Sprawozdanie nr 6 Programowanie Usług Sieciowych

Mateusz Snoch

Zadanie

Na laboratoriach należało napisać program sterowany przez daemona. Przerobiłem program z laboratoriów nr 3.

Kod programu:

Server_demon.c

```
1.  #include        "headers.h"
2.  #include        "demon.h"
3.  #include        <time.h>
4.  int main(int argc, char **argv)
5.  {
6.  daemon_init(argv[0], 0);
7.  int listenfd, connfd;
8.  pid_t childpid;
9.  struct sockaddr_in     servaddr;
10. char                   buff[MAXLINE];
11. time_t                 ticks;
12. listenfd = socket(AF_INET, SOCK_STREAM, 0);
13. bzero(&servaddr, sizeof(servaddr));
14. servaddr.sin_family      = AF_INET;
15. servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
16. servaddr.sin_port        = htons(4000);
17. bind(listenfd, (SA *) &servaddr, sizeof(servaddr));
18. listen(listenfd, LISTENQ);
19. signal(SIGCHLD,sig_chld);
20. for ( ; ; ) {
21. connfd = accept(listenfd, (SA *) NULL, NULL);
22. if ( (childpid = fork()) == 0)
23. {
24. close(listenfd);
25. data _data;
26. int n = read(connfd,&_data,sizeof(data));
27. if(n==sizeof(data))
28. {
29. printf("Nawiazano polaczenie\n");
30. int i;
31. for(i=0;i<9;i++){
32. _data.arr[i]*=1.5;
33. }
34. }
```

```
35. else printf("Nie udalo sie nawiazac polaczenia\n");
36. write(connfd, &_data, sizeof(data));
37. printf("Wyslano dane \n");
38. exit(0);
39. }
40. close(connfd);
41. };
42. }
```

W serwerze do kodu z laboratorium 3 została dodana funkcja daemon_init().

Client.c

```
1.  #include        "headers.h"
2.  int main(int argc, char **argv)
3.  {
4.  int                                     sockfd, n;
5.  struct sockaddr_in      servaddr;
6.  if (argc != 2)
7.  err_sys("Aby uruchomic : Adres IP");
8.  if ( (sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
9.  err_sys("Blad utworzenia polaczenia");
10. bzero(&servaddr, sizeof(servaddr));
11. servaddr.sin_family = AF_INET;
12. servaddr.sin_port   = htons(4000);
13. if (inet_pton(AF_INET, argv[1], &servaddr.sin_addr) <= 0)
14. err_sys("Blad konwersji do adresu IP dla %s", argv[1]);
15. if (connect(sockfd, (SA *) &servaddr, sizeof(servaddr)) < 0)
16. err_sys("Blad polaczenia");
17. FILE *d=fopen("p1.txt","r");
18. data _plik;
19. int max = 0;
20. while(!feof(d))
21. {
22. fscanf(d,"%f",&_plik.arr[max++]);
23. }
24. fclose(d);
25. write(sockfd,&_plik,sizeof(data));
26. data _plik2;
27. n = read(sockfd, &_plik2, sizeof(data));
28. if(n==sizeof(data))
29. {
30. d = fopen("p2.txt","w");
31. printf("Odebrane dane\n");
32. int i;
33. for(i=0;i<9;i++)
34. {
35. fprintf(d,"%f\t",_plik2.arr[i]);
36. if((i+1)%3==0) fprintf(d,"\n");
37. }
```

```
38. fclose(d);
39. }
40. exit(0);
41. }
```

Program klienta nie zmienił się w porównaniu do zajęć 3.

