Esercitazione 8

RPC

Gruppo 23:

- Bertoni Marco
- Di Simone Paolo
- Iftimie Dragos Mihaita
- Vivarelli Giulia

APPLICAZIONE C/S PER EFFETTUARE OPERAZIONI REMOTE

- dato un file testo in input contare le righe, parole e caratteri contenuti
- dato un direttorio e una soglia numerica in input contare il numero di file la cui dimensione è maggiore della soglia

scan.x

eXternal Data Represantation

```
const MAXLENFILE = 4096;****
const MAXLENDIR = 4096;****
struct Dir_input{ char dir[MAXLENDIR]; int soglia;};
struct File_input{ char file[MAXLENFILE];};
struct File_res{ int chars; int words; int lines;};
program SCANPROG {
   version SCANVERS {
   File_res FILE_SCAN(File_input) = 1;
   int DIR_SCAN(Dir_input) = 2;
   } = 1;
} = 0x20000020;
```

Server

File scan:

```
File_res *file_scan_1_svc(File_input * input, struct svc_req * rq){
   char separators[] = {' ', '\n', '.', ',','\0'};
   char buff[BUFFLEN];****
   static File_res result;
   result.chars = 0;result.words = 0;result.lines = 0;
   …
   if((fd = open(input->file,O_RDONLY)) == -1){
       result.chars = -1;
       result.lines = errno;
       return &result;
   }
```

Server

File scan:

```
while((nread = read(fd, buff, BUFFLEN)) > 0){
    for(int i = 0; i < nread; i++){
        char tmp = buff[i];
        result.words = (strchr(separators, tmp) != NULL)?(result.words + 1):result.words;
        result.chars++;
        result.lines = (buff[i] == '\n')?(result.lines + 1):result.lines;
    }
}
…
close(fd);
return &result;
```

Server

Dir scan:

```c
int * dir_scan_1_svc(Dir_input * input, struct svc_req * rq){
    static int result;
    result = 0;
    DIR *dir;
    struct dirent *entry;
    char path[MAXLENDIR+257];
    …
    if((dir = opendir(input->dir)) == NULL){
        result = -1;
        return (&result);
    }
```

Server

Dir scan:

```
while((entry = readdir(dir)) != NULL){
if(espressione){
       result = (lseek(fd,0,SEEK_END) > input->soglia) ? (result + 1) : result;
       close(fd);
       }
    }
return (&result);
```

```
espressione = (entry->d_type == DT_REG) &&
              sprintf(path,"%s/%s",input->dir,entry->d_name) &&
              (fd = open(path,O_RDONLY)) != -1;
```

Client

> client HostServer

```
CLIENT *c; File_input f_in;File_res *f_out;Dir_input d_in; int *d_out;
char tmp[LINE], pathfile[MAXLENFILE], pathdir[MAXLENDIR];
c = clnt_create(server, SCANPROG, SCANVERS, "udp");
if (c == NULL){
    clnt_pcreateerror(server);
    exit(1);
}
```

```
printf(...comunica all’utente che ti aspetti D o F...);
while(gets(tmp)){
printf(...comunica all’utente che ti aspetti un path);
…
}

clnt_destroy(c);
printf("See you...\n");
return 0;
```

Client

> Client HostServer

```
if(strcmp(tmp, "F") == 0){
   if(!(gets(pathfile)))
      break;
   strcpy(f_in.file, pathfile);
   f_out = file_scan_1(&f_in,c);
   if(f_out == NULL){
      clnt_perror(c, server); exit(1);
   }
   if(f_out->chars == -1){
      errno = f_out->lines;
      perror("errore in apertura file:\n");
      exit(1);
   }
   printf("chars: %d, words: %d,lines:%d\n", f_out->chars,f_out->words,f_out->lines);
}
```

