



# 우분투 리눅스

시스템 & 네트워크

Chapter 06. 프로세스 관리하기

# 목차

00. 개요

01. 프로세스의 개념

02. 프로세스 관리 명령

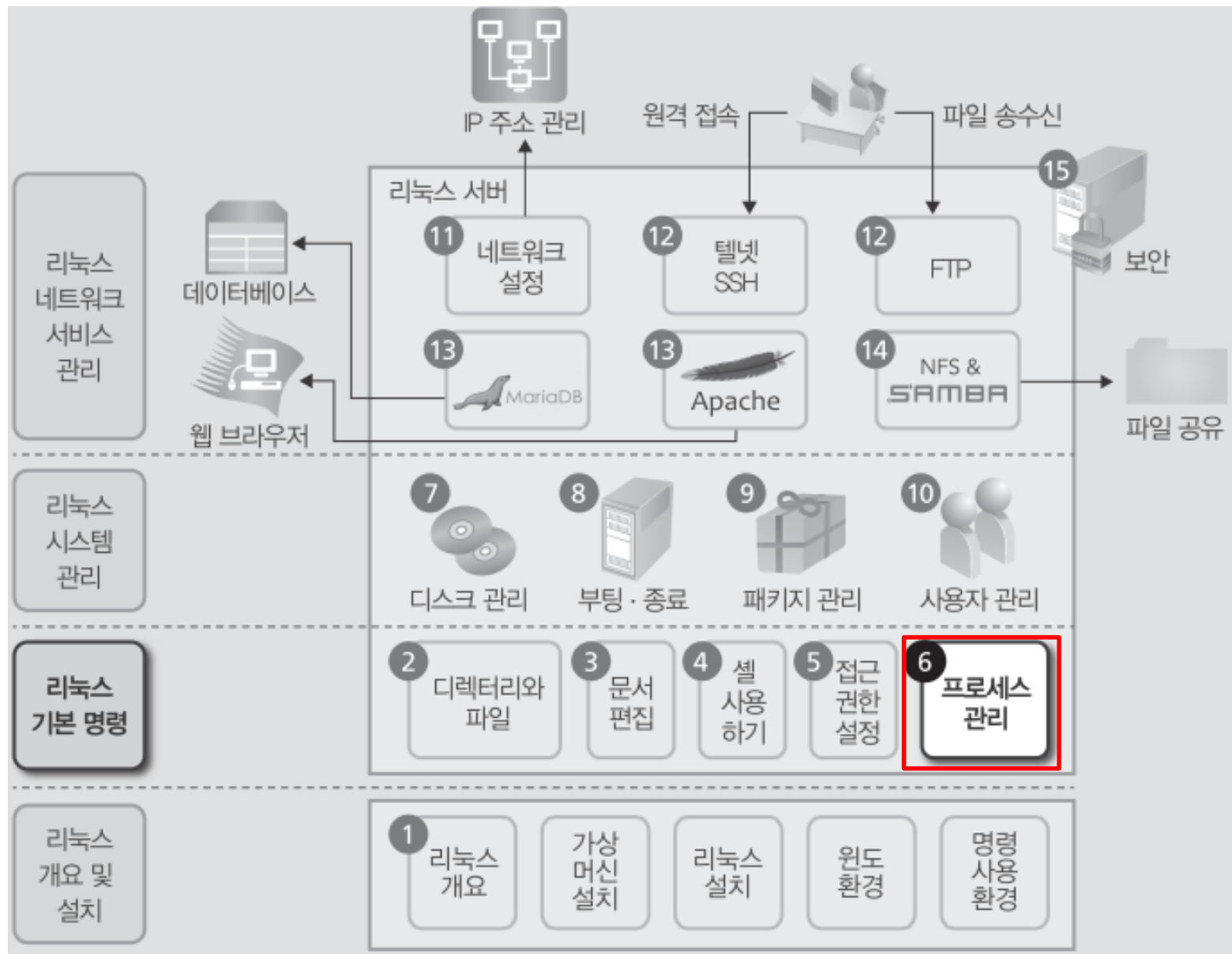
03. 포그라운드, 백그라운드 프로세스와 작업 제어

04. 작업 예약

# 학습목표

- 프로세스가 무엇인지 설명할 수 있다.
- 프로세스 목록을 확인하고 특정 프로세스를 검색할 수 있다.
- 프로세스를 강제로 종료할 수 있다.
- 프로세스 관리 도구로 전체 프로세스의 상태를 확인할 수 있다.
- 포그라운드와 백그라운드 작업의 차이를 설명할 수 있다.
- 백그라운드로 작업을 실행하고 포그라운드로 변환할 수 있다.
- 정해진 시간에 혹은 주기적으로 명령이 실행되도록 설정할 수 있다.

