Department of Computer Science and Engineering

S.G.Shivanirudh , 185001146, Semester V

16 September 2020

UCS1511 - Networks Laboratory

Exercise 6: Domain Name Server using UDP

Objective:

Simulate the concept of **Domain Name Server** using UDP.

Code:

DNS Table structure:

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
```

```c
#include<unistd.h>
#include<arpa/inet.h>

#define ADDR_LIMIT 5
#define DOMAIN_LIMIT 10

//Record of each domain-address pair
struct record{
    char *domain;
    char *address[ADDR_LIMIT];
};

typedef struct record Record;

void init(Record *r){
    r->domain = (char*)calloc(100, sizeof(char));
    for(int i =0;i<ADDR_LIMIT;i++)
        r->address[i] = (char*)calloc(100, sizeof(char));
}

//Print DNS Table
void DNSTable(Record table[DOMAIN_LIMIT]){
    printf(" _____ \n");
    printf("|___Domain Name___|________Address_______|\n");

    for (int i = 0; i < DOMAIN_LIMIT; i++){
        if (table[i].domain[0]){
            printf("| %-15s | %-20s |\n", table[i].domain,
    table[i].address[0]);

            for (int j = 1; j < ADDR_LIMIT && table[i].
    address[j][0]; j++)
                printf("| %-15s | %-20s |\n", "", table[i].
    address[j]);
            printf("|_________________|______________________
    |\n");
        }
     }
    printf("\n");
}

//Check if newly specified address is a valid address
int checkAddress(Record *table, char *address){

    char* addr_copy = (char*)calloc(100, sizeof(char));
```

```
48      strcpy(addr_copy, address);
49      char *split;
50      int val;
51      split = strtok(addr_copy, ".");
52      //Check if all octets lie within 0 and 255.
53      while (split){
54          val = atoi(split);
55          if (val < 0 || val > 255){
56              printf("\nError: Invalid Address.\n");
57              return 0;
58          }
59          split = strtok(NULL, ".");
60      }
61
62      //Check if new address already exists in the table
63      for (int i = 0; i < DOMAIN_LIMIT; i++){
64          if (!table[i].domain[0])
65              continue;
66
67          for (int j = 0; j < ADDR_LIMIT && table[i].address[j
    ][0]; j++)
68              if (strcmp(address, table[i].address[j]) == 0){
69                  printf("\nError: IP address already exists.\n
    ");
70                  return 0;
71              }
72      }
73
74      return 1;
75  }
76
77  //Create DNS-address pair in the table
78  int createRecord(Record table[DOMAIN_LIMIT], char *domain,
        char *address){
79
80      int ix = -1;
81      int flag = 0;
82      //Check if entry exists already
83      int addr_valid = checkAddress(table, address);
84      if (!addr_valid)
85          return flag;
86
87      for (int i = 0; i < DOMAIN_LIMIT; i++){
88          if (strcmp(table[i].domain, domain) == 0){
89              for (int j = 0; j < DOMAIN_LIMIT; j++)
```

```
                if (!table[i].address[j][0]){
                    strcpy(table[i].address[j], address);
                    flag = 1;
                    break;
                }
            break;
        }
        if (!table[i].domain[0] && ix == -1)
            ix = i;
    }

    // If record can be created
    if (!flag){
        strcpy(table[ix].domain, domain);
        strcpy(table[ix].address[0], address);
        flag = 1;
    }

    return flag;
}

char *getAddress(Record *table, char *domain){

    char* addresses = (char*)calloc(ADDR_LIMIT*20, sizeof(
    char));

    for (int i = 0; i < DOMAIN_LIMIT; i++){
        if (strcmp(table[i].domain, domain) == 0){
            for (int j = 0; j < ADDR_LIMIT; j++)            {
                strcat(addresses, table[i].address[j]);
                strcat(addresses, " ");
            }
            break;
        }
    }
    return addresses;
}
```

Server:

```
#include "DNSTable.h"

```

```
int main(int argc, char **argv){

    Record table[DOMAIN_LIMIT];
    char *addresses = (char*)calloc(ADDR_LIMIT*2, sizeof(char));
    for(int i =0; i<DOMAIN_LIMIT;i++){
        init(&table[i]);
    }

    if (argc > 1){
        perror("\nError: No arguments needed for server.\n");
        exit(0);
    }

    //Server and Client addresses
    struct sockaddr_in server_address, client_address;
    //Buffer to handle messages
    char buffer[1024];

    char *domain = (char*)calloc(100, sizeof(char));
    char *address = (char*)calloc(100, sizeof(char));

