Model practical Exam

Reg no : 185001161

Date : 11/10/2020

Code:

server :

local:

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<string.h>

#include<ctype.h>

#include<netinet/in.h>

#include<sys/socket.h>

struct table

{

char ip[100];

char server[100];

};

int k = 0;

struct table \*create\_table(int n)

{

int i;

char test[100];

struct table \*t;

t = (struct table\*)malloc(sizeof(struct table)\*n);

return t;

}

void update\_table(struct table \*t,char ip[],char domain[])

{

strcpy(t[k].ip,ip);

strcpy(t[k].server,domain);

k++;

}

char \*get\_ip(char buff[],struct table t[],int n)

{

int i;

for(i=0;i<n;i++)

{

if(strcmp(buff,t[i].server)==0)

{

return t[i].ip;

}

}

strcpy(buff,"");

return buff;

}

void print\_table(int n,struct table t[])

{

int i;

printf("\n\tDomain Name\t\t\tIp Adrress\n");

for(i=0;i<n;++i)

{

printf("\t%-20s\t%s\n",t[i].server,t[i].ip);

}

}

int main(int argc,char \*\*argv)

{

int len,ret;

int sockfd,n,rootfd,tldfd,authofd;

struct sockaddr\_in serveraddr,clientaddr,rootaddr,tldaddr,authoaddr;

char buff[1024],str[1024],buff2[1024];

sockfd = socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd < 0)

{

printf("-----Error in Connection-----.\n");

exit(1);

}

printf("-----Local server Socket is Created!.\n");

bzero(&serveraddr,sizeof(serveraddr));

bzero(&clientaddr,sizeof(clientaddr));

bzero(&rootaddr,sizeof(rootaddr));

bzero(&tldaddr,sizeof(tldaddr));

bzero(&authoaddr,sizeof(authoaddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

serveraddr.sin\_port = htons(7228);

rootaddr.sin\_family = AF\_INET;

rootaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

rootaddr.sin\_port = htons(7227);

tldaddr.sin\_family = AF\_INET;

tldaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

tldaddr.sin\_port = htons(7226);

authoaddr.sin\_family = AF\_INET;

authoaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

authoaddr.sin\_port = htons(7225);

if(bind(sockfd,(struct sockaddr\*)&serveraddr,sizeof(serveraddr))<0)

{

printf("\nbind error\n");

exit(1);

}

rootfd =socket(PF\_INET,SOCK\_DGRAM,0);

tldfd =socket(PF\_INET,SOCK\_DGRAM,0);

authofd=socket(PF\_INET,SOCK\_DGRAM,0);

int l1,l2,l3;

len = sizeof(clientaddr);

l1 = sizeof(rootaddr);

l2 = sizeof(tldaddr);

l3 = sizeof(authoaddr);

struct table \*t;

t = create\_table(100);

char domain[100];

print\_table(k,t);

int stop ;

while(1)

{

n = recvfrom(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,&len);

strcpy(domain,buff);

if(strcmp(get\_ip(domain,t,k),"")!=0)

{

strcpy(buff2,get\_ip(domain,t,k));

printf("\nfound\n");

print\_table(k,t);

n = sendto(sockfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&clientaddr,len);

}

printf("\nThe requested host name is : %s\n",buff);

stop = 0;

n = sendto(rootfd,&buff,sizeof(buff),0,(struct sockaddr\*)&rootaddr,l1);

while(1)

{

n = recvfrom(rootfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&rootaddr,&l1);

if(strcmp(buff2,"")!=0)

{

update\_table(t,buff2,domain);

print\_table(k,t);

n = sendto(sockfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&clientaddr,len);

stop =1;

}

else

break;

}

if(stop!=1){

stop = 0;

n = sendto(tldfd,&buff,sizeof(buff),0,(struct sockaddr\*)&tldaddr,l2);

while(1)

{

n = recvfrom(tldfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&tldaddr,&l2);

if(strcmp(buff2,"")!=0)

{

update\_table(t,buff2,domain);

print\_table(k,t);

n = sendto(sockfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&clientaddr,len);

stop =1;

}

else

break;

}}

if(stop!=1){

stop = 0;

n = sendto(authofd,&buff,sizeof(buff),0,(struct sockaddr\*)&authoaddr,l3);

while(1)