# 리눅스 실습 스터디 맵



# 00 개요



[그림 6-1] 6장의 내용 구성

# 01 프로세스의 개념

## ■ 프로세스: 현재 시스템에서 실행 중인 프로그램

## ■ 프로세스의 부모-자식 관계

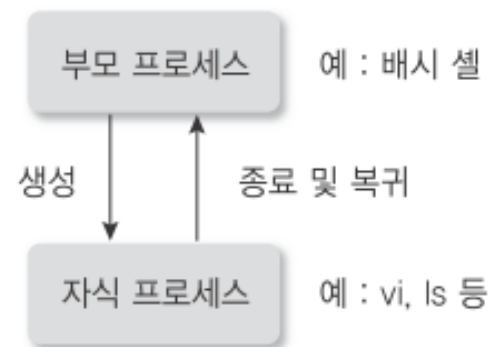
- 프로세스는 부모-자식 관계를 가지고 있음
- 필요에 따라 부모 프로세스(parent process)는 자식 프로세스(child process)를 생성하고, 자식 프로세스는 또 다른 자식 프로세스 생성 가능
- 부팅할 때 스케줄러가 실행한 프로세스인 systemd와 kthreadd 프로세스를 제외하면 모든 프로세스는 부모 프로세스를 가지고 있음
- 자식 프로세스는 할 일이 끝나면 부모 프로세스에 결과를 돌려주고 종료

## ■ 프로세스의 번호

- 각 프로세스는 고유한 번호를 가지고 있는데 이것이 PID

## ■ 프로세스의 종류

- 데몬 프로세스
  - 특정 서비스를 제공하기 위해 존재하며 리눅스 커널에 의해 실행
- 고아 프로세스
  - 자식 프로세스가 아직 실행 중인데 부모 프로세스가 먼저 종료된 자식 프로세스는 고아(orphan) 프로세스
  - 1번 프로세스가 고아 프로세스의 새로운 부모 프로세스가 되어 고아 프로세스의 작업 종료 지원
- 좀비 프로세스
  - 자식 프로세스가 실행을 종료했는데도 프로세스 테이블 목록에 남아 있는 경우
  - 좀비 프로세스는 프로세스 목록에 defunct 프로세스라고 나오기도함
  - 좀비 프로세스가 증가하면 프로세스 테이블의 용량이 부족해서 일반 프로세스가 실행되지 않을 수도 있음



[그림 6-2] 부모 프로세스와 자식 프로세스의 관계

## 02 프로세스 관리 명령

### ■ 프로세스 목록 보기

- 현재 실행 중인 프로세스의 목록을 보는 명령: ps
  - 유닉스(SVR4) 옵션 : 묶어서 사용할 수 있고, 붙임표로 시작한다(예 : -ef).
  - BSD 옵션 : 묶어서 사용할 수 있고, 붙임표로 시작하지 않는다(예 : aux).
  - GNU 옵션 : 붙임표 두 개로 시작한다(예 : --pid).

#### ps

**기능** 현재 실행 중인 프로세스에 대한 정보를 출력한다.

**형식** ps 옵션

**옵션**

〈유닉스 옵션〉	-e : 시스템에서 실행 중인 모든 프로세스의 정보를 출력한다. -f : 프로세스에 대한 자세한 정보를 출력한다. -u uid : 특정 사용자에게 대한 모든 프로세스의 정보를 출력한다. -p pid : pid로 지정한 특정 프로세스의 정보를 출력한다.
〈BSD 옵션〉	a : 터미널에서 실행한 프로세스의 정보를 출력한다. u : 프로세스 소유자의 이름, CPU 사용량, 메모리 사용량 등 상세 정보를 출력한다. x : 시스템에서 실행 중인 모든 프로세스의 정보를 출력한다.
〈GNU 옵션〉	--pid PID 목록 : 목록으로 지정한 특정 PID 정보를 출력한다.

**사용 예** ps                  ps -ef                  ps aux

## 02 프로세스 관리 명령

### ■ 현재 단말기의 프로세스 목록 출력하기 : ps

- ps 명령을 옵션 없이 사용하면 현재 셸이나 터미널에서 실행한 사용자 프로세스에 대한 정보를 출력

```
user1@myubuntu:~$ ps
  PID TTY          TIME CMD
 5501 pts/1        00:00:00 bash
 6162 pts/1        00:00:00 ps
user1@myubuntu:~$
```

### ■ 프로세스의 상세 정보 출력하기 : -f 옵션

- 프로세스의 상세한 정보를 출력: PPID와 터미널 번호, 시작 시간 등

```
user1@myubuntu:~$ ps -f
UID          PID  PPID  C STIME TTY          TIME CMD
user1        5501  5500  0 00:32 pts/1        00:00:00 -bash
user1        6163  5501  0 04:33 pts/1        00:00:00 ps -f
user1@myubuntu:~$
```

[표 6-1] ps -f의 출력 정보

항목	의미	항목	의미
UID	프로세스를 실행한 사용자 ID	STIME	프로세스의 시작 날짜나 시간
PID	프로세스 번호	TTY	프로세스가 실행된 터미널의 종류와 번호
PPID	부모 프로세스 번호	TIME	프로세스 실행 시간
C	CPU 사용량(% 값)	CMD	실행되고 있는 프로그램 이름(명령)



## 02 프로세스 관리 명령

### ■ 터미널에서 실행한 프로세스의 정보 출력하기 : a 옵션

- 터미널에서 실행한 프로세스의 정보를 출력

```
user1@myubuntu:~$ ps a
  PID  TTY      STAT  TIME COMMAND
  860  tty4      Ss+    0:00 /sbin/getty -8 38400 tty4
  864  tty5      Ss+    0:00 /sbin/getty -8 38400 tty5
  872  tty2      Ss+    0:00 /sbin/getty -8 38400 tty2
  876  tty3      Ss+    0:00 /sbin/getty -8 38400 tty3
  883  tty6      Ss+    0:00 /sbin/getty -8 38400 tty6
  988  tty7      Ss+    1:39 /usr/bin/X -core :0 -auth /var/run/lightdm/root/:0 -n
  993  tty1      Ss+    0:00 /sbin/getty -8 38400 tty1
(생략)
user1@myubuntu:~$
```

