Laborator 1 – Retele de calculatoare
Clienti TCP/UDP

1. Folositi o comanda de tip **nslookup** / **dig** / **host** pentru a afla care este serverul de mail al facultatii. Incercati apoi sa va trimiteti un mail pe adresa proprie folosind protocolul SMTP (Simple Mail Transfer Protocol) asa cum ati vazut in clasa.

2. Instalati pe calculator superserverul internet (**inetd** / **xinetd**). Activati serviciile TCP de **daytime** si **echo**, respectiv serviciul UDP **time (***Indicatie*: setati campul *disable* pe *no* in fisierele de configurare ale serviciilor si apoi folositi comanda **service** ca superuser). Cautati in fisierul **/etc/services** porturile pe care functioneaza aceste servicii si incercati sa vedeti cum functioneaza folosind urmatoarea comanda

telnet localhost <port>

Ce se intampla?

3. Folositi apelurile sistem **socket** si **connect** pentru a scrie programe care se conecteaza la serviciile de mai sus. Pentru a putea efectua apelul connect, completati campurile structurii urmatoare in mod corespunzator:

```
struct sockaddr_in
{
    short   sin_family;         // AF_INET
    unsigned short integer sin_port;
    struct  in_addr    sin_addr;
    char    sin_zero[8];
};

struct in_addr
{
    unsigned long integer s_addr;
};
```

Ce se intampla in cazul clientului de **echo** si al trimiterii unor mesaje lungi daca apelati **read** o singura data pentru a primi mesajul serverului?

Indicatie: pt a tipari pe ecran versiunea lizibila a valorii intoarse de serverul de *time*, folositi functia *ctime* (man ctime).

Obs: pentru acest laborator vom folosi numai clienti UDP conectati (adica programe care apeleaza **connect** pentru socketi UDP pentru a putea folosi primitivele uzuale de citire/scriere)

4. Modificati clientul de **echo** a.i. sa verifice daca raspunsul serverului e intr-adevar copia string-ului trimis de client. Adaugati si facilitatea de a numara octetii trimisi respectiv primiti de la server pentru a completa testul de corectitudine.

5. Modificati clientul **echo** sa foloseasca UDP ca protocol de transport. Trimiteti o datagrama UDP catre server si masurati RTT-ul (round trip time). Daca raspunsul nu vine in termen de S secunde, declarati hostul destinatie (serverul) ca fiind imposibil de accesat *(unreachable*). Simulati situatia

cand serverul e unreachable precum si un protocol simplu de retransmisie pentru a simula pierderea de datagrame UDP in Internet.

De cate ori se apeleaza functia **read** spre deosebire de clientul TCP?

6. Modificati programele de la punctul 3 a.i. sa foloseasca functiile **gethostbyname** (pentru a converti un nume de host intr-o adresa de IP), **getservbyname** (pentru a cauta un serviciu de internet dupa nume) si respectiv **getprotobyname** (pentru a cauta un protocol dupa nume). Folositi informatia furnizata de aceste functii pentru a completa structura *sockaddr_in* de mai sus. Structurile de date pe care le veti folosi pentru apeluri arata in felul urmator:

```
struct hostent {
        char  *h_name; /* official host name */
        char  **h_aliases; /* other aliases */
        int h_addrtype; /* address type */
        int h_length; /* address length */
        char **h_addr_list;  /* list of addresses */
};
#define h_addr          h_addr_list[0]

struct servent {
        char *s_name; /* official service name */
        char **s_aliases; /* other aliases */
        int sort; /* port for this service */
        char *s_proto; /* protocol to use */
};

struct protoent {
        char *p_name; / * official protocol name * /
        char **p_aliases; /* list of aliases allowed */
        int p_proto; /* official protocol number */
};
```