Vasabi Note: ตัวอย่างของโปรแกรม TCP client และ server

โปรแกรม TCP client

กำหนดให้ client โปรแกรมชื่อว่า mclient.c ในส่วนแรกของโปรแกรมเป็น list ของ header ถัดจากนั้นเป็น prototypes ของฟังก์ชั่นที่เราสร้างขึ้นมาครอบ slow system calls เช่น connect() และ write() และ read() เพื่อทำให้ฟังก์ชั่นกลับมาทำงานของมันอีกครั้งในกรณีที่ถูกขัดจังหวะโดย signal interrupt และรีเทรินด้วยค่า errno = EINTR (มีรายละเอียดของฟังชั่งเหล่านี้ในโปรแกรม)

```
 1 #include <stdlib.h>
 2 #include <stdio.h>
 3
 4 #include <sys/types.h>
 5 #include <sys/socket.h>
 6 #include <netinet/in.h>
 7 #include <arpa/inet.h>
 8
 9 #include <string.h>
10 #include <errno.h>
11 #include <unistd.h>
12 #include <stdint.h>
13 #include <fcntl.h>
14
15 int connect_cr(int fd, const struct sockaddr *addr, socklen_t len);
16 int write_full(int fd, const void *buf, size_t count);
17 int read_full(int fd, void *buf, size_t count);
```

ในโปรแกรมนี้เราสร้าง client ให้สร้าง connection กับ TCP server ที่ bind อยู่ที่ port 7001 ของเครื่องที่มี IP 127.0.0.1 หรือเครื่อง localhost นั่นเอง ข้อมูลเกี่ยวกับ TCP server ที่ต้องการติดต่ออ้วยเหล่านี้ต้องได้รับการกำหนดใน sockaddr structure แล้วผ่านไปให้กับ kernel เป็น parameter หนึ่งของ connect system call

```
18
19 #define         SERV_PORT       7001
20 #define         SERV_IP_ADDR    "127.0.0.1"
21
22 #define         DATA_LENGTH     4096
23
24 // socket structure definitions
25 int client_fd;
26 struct sockaddr_in serv_addr;
27
28 main(int argc, char* argv[]){
29
30         int i, ret;
31         uint8_t *p;
32
33         client_fd = socket(AF_INET, SOCK_STREAM, 0);
34
35         serv_addr.sin_family = AF_INET;
36         serv_addr.sin_port = htons(SERV_PORT);
37         serv_addr.sin_addr.s_addr = inet_addr(SERV_IP_ADDR);
38
39         if((ret = connect(client_fd, (struct sockaddr *) &serv_addr, sizeof(serv_addr))) < 0){
40                 printf("Error occured errno = %d\n", errno);
41                 exit(1);
42         }
43
```

หลังจากนั้นเรา allocate memory กำหนดข้อมูลลงในนั้นและส่งออกไป โดยใช้ write_full (wrapper) ฟังก์ชั่น

```
44          p = malloc(sizeof(uint8_t) * DATA_LENGTH);
45
46          for(i = 0; i < 9; i++){
47                  memset(p, i, sizeof(uint8_t) * DATA_LENGTH); // fill p with i's value
48
49                  write_full(client_fd, p, (sizeof(uint8_t) * DATA_LENGTH));
50          }
51
52          close(client_fd);
53
54  }
```

ข้างล่างเป็น รายละเอียดของ wrapper สามอันที่ ครอบ slow system call ได้แก่ connect write และ read

```
56  int connect_cr(int fd, const struct sockaddr *addr, socklen_t len){
57          int ret;
58  repeat_connect:
59          ret = connect(fd, addr, len);
60          if (ret < 0) {
61              if (errno == EINTR){
62                  goto repeat_connect;
63              }
64              printf("connect error errno=%d fd=%d\n", errno, fd);
65          }
66          return ret;
67
68  }
69  // The write_full function below is taken from qemu's code.
70  int write_full(int fd, const void *buf, size_t count){
71      ssize_t ret = 0;
72      ssize_t total = 0;
73
74      while (count) {
75          ret = write(fd, buf, count);
76          if (ret < 0) {
77              if (errno == EINTR){
78                  continue;
79              }
80              printf("write error errno=%d fd=%d\n", errno, fd);
81              break;
82          }
83
84          count -= ret;
85          buf += ret;
86          total += ret;
87      }
88
89      return total;
90  }
```

