Introduction

- A Simple Daytime Client
- A Simple Daytime Server
- Error handling: wrapper functions
- Types of Networking APIs
- BSD networking history
- Discover details of your local network

A Simple Daytime Client

- Create TCP socket: get a file descriptor
- Prepare server address structure: fill-in IP address and port number
- Connect to the server: bind the file descriptor with the remote server
- Read/write from/to server
- Close socket

Client Program Usage

```
solaris %a.out 206.168.112.96
Mon May 26 20:58:40 2003
```

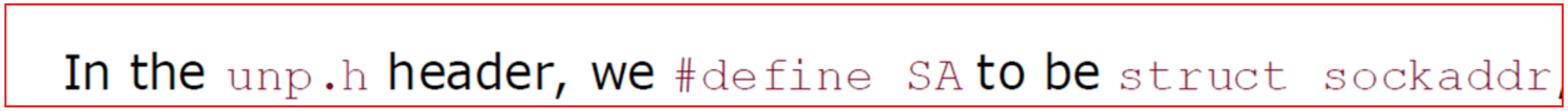
our input

the program's output

```
#include"unp.h  “

int
main(int argc, char **argv)
{
	int sockfd, n;
	char			recvline[MAXLINE + 1];
	struct sockaddr_in	servaddr;

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

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

intro/daytimetcpcli.c

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


In the unp.h header, we #define SA to be struct sockaddr,

```
  if (connect(sockfd, (SA *) &servaddr, sizeof(servaddr)) < 0)
            err_sys("connect error");
  while ( (n = read(sockfd, recvline, MAXLINE)) > 0) {
            recvline[n] = 0;      /* null terminate */
            if (fputs(recvline, stdout) == EOF)
                  err_sys("fputs error");
   }
   if (n < 0)
       err_sys("read error");
   exit(0);
}
```

```
#include	"unp.h"

int
main(int argc, char **argv)
{
	int					sockfd, n;
	struct sockaddr_in6	servaddr;
	char				recvline[MAXLINE + 1];

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

	if ( (sockfd = socket(AF_INET6, SOCK_STREAM, 0)) < 0)
		err_sys("socket error");
bzero(&servaddr, sizeof(servaddr));
 servaddr.sin6_family = AF_INET6;
 servaddr.sin6_port   = htons(13);/* daytime server */
 if (inet_pton(AF_INET6, argv[1], &servaddr.sin6_addr) <= 0)
 err_quit("inet_pton error for %s", argv[1]);

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

	while ( (n = read(sockfd, recvline, MAXLINE)) > 0) {
		recvline[n] = 0;	/* null terminate */
		if (fputs(recvline, stdout) == EOF)
			err_sys("fputs error");
	}
	if (n < 0)
		err_sys("read error");
	exit(0);
}
```

A Simple Daytime Server

- Create TCP socket: get a file descriptor
- Bind the socket with its local port
- Listen: convert the socket to a listening descriptor
- Accept blocks to sleep
- Accept returns a connected descriptor
- Read/write
- Close socket

Problems with the the Simple Server

- Protocol dependency on IPv4
- Iterative server: no overlap of service times of different clients
- Need for concurrent server: fork, pre-fork, or thread
- Need for daemon: background, unattached to a terminal

intro/daytimetcpsrv.c

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

int
main(int argc, char **argv)
{
 int listenfd, connfd;
 struct sockaddr_in servaddr;
 char buff[MAXLINE];
 time_t ticks;

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

intro/daytimetcpsrv.c

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

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

 Listen(listenfd, LISTENQ);

 for ( ; ; ) {
    connfd = Accept(listenfd, (SA *) NULL, NULL);
    ticks = time(NULL);
     snprintf(buff, sizeof(buff), "%.24s\r\n", ctime(&ticks));
    Write(connfd, buff, strlen(buff));
     Close(connfd);
  }
}
```

Error Handling: wrappers

lib/wrapsock.c

```
/* include Socket */
int
Socket(int family, int type, int protocol)
{
	int		n;

	if ( (n = socket(family, type, protocol)) < 0)
		err_sys("socket error");
	return(n);
}
/* end Socket */
```

Types of Networking APIs

- TCP socket
- UDP socket
- raw socket over IP (bypass TCP/UDP)
- datalink (bypass IP)
  - BPF (BSD Packet Filter)
  - DLPI (SVR4 Data Link Provider Interface)

BSD Networking History

- BSD releases:
  - Licensed: 4.2 BSD with TCP/IP and socket APIs(1983), 4.3 BSD (1986), 4.3 BSD Tahoe (1988), 4.3 Reno (1990), 4.4 BSD (1993)
  - Non-licensed: Net/1 (Tahoe) (1989), Net/2 (Reno) (1991), 4.4 BSD-Lite (Net/3) (1994), 4.4 BSD-Lite2 (1995) ---> BSD/OS, FreeBSD, NetBSD, OpenBSD
- System V Release 4: Solaris and Linux
- UNIX Standards: POSIX and The Open Group

Discovering Details of Your Local Network

- To find out interfaces: netstat -ni
- To find out routing table: netstat -rn
- To find out details of an interface: ifconfig
- To discover hosts on a LAN: ping