    //Server socket file descriptor
    int sockfd = socket(AF_INET, SOCK_DGRAM, 0);//domain = IPv4, type = UDP, protocol = ip
    if (sockfd < 0)
        perror("\nError: Socket creation failed!\n");

    //Filling server_address with null bytes
    bzero(&server_address, sizeof(server_address));

    server_address.sin_family = AF_INET;// Uses Internet address family
    server_address.sin_addr.s_addr = htonl(INADDR_ANY);//Use any of the available addresses
    server_address.sin_port = htons(7894);//Use port 7894

    //Bind socket to the specified port
    if ((bind(sockfd, (struct sockaddr *)&server_address, sizeof(server_address)))< 0)
        perror("\nError: Socket bind failed!\n");

    int len = sizeof(client_address);

    //Create record in table
```

```
43      createRecord(table, "google.com", "192.168.1.1");
44      createRecord(table, "yahoo.com", "194.12.34.12");
45      createRecord(table, "google.com", "17.10.23.123");
46
47      //Allow modification of table
48      char opt='n';
49      do{
50          DNSTable(table);
51
52          printf("\nDo you want to update table? y/n: ");scanf(
        " %c", &opt);
53
54          if(opt == 'y' || opt == 'Y'){
55              printf("\nEnter domain: ");scanf(" %[^\n]",
        domain);
56              printf("\nEnter address: ");scanf(" %[^\n]",
        address);
57
58              int rval = createRecord(table, domain, address);
59              if(rval)
60                  printf("\nSuccessfully added entry!!\n");
61          }
62      }while(opt == 'y'||opt == 'Y');
63
64      printf("\nDNS Server set up\n");
65
66      while(1){
67          bzero(&buffer, sizeof(buffer));
68          recvfrom(sockfd, buffer, sizeof(buffer), MSG_WAITALL,
        (struct sockaddr *)&client_address, &len);
69
70          strcpy(addresses, getAddress(table, buffer));
71          sendto(sockfd, addresses, sizeof(buffer), MSG_CONFIRM
        , (struct sockaddr *)&client_address, len);
72      }
73      close(sockfd);
74  }
```

Client:

```
1  #include "DNSTable.h"
2
```

```
3  int main(int argc, char **argv){
4
5      if (argc < 2){
6          perror("\nError: IP address is to be passed as
       argument.\n");
7          exit(0);
8      }
9
10     //Server addresses
11     struct sockaddr_in server_address;
12     //Buffer to handle messages
13     char buffer[1024];
14     //Query
15     Record *query = (Record*)malloc(sizeof(Record));
16
17     //Server socket file descriptor
18     int sockfd = socket(AF_INET, SOCK_DGRAM, 0);
19     if (sockfd < 0){
20         perror("\nError: Socket creation failed!\n");
21     }
22     //Filling server_address with null bytes
23     bzero(&server_address, sizeof(server_address));
24
25     server_address.sin_family = AF_INET;// Uses Internet
       address family
26     server_address.sin_addr.s_addr = INADDR_ANY;//Use any of
       the available addresses
27     server_address.sin_port = htons(7894);//Use port 7894
28
29     int len = sizeof(Record);
30     while(1){
31         init(query);
32
33         printf("\nEnter the domain name: ");scanf(" %[^\n]",
       query->domain);
34
35         if (strcmp(query->domain, "end") == 0)
36             break;
37         //Send requested domain name
38         sendto(sockfd, query->domain, sizeof(buffer),
       MSG_CONFIRM, (struct sockaddr *)&server_address, sizeof(
       server_address));
39
40         bzero(&buffer, sizeof(buffer));
```

```
41          //Recieve IP address(es) of requested domain if it
    exists.
42          recvfrom(sockfd, buffer, sizeof(buffer), MSG_WAITALL,
     (struct sockaddr *)&server_address, &len);
43
44          char* split = strtok(buffer, " ");
45          if(split){
46              printf("\nThe IP Address of the requested domain
    is: ");
47              while(split){
48                  printf("\n%s", split);
49                  split = strtok(NULL, " ");
50              }
51              printf("\n");
52          }
53          else{
54              printf("\nNo address in DNS.\n");
55          }
56      }
57
58      close(sockfd);
59  }
```

Output:

Server:

```
 ________________________________________
|___Domain Name___|________Address_______|
| google.com      | 192.168.1.1          |
|                 | 17.10.23.123         |
|_________________|______________________|
| yahoo.com       | 194.12.34.12         |
|_________________|______________________|


