Echo Client-Server program usin UDP Protocol

Prof. Anand D Dave
Department of Information Technology,
Dharmsinh Desai University, Nadiad.

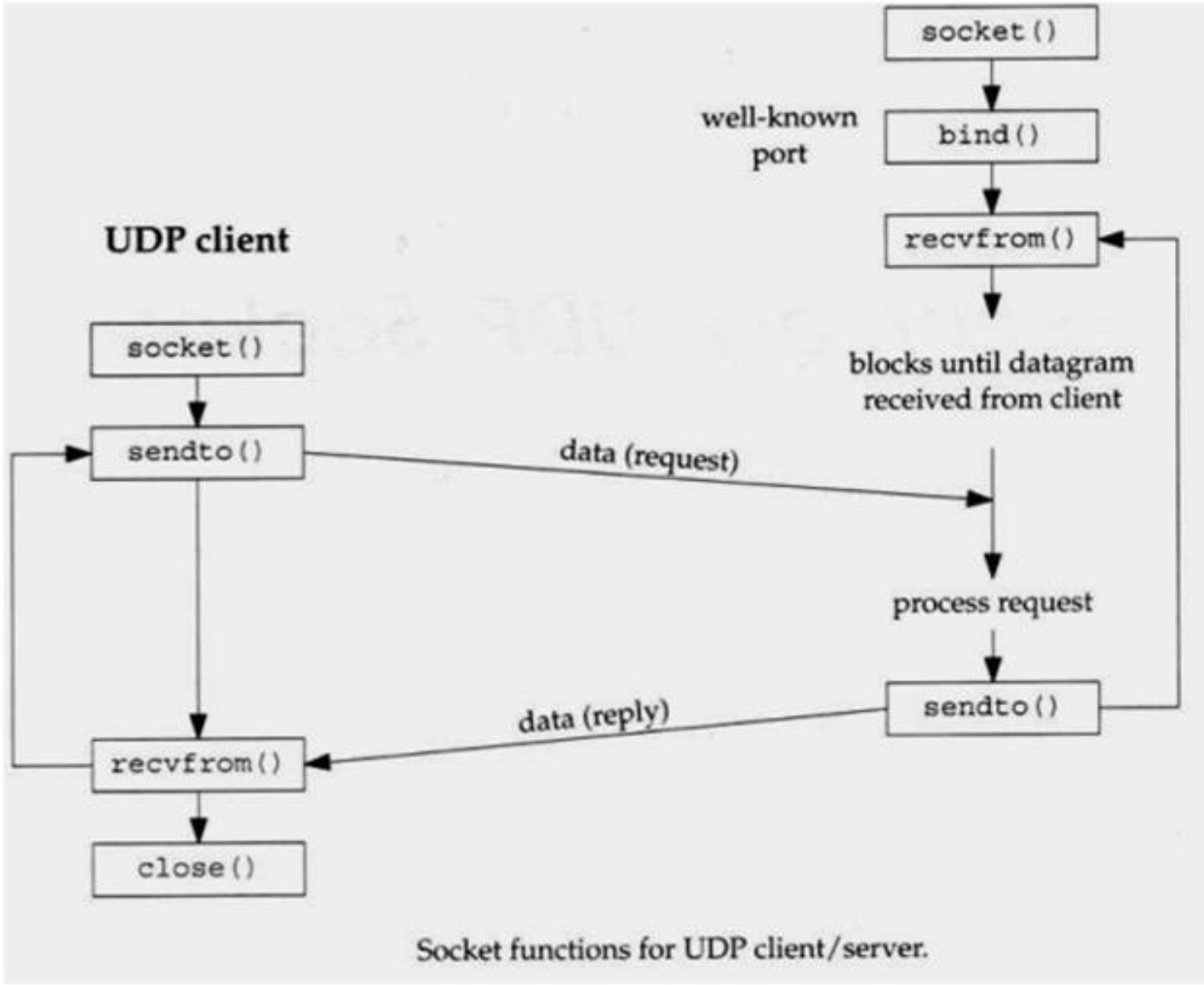
Client-Server Using UDP Socket

- UDP is
  - Connectionless
  - Unreliable
  - datagram protocol
- Some popular applications are built using UDP
  - DNS(the Domain Name System)
  - NFS(the Network File System)
  - SNMP(Simple Network Management Protocol)

Structure of UDP client-server program

- Client does not establish a connection to the server
  - client just sends a datagram to the server using sendto() function.
- similarly, server does not accept a connection from client.
  - It instead simply calls recvfrom() function which waits for data to arrive.