{

n = recvfrom(authofd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&authoaddr,&l3);

if(strcmp(buff2,"")!=0)

{

update\_table(t,buff2,domain);

print\_table(k,t);

n = sendto(sockfd,&buff2,sizeof(buff2),0,(struct sockaddr\*)&clientaddr,len);

stop =1;

}

else

break;

}}

}

close(sockfd);

return 0;

}

root:

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<string.h>

#include<ctype.h>

#include<netinet/in.h>

#include<sys/socket.h>

struct table

{

char ip[100];

char server[100];

};

int k =0;

struct table \*create\_table(int n)

{

int i;

char test[100];

struct table \*t;

t = (struct table\*)malloc(sizeof(struct table)\*n);

return t;

}

void update\_table(struct table \*t,char ip[],char domain[])

{

strcpy(t[k].ip,ip);

strcpy(t[k].server,domain);

k++;

}

char \*get\_ip(char buff[],struct table t[],int n)

{

int i;

for(i=0;i<n;i++)

{

if(strcmp(buff,t[i].server)==0)

{

return t[i].ip;

}

}

strcpy(buff,"");

return buff;

}

void print\_table(int n,struct table t[])

{

int i;

printf("\n\tDomain Name\t\t\tIp Adrress\n");

for(i=0;i<n;++i)

{

printf("\t%-20s\t%s\n",t[i].server,t[i].ip);

}

}

int main(int argc,char \*\*argv)

{

int len;

int sockfd,n;

struct sockaddr\_in serveraddr,clientaddr;

char buff[1024];

char str[1000];

sockfd = socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd <0)

{

printf("-----Error in Connection-----.\n");

exit(1);

}

printf("-----Root server Socket is Created!.\n");

bzero(&serveraddr,sizeof(serveraddr));

bzero(&clientaddr,sizeof(clientaddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

serveraddr.sin\_port = htons(7227);

if(bind(sockfd,(struct sockaddr\*)&serveraddr,sizeof(serveraddr))<0)

{

printf("\nbind error\n");

exit(1);

}

len = sizeof(clientaddr);

struct table \*t;

t = create\_table(100);

print\_table(k,t);

while(1)

{

n= recvfrom(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,&len);

printf("\nThe Request is : %s\n",buff);

strcpy(buff,get\_ip(buff,t,k));

if (strcmp(buff,"")==0)

{

printf("\n Not available in root dns\n");

strcpy(buff,"");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

else

{

printf("\nfound\n");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

}

close(sockfd);

return 0;

}

tld:

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<string.h>

#include<ctype.h>

#include<netinet/in.h>

#include<sys/socket.h>

struct table

{

char ip[100];

char server[100];

};

int k =0;

struct table \*create\_table(int n)

{

int i;

char test[100];

struct table \*t;

t = (struct table\*)malloc(sizeof(struct table)\*n);

return t;

}

void update\_table(struct table \*t,char ip[],char domain[])

{

strcpy(t[k].ip,ip);

strcpy(t[k].server,domain);

k++;

}

char \*get\_ip(char buff[],struct table t[],int n)

{

int i;

for(i=0;i<n;i++)

{

if(strcmp(buff,t[i].server)==0)

{

return t[i].ip;

}

}

strcpy(buff,"");

return buff;

}

void print\_table(int n,struct table t[])

{

int i;

printf("\n\tDomain Name\t\t\tIp Adrress\n");

for(i=0;i<n;++i)

{

printf("\t%-20s\t%s\n",t[i].server,t[i].ip);

}

}

int main(int argc,char \*\*argv)

{

int len;

int sockfd,n;

struct sockaddr\_in serveraddr,clientaddr;

char buff[1024];

char str[1000];

sockfd = socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd <0)

{

printf("-----Error in Connection-----.\n");

exit(1);

}

printf("-----Root server Socket is Created!.\n");

bzero(&serveraddr,sizeof(serveraddr));

bzero(&clientaddr,sizeof(clientaddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

serveraddr.sin\_port = htons(7226);

if(bind(sockfd,(struct sockaddr\*)&serveraddr,sizeof(serveraddr))<0)

{

printf("\nbind error\n");

exit(1);

}

len = sizeof(clientaddr);

struct table \*t;

t = create\_table(100);

print\_table(k,t);

while(1)