[표 6-2] STAT에 사용되는 문자의 의미

문자	의미	비고	문자	의미	비고
R	실행 중(running)		STIME	프로세스의 시작 날짜나 시간	
S	인터럽트가 가능한 대기(sleep) 상태		s	세션 리더 프로세스	BSD 형식
T	작업 제어에 의해 정지된(stopped) 상태		+	포그라운드 프로세스 그룹	
Z	좀비 프로세스(defunct)		l(소문자 L)	멀티 스레드	

## 02 프로세스 관리 명령

### ■ 터미널에서 실행한 프로세스의 상세 정보 출력하기 : a 옵션과 u 옵션

- a 옵션과 u 옵션을 함께 사용하면 터미널에서 실행한 프로세스의 상세 정보를 출력: CPU와 메모리 사용량 등

```
user1@myubuntu:~$ ps au
USER      PID %CPU %MEM    VSZ   RSS  TTY      STAT START   TIME COMMAND
root        860  0.0  0.0   4668    864  tty4      Ss+   2월22    0:00 /sbin/getty -
root        864  0.0  0.0   4668    860  tty5      Ss+   2월22    0:00 /sbin/getty -
root        872  0.0  0.0   4668    860  tty2      Ss+   2월22    0:00 /sbin/getty -
root        876  0.0  0.0   4668    864  tty3      Ss+   2월22    0:00 /sbin/getty -
(생략)
user1      5638  0.0  0.3   8304   3356  pts/7     Ss+   00:32    0:00 -bash
user1      6166  0.0  0.1   6460   1176  pts/1     R+    04:37    0:00 ps au
user1@myubuntu:~$
```

[표 6-3] ps au의 출력 정보

항목	의미	항목	의미
USER	사용자 계정 이름	VSZ	사용하고 있는 가상 메모리의 크기(KB)
%CPU	CPU 사용량을 퍼센트로 표시	RSS	사용하고 있는 물리적 메모리의 크기(KB)
%MEM	물리적 메모리 사용량을 퍼센트로 표시	START	프로세스 시작 시간

## 02 프로세스 관리 명령

### ■ 전체 프로세스 목록 출력하기(유닉스 옵션) : -e 옵션

- -e 옵션은 시스템에서 실행 중인 모든 프로세스를 출력
- TTY의 값이 ?인 것은 대부분 데몬으로 시스템이 실행한 프로세스

```
user1@myubuntu:~$ ps -e | more
  PID TTY          TIME CMD
    1 ?            00:00:02 init
    2 ?            00:00:00 kthreadd
    3 ?            00:00:01 ksoftirqd/0
    5 ?            00:00:00 kworker/0:0H
(생략)
   22 ?            00:00:00 devfreq_wq
   23 ?            00:00:07 kworker/0:1
   25 ?            00:00:00 khungtaskd
--More--
```

- -ef 옵션 사용: 전체 프로세스의 더 자세한 정보 출력

```
[user1@myubuntu:~$ ps -ef | more
UID          PID  PPID  C STIME TTY          TIME CMD
root           1      0  0  2월22 ?       00:00:02 /sbin/init
root           2      0  0  2월22 ?       00:00:00 [kthreadd]
root           3      2  0  2월22 ?       00:00:01 [ksoftirqd/0]
(생략)
root          25      2  0  2월22 ?       00:00:00 [khungtaskd]
--More--
```

## 02 프로세스 관리 명령

### ■ 전체 프로세스 목록 출력하기(BSD 옵션) : ax 옵션

- 시스템에서 실행 중인 모든 프로세스를 출력

```
user1@myubuntu:~$ ps ax | more
  PID TTY          STAT TIME  COMMAND
    1 ?           Ss      0:02  /sbin/init
    2 ?           S        0:00  [kthreadd]
    3 ?           S        0:01  [ksoftirqd/0]
(생략)
   23 ?           S        0:07  [kworker/0:1]
   25 ?           S        0:00  [khungtaskd]
--More--
```

- aux 옵션은 -ef처럼 시스템에서 실행 중인 모든 프로세스에 대한 자세한 정보를 출력

```
user1@myubuntu:~$ ps aux | more
USER          PID  %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1   0.0   0.2   4052   2320 ?        Ss     2월22    0:02 /sbin/init
root           2   0.0   0.0     0     0  ?        S      2월22    0:00 [kthreadd]
root           3   0.0   0.0     0     0  ?        S      2월22    0:01 [ksoftirqd/0]
(생략)
root          22   0.0   0.0     0     0  ?        S<     2월22    0:00 [devfreq_wq]
root          23   0.0   0.0     0     0  ?        S      2월22    0:07 [kworker/0:1]
--More--
```