Structure of UDP client-server program



Socket functions for UDP client/server.

UDP Socket functions

sendto and recvfrom functions

```
#include <sys/socket.h>
ssize_t sendto(int sockfd, const void *buff, size_t nbytes, int
flags, const struct sockaddr *to, socklen_t addrlen);

ssize_t recvfrom(int sockfd, void *buff, size_t nbytes, int flags,
struct sockaddr *from, socklen_t *addrlen);
```

- Both return: number of bytes read or written if OK, -1 on error
- First three arguments, sockfd, buff, and nbytes, are identical to the first three arguments for read and write calls.

UDP Socket functions

sendto and recvfrom functions

ssize_t sendto(int sockfd, const void *buff, size_t nbytes, int flags, const struct sockaddr *to, socklen_t addrlen);

– flags: set this to 0 for simple UDP programs.
– to: socket address structure containing the protocol address (IPaddress and port #) of where data is to be sent.

– addrlen: Size of socket address structure (integer value argument)

UDP Socket functions

ssize_t recvfrom(int sockfd, void *buff, size_t nbytes, int flags,
struct sockaddr *from, socklen_t *addrlen);

– flags: set this to 0 for simple UDP programs.
– from: socket address structure that is filled in by recvfrom with the protocol address (IP address and port #) of where data came from

– *addrlen: pointer to integer value. (Value-result argument).

• If the from argument is a null pointer, then the corresponding length argument (addrlen) must also be a null pointer. I.e., we are not interested in knowing who sent us data.
• Final two arguments to sendto are similar to the final two arguments to connect.

UDP Socket functions

sendto and recvfrom functions

- Writing a datagram of length 0 is OK.
  – In the case of UDP
- This results in an **IP datagram** containing an IP header (20 bytes IPv4), an 8 byte **UDP header** and no data.

- This means return value of 0 from recvfrom is OK for a datagram protocol.

  – It does not mean that peer has closed the connection (as happens in case of TCP).
  – UDP is connectionless. No meaning of closing the connection.

UDP Client-Server Program - Echo client-server application: echoUDPServer.c

- ```
  #include <sys/types.h>
  #include <sys/socket.h>
  #include <stdio.h>
  #include <netinet/in.h>
  #define MAXLINESIZE 100
  #define SERV_PORT 8000
  int main(int argc, char** argv)
  {
  int sockfd;
  struct sockaddr_in servaddr, cliaddr;
  int n;
  socklen_t len;
  char mesg [MAXLINESIZE];
  ```
  ```

UDP Client-Server Program - Echo client-server application: echoUDPServer.c

```
/*Create socket*/

    sockfd = socket(AF_INET,SOCK_DGRAM,0);

    /*Initialize socket address structure*/

    bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(SERV_PORT);

    /*bind socket address to the socket*/

    bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
```

UDP Client-Server Program - Echo client-server application: echoUDPServer.c

```
for(; ; ) {
    len=sizeof(cliaddr);
    /*wait for datagram to arrive, receive a datagram*/
    n=recvfrom(sockfd,mesg,MAXLINESIZE,0,(struct sockaddr*)&cliaddr,&len);
    /*send back the datagram to the client from which it arrived*/
    sendto(sockfd,mesg,n,0,(struct sockaddr*)&cliaddr,len);
     }
    }
```

UDP Client-Server Program - Echo client-server application: echoUDPClient.c

```
#include <sys/types.h>
 #include <sys/socket.h>
 #include <stdio.h>
 #include <netinet/in.h>
 #define MAXLINE 100
 #define SERV_PORT 8000
 int main(int argc, char** argv)
 {
 int sockfd;
 struct sockaddr_in servaddr;
 int n;
 char sendline[MAXLINE],recvline[MAXLINE];
```

UDP Client-Server Program - Echo client-server application: echoUDPClient.c

```
  if(argc!=2){

fprintf(stderr,"Usage: %s IP-Address\n",argv[0]);
exit(-1);
}

/*Initialize socket address structure*/

bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family = AF_INET;
if(inet_pton(AF_INET,argv[1], &servaddr.sin_addr)<0)
fprintf(stderr,"Error in inet_pton");
servaddr.sin_port = htons(SERV_PORT);
```

UDP Client-Server Program - Echo client-server application: echoUDPClient.c