Client

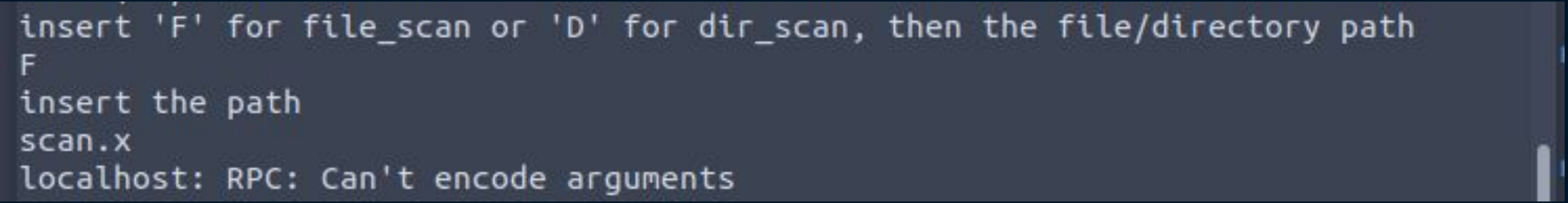
> Client HostServer

```
else if(strcmp(tmp, "D") == 0){
          if(!(gets(pathdir)))
              break;
          printf("insert an int\n");
          scanf("%d",&(d_in.soglia));
          getchar();
          strcpy(d_in.dir, pathdir);
          d_out = dir_scan_1(&d_in,c);
          if(d_out == NULL){
              clnt_perror(c, server); exit(1);
          }
          if((*d_out) == -1){
              printf("errore in ricerca directory\n");
              exit(1);
          }
          printf("%d files sopra soglia\n",(*d_out));
      }
```

Esecuzione

```
rpcgen scan.x;
gcc scan_client.c scan_clnt.c scan_xdr.c -g -o client;
gcc scan_svc.c scan_proc.c scan_xdr.c -g -o server;
```

```
server &
client localhost
```



```
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
F
insert the path
scan.x
localhost: RPC: Can't encode arguments
```

Esecuzione

Problema -> capienza UDP; numero di bytes massimo: 2190

```
#! /bin/bash
MIN=2000;
MAX=2500;
HOST="localhost";
for i in `seq $MIN $MAX`;
do
        rpcgen -DMAXLENFILE=$i -DMAXLENDIR=$i scan.x;
        gcc -D MAXLENFILE=$i scan_client.c scan_clnt.c scan_xdr.c -o client;
        gcc -D MAXLENDIR=$i scan_svc.c scan_proc.c scan_xdr.c -o server;
        echo "F\nscan.x\n\0" | ./client $HOST 2>&1 >/dev/null | awk  '($4 != "encode")
{exit 1}' && echo "$i" > max_byte_plus_one.txt && rm client s* && echo "numero di bytes
minimo trovato" &&  exit;
done;
```

SCAN.X diventa

```
const MAXLENFILE = 2190;
const MAXLENDIR = 2186;
```

Buffer

```
buffer: 2K,4K,8K,16K,32K
file->file di testo le 120 giornate di sodoma o la scuola del libertinaggio
88 MB, 880 MB, 1760 MB
misurazione tempo medio di risposta
```

```
2K
X,Y
88,0.751562
880,7.89383
1760,14.8877
```