{

n= recvfrom(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,&len);

printf("\nThe Request is : %s\n",buff);

strcpy(buff,get\_ip(buff,t,k));

if (strcmp(buff,"")==0)

{

printf("\n Not available in tld dns\n");

strcpy(buff,"");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

else

{

printf("\nfound\n");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

}

close(sockfd);

return 0;

}

authoritative :

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<string.h>

#include<ctype.h>

#include<netinet/in.h>

#include<sys/socket.h>

struct table

{

char ip[100];

char server[100];

};

int k =0;

struct table \*create\_table(int n)

{

int i;

char test[100];

struct table \*t;

t = (struct table\*)malloc(sizeof(struct table)\*n);

strcpy(t[0].server,"www.yahoo.com");

strcpy(t[0].ip,"10.2.45.67");

strcpy(t[1].server,"www.annauniv.edu");

strcpy(t[1].ip,"197.34.53.122");

strcpy(t[2].server,"www.google.com");

strcpy(t[2].ip,"142.89.78.66");

return t;

}

void update\_table(struct table \*t,char ip[],char domain[])

{

strcpy(t[k].ip,ip);

strcpy(t[k].server,domain);

k++;

}

char \*get\_ip(char buff[],struct table t[],int n)

{

int i;

for(i=0;i<n;i++)

{

if(strcmp(buff,t[i].server)==0)

{

return t[i].ip;

}

}

strcpy(buff,"");

return buff;

}

void print\_table(int n,struct table t[])

{

int i;

printf("\n\tDomain Name\t\t\tIp Adrress\n");

for(i=0;i<n;++i)

{

printf("\t%-20s\t%s\n",t[i].server,t[i].ip);

}

}

int main(int argc,char \*\*argv)

{

int len;

int sockfd,n;

struct sockaddr\_in serveraddr,clientaddr;

char buff[1024];

char str[1000];

sockfd = socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd <0)

{

printf("-----Error in Connection-----.\n");

exit(1);

}

printf("-----Root server Socket is Created!.\n");

bzero(&serveraddr,sizeof(serveraddr));

bzero(&clientaddr,sizeof(clientaddr));

serveraddr.sin\_family = AF\_INET;

serveraddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

serveraddr.sin\_port = htons(7225);

if(bind(sockfd,(struct sockaddr\*)&serveraddr,sizeof(serveraddr))<0)

{

printf("\nbind error\n");

exit(1);

}

len = sizeof(clientaddr);

struct table \*t;

t = create\_table(3);

k = 3;

print\_table(k,t);

while(1)

{

n= recvfrom(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,&len);

printf("\nThe Request is : %s\n",buff);

strcpy(buff,get\_ip(buff,t,k));

if (strcmp(buff,"")==0)

{

printf("\n NOt avaulable in root dns\n");

strcpy(buff,"");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

else

{

printf("\nfound\n");

n = sendto(sockfd,&buff,sizeof(buff),0,(struct sockaddr\*)&clientaddr,len);

}

}

close(sockfd);

return 0;

}

client:

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

#include<netdb.h>

#include<unistd.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<string.h>

#include<ctype.h>

#include<netinet/in.h>

#include<sys/socket.h>

//#define PORT 7228

int main(int argc,char \*\*argv)

{

int sockfd,ret,len,n;

struct sockaddr\_in serverAddr;

char buffer[1024];

char str[1000];

sockfd = socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd < 0)

{

printf("-----Error in Connection-----.\n");

exit(1);

}

printf("-----Client Socket is Created!.\n");

bzero(&serverAddr,sizeof(serverAddr));

serverAddr.sin\_family = AF\_INET;

serverAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

serverAddr.sin\_port = htons(7228);

len = sizeof(serverAddr);

int x;

char domain[100];

while(1)

{

printf("\n\n1.Request\n2.Exit \n Enter choice:");

scanf("%d",&x);

if (x==1)

{

printf("\nEnter the domain name:");

scanf("%s",buffer);

sendto(sockfd,buffer,sizeof(buffer),0,(struct sockaddr \*)&serverAddr,len);

recvfrom(sockfd,buffer,sizeof(buffer),0,(struct sockaddr\*)&serverAddr,&len);

printf("\nIP address :%s",buffer);

}

else if(x==2)

{

break;

}

}

}

Output Screenshot:

Everything works fine as heirarchy.

