9장 프로세스 간 통신

- 프로세스간 통신
- 시그널
- 파이프
- 소켓

한빛미디어(주)

프로세스 간 통신

- Inter-process communication

- 프로세스들은 여러 가지 필요에 의해서 서로 데이터를 주고 받는다.
  - 주고받는 데이터의 크기가 다양하다.
  - 프로세스가 동일한 시스템에 있거나 서로 다른 시스템에 있다.
  - 통신의 주체인 양쪽 프로세스가 사용자 프로세스일 수 있고 시스템의 커널일 수도 있다.

- 프로세스가 수행할 통신의 유형에 따라 적절한 방법이 선택되어 데이터를 주고받는데 사용된다.
  - 시그널(signal), 파이프(pipe), 소켓(socket)

시그널 (signal)

- 특정 이벤트가 발생했을 때 프로세스에게 전달하는 신호
  - 연산 오류 발생, 자식 프로세스의 종료, 사용자의 종료 요청 등
  - 굉장히 작은 값이다.
  - 인터럽트(interrupt)라고 부르기도 한다.

- 시그널은 여러 종류가 있고 각각에 유일한 번호가 붙여져 있다.
  - 프로그램 내에서는 매크로 상수를 사용한다.
  - 예1) Ctrl + C를 누를 때 SIGTERM이 전달된다.
  - 예2) kill 명령을 사용하면 해당 프로세스에게 SIGTERM이 전달된다.

시그널을 수신한 프로세스의 반응

1. 시그널에 대해 기본적인 방법으로 대응한다. 대부분의 시그널에 대해서 프로세스는 종료하게 된다.
2. 시그널을 무시한다. 단, SIGKILL과 SIGSTOP은 무시될 수 없다.
3. 프로그래머가 지정한 함수를 호출한다.

시그널 핸들러(signal handler)

- 프로세스가 특정 시그널을 포착했을 때 수행해야 할 별도의 함수
- 프로세스는 시그널을 포착하면 현재 작업을 일시 중단하고 시그널 핸들러를 실행한다. 시그널 핸들러의 실행이 끝나면 중단된 작업을 재개한다

시그널 종류

- "/usr/include/asm/signal.h" 또는 교재 참고

실습 : 프로세스에게 시그널을 보내는 간단한 방법

- [Ctrl + C]나 [Ctrl + Z]를 눌러서 전면에서 실행 중인 프로세스에게 SIGINT나 SIGSTP 시그널을 보낸다.

- 두 신호에 대한 프로세스의 반응은 무엇인가?

파이프 (pipe)

- 리눅스 이전에 유닉스 환경에서부터 요긴하게 사용된 프로스스 간 통신법
- 파이프(|)의 앞에 놓인 프로세스가 표준 출력하는 내용을 파이프의 뒤에 놓인 프로세스가 표준 입력으로 받아들인다.

파이프는 파일이다.

- 파이프는 특수한 형태의 파일을 경유지로 하여 두 개의 프로세스가 데이터를 주고받는다.
- 보통 앞에 놓인 프로세스는 파이프에 대해 쓰기만 수행하고, 뒤에 놓인 프로세스는 파이프에 대해 읽기만 수행한다.
- 임시 파이프(anonymous pipe)와 네임드 파이프(named pipe, FIFO)로 나뉜다.

임시 파이프 (anonymous pipe)

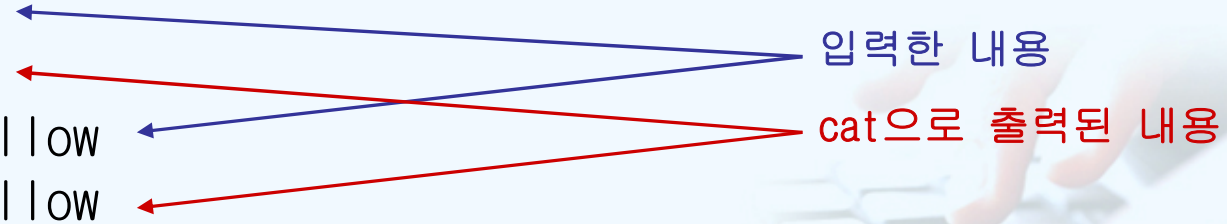
```
$ ls / | wc -w
      20
$ ls -l / | wc -l
      20
$
```

- 명령어 라인을 수행하기 위해 각각 한 개의 임시 파일이 만들어져서 파이프로 사용된다.
- 명령어 라인의 수행이 끝나면 파이프 역시 사라진다.
- 하나의 명령어 라인을 실행하기 위해 임시로 만들어지기 때문에 이름을 가지지 않고, 그 존재를 쉽게 확인할 수 없다.
- "ls" 프로세스는 파이프에 대해 쓰기 작업만 수행하고, "wc" 프로세스는 읽기 작업만 수행한다.