การ compile program ใช้

$ gcc -o mcli mclient.c

```
91
92 int read_full(int fd, void *buf, size_t count){
93     ssize_t ret = 0;
94     ssize_t total = 0;
95
96     while (count) {
97         ret = read(fd, buf, count);
98         if (ret < 0) {
99             if (errno == EINTR){
100                 continue;
101             }
102             printf("read error errno=%d fd=%d\n", errno, fd);
103             break;
104         }
105
106         count -= ret;
107         buf += ret;
108         total += ret;
109     }
110
111     return total;
112 }
```

โปรแกรม TCP server

ในส่วนของ server เริ่มต้นด้วยชุดของ include ตามด้วย function prototypes ของ wrapper functions และ definitions โปรแกรม server ได้รับการกำหนดให้ใช้พอร์ต 7001 โปรแกรม bind ตัวเองบน port 7001 และรอรับการสร้าง connection ที่ accept system call หลังจาก accept return เราจะสื่อสารกับ client ผ่าน conn_fd file descriptor

```
 1 #include <stdlib.h>
 2 #include <stdio.h>
 3
 4 #include <string.h>
 5 #include <errno.h>
 6 #include <unistd.h>
 7 #include <stdint.h>
 8 #include <fcntl.h>
 9
10 #include <sys/types.h>
11 #include <sys/socket.h>
12 #include <netinet/in.h>
13 #include <arpa/inet.h>
14
15 int accept_cr(int fd, struct sockaddr *addr, socklen_t *len);
16 int write_full(int fd, const void *buf, size_t count);
17 int read_full(int fd, void *buf, size_t count);
```

```
19 #define         SERV_PORT       7001
20 #define         SERV_IP_ADDR    "localhost"
21 #define         BACKLOG         5
22
23 #define         DATA_LENGTH     4096
24
25 // socket structure definitions
26 int server_fd;
27 int conn_fd;
28 struct sockaddr_in serv_addr;
29
30 main(int argc, char* argv[]){
31
32         int i, ret;
33         uint8_t *p;
34
35         server_fd = socket(AF_INET, SOCK_STREAM, 0);
36
37         memset(&serv_addr, 0, sizeof(serv_addr));
38         serv_addr.sin_family = AF_INET;
39         serv_addr.sin_port = htons(SERV_PORT);
40         serv_addr.sin_addr.s_addr = INADDR_ANY;
41
42         bind(server_fd, (struct sockaddr *) &serv_addr, sizeof(serv_addr));
43
44         listen(server_fd, BACKLOG);
45
46         if((conn_fd = accept_cr(server_fd, NULL, NULL)) < 0){
47                 printf("Accept: Error occured\n");
48                 exit(1);
49         }
```

```
51         p = malloc(sizeof(uint8_t) * DATA_LENGTH);
52
53         for(i = 0; i < 9; i++){
54                 read_full(conn_fd, p, (sizeof(uint8_t) * DATA_LENGTH));
55
56                 printf("check out 0 %d 1 %d 100 %d\n", (uint8_t)*p, (uint8_t)*(p+1), (uint8_t)*(p+100));
57         }
58
59         close(conn_fd);
60         close(server_fd);
61
62 }
63
64 int accept_cr(int fd, struct sockaddr *addr, socklen_t *len){
65         int ret;
66 repeat_accept:
67         ret = accept(fd, addr, len);
68         if (ret < 0) {
69             if (errno == EINTR){
70                 goto repeat_accept;
71             }
72             printf("accept error errno=%d fd=%d\n", errno, fd);
73         }
74         return ret;
75 }
```

ข้อมูลที่ได้รับจะเป็นแบบ binary ซึ่ง client ใช้ memset system call กำหนดไว้

การ compile program ใช้

$ gcc -o mserv mserver.c

ตอน run ให้เปิดสอง window แล้วรัน

$ ./mserv

ก่อนบน window หนึ่งแล้วหลังจากนั้น ค่อยรัน

$ ./mcli บนอีก window หนึ่ง