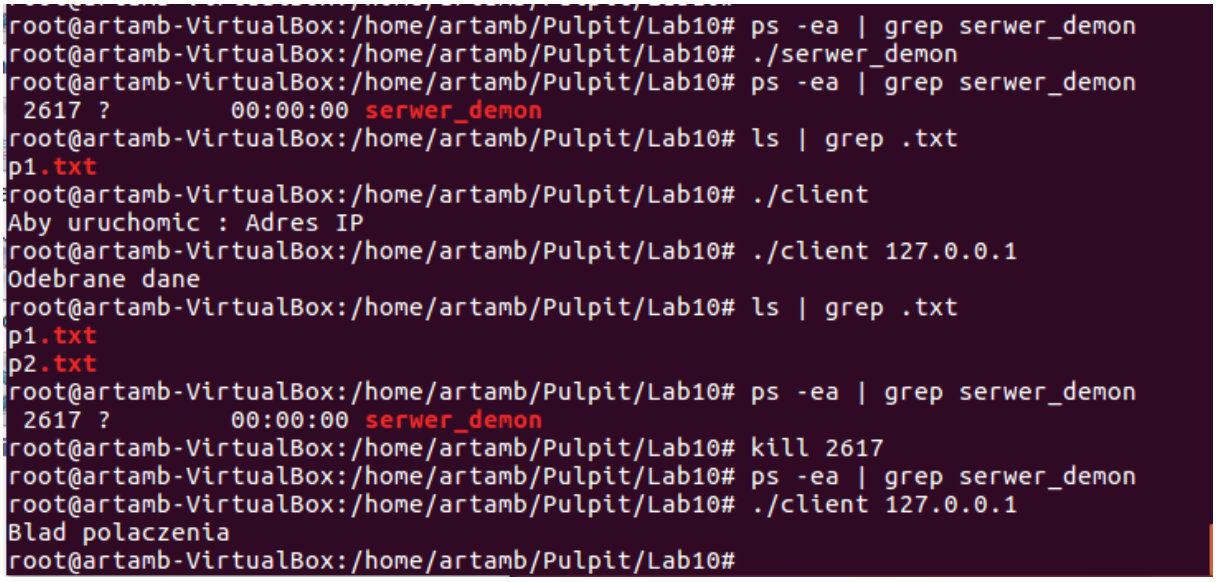
Headers.h

```
1.  #include    <netdb.h>
2.  #include    <signal.h>
3.  #include    <stdio.h>
4.  #include    <stdlib.h>
5.  #include    <string.h>
6.  #include    <sys/stat.h>
7.  #include    <sys/uio.h>
8.  #include    <unistd.h>
9.  #include    <sys/wait.h>
10. #include    <sys/un.h>
11. #include    <time.h>
12. #include    <netinet/in.h>
13. #include    <arpa/inet.h>
14. #include    <errno.h>
15. #include    <fcntl.h>
16. #define     LISTENQ         1024
17. #define     MAXLINE         4096
18. #define     MAXSOCKADDR     128
19. #define     BUFFSIZE    8192
20. #define     SA      struct sockaddr
21. #include    "funkcje.h"
22. void sig_chld(int signo)
23. {
24. pid_t pid;
25. int stat;
26. pid = wait (&stat);
27. while((pid=waitpid(-1,&stat,WNOHANG))>0)
28. printf("Potemek %d zakonczony\n", pid);
29. return;
30. }
31. typedef struct
32. {
33. float arr[9];
34. } data;
35. void err_sys(const char *, ...);
36. void err_sys(const char *msg, ...)
37. {
38. printf("%s\n",msg);
39. exit(1);
40. }
```

Demon.h:

```
1.  #include        <syslog.h>
2.  #define         MAXFD       64
3.  void daemon_init(const char *pname, int facility)
4.  {
5.  int             i;
6.  pid_t   pid;
7.  if ( (pid = fork()) != 0)
8.  exit(0);
9.  setsid();
10. signal(SIGHUP, SIG_IGN);
11. if ( (pid = fork()) != 0)
12. exit(0);
13. chdir("/");
14. umask(0);
15. for (i = 0; i < MAXFD; i++)
16. close(i);
17. openlog(pname, LOG_PID, facility);
18. }
```

Screeny z działania programu:



Demon działa poprawnie. Na screenie widać również, że jeśli użyjemy polecenia kill i zniszczymy demona to klient nie działa. Niestety nie udało mi się, aby demon działał na inetd.