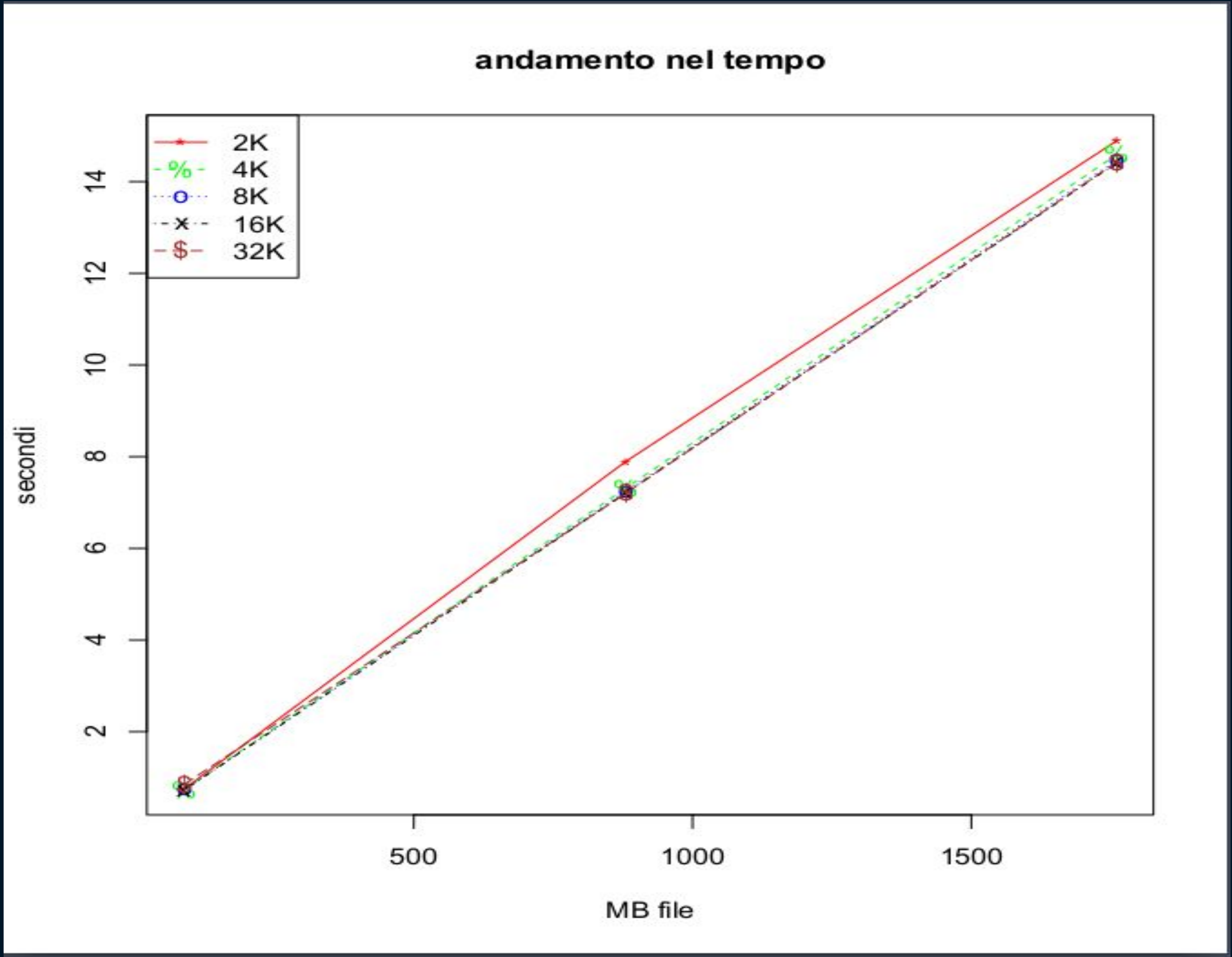
```
4K
X,Y
88,0.736136
880,7.31167
1760,14.5864
```

```
8K
X,Y
88,0.732112
880,7.24542
1760,14.4627
```

```
16K
X,Y
88,0.725377
880,7.21031
1760,14.4059
```

