TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

2. 네트워크 프로그래밍 – chat_server (글 도배 방지 1)

```c
#include "load_test.h"

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <signal.h>
#include <stdbool.h>
#include <pthread.h>
#include <arpa/inet.h>
#include <sys/socket.h>

#define BUF_SIZE        128
#define MAX_CLNT        256

typedef struct sockaddr_in      si;
typedef struct sockaddr *       sp;

int clnt_cnt = 0;
int clnt_socks[MAX_CLNT];
int cnt[MAX_CLNT];
pthread_mutex_t mtx;

// Black List
int black_cnt;
char black_list[MAX_CLNT][16];

// Information of Thread
typedef struct __iot{
        int sock;
        char ip[16];
        int cnt;
} iot;

iot info[BUF_SIZE];

void err_handler(char *msg)
{
        fputs(msg, stderr);
        fputc('\n', stderr);
        exit(1);
}
```

```c
void proc_msg(char *msg, int len, int sock)
{
        int i;

        pthread_mutex_lock(&mtx);

        for(i = 0; i < clnt_cnt; i++)
        {
                if(info[i].sock == sock)
                        continue;
                write(info[i].sock, msg, len);
        }

        pthread_mutex_unlock(&mtx);
}

void add_black_list(char *ip)
{
        pthread_mutex_lock(&mtx);
        strcpy(black_list[black_cnt++], ip);
        printf("black_list = %s\n", black_list[black_cnt - 1]);
        pthread_mutex_unlock(&mtx);
}

bool check_black_list(char *ip)
{
        int i;

        pthread_mutex_lock(&mtx);

        printf("Here\n");

        for(i = 0; i < black_cnt; i++)
        {
                if(!strcmp(black_list[i], ip))
                {
                        pthread_mutex_unlock(&mtx);
                        return true;
                }
        }

        pthread_mutex_unlock(&mtx);

        return false;
}
```

```c
void *clnt_handler(void *arg)
{
        iot thread_info = *((iot *)arg);
        int len = 0, i;
        char msg[BUF_SIZE] = {0};

        tv start, end;
        double runtime = 0.0;
        double load_ratio;

        for(;;)
        {
                gettimeofday(&start, NULL);
                //len = read(clnt_sock, msg, sizeof(msg));
                len = read(thread_info.sock, msg, sizeof(msg));
                proc_msg(msg, len, thread_info.sock);
                gettimeofday(&end, NULL);

                runtime = get_runtime(start, end);

                load_ratio = 1.0 / runtime;
                printf("load_ratio = %lf\n", load_ratio);

                if(load_ratio > 1.5)
                        thread_info.cnt++;

                if(thread_info.cnt > 10)
                {
                        write(thread_info.sock, "You're Fired!!!\n", 16);
                        add_black_list(thread_info.ip);
                        goto end;
                }
        }

#if 0
        while((str_len = read(clnt_sock, msg, sizeof(msg))) != 0)
                proc_msg(msg, str_len, i);
#endif

end:
        pthread_mutex_lock(&mtx);

        for(i = 0; i < clnt_cnt; i++)
        {
                if(thread_info.sock == info[i].sock)
```

```c
                {
                        while(i++ < clnt_cnt - 1)
                                info[i].sock = info[i + 1].sock;
                        break;
                }
        }

#if 0
        for(i = 0; i < clnt_cnt; i++)
        {
                if(clnt_sock == clnt_socks[i])
                {
                        while(i++ < clnt_cnt - 1)
                                clnt_socks[i] = clnt_socks[i + 1];
                        break;
                }
        }
#endif

        clnt_cnt--;
        pthread_mutex_unlock(&mtx);
        close(thread_info.sock);

        return NULL;
}
```

