Exp No: 7 **Date:** 11/10/2020

Domain Name Server **Name:** Swetha Saseendran

**Register Number**: 185001183

# Code:

## Client:

#include "table.h"

int main()

{

    char buffer[MAXLINE],IP[MAXLINE];

    char hostname[200] ;

    int sockfd;

    struct sockaddr\_in serveraddr;

    socklen\_t addr\_size;

    //Conn with Local Server

    bzero(&serveraddr, sizeof(serveraddr));

    sockfd = setConn(&serveraddr, cliPORT, 0, "local server");

    addr\_size = sizeof(serveraddr);

    bzero(&buffer, sizeof(buffer));

    while(1)

    {

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

            scanf("%s", hostname);

            strcpy(buffer,hostname);

            //send hostname to local

            sendto(sockfd, &buffer, sizeof(buffer), 0, (struct sockaddr\*)&serveraddr, addr\_size);

            //recv IP from local

            recvfrom(sockfd, &buffer, sizeof(buffer), 0, (struct sockaddr\*)&serveraddr, &addr\_size);

            printf("The IP Address is :\t\t%s\n", buffer);

    }

    close(sockfd);

    return 0;

}

## Authoritative Server:

//Only .COM auth

#include "table.h"

int main()

{

    struct sockaddr\_in local;

    int sockfd, n;

    socklen\_t addr\_size, len;

    char buffer[MAXLINE], reply[50], \*ip;

    struct hostent \*he;

    sockfd = setConn(&local, authPORT, 1, "local server");

    addr\_size = sizeof(local);

    while(1)

    {

            bzero(&buffer,sizeof(buffer));

            //recv hostname from local

            recvfrom(sockfd, &buffer, sizeof(buffer), 0, (struct sockaddr\*)&local, &addr\_size);

            printf("\nReceived IP request from local server for %s",buffer);

            he = gethostbyname(buffer);

            ip = inet\_ntoa(\*((struct in\_addr \*)he->h\_addr\_list[0]));//get the IP address from host entry

            //sending IP to local

            strcpy(reply, ip);

            sendto(sockfd,&reply, sizeof(reply), 0, (struct sockaddr\*) &local, addr\_size);

            printf("\nReplied back with IP %s to local server.\n", ip);

   }

   close(sockfd);

   return 0;

}

## Root Server:

//Root will be like a server to local

#include "table.h"

#define PORT "4004"

int main()

{

    int sockfd;

    char buffer[MAXLINE];

    socklen\_t addr\_size, len;

    struct sockaddr\_in local;

    sockfd = setConn(&local, rootPORT, 1, "local server");

    addr\_size = sizeof(local);

    while(1)

    {

            //recv hostname from local

            bzero(buffer,MAXLINE);

            recvfrom(sockfd, &buffer, MAXLINE, 0, (struct sockaddr\*)&local, &addr\_size);

            printf("\nReceived request from local DNS server for %s",buffer);

        strcpy(buffer,PORT);

        printf("\nReplied back with port address [%s] of TLD \n",buffer);

        //send to local server

        sendto(sockfd, &buffer, sizeof(buffer), 0, (struct sockaddr\*) &local, addr\_size);

    }

    close(sockfd);

    return 0;

}

## TLD Server:

//Only .COM TLD

#include "table.h"

#define PORT "4008" //here we only use .com auth

int main()

{

    int sockfd;

    char buffer[MAXLINE];

    socklen\_t addr\_size, len;

    struct sockaddr\_in local;

    sockfd = setConn(&local, tldPORT, 1, "local server");

    addr\_size = sizeof(local);

    while(1)

    {

            bzero(&buffer,sizeof(buffer));

            //recv hostname from local

            recvfrom(sockfd, &buffer, sizeof(buffer), 0, (struct sockaddr\*)&local, &addr\_size);

            printf("\nReceived request from local server for %s",buffer);

        strcpy(buffer,PORT);

        //send to local server

        sendto(sockfd,&buffer, sizeof(buffer), 0, (struct sockaddr\*) &local, addr\_size);

        printf("\nReplied back with port address [%s] of Authoritative Server.\n",buffer);

    }

    close(sockfd);

    return 0;

}

## Local Server:

#include "table.h"

Table lookup[30];

int cli\_fd, root\_fd, tld\_fd, auth\_fd;

struct sockaddr\_in client\_addr, local\_addr, root\_addr, tld\_addr, auth\_addr;

int main()