## 02 프로세스 관리 명령

### ■ 특정 사용자의 프로세스 목록 출력하기 : -u 옵션

```
user1@myubuntu:~$ ps -u user1
  PID TTY          TIME CMD
 1646 ?            00:00:01 gnome-keyring-d
 1648 ?            00:00:01 init
 1717 ?            00:00:00 ssh-agent
 1727 ?            00:00:04 dbus-daemon
 1733 ?            00:00:00 upstart-event-b
(생략)
```

- 더 상세한 정보를 보고 싶으면 -f 옵션을 함께 사용

```
user1@myubuntu:~$ ps -fu user1
UID          PID  PPID  C  STIME TTY          TIME CMD
user1      1646     1   0   2월22 ?        00:00:01 /usr/bin/gnome-keyring-daemon --
user1      1648   1378   0   2월22 ?        00:00:01 init --user
user1      1717   1648   0   2월22 ?        00:00:00 ssh-agent
user1      1727   1648   0   2월22 ?        00:00:04 dbus-daemon --fork --session --a
user1      1733   1648   0   2월22 ?        00:00:00 upstart-event-bridge
(생략)
```

## 02 프로세스 관리 명령

### ■ 특정 프로세스 정보 출력하기 : -p 옵션

- -p 옵션과 함께 특정 PID를 지정하면 해당 프로세스의 정보를 출력

```
user1@myubuntu:~$ ps -fp 5501
UID          PID    PPID  C STIME TTY          TIME CMD
user1      5501    5500   0 00:32 pts/1    00:00:00 -bash
user1@myubuntu:~$
```

### ■ ps 명령을 이용해 특정 프로세스 정보 검색하기

- ps 명령과 grep 명령을 |로 연결하여 특정 프로세스에 대한 정보를 검색

```
user1@myubuntu:~$ ps -ef | grep bash
user1      2205    2197   0  2월22 pts/5    00:00:00 bash
user1      5501    5500   0 00:32 pts/1    00:00:00 -bash
user1      5638    5637   0 00:32 pts/7    00:00:00 -bash
user1      6185    5501   0 04:50 pts/1    00:00:00 grep --color=auto bash
user1@myubuntu:~$
```

## 02 프로세스 관리 명령

### ■ pgrep 명령을 이용해 특정 프로세스 정보 검색하기

#### pgrep

**기능**     지정한 패턴과 일치하는 프로세스에 대한 정보를 출력한다.

**형식**     pgrep [옵션] [패턴]

**옵션**     -x : 패턴과 정확히 일치하는 프로세스 정보를 출력한다.  
          -n : 패턴을 포함하고 있는 가장 최근의 프로세스 정보를 출력한다.  
          -u 사용자 이름 : 특정 사용자에게 대한 모든 프로세스를 출력한다.  
          -l : PID와 프로세스 이름을 출력한다.  
          -t term : 특정 단말기와 관련된 프로세스 정보를 출력한다.

**사용 예** pgrep bash

- bash 패턴을 지정하여 검색한 예

```
user1@myubuntu:~$ pgrep -x bash
2205
5501
5638
user1@myubuntu:~$
```

## 02 프로세스 관리 명령

### ■ pgrep 명령을 이용해 특정 프로세스 정보 검색하기

- pgrep의 경우 -l 옵션을 지정해도 단지 PID와 명령 이름만 출력

```
user1@myubuntu:~$ pgrep -l bash
2205 bash
5501 bash
5638 bash
user1@myubuntu:~$
```

- 더 자세한 정보를 검색하려면 pgrep 명령을 ps 명령과 연결하여 사용

```
user1@myubuntu:~$ ps -fp $(pgrep -x bash)
UID          PID  PPID  C STIME TTY          STAT      TIME CMD
user1        2205  2197   0  2월22 pts/5      Ss+        0:00 bash
user1        5501  5500   0 00:32 pts/1      Ss         0:00 -bash
user1        5638  5637   0 00:32 pts/7      Ss+        0:00 -bash
user1@myubuntu:~$
```

- -u 옵션으로 사용자명을 지정하여 검색

```
user1@myubuntu:~$ ps -fp $(pgrep -u user1 bash)
UID          PID  PPID  C STIME TTY          STAT      TIME CMD
user1        2205  2197   0  2월22 pts/5      Ss+        0:00 bash
user1        5501  5500   0 00:32 pts/1      Ss         0:00 -bash
user1        5638  5637   0 00:32 pts/7      Ss+        0:00 -bash
user1@myubuntu:~$
```



## 02 프로세스 관리 명령

### ■ kill 명령을 이용해 프로세스 종료하기

#### kill

**기능**     지정한 시그널을 프로세스에 보낸다.

**형식**     kill [시그널] PID...

**시그널**   -2 : 인터럽트 시그널을 보낸다(**Ctrl**+C).

          -9 : 프로세스를 강제로 종료한다.

          -15 : 프로세스가 관련된 파일을 정리하고 프로세스를 종료한다. 종료되지 않는 프로세스가 있을 수 있다.

