UDP DNS Server Program

19Z510 – COMPUTER NETWORKS LABORATORY

Anandkumar NS (22Z209)

**BACHELOR** **OF** **ENGINEERING**



Date: 19/08/2024

**DEPARTMENT** **OF** **COMPUTER** **SCIENCE** **ENGINEERING** **PSG** **COLLEGE** **OF** **TECHNOLOGY**

(Autonomous Institution)

**COIMBATORE** **– 641** **004**

Client:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define MAX 80

#define PORT 3000

#define SA struct sockaddr

void send\_request(int sockfd, struct sockaddr\_in \*servaddr) {

while(1){

char buffer[MAX];

char response[MAX];

socklen\_t len = sizeof(\*servaddr);

// Get URL from user

printf("Enter the URL: ");

fgets(buffer, sizeof(buffer), stdin);

buffer[strcspn(buffer, "\n")] = '\0'; // Remove newline character

if (strcmp(buffer,"exit")==0){

break;

}

else{ // Send URL to server

sendto(sockfd, buffer, strlen(buffer), 0, (SA\*)servaddr, len);

printf("URL Sent\n");

// Receive response from server

recvfrom(sockfd, response, sizeof(response), 0, NULL, NULL);

printf("Server response: %s\n", response);

}

}

}

int main() {

int sockfd;

struct sockaddr\_in servaddr;

// Create UDP socket

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

if (sockfd < 0) {

perror("Socket creation failed");

exit(EXIT\_FAILURE);

}

printf("UDP socket created\n");

// Initialize server address

memset(&servaddr, 0, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

// Send request and receive response

send\_request(sockfd, &servaddr);

close(sockfd);

return 0;

}

Server:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define MAX 80

#define PORT 3000

#define SA struct sockaddr

// Function to look up the IP address for the given URL

void handle\_request(int sockfd, struct sockaddr\_in \*client\_addr, socklen\_t client\_len) {

char buffer[MAX];

char response[MAX];

FILE \*fp;

char url[MAX];

char ip[MAX];

socklen\_t len = sizeof(\*client\_addr);

// Receive URL from client

ssize\_t recv\_len = recvfrom(sockfd, buffer, sizeof(buffer) - 1, 0, (SA\*)client\_addr, &client\_len);

if (recv\_len < 0) {

perror("Receive failed");

return;

}

buffer[recv\_len] = '\0'; // Null-terminate the received string

// Debugging: Print the received URL

printf("Received URL: %s\n", buffer);

// Open the text file containing URL-IP pairs

fp = fopen("data.txt", "r");

if (fp == NULL) {

perror("Unable to open file");

snprintf(response, sizeof(response), "Server error");

sendto(sockfd, response, strlen(response), 0, (SA\*)client\_addr, len);

return;

}

// Initialize response with "URL not found" message

snprintf(response, sizeof(response), "URL not found");

// Read the file line by line

while (fscanf(fp, "%s %s", url, ip) != EOF) {

if (strcmp(buffer, url) == 0) {

snprintf(response, sizeof(response), "%s", ip);

break;

}

}

fclose(fp);

// Debugging: Print the response to be sent

printf("Sending response: %s\n", response);

// Send the response to the client

sendto(sockfd, response, strlen(response), 0, (SA\*)client\_addr, len);

}

int main() {

int sockfd;

struct sockaddr\_in servaddr, cli;

socklen\_t len = sizeof(cli);

// Create UDP socket

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

if (sockfd < 0) {

perror("Socket creation failed");

exit(EXIT\_FAILURE);

}

printf("UDP socket created\n");

// Initialize server address

memset(&servaddr, 0, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

servaddr.sin\_port = htons(PORT);

// Bind the socket

if (bind(sockfd, (SA\*)&servaddr, sizeof(servaddr)) < 0) {

perror("Bind failed");

close(sockfd);

exit(EXIT\_FAILURE);

}

printf("UDP server listening on port %d\n", PORT);

// Handle incoming requests

while (1) {

handle\_request(sockfd, &cli, len);

}

close(sockfd);

return 0;

}