{

    int position, k = 0;

    char buffer[MAXLINE],hostname[1024],IPs[1024];

    socklen\_t addr\_size, len;

    cli\_fd  = setConn(&local\_addr, cliPORT, 1, "client");  //bind here since server

    root\_fd = setConn(&root\_addr, rootPORT, 0, "root server");   //do not bind here

    tld\_fd  = setConn(&tld\_addr, tldPORT, 0, "TLD server");   //do not bind here

    auth\_fd = setConn(&auth\_addr, authPORT, 0, "authoritative server"); //do not bind here

    addr\_size = sizeof(client\_addr);

    while(1)

    {

        bzero(&buffer, sizeof(buffer));

        //recv hostname from clients

    recvfrom(cli\_fd, &buffer, sizeof(buffer), 0, (struct sockaddr\*)&client\_addr,&addr\_size);

    printf("\n\nReceived a request for IP Address of %s from a client.",buffer);

    strcpy(hostname, buffer);

    position = checkTable(lookup, buffer, k);

    if (position == -1)   //if not in lookup

    {

      printf("\nLocal table does not have an entry for %s.\n\tRequesting Root Server.............",hostname);

      //store domain name in lookUp table

      strcpy(lookup[k].domainName,hostname);

      /\*

        Communicating with ROOT

      \*/

      len = sizeof(root\_addr);

      //send hostname to root

      bzero(buffer, MAXLINE);

      sendto(root\_fd, &hostname, sizeof(hostname), 0, (struct sockaddr\*)&root\_addr, len);

      //recv tld addr from root

      recvfrom(root\_fd, &buffer, MAXLINE, 0, (struct sockaddr\*)&root\_addr, &len);

      printf("\nRoot replied with address port [%s] to request the TLD Server.\n\tRequesting TLD Server.............",buffer);

      /\*

        Communicating with TLD

      \*/

      len = sizeof(tld\_addr);

      //send hostname to tld

      bzero(buffer, MAXLINE);

      sendto(tld\_fd, &hostname, sizeof(hostname), 0, (struct sockaddr\*) &tld\_addr, len);

      //recv authaddr from tld

      recvfrom(tld\_fd, buffer, MAXLINE, 0, (struct sockaddr\*)&tld\_addr, &len);

      printf("\nTLD replied with address port [%s] to request the Authoritative Server.\n\tRequesting Authoritative Server.............",buffer);

      /\*

        Communicating with Auth

      \*/

      len = sizeof(auth\_addr);

      //send hostname to auth

      bzero(&buffer, MAXLINE);

      sendto(auth\_fd, &hostname, sizeof(hostname), 0, (struct sockaddr\*) &auth\_addr, len);

      //recv authaddr from tld

      recvfrom(auth\_fd,&buffer, MAXLINE, 0, (struct sockaddr\*)&auth\_addr, &len);

      printf("\nAuthoritative Server replied with IP address: %s.",buffer);

      //Setting IP's in table

      strcpy(lookup[k].ip,buffer);

      k++;

       }

       else //IP in lookup

       {

          printf("\nLocal table has an entry for %s.\n",hostname);

          bzero(&buffer, MAXLINE);

          strcpy(buffer,lookup[position].ip);

          printf("\nLocal Server replied with IP address: %s.",buffer);

       }

      //send IPs to client

      addr\_size = sizeof(client\_addr);

      sendto(cli\_fd,&buffer,sizeof(buffer), 0, (struct sockaddr\*)&client\_addr, addr\_size);

   }

   close(cli\_fd);

   return 0;

}

## “table” Header File:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#include <stdlib.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <netdb.h>

#define MAXLINE 1024

//Named relative to local server

#define cliPORT 4000

#define rootPORT 4002

#define tldPORT 4004 //here we only use .com TLD

#define authPORT 4008 //here we only use .com auth

struct Table

{

    char domainName[200];

    char ip[10]; //IP

};

typedef struct Table Table;

//find if a string matches to domainName of list of Tables of size k

int error(char \*msg)

{

    perror(msg);

    exit(1);

}

int setConn(struct sockaddr\_in \*conn, int port, int is\_bound, char \*conn\_name)

{

    printf("\nConnecting to %s",conn\_name);

    int sockfd;