**사용 예**   kill 1001               kill -15 1001           kill -9 1001

- kill 예: man을 실행시킨 프로세스를 찾아서 종료시키기

```
user1@myubuntu:~$ ps -fp $(pgrep -x man)
UID          PID  PPID  C STIME TTY          TIME CMD
user1        6193  5501  0 04:56 pts/1    00:00:00 man ps
user1@myubuntu:~$ kill 6193
user1@myubuntu:~$
```

## 02 프로세스 관리 명령

### ■ 프로세스 강제로 종료하기

- 단순히 kill 명령으로는 종료되지 않는 경우 강제 종료 시그널인 9번을 보낸다.
- 강제종료 예: kill 명령으로 종료되지 않음

```
user1@myubuntu:~$ ps -fp $(pgrep -x sh)
UID          PID    PPID    C  STIME TTY          STAT    TIME CMD
user1        2087   1974    0   2월22 ?        Ss      0:00 /bin/sh -c /usr/bin/gtk-windo
user1        6230   5501    0   04:59 pts/1      S+      0:00 sh
user1@myubuntu:~$ kill 6230
user1@myubuntu:~$ ps -fp $(pgrep -x sh)
UID          PID    PPID    C  STIME TTY          STAT    TIME CMD
user1        2087   1974    0   2월22 ?        Ss      0:00 /bin/sh -c /usr/bin/gtk-windo
user1        6230   5501    0   04:59 pts/1      S+      0:00 sh
user1@myubuntu:~$
```

- 강제 종료 시그널인 9번을 보내 강제로 종료

```
user1@myubuntu:~$ kill -9 6230
user1@myubuntu:~$
```

## 02 프로세스 관리 명령

### ■ pkill 명령을 이용해 프로세스 종료하기

- PID가 아니라 프로세스의 명령 이름(CMD)으로 프로세스를 찾아 종료

```
user1@myubuntu:~$ ps -fp $(pgrep -x man)
UID          PID  PPID  C  STIME TTY          STAT TIME  CMD
user1        6396  5501  0  05:07 pts/1    S+   0:00  man pkill
user1        6412  5638  0  05:07 pts/7    S+   0:00  man pkill
user1@myubuntu:~$ pkill -x man
user1@myubuntu:~$ pgrep -x man
user1@myubuntu:~$
```

## 02 프로세스 관리 명령

### ■ 프로세스 관리 도구

- top 명령: 현재 실행 중인 프로세스에 대한 정보를 주기적으로 출력

[표 6-4] top의 출력 정보

항목	의미	항목	의미
PID	프로세스 ID	SHR	프로세스가 사용하는 공유 메모리 크기
USER	사용자 계정	%CPU	CPU 사용량
PR	우선순위	%MEM	메모리 사용량(%)
NI	Nice 값	TIME+	CPU 누적 이용 시간
VIRT	프로세스가 사용하는 가상 메모리 크기	COMMAND	명령 이름
RES	프로세스가 사용하는 메모리 크기		

[표 6-5] top 내부명령

항목	의미	항목	의미
<span>Enter</span> , <span>Space Bar</span>	화면을 즉시 다시 출력한다.	u	사용자에 따라 정렬하여 출력한다.
h, ?	도움말 화면을 출력한다.	M	사용하는 메모리 크기에 따라 정렬하여 출력한다.
k	프로세스를 종료한다. 종료할 킬 프로세스의 PID를 물어본다.	p	CPU 사용량에 따라 정렬하여 출력한다.
n	출력한 프로세스의 개수를 바꾼다.	q	top 명령을 종료한다.