2. 네트워크 프로그래밍 – chat_server (글 도배 방지 2)

```
int main(int argc, char **argv)
{
        int serv_sock, clnt_sock;
        si serv_addr, clnt_addr;
        socklen_t addr_size;
        pthread_t t_id;
        int idx = 0;

        if(argc != 2)
        {
                printf("Usage: %s <port>\n", argv[0]);
                exit(1);
        }

        srand(time(NULL));

        pthread_mutex_init(&mtx, NULL);

        serv_sock = socket(PF_INET, SOCK_STREAM, 0);

        if(serv_sock == -1)
                err_handler("socket() error");

        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family = AF_INET;
        serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
        serv_addr.sin_port = htons(atoi(argv[1]));

        if(bind(serv_sock, (sp)&serv_addr, sizeof(serv_addr)) == -1)
                err_handler("bind() error");

        if(listen(serv_sock, MAX_CLNT) == -1)
                err_handler("listen() error");
```

```
        for(;;)
        {
                addr_size = sizeof(clnt_addr);
                clnt_sock = accept(serv_sock, (sp)&clnt_addr, &addr_size);

                printf("Check Black List\n");

                if(check_black_list(inet_ntoa(clnt_addr.sin_addr)))
                {
                        write(clnt_sock, "Get out of my server!!!\n", 23);
                        close(clnt_sock);
                        continue;
                }

                pthread_mutex_lock(&mtx);

                info[clnt_cnt].sock = clnt_sock;
                strcpy(info[clnt_cnt].ip, inet_ntoa(clnt_addr.sin_addr));
                info[clnt_cnt++].cnt = 0;

                pthread_mutex_unlock(&mtx);

                //pthread_create(&t_id, NULL, clnt_handler, (void *)&clnt_sock);
                pthread_create(&t_id, NULL, clnt_handler, (void *)&info[clnt_cnt - 1]);
                pthread_detach(t_id);
                printf("Connected Client IP: %s\n", inet_ntoa(clnt_addr.sin_addr));
        }

        close(serv_sock);

        return 0;
}
```

- gettimeofday를 이용하여 시작시간과 read가 끝난 시간을 파악하여 만들어둔 get_runtime함수를 이용해서 시작시간과 끝난 시간의 차이를 구한다.
- ratio의 비율이 정해둔 이상을 넘어갈 때마다 해당 client의 count를 올려 잦은 횟수로 ratio를 넘으면 해당 ip를 저장하고 close를 시킨다.
- black_list에 추가된 ip는 다시 접속하면 black_list_check함수에서 걸러져 접속을 금지 당한다.

2. 네트워크 프로그래밍 – chat_client (글 도배용 1)

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/epoll.h>

#define BUF_SIZE                128
#define NAME_SIZE               32

typedef struct sockaddr_in      si;
typedef struct sockaddr *       sp;

char name[NAME_SIZE] = "[내가이긴다]";
char msg[2048];

void err_handler(char *msg)
{
        fputs(msg, stderr);
        fputc('\n', stderr);
        exit(1);
}

void make_rand_str(char *tmp)
{
        int i, end = rand() % 7 + 3;

        for(i = 0; i < end; i++)
                tmp[i] = rand() % 26 + 65;
}
```

```c
void *send_msg(void *arg)
{
        int sock = *((int *)arg);
        char msg2[] = "https://kr.battle.net/heroes/ko/  <<== 지금 당장 접속하세요!!\n";
        srand(time(NULL));

        char tmp1[32] = {0};

        for(;;)
        {
#if PASSIVE
                fgets(msg, BUF_SIZE, stdin);

                write(sock, msg, strlen(msg));
#endif
#if ATTACK
                make_rand_str(tmp1);

                printf("%s\n", msg);
                sprintf(msg, "%s %s %s", name, tmp1, msg2);
                printf("tmp1 = %s\n", tmp1);
                write(sock, msg, strlen(msg));
                sleep(5);
#endif
        }

        return NULL;
}

void *recv_msg(void *arg)
{
        int sock = *((int *)arg);
        char msg[NAME_SIZE + 2048];
        int str_len;

        for(;;)
        {
                str_len = read(sock, msg, NAME_SIZE + 2047);

                msg[str_len] = 0;
                fputs(msg, stdout);
        }

        return NULL;
}
```

```c
int main(int argc, char **argv)
{
        int sock;
        si serv_addr;
        pthread_t snd_thread, rcv_thread;
        void *thread_ret;

        sock = socket(PF_INET, SOCK_STREAM, 0);

        if(sock == -1)
                err_handler("socket() error");

        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family = AF_INET;
        serv_addr.sin_addr.s_addr = inet_addr(argv[1]);
        serv_addr.sin_port = htons(atoi(argv[2]));

        if(connect(sock, (sp)&serv_addr, sizeof(serv_addr)) == -1)
                err_handler("connect() error");

        pthread_create(&snd_thread, NULL, send_msg, (void *)&sock);
        pthread_create(&rcv_thread, NULL, recv_msg, (void *)&sock);
        pthread_join(snd_thread, &thread_ret);
        pthread_join(rcv_thread, &thread_ret);

        close(sock);

        return 0;
}
```

2. 네트워크 프로그래밍 – chat_client (글 도배용 2)

- send_smg함수에서 make_rand_str함수를 이용하여 랜덤하게 3부터 7까지의 범위를 지정해서 알파벳을 바꾸고 계속 변화하는 메시지(msg)를 만든다.

- 마지막으로 sleep을 걸어 5초마다 일정하게 메시지를 write하는 Dos공격 형태를 만든 코드이다.