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

    if(sockfd < 0)

        error("Error in creating socket.\n");

    bzero(conn, 16);

    conn->sin\_family = AF\_INET;

    conn->sin\_addr.s\_addr = inet\_addr("127.0.0.1");

    conn->sin\_port = htons(port);

    if(is\_bound)

        if(bind(sockfd, (struct sockaddr \*)conn, 16) < 0)

            error("Error in binding.\n");

        else

            printf("\nConnection to %s is successful.\n", conn\_name);

    return sockfd;

}

int checkTable(Table lookup[50], char \*hostname , int k)

{

    for(int i = 0; i < k; i++)

        if(strcmp(hostname, lookup[i].domainName) == 0)

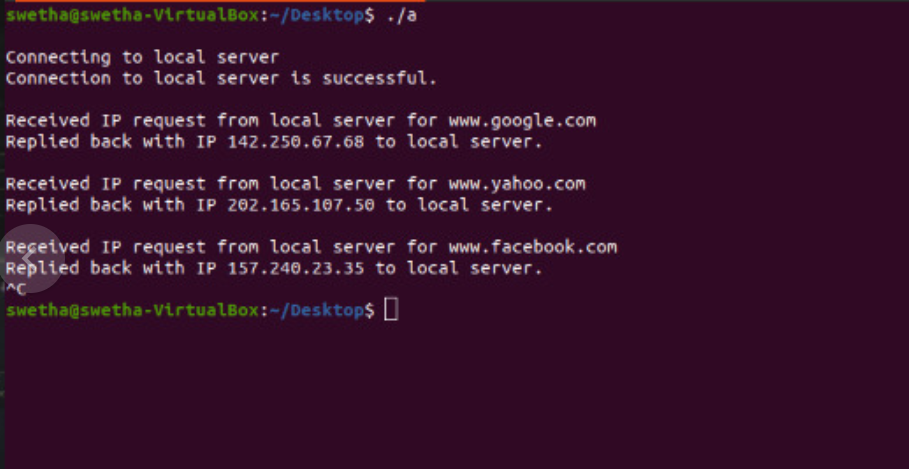
              return i;

    return -1;

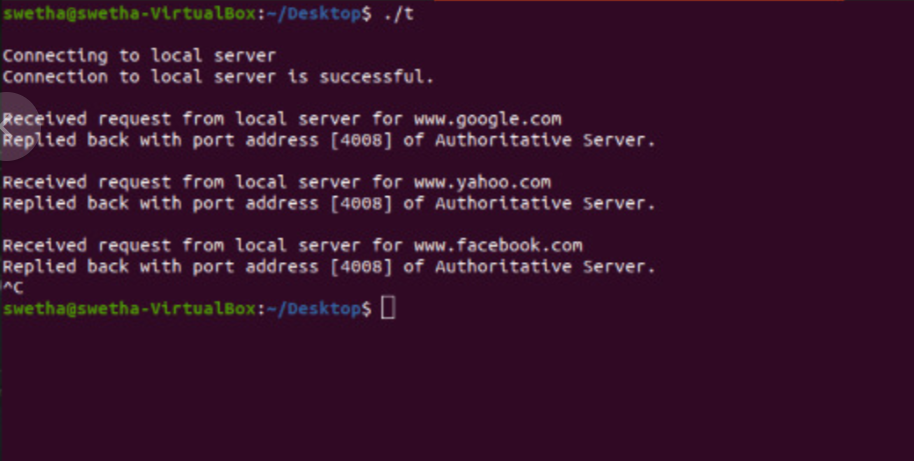
}

# Sample Input Output:

## Authoritative Server:



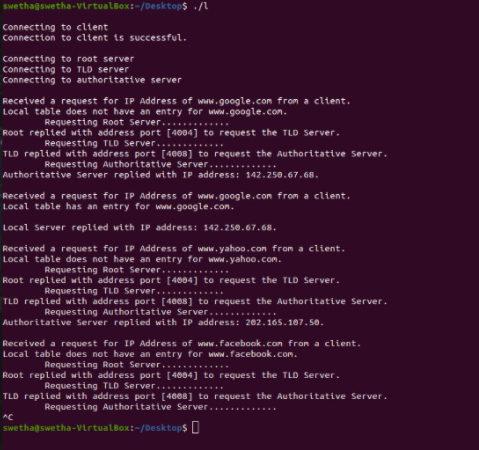
**TLD Server:**



## Root Server:



## Local Server:



## Client:

