 21, * : * } = socket pair for server
  - local address not specified
    - can receive connections from any network interface (address)
  - foreign address and port number not specified
    - in LISTENING mode

Concurrent Server

Figure 2.12. Connection request from client to server.

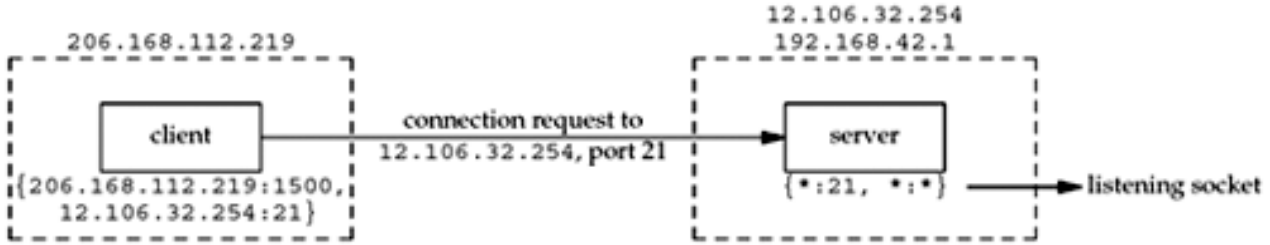
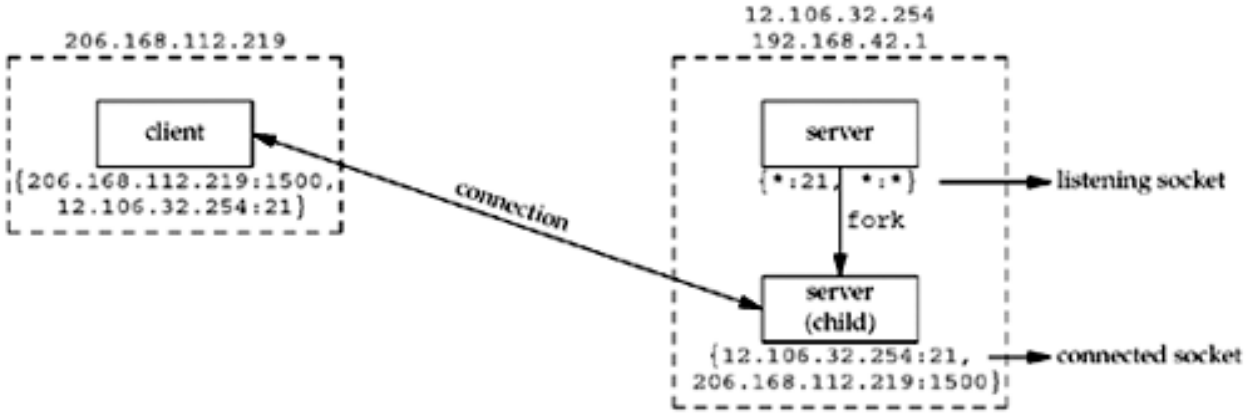
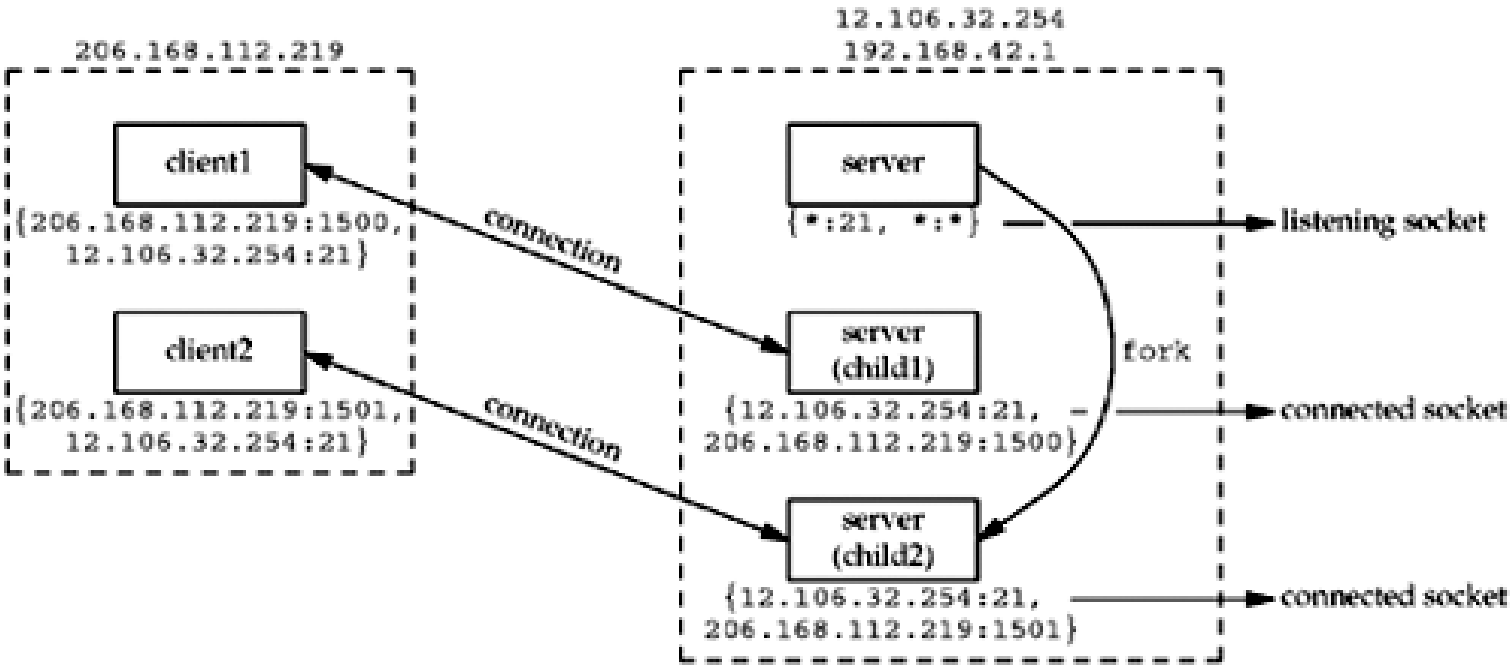


