4주차_
멀티프로세스

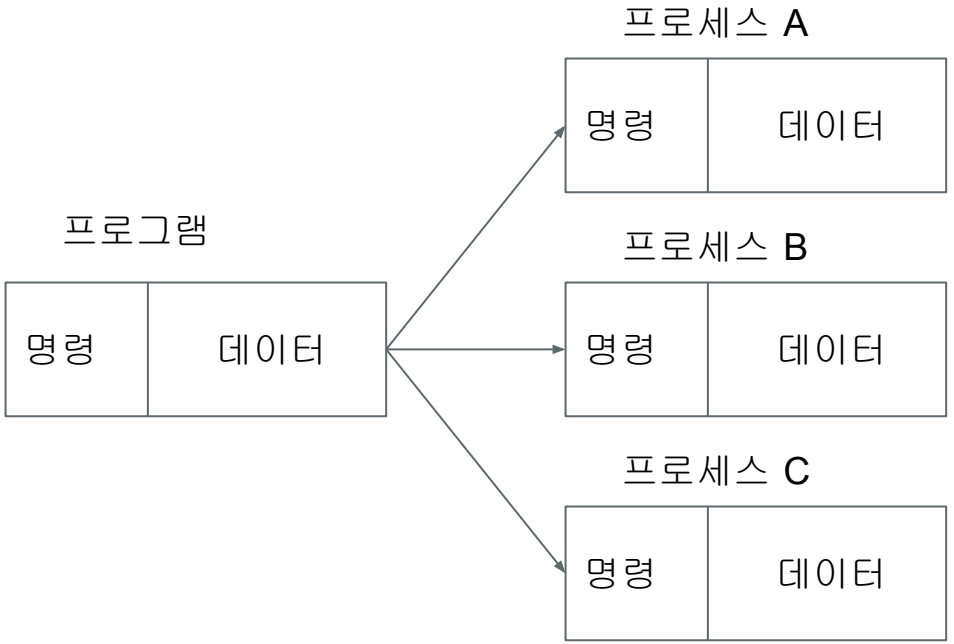
데이터 네트워크연구실
이현호
leeo75@cs-cnu.org

Goals

- 멀티프로세스 이해하기
- 네트워크에서 멀티프로세스

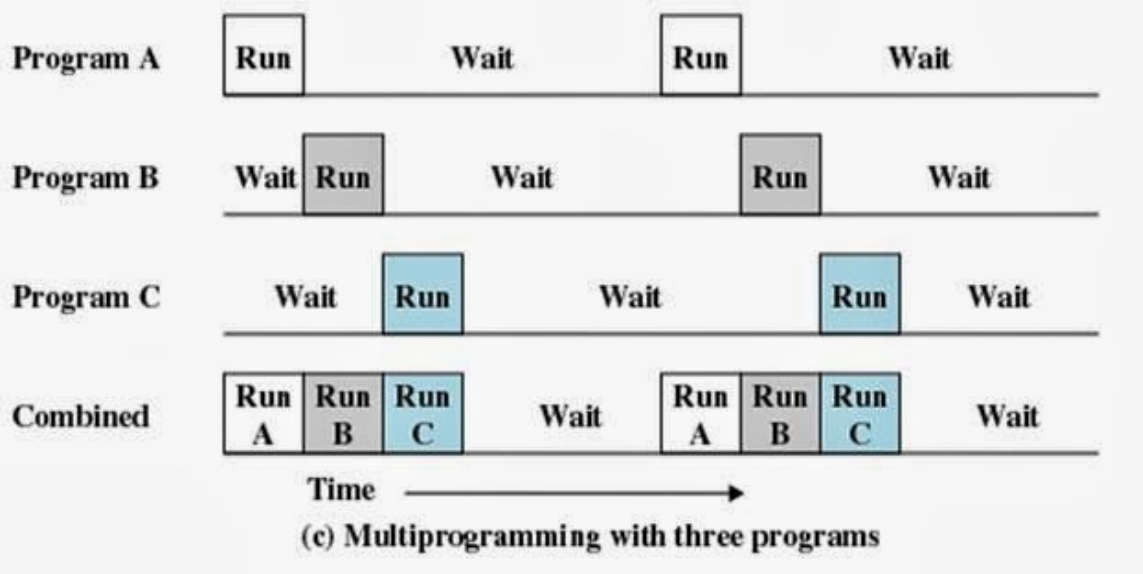
Process

- 프로그램 실행시 메모리에 올라가 프로세스가 동작
- 같은 프로세스 여러개 실행 가능
- 고유 번호가 존재



Process

- 멀티 프로세스의 필요성



(c) Multiprogramming with three programs

Process

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char **argv)
{
        int i = 0;
        while(1)
        {
                printf("%d\n", i);
                i++;
                sleep(1);
        }
}

```

- ps -aux | grep proc_test

```
hyunholee@DNLAB:~$ ps -aux | grep proc_test
hyunhol+ 32106  0.0  0.0   4356   652 pts/8    S+   17:32   0:00 ./proc_test
hyunhol+ 32159  0.0  0.0  12944   984 pts/9    S+   17:34   0:00 grep --color
hyunholee@DNLAB:~$ ps -aux | grep proc_test
hyunhol+ 32161  0.0  0.0  12944   968 pts/9    S+   17:34   0:00 grep --color
```

Test fork

```
 1 #include <unistd.h>
 2 #include <stdlib.h>
 3 #include <stdio.h>
 4
 5 int main(int argc, char **argv)
 6 {
 7         pid_t pid;
 8         int i = 100;
 9
10         printf("Program Start!!\n");
11
12         pid = fork();
13         printf("fork !!!\n");
14         if (pid < 0)
15         {
16                 printf("fork failure\n");
17                 return 1;
18         }
```

Test fork

```
20          if (pid == 0)
21          {
22                  printf("Im parent Process %d\n", getpid());
23                  while(1)
24                  {
25                          printf("P : %d\n", i);
26                          i++;
27                          sleep(1);
28                  }
29          }
30          else if (pid > 0)
31          {
32                  printf("Im Child Process %d\n", getpid());
33                  while(1)
34                  {
35                          printf("C : %d\n", i);
36                          i+=2;
37                          sleep(1);
38                  }
39          }
40  }
41
```

Test fork

- Program Start는 몇번 출력될까? 1번? 2번?
- fork는 몇번 출력될까? 1번? 2번?
- P와 C는 각각 어떻게 출력될까?

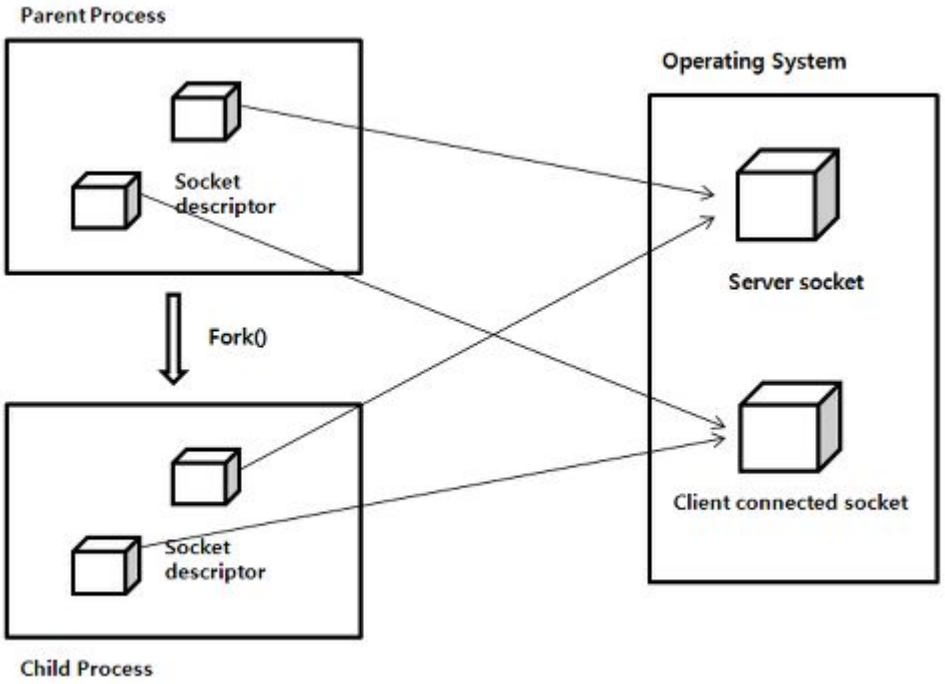
Result

```
hyunholee@DNLAB:~$ ps -aux | grep fork_test
hyunhol+ 32221  0.0  0.0   4356   684 pts/8    S+   17:42   0:00 ./fork_test
hyunhol+ 32222  0.0  0.0   4356    88 pts/8    S+   17:42   0:00 ./fork_test
```

```
hyunholee@DNLAB:~/temp/Multi_process/process$ ./fork_test
Program Start!!
fork !!!
Im Child Process 32226
C : 100
fork !!!
Im parent Process 32227
P : 100
C : 102
P : 101
C : 104
P : 102
C : 106
P : 103
```

MultiProcess

- 자식 프로세스 생성
- 부모와 다른 영역의 메모리를 사용
- 왜 필요한가?
  - 여러가지 일을 처리하기 위해서



Server

Multi process server

```
 1 #include <sys/types.h>
 2 #include <sys/stat.h>
 3 #include <sys/socket.h>
 4 #include <netinet/in.h>
 5 #include <arpa/inet.h>
 6 #include <stdio.h>
 7 #include <string.h>
 8 #include <unistd.h>
 9 #include <signal.h>
10
11 #define MAXLINE 1024
12 #define PORTNUM 6292
13
14 int main(int argc, char **argv)
15 {
16         int listen_fd, client_fd;
17         pid_t pid;
18         socklen_t addrlen;
19         int readn;
```

Multi process server

```
20          char buf[MAXLINE];
21          struct sockaddr_in client_addr, server_addr;
22
23          if( (listen_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
24          {
25                  return 1;
26          }
27          memset((void *)&server_addr, 0x00, sizeof(server_addr));
28          server_addr.sin_family = AF_INET;
29          server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
30          server_addr.sin_port = htons(PORTNUM);
31
32          if(bind(listen_fd, (struct sockaddr *)&server_addr, sizeof(server_addr)) ==-1)
33          {
34                  perror("bind error");
35                  return 1;
36          }
37          if(listen(listen_fd, 5) == -1)
38          {
```

Multi process server

```
39                perror("listen error");
40                return 1;
41        }
42
43        signal(SIGCHLD, SIG_IGN);
44        while(1)
45        {
46                addrlen = sizeof(client_addr);
47                client_fd = accept(listen_fd,
48                        (struct sockaddr *)&client_addr, &addrlen);
49                if(client_fd == -1)
50                {
51                        printf("accept error\n");
52                        break;
53                }
54                pid = fork();
55                if(pid == 0)
56                {
57                        memset(buf, 0x00, MAXLINE);
```

Multi process server

```
58                      while((readn = read(client_fd, buf, MAXLINE)) > 0)
59                      {
60                              printf("Read Data %s(%d) : %s",
61                                              inet_ntoa(client_addr.sin_addr),
62                                              client_addr.sin_port,
63                                               buf);
64                              write(client_fd, buf, strlen(buf));
65                              memset(buf, 0x00, MAXLINE);
66                      }
67                      close(client_fd);
68                      return 0;
69              }
70      }
71      return 0;
72 }
```

Result

```
hyunholee@DNLAB:~/temp/Multi_process$ ./echo_server_fork
Read Data 127.0.0.1(31979) : hi
Read Data 127.0.0.1(32491) : bye
Read Data 127.0.0.1(33003) : call me
Read Data 127.0.0.1(31979) : my name is leo
Read Data 127.0.0.1(32491) : welcome to network class

hyunholee@DNLAB:~/temp/Multi_process$ ps -ef | grep echo_server_fork
hyunhol+ 32486 32046  0 18:00 pts/8    00:00:00 ./echo_server_fork
hyunhol+ 32488 32486  0 18:00 pts/8    00:00:00 ./echo_server_fork
hyunhol+ 32493 32486  0 18:00 pts/8    00:00:00 ./echo_server_fork
hyunhol+ 32495 32486  0 18:00 pts/8    00:00:00 ./echo_server_fork
```