```
/*Create socket*/
  sockfd = socket (AF_INET,SOCK_DGRAM,0);
  while(fgets(sendline,MAXLINE,stdin)!=NULL)
  {
  /*Send the message to the server*/

  sendto(sockfd,sendline,strlen(sendline),0,(struct  sockaddr*)&servaddr,sizeof(servaddr));

  /*wait for response and receive the response*/

  n = recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);

  recvline[n]  = 0;

/*Display what the server  sent in reply*/

  fputs(recvline,stdout);
      }
  }
```

UDP Client-Server Program - Echo client-server application: echoUDPClient.c

```
/*Create socket*/
  sockfd = socket (AF_INET,SOCK_DGRAM,0);
  while(fgets(sendline,MAXLINE,stdin)!=NULL)
  {
  /*Send the message to the server*/

  sendto(sockfd,sendline,strlen(sendline),0,(struct  sockaddr*)&servaddr,sizeof(servaddr));

  /*wait for response and receive the response*/

  n = recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);

  recvline[n]  = 0;

/*Display what the server  sent in reply*/

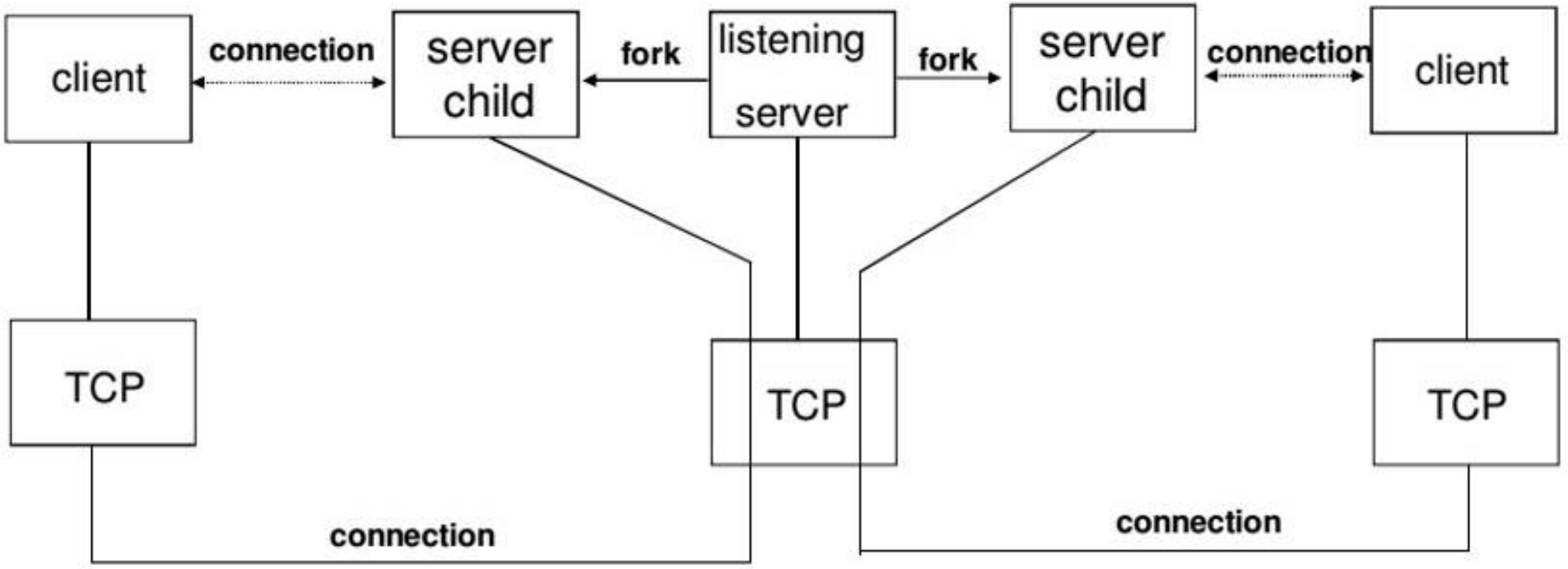
  fputs(recvline,stdout);
      }
  }
```

UDP Client-Server Program - Echo client-server application: echoUDPClient.c

- Compile echoTCPServer.c and echoTCPClient.c using gcc.
- First run the server.
- Run clients and test.

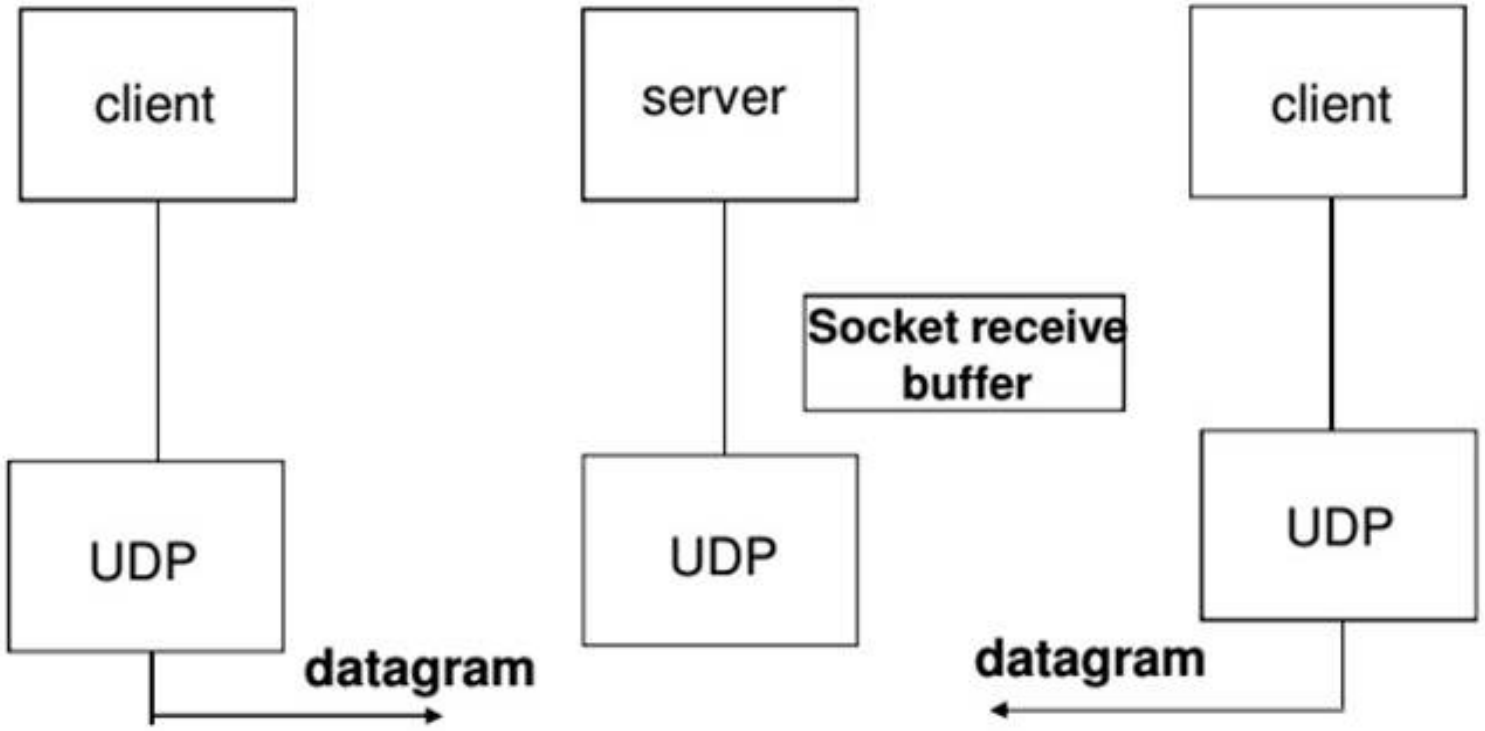
Comparing TCP and UDP based programs

- TCP client server with two clients



Comparing TCP and UDP based programs

- UDP client-server with two clients.



References

- Sources :
- https://www.javatpoint.com/
- https://www.tutorialspoint.com/software_architecture_design/distributed_architecture.htm
- https://sites.google.com/site/prajapatiharshadb/class-notes-for-students
- UNIX Network Programming by W. Richard Stevens, Prentice Hall Publication
- Distributed Computing: Concepts &Applications: by M. L. Liu Addison Wiselly