네임드 파이프 (named pipe, FIFO)

- "mkfifo" 명령을 사용하여 특수 파일인 네임드 파이프를 생성한다.
  - "ls" 명령으로 확인할 수 있다.
- 예제

```
$ mkfifo fifo
$ ls -l fifo
prw-r--r--    1 usp      student          0 Nov 19 03:22 fifo
$ cat < fifo &
[1] 9919
$ cat > fifo
apple is red
apple is red
banana is yellow
banana is yellow
[1]+  Done                    cat <fifo
$
```



실습 : 네임드 파이프로 채팅하기

- 두 개의 네임드 파이프를 생성하여 하나는 A → B 프로세스로 나머지 하나는 B → A 프로세스로 데이터가 흐르도록 한다.
- 파이프는 흐름당 하나씩 생성하고 하나의 프로세스는 쓰기만 수행하고 나머지 하나는 읽기만 수행하도록 한다.

```
$ ll
total 0
prw-r--r--    1 usp      student          0 Nov 19 03:47 fifo1
prw-r--r--    1 usp      student          0 Nov 19 03:47 fifo2
```

```
$ cat < fifo2 &
[1] 20143
$ cat > fifo1
hello                    →
hi~                      ←
nice to meet you~        →
me too~                  ←
```

```
$ cat < fifo1 &
[1] 20142
$ cat > fifo2
hello
hi~
nice to meet you~
me too~
```

네트워크 장치를 경유하는 통신 방법

- 서로 다른 시스템의 프로세스끼리 데이터를 주고 받을 수 있다.
- 시그널, 파이프는 동일 시스템 내에서만 가능하다.

연결형 모델과 비연결형 모델

연결형 모델	비연결형 모델
통신을 위해서 연결을 설정하는 작업이 필요하다.	연결을 설정하는 작업이 필요 없다.
통신이 끝난 뒤 연결을 해제하는 작업이 필요하다.	연결 설정이 없으므로 연결 해제 역시 필요 없다.
통신 수행 시 확실한 메시지 전달이 프로토콜 수준에서 보장된다.	메시지 전달이 확실하지 않다. 실패할 수도 있다.
TCP (transmission control protocol)	UDP (user datagram protocol)

인터넷 주소 (internet address)

- IP 주소라고도 한다.
- 네트워크 상에서 컴퓨터 하나를 유일하게 식별하기 위한 주소
- 4바이트 바이너리로 표현된다. (IP version4)
  - 사용자 입장에서는 편하게 “222.31.200.87”과 같은 문자열 형태로 표현한다.
  - 프로세스는 바이너리로 된 IP 주소를 사용한다. (변환이 필요하다.)
- IP (internet protocol)는 주소를 이용하여 컴퓨터를 식별하는 프로토콜이다.

포트 번호 (port)

- 인터넷 주소로 선택된 컴퓨터 내에서 실행 중인 통신 프로그램 중 하나를 선택하기 위한 번호
- 2바이트의 크기를 가진다.
- 하나의 시스템 내에서 중복될 수 없다. (사실 중복될 수 있으나 이는 논외로 한다.)
  - 사용하지 않는 번호 중에 하나를 가져와 사용한다.
- 잘 알려진 포트 (well-known port)
  - 잘 알려진 인터넷 서비스를 위해 미리 약속한 포트

서비스	포트 번호
daytime	13
ftp	21
telnet	23
http	80

소켓 (socket)

- 소켓은 파일 기술자와 같다.
- 프로세스가 다른 프로세스와 데이터를 주고 받기 위해 필요하다.
- 소켓과 파일 기술자가 다른 점은…
  - 파일 기술자 너머에는 파일이 있지만, 소켓 너머에는 자신처럼 소켓을 가진 프로세스가 있다.
- 프로세스는 소켓에 데이터를 쓰거나, 소켓에서 데이터를 읽어서 다른 프로세스와 통신한다.
- 소켓에는 상태 프로세스에 대한 인터넷 주소, 포트 번호가 등의 정보가 담겨져 있다.