Do you want to update table? y/n: y

Enter domain: google.com

Enter address: 255.254.253.252
```

```
Successfully added entry!!
 ________________________________________
|___Domain Name___|________Address_______|
| google.com      | 192.168.1.1          |
|                 | 17.10.23.123         |
|                 | 255.254.253.252      |
|_________________|______________________|
| yahoo.com       | 194.12.34.12         |
|_________________|______________________|


Do you want to update table? y/n: y

Enter domain: youtube.com

Enter address: 111.234.15.1

Successfully added entry!!
 ________________________________________
|___Domain Name___|________Address_______|
| google.com      | 192.168.1.1          |
|                 | 17.10.23.123         |
|                 | 255.254.253.252      |
|_________________|______________________|
| yahoo.com       | 194.12.34.12         |
|_________________|______________________|
| youtube.com     | 111.234.15.1         |
|_________________|______________________|


Do you want to update table? y/n: n

DNS Server set up
```

Client 1:

```
Enter the domain name: google.com

The IP Address of the requested domain is:
192.168.1.1
17.10.23.123
```

```
255.254.253.252

Enter the domain name: youtube.com

The IP Address of the requested domain is:
111.234.15.1

Enter the domain name: end
```

Client 2:

```
Enter the domain name: youtube.com

The IP Address of the requested domain is:
111.234.15.1

Enter the domain name: yahoo.com

The IP Address of the requested domain is:
194.12.34.12

Enter the domain name: end
```

Client 3:

```
The IP Address of the requested domain is:
194.12.34.12

Enter the domain name: google.com

The IP Address of the requested domain is:
192.168.1.1
17.10.23.123
255.254.253.252

Enter the domain name: end
```

Objective:

Simulate the concept of **Recursive Domain Name Server** using UDP.

Code:

Server:

```
#include "DNSTable.h"

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

    if (argc > 1){
        perror("\nError: No arguments needed for server.\n");
        exit(0);
    }
    //IP address(es) retrieved
    char *addresses = (char*)calloc(ADDR_LIMIT*2, sizeof(char
    ));
    //Table at authoritative level
    Record auth_table[DOMAIN_LIMIT];
    for(int i =0; i<DOMAIN_LIMIT;i++){
        init(&auth_table[i]);
    }
    //Table at local DNS server
    Record local_table[DOMAIN_LIMIT];
    for(int i =0; i<DOMAIN_LIMIT;i++){
        init(&local_table[i]);
    }

    //Server and Client addresses
    struct sockaddr_in server_address, client_address;
    //Buffer to handle messages
    char buffer[1024];

    char *domain = (char*)calloc(100, sizeof(char));
    char *address = (char*)calloc(100, sizeof(char));

    //Server socket file descriptor
```

```
31      int sockfd = socket(AF_INET, SOCK_DGRAM, 0);//domain =
    IPv4, type = UDP, protocol = ip
32      if (sockfd < 0)
33          perror("\nError: Socket creation failed!\n");
34
35      //Filling server_address with null bytes
36      bzero(&server_address, sizeof(server_address));
37
38      server_address.sin_family = AF_INET;// Uses Internet
    address family
39      server_address.sin_addr.s_addr = htonl(INADDR_ANY);//Use
    any of the available addresses
40      server_address.sin_port = htons(7894);//Use port 7894
41
42      //Bind socket to the specified port
43      if ((bind(sockfd, (struct sockaddr *)&server_address,
    sizeof(server_address)))< 0)
44          perror("\nError: Socket bind failed!\n");
45
46      int len = sizeof(client_address);
47
48      //Create record in table
49      createRecord(auth_table, "google.com", "192.168.1.1");
50      createRecord(auth_table, "yahoo.com", "194.12.34.12");
51      createRecord(auth_table, "google.com", "17.10.23.123");
52
53      //Allow modification of table
54      char opt='n';
55      do{
56          DNSTable(auth_table);
57
58          printf("\nDo you want to update table? y/n: ");scanf(
    " %c", &opt);
59
60          if(opt == 'y' || opt == 'Y'){
61              printf("\nEnter domain: ");scanf(" %[^\n]",
    domain);
62              printf("\nEnter address: ");scanf(" %[^\n]",
    address);
63
64              int rval = createRecord(auth_table, domain,
    address);
65              if(rval)
66                  printf("\nSuccessfully added entry!!\n");
67          }
```