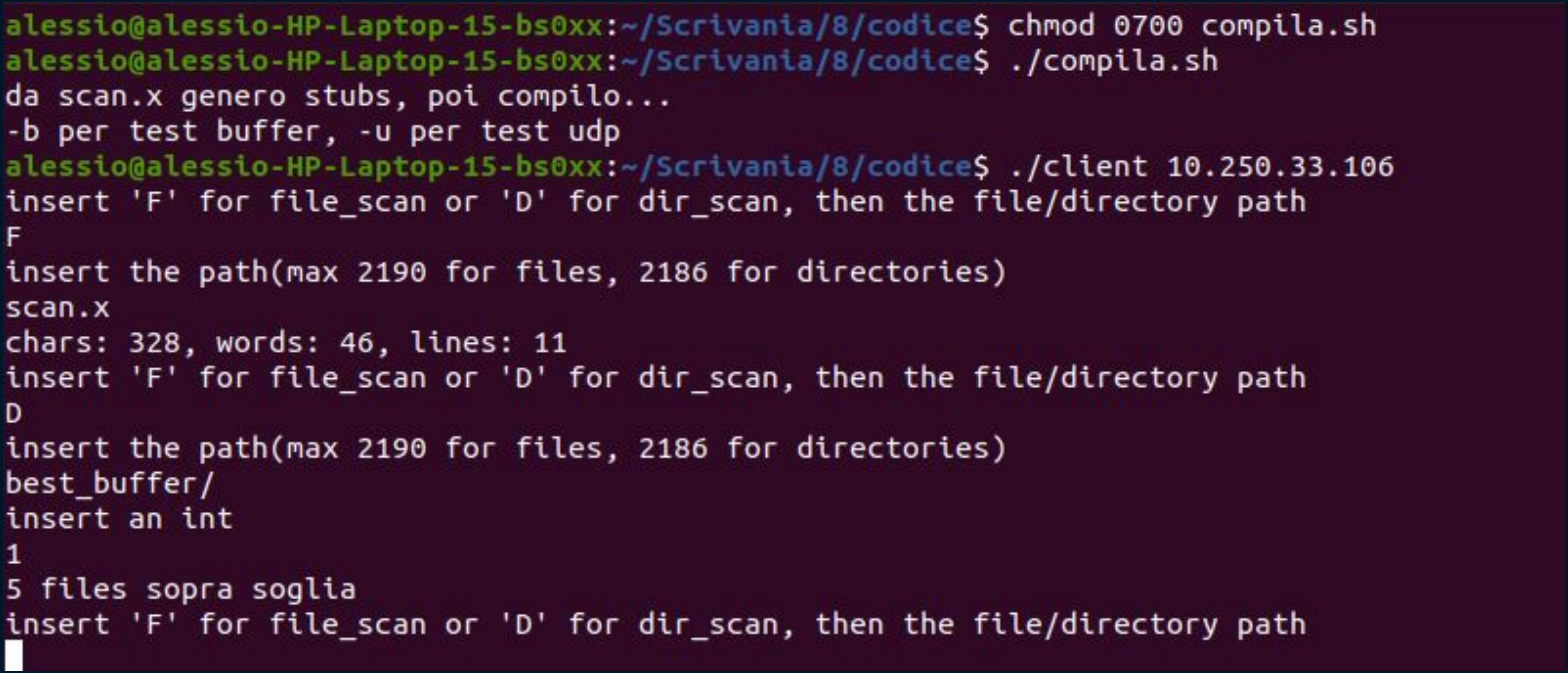
```
32K
X,Y
88,0.850716
880,7.20059
1760,14.4233
```

Buffer



Esercitazione

Proof of work (rete locale)