## 02 프로세스 관리 명령

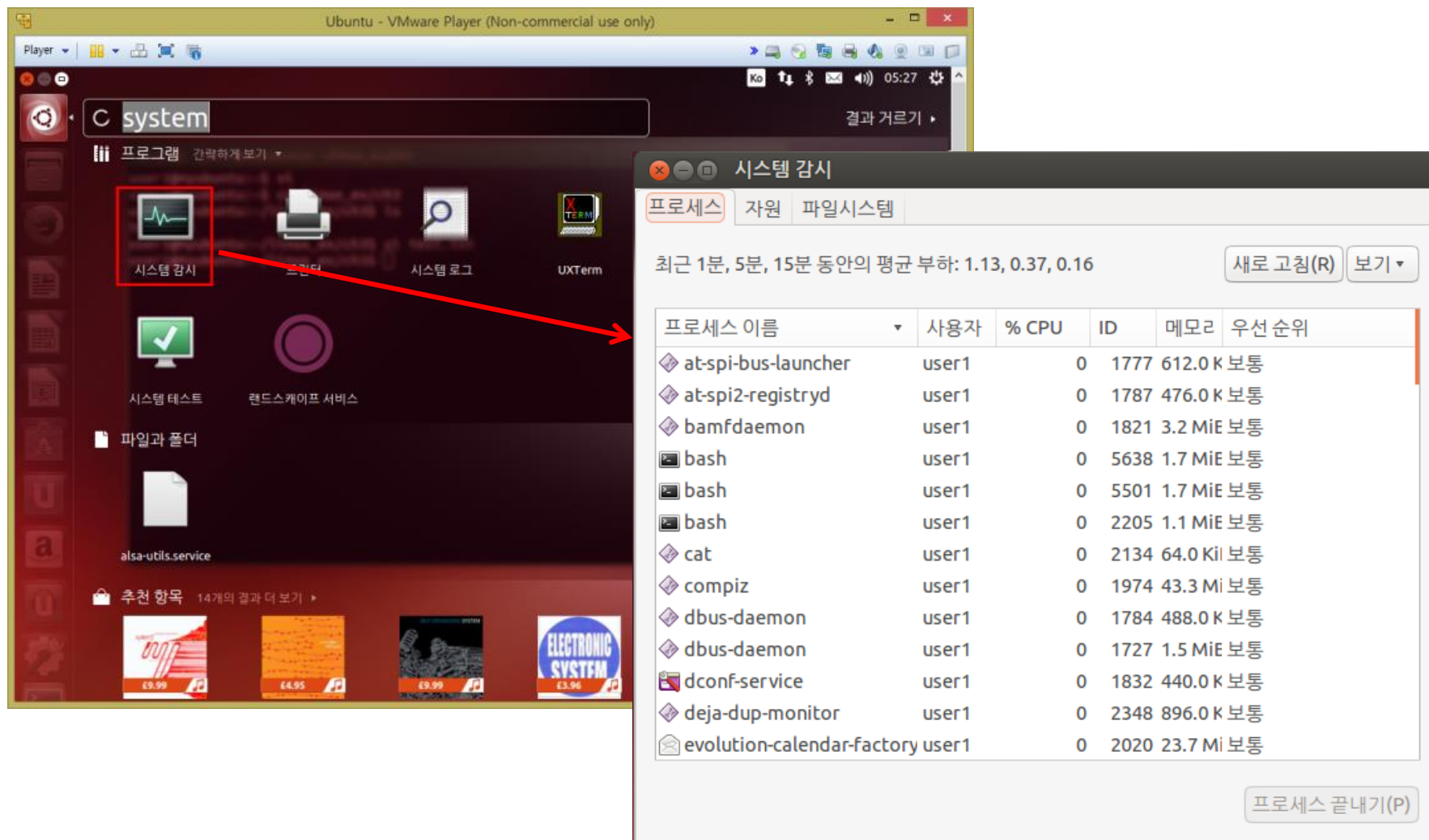
### ■ top 실행 화면

```
user1@myubuntu:~$ top
top - 05:20:23 up 1 day, 18:50, 5 users, load average: 0.03, 0.04, 0.05
Tasks: 196 total, 1 running, 195 sleeping, 0 stopped, 0 zombie
%Cpu(s):  0.1 us,  0.1 sy,  0.0 ni, 99.6 id,  0.2 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:  1025844 total,  828016 used, 197828 free, 122420 buffers
KiB Swap: 1046524 total,    80 used,1046444 free, 360268 cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6148	user1	20	0	133m	26m	20m	S	3.3	2.6	2:04.31	gnome-system-mo
5500	user1	20	0	10740	1816	1052	S	0.3	0.2	0:01.08	sshd
1	root	20	0	4052	2320	1340	S	0.0	0.2	0:02.74	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.01	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:01.54	ksoftirqd/0
5	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H
7	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
9	root	20	0	0	0	0	S	0.0	0.0	0:06.94	rcu_sched
10	root	rt	0	0	0	0	S	0.0	0.0	0:10.25	watchdog/0
11	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	khelper
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
13	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	netns
14	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	writeback
15	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	kintegrityd
16	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	bioset
17	root	0 -20		0	0	0	S	0.0	0.0	0:00.00	kworker/u17:0

## 02 프로세스 관리 명령

### ■ 시스템 정보: GNOME



Ubuntu - VMware Player (Non-commercial use only)

Player

system

결과 거르기

프로그램 간략하게 보기

시스템 감시

시스템 테스트

런드스케이프 서비스

파일과 폴더

alsa-utils.service

추천 항목 14개의 결과 더 보기

시스템 감시

프로세스 자원 파일시스템

최근 1분, 5분, 15분 동안의 평균 부하: 1.13, 0.37, 0.16

새로 고침(R) 보기

프로세스 이름	사용자	% CPU	ID	메모리	우선 순위
at-spi-bus-launcher	user1	0	1777	612.0 K	보통
at-spi2-registryd	user1	0	1787	476.0 K	보통
bamfdaemon	user1	0	1821	3.2 MiE	보통
bash	user1	0	5638	1.7 MiE	보통
bash	user1	0	5501	1.7 MiE	보통
bash	user1	0	2205	1.1 MiE	보통
cat	user1	0	2134	64.0 Ki	보통
compiz	user1	0	1974	43.3 Mi	보통
dbus-daemon	user1	0	1784	488.0 K	보통
dbus-daemon	user1	0	1727	1.5 MiE	보통
dconf-service	user1	0	1832	440.0 K	보통
deja-dup-monitor	user1	0	2348	896.0 K	보통
evolution-calendar-factory	user1	0	2020	23.7 Mi	보통

프로세스 끝내기(P)

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 포그라운드 작업

- 포그라운드 프로세스: 사용자가 입력한 명령이 실행되어 결과가 출력될 때까지 기다려야 하는 포그라운드 방식으로 처리되는 프로세스
- 이를 작업 제어에서는 포그라운드 작업이라고 함

```
user1@myubuntu:~$ sleep 100
```

포그라운드 작업  
sleep 명령이 끝날 때까지 기다려야 한다.

### ■ 백그라운드 작업

- 백그라운드 프로세스: 명령을 실행하면 명령의 처리가 끝나는 것과 관계없이 곧바로 프롬프트가 출력되어 사용자가 다른 작업을 계속할 수 있음
- 작업 제어에서는 백그라운드 작업이라고 함

```
user1@myubuntu:~$ sleep 100&
```

백그라운드 작업

```
[1] 6852
```

```
user1@myubuntu:~$
```

프롬프트가 바로 나와 다른 명령을 실행할 수 있다.

### ■ 백그라운드 작업과 출력 방향 전환하기

- 백그라운드로 처리할 때는 주로 출력과 오류 방향 전환을 하여 실행 결과와 오류 메시지를 파일로 저장

```
user1@myubuntu:~$ find / -name passwd > pw.dat 2>&1 &
```

pw.dat에 결과와 오류를 저장한다.

```
[2] 6853
```

```
user1@myubuntu:~$
```

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 작업 제어

- 작업 제어는 작업 전환과 작업 일시 중지, 작업 종료를 의미
- 작업 전환: 포그라운드 작업->백그라운드 작업, 백그라운드 작업->포그라운드 작업으로 전환
- 작업 일시 중지: 작업을 잠시 중단
- 작업 종료: 프로세스를 종료하는 것처럼 작업을 종료

### ■ 작업 목록 보기 : jobs

#### jobs

기능	백그라운드 작업을 모두 보여준다. 특정 작업 번호를 지정하면 해당 작업의 정보만 보여준다.	
형식	jobs [ %작업 번호 ]	
%작업 번호	%번호 : 해당 번호의 작업 정보를 출력한다.	
	%+ 또는 %* : 작업 순서가 +인 작업 정보를 출력한다.	
	% - : 작업 순서가 -인 작업 정보를 출력한다.	
사용 예	jobs %1	jobs



## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ jobs 명령 예

```
user1@myubuntu:~$ jobs
[1]-  실행 중                sleep 100 &
[2]+  실행 중                find / -name passwd > pw.dat 2>&1 &
user1@myubuntu:~$
```

[표 6-6] jobs 명령의 출력 항목

항목	출력 예	의미
작업 번호	[1]	작업 번호로서 백그라운드로 실행할 때마다 순차적으로 증가한다([1], [2], [3], ...).
작업 순서	+	작업 순서를 표시한다. <ul style="list-style-type: none"><li>• +: 가장 최근에 접근한 작업</li><li>• -: + 작업보다 바로 전에 접근한 작업</li><li>• 공백: 그 외의 작업</li></ul>
상태	실행 중	작업의 상태를 표시한다. <ul style="list-style-type: none"><li>• 실행 중(Running): 현재 실행 중이다.</li><li>• 완료됨(Done): 작업이 정상적으로 종료된다.</li><li>• 종료됨(Terminated): 작업이 비정상적으로 종료된다.</li><li>• 정지(Stopped): 작업이 잠시 중단된다.</li></ul>
명령	sleep 100 &	백그라운드로 실행 중인 명령

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 작업 전환하기

[표 6-7] 작업 전환 명령

명령	의미
<code>[Ctrl]+z</code> 또는 <code>stop [%작업 번호]</code>	포그라운드 작업을 중지한다(종료하는 것이 아니라 잠시 중단하는 것이다).
<code>bg [%작업 번호]</code>	작업 번호가 지시하는 작업을 백그라운드 작업으로 전환한다.
<code>fg [%작업 번호]</code>	작업 번호가 지시하는 작업을 포그라운드 작업으로 전환한다.

- 작업전환 예: 포그라운드 -> 백그라운드

```
user1@myubuntu:~$ jobs
user1@myubuntu:~$ sleep 100
^Z
[1]+  정지됨      sleep 100
user1@myubuntu:~$ bg %1
[1]+  sleep 100 &
user1@myubuntu:~$ jobs
[1]+  실행중      sleep 100 &
user1@myubuntu:~$
```

백그라운드 작업이 없다.  
포그라운드로 실행한다.  
Ctrl+z로 일시 중지한다.  
일시 중지된 상태이다.  
백그라운드로 전환한다.  
백그라운드로 실행 중이다.

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 작업 전환하기

- 작업전환 예: 백그라운드 -> 포그라운드

```
user1@myubuntu:~$ jobs
[1]+  실행중                sleep 100 &
user1@myubuntu:~$ fg          포그라운드로 전환한다.
sleep 100   포그라운드로 실행 중이다.
```

### ■ 작업 종료하기 : Ctrl+c

- 포그라운드 작업은 Ctrl+c를 입력하면 대부분 종료

```
user1@myubuntu:~$ sleep 100          포그라운드로 실행 중이다.
^C                                    강제 종료한다.
user1@myubuntu:~$
```

- 백그라운드 작업은 kill 명령으로 강제 종료: PID 또는 '%작업 번호'

```
user1@myubuntu:~$ sleep 100&   백그라운드로 실행 중이다.
[1] 6403
user1@myubuntu:~$ kill %1      강제 종료한다.
user1@myubuntu:~$
[1]+  종료됨      sleep 100      Enter를 한 번 더 입력해야 메시지가 출력된다.
user1@myubuntu:~$
```

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 로그아웃 후에도 백그라운드 작업 계속 실행하기 : nohup

- 로그아웃한 다음에도 작업이 완료될 때까지 백그라운드 작업을 실행해야 할 경우가 있다. 이때 nohup 명령을 사용

#### nohup

**기능** 로그아웃한 뒤에도 백그라운드 작업을 계속 실행한다.

**형식** nohup 명령&

- nohup 명령 사용 예

```
[user1@myubuntu:~$ nohup find / -name passwd &  
[1] 6867  
user1@myubuntu:~$ nohup: 입력 무시 및 'nohup.out' 에 출력 추가
```

- 다시 로그인하여 파일 내용 확인

```
user1@myubuntu:~$ more nohup.out  
/usr/share/bash-completion/completions/passwd  
/usr/share/lintian/overrides/passwd  
/usr/share/doc/passwd  
/usr/bin/passwd  
find: '/root': 허가 거부  
find: '/proc/tty/driver': 허가 거부  
(생략)
```

## 03 포그라운드, 백그라운드 프로세스와 작업 제어

### ■ 로그아웃 후에도 백그라운드 작업 계속 실행하기 : nohup

- 명령 실행 시 다음 예와 같이 출력 방향 전환을 하면 nohup.out 파일을 생성하지 않고 지정한 파일에 결과와 오류 메시지를 출력

```
user1@myubuntu:~$ nohup find / -name passwd > pw.dat 2>&1 &  
[1] 6874  
user1@myubuntu:~$ exit
```

- 다시 로그인하여 파일 내용 확인

```
user1@myubuntu:~$ more pw.dat  
nohup: 입력 무시  
/usr/share/bash-completion/completions/passwd  
/usr/share/lintian/overrides/passwd  
/usr/share/doc/passwd  
/usr/bin/passwd  
find: '/root': 허가 거부  
find: '/proc/tty/driver': 허가 거부  
(생략)
```

## 04 작업 예약

### ■ 특정한 시간에 작업을 수행하도록 예약할 수 있는 두 가지 방법

- 정해진 시간에 한 번만 수행
- 정해진 시간에 반복 수행

### ■ 정해진 시간에 한 번 실행

#### at

**기능** 예약한 명령을 정해진 시간에 실행한다.

**형식** at [옵션] 시간

**옵션**

- l : 현재 실행 대기 중인 명령의 전체 목록을 출력한다(atq 명령과 동일).
- r 작업 번호 : 현재 실행 대기 중인 명령에서 해당 작업 번호를 삭제한다(atrm과 동일).
- m : 출력 결과가 없더라도 작업이 완료되면 사용자에게 메일로 알려준다.
- f 파일 : 표준 입력 대신 실행할 명령을 파일로 지정한다.

**사용 예** at -m 0730 tomorrow                      at 10:00 pm                      at 8:15 am May 30

## 04 작업 예약

- at 명령 설치 : `sudo apt-get install at`, `sudo apt-get install mailutils`
- at 명령 설정하기

- at 명령을 사용하여 정해진 시간에 명령을 실행하도록 예약하려면 at 명령 뒤에 시간을 명시

```
user1@myubuntu:~/linux_ex/ch6$ at 09:00 am
warning: commands will be executed using /bin/sh
at>
```

- 시간을 지정하는 형식
  - at 4pm + 3 days : 지금부터 3일 후 오후 4시에 작업을 수행한다.
  - at 10am Jul 31 : 7월 31일 오전 10시에 작업을 수행한다.
  - at 1am tomorrow : 내일 오전 1시에 작업을 수행한다.
  - at 10:00am today : 오늘 오전 10시에 작업을 수행한다.
- at로 실행할 명령은 기본적으로 표준 입력으로 지정: 명령의 입력을 마치려면 ctrl+d 입력

```
user1@myubuntu:~/linux_ex/ch6$ at 09:00 am
at> ls -l ~user1
at> <EOT>
job 1 at Tue Feb 25 09:00:00 2014
user1@myubuntu:~/linux_ex/ch6$
```

시간을 지정한다.  
실행할 명령을 지정한다.  
Ctrl+d를 입력하여 종료한다.  
작업 예약이 완료되었다.

## 04 작업 예약

### ■ at 명령의 실행 결과 확인하기

- at 명령의 실행 결과는 메일로 전달

```
user1@myubuntu:~$  
You have mail in /var/mail/user1  
user1@myubuntu:~/linux_ex/ch6$ mail  
"/var/mail/user1": 1 message 1 new  
>N 1 user1          월 2월 24 14: 27/1198  Output from your job  
?  
Message 1:  
Return-Path: <user1@myubuntu>  
X-Original-To: user1  
Delivered-To: user1@myubuntu  
Received: by myubuntu (Postfix, from userid 1000)  
        id 1236F65222; Mon, 24 Feb 2014 14:22:01 +0900 (KST)  
Subject: Output from your job          5  
To: user1@myubuntu  
Message-Id: <20140224052201.1236F65222@myubuntu>  
Date: Mon, 24 Feb 2014 14:22:01 +0900 (KST)  
From: user1@myubuntu (user1)  
합계 148  
-rw-r--r--  1 user1 user1   8980  2월 20 21:19 examples.desktop  
drwxrwxr-x  7 user1 user1   4096  2월 24 06:28 linux_ex
```

(생략)



## 04 작업 예약

### ■ at 작업 파일 확인하기

- at로 생성된 작업 파일은 /var/spool/at 디렉터리에 저장

```
user1@myubuntu:~/linux_ex/ch6$ at 11:10 am
warning: commands will be executed using /bin/sh
at> ls
at> <EOT>
job 6 at Tue Feb 25 11:10:00 2014
user1@myubuntu:~/linux_ex/ch6$
```



```
user1@myubuntu:~/linux_