```
68      }while(opt == 'y'||opt == 'Y');
69
70      printf("\nDNS Server set up\n");
71
72      while(1){
73          printf("\n%50s\n","-");
74          bzero(&buffer, sizeof(buffer));
75          recvfrom(sockfd, buffer, sizeof(buffer), MSG_WAITALL,
        (struct sockaddr *)&client_address, &len);
76
77          strcpy(addresses, getAddress(local_table, buffer));
78
79          char *copy = (char*)calloc(ADDR_LIMIT*20, sizeof(char
        ));
80          strcpy(copy, addresses);
81
82          printf("\nChecking local DNS server...");
83          char* split = strtok(copy, " ");
84          if(split){
85              printf("Available in local DNS server. \n");
86              sendto(sockfd, addresses, sizeof(buffer),
        MSG_CONFIRM, (struct sockaddr *)&client_address, len);
87          }
88          else{
89              printf("\nNot found in local DNS server.\n");
90
91              printf("\nChecking root DNS server...");
92              printf("\nNot found in root DNS server. \n");
93
94              printf("\nChecking top level DNS server...");
95              printf("\nNot found in top level DNS server. \n")
        ;
96
97              printf("\nChecking authoritative DNS server...");
98              strcpy(addresses, getAddress(auth_table, buffer))
        ;
99
100             strcpy(copy, addresses);
101             split = strtok(copy, " ");
102             if(split){
103                 while(split){
104                     int val = createRecord(local_table,
        buffer, split);
105                     if(val)
```

```
106                         printf("\nSuccessfully added entry in
     local DNS server.\n");
107                     split = strtok(NULL, " ");
108                 }
109                 sendto(sockfd, addresses, sizeof(buffer),
    MSG_CONFIRM, (struct sockaddr *)&client_address, len);
110             }
111             else{
112                 sendto(sockfd, addresses, sizeof(buffer),
    MSG_CONFIRM, (struct sockaddr *)&client_address, len);
113             }
114         }
115 
116 
117     }
118     close(sockfd);
119 }
```

Output:

Server:

```
 ----------------------------------------
|___Domain Name___|________Address_______|
| google.com      | 192.168.1.1          |
|                 | 17.10.23.123         |
|_________________|______________________|
| yahoo.com       | 194.12.34.12         |
|_________________|______________________|


Do you want to update table? y/n: y

Enter domain: youtube.com

Enter address: 255.254.253.252

Successfully added entry!!
 ----------------------------------------
|___Domain Name___|________Address_______|
| google.com      | 192.168.1.1          |
```

```
|                  | 17.10.23.123          |
|__________________|_______________________|
| yahoo.com        | 194.12.34.12          |
|__________________|_______________________|
| youtube.com      | 255.254.253.252       |
|__________________|_______________________|


Do you want to update table? y/n: n

DNS Server set up

-------------------------------------------------

Checking local DNS server...
Not found in local DNS server.

Checking root DNS server...
Not found in root DNS server.

Checking top level DNS server...
Not found in top level DNS server.

Checking authoritative DNS server...
Successfully added entry in local DNS server.

-------------------------------------------------

Checking local DNS server...
Not found in local DNS server.

Checking root DNS server...
Not found in root DNS server.

Checking top level DNS server...
Not found in top level DNS server.

Checking authoritative DNS server...
Successfully added entry in local DNS server.

-------------------------------------------------

Checking local DNS server...
Not found in local DNS server.

```

```
Checking root DNS server...
Not found in root DNS server.

Checking top level DNS server...
Not found in top level DNS server.

Checking authoritative DNS server...
Successfully added entry in local DNS server.

-------------------------------------------------

Checking local DNS server...Available in local DNS server.

-------------------------------------------------

Checking local DNS server...Available in local DNS server.

-------------------------------------------------

Checking local DNS server...Available in local DNS server.
```

Client 1:

```
Enter the domain name: yahoo.com

The IP Address of the requested domain is:
194.12.34.12

Enter the domain name: google.com

The IP Address of the requested domain is:
192.168.1.1
17.10.23.123

Enter the domain name: end
```

Client 2:

```
Enter the domain name: google.com

The IP Address of the requested domain is:
192.168.1.1
17.10.23.123

Enter the domain name: youtube.com

The IP Address of the requested domain is:
255.254.253.252

Enter the domain name: end
```

Client 3:

```
Enter the domain name: youtube.com

The IP Address of the requested domain is:
255.254.253.252

Enter the domain name: yahoo.com

The IP Address of the requested domain is:
194.12.34.12

Enter the domain name: end
```