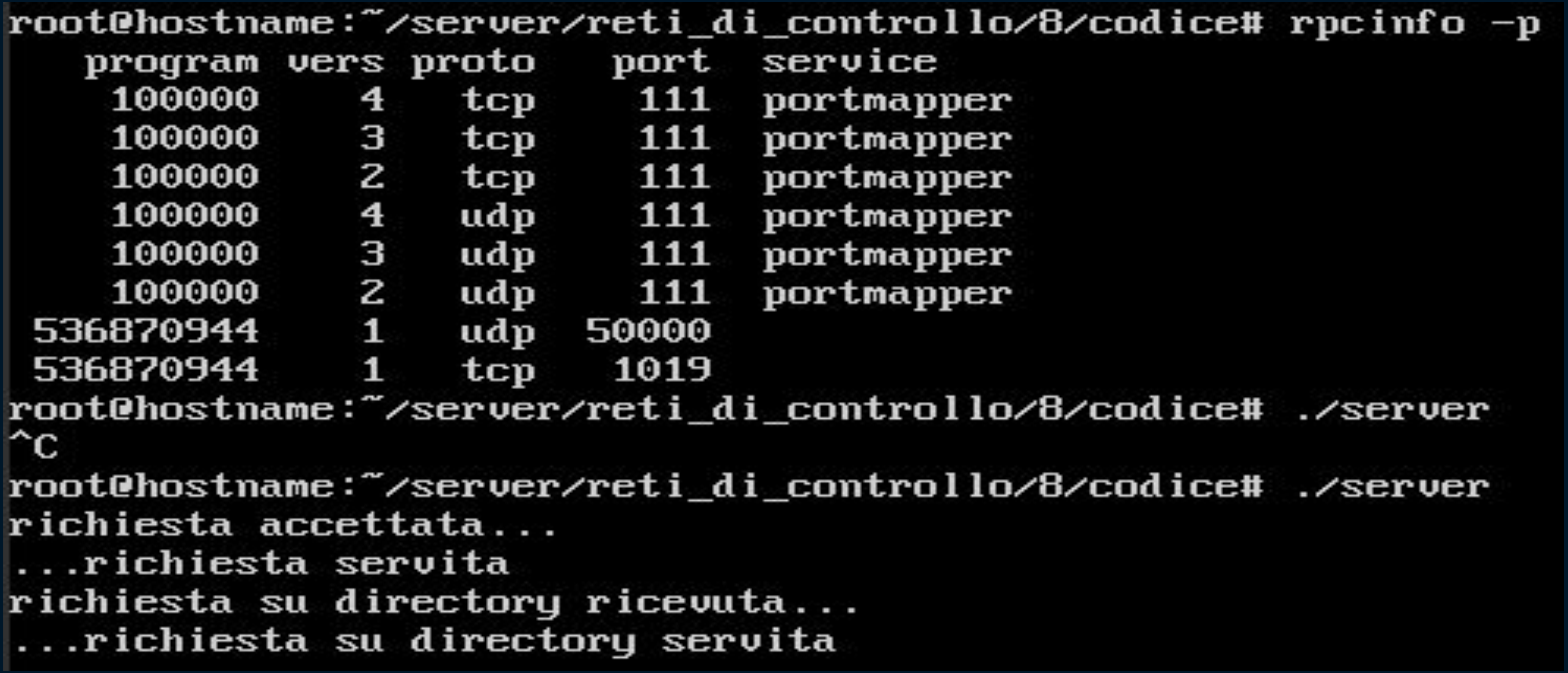
```
alessio@alessio-HP-Laptop-15-bs0xx:~/Scrivania/8/codice$ chmod 0700 compila.sh
alessio@alessio-HP-Laptop-15-bs0xx:~/Scrivania/8/codice$ ./compila.sh
da scan.x genero stubs, poi compilo...
-b per test buffer, -u per test udp
alessio@alessio-HP-Laptop-15-bs0xx:~/Scrivania/8/codice$ ./client 10.250.33.106
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
F
insert the path(max 2190 for files, 2186 for directories)
scan.x
chars: 328, words: 46, lines: 11
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
D
insert the path(max 2190 for files, 2186 for directories)
best_buffer/
insert an int
1
5 files sopra soglia
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
```

Proof of work (distribuito)

```
Modifica stub server per ascoltare alla porta 50000
apertura porte 111, 50000 nel security group del server
```