Figure 2.13. Concurrent server has child handle client.



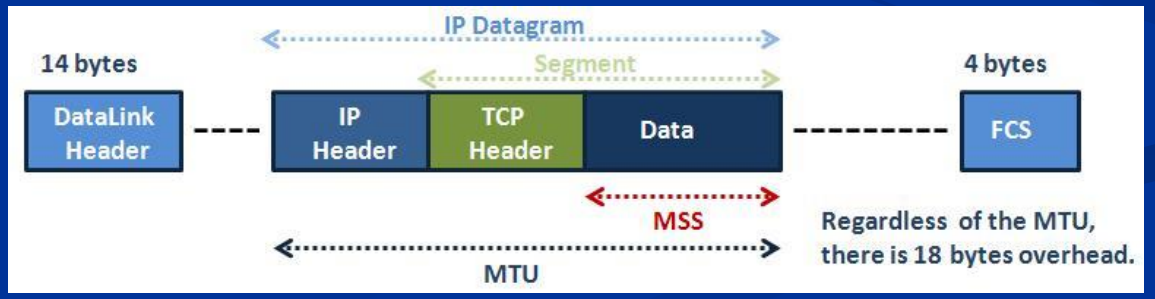
Concurrent Server (cont.)

Figure 2.14. Second client connection with same server.



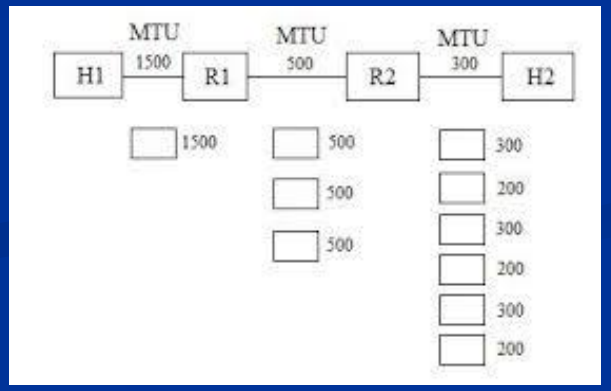
Buffer Sizes and Limitations

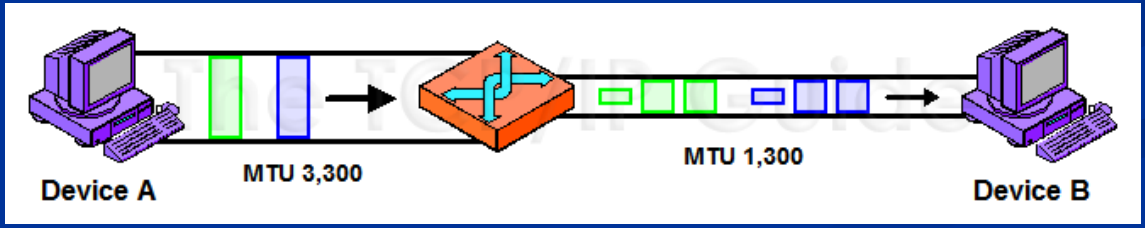
- Maximum Transmission Unit (MTU) size dependent on network technology
  - Ethernet = 1,500 bytes
  - PPP = configurable size
  - IPv4 = 68 - 65,535 bytes (including header)
  - IPv6 = 1,280 - 65,575 bytes (including 40-byte header)
    - IPv6 can work over smaller MTU links but requires fragmentation and reassembly
    - IPv6 has jumbo payload option for datalinks with MTU size > 65,535 bytes



MTU Size and Fragmentation/Reassembly

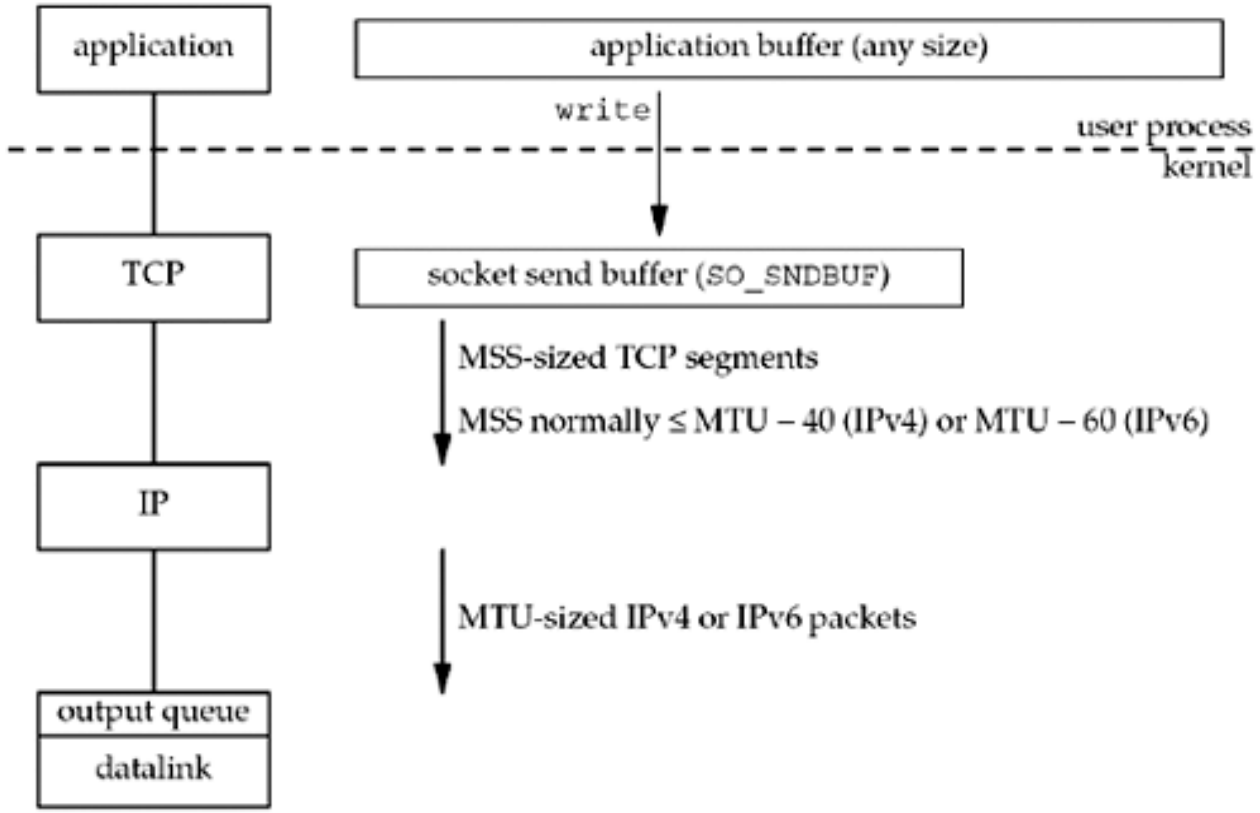
- Path MTU
  - smallest MTU size along the path between 2 hosts
  - need not be identical for A → B and B → A (asymmetric links)
- IPv4 datagram size > path MTU size
  - IPv4 host/router performs fragmentation
- IPv6 datagram size > path MTU size
  - only IPv6 host performs fragmentation, not routers
  - path MTU discovery used to determine min MTU size
- Reassembly only done at final destination
- Don't Fragment (DF) flag
  - cause routers to drop packets exceeding MTU size on outgoing interface





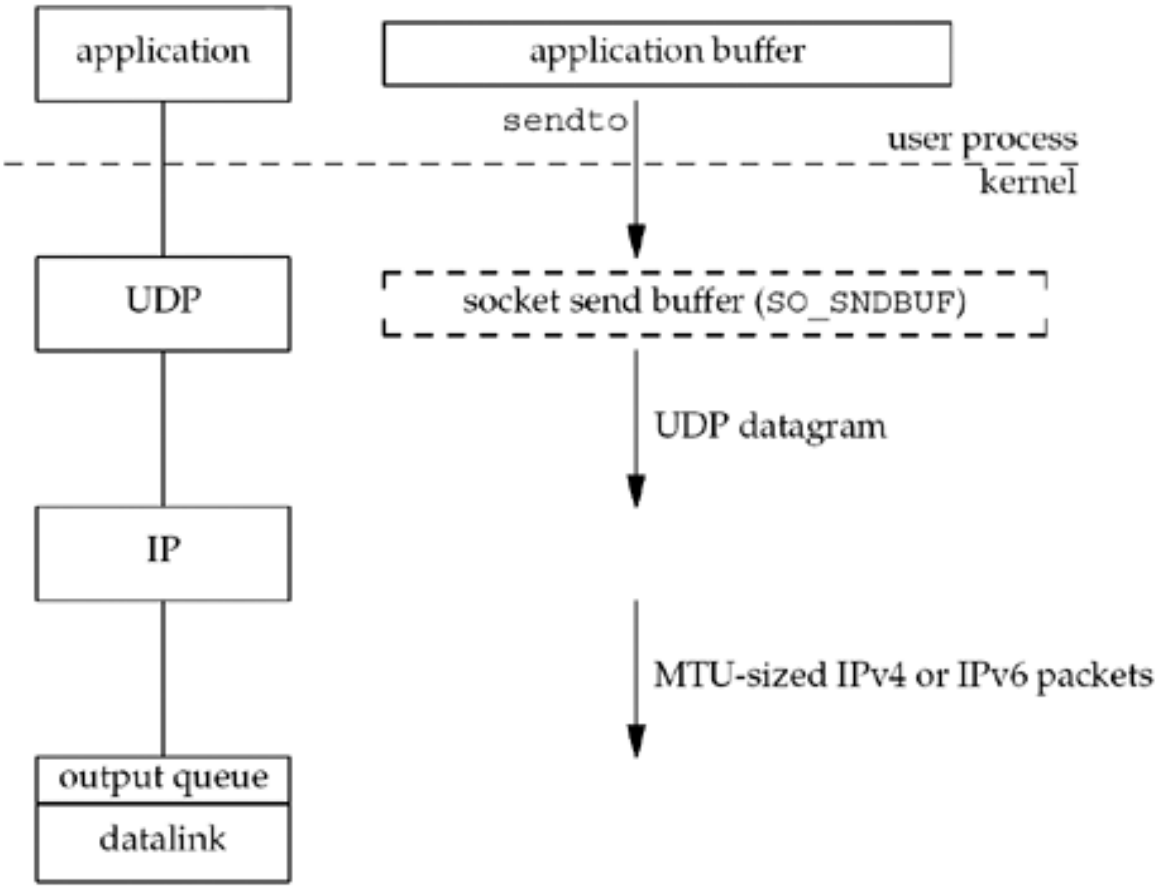
TCP Application Output

Figure 2.15. Steps and buffers involved when an application writes to a TCP socket.



UDP Application Output

Figure 2.16. Steps and buffers involved when an application writes to a UDP socket.



Standard Internet Services

Figure 2.18. Standard TCP/IP services provided by most implementations.

Name	TCP port	UDP port	RFC	Description
echo	7	7	862	Server returns whatever the client sends.
discard	9	9	863	Server discards whatever the client sends.
daytime	13	13	867	Server returns the time and date in a human-readable format.
chargen	19	19	864	TCP server sends a continual stream of characters, until the connection is terminated by the client. UDP server sends a datagram containing a random number of characters (between 0 and 512) each time the client sends a datagram.
time	37	37	868	Server returns the time as a 32-bit binary number. This number represents the number of seconds since midnight January 1, 1900, UTC.

Application Protocol Usage

Figure 2.19. Protocol usage of various common Internet applications.

Application	IP	ICMP	UDP	TCP	SCTP
ping		•			
traceroute		•	•		
OSPF (routing protocol)	•				
RIP (routing protocol)			•		
BGP (routing protocol)				•	
BOOTP (bootstrap protocol)			•		
DHCP (bootstrap protocol)			•		
NTP (time protocol)			•		
TFTP			•		
SNMP (network management)			•		
SMTP (electronic mail)				•	
Telnet (remote login)				•	
SSH (secure remote login)				•	
FTP				•	
HTTP (the Web)				•	
NNTP (network news)				•	
LPR (remote printing)				•	
DNS			•	•	
NFS (network filesystem)			•	•	
Sun RPC			•	•	
DCE RPC			•	•	
IUA (ISDN over IP)					•
M2UA,M3UA (SS7 telephony signaling)					•
H.248 (media gateway control)			•	•	•
H.323 (IP telephony)			•	•	•
SIP (IP telephony)			•	•	•