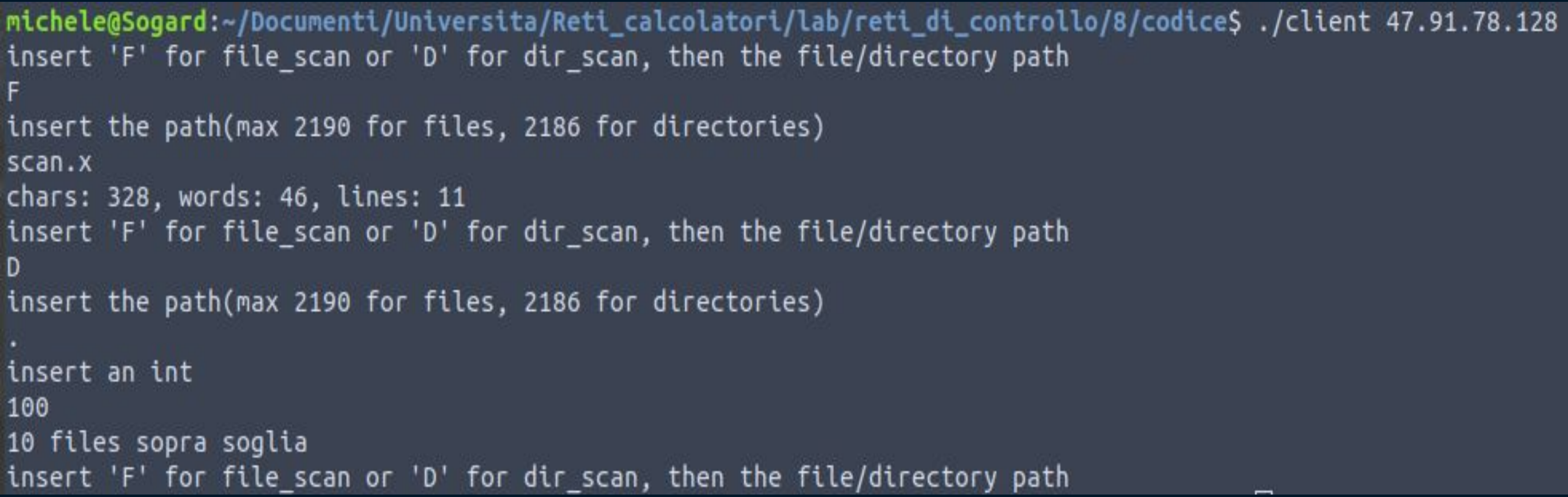
```
scan_svc.c

//creazione socket datagram
//creazione struttura dati servaddr con porta 50000
//bind socket
transp = svcudp_create(sockfd);
```

Proof of work (distribuito) Server



```
root@hostname:~/server/reti_di_controllo/8/codice# rpcinfo -p
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100000    3   tcp    111  portmapper
    100000    2   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100000    3   udp    111  portmapper
    100000    2   udp    111  portmapper
 536870944    1   udp  50000
 536870944    1   tcp   1019
root@hostname:~/server/reti_di_controllo/8/codice# ./server
^C
root@hostname:~/server/reti_di_controllo/8/codice# ./server
richiesta accettata...
...richiesta servita
richiesta su directory ricevuta...
...richiesta su directory servita
```

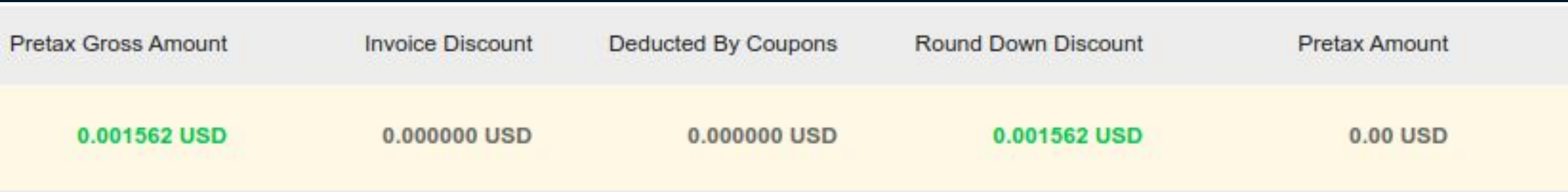
Proof of work (distribuito) Client



```
michele@Sogard:~/Documenti/Universita/Reti_calcolatori/lab/reti_di_controllo/8/codice$ ./client 47.91.78.128
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
F
insert the path(max 2190 for files, 2186 for directories)
scan.x
chars: 328, words: 46, lines: 11
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
D
insert the path(max 2190 for files, 2186 for directories)
.
insert an int
100
10 files sopra soglia
insert 'F' for file_scan or 'D' for dir_scan, then the file/directory path
```

Esercitazione

Per la scienza



Pretax Gross Amount	Invoice Discount	Deducted By Coupons	Round Down Discount	Pretax Amount
0.001562 USD	0.000000 USD	0.000000 USD	0.001562 USD	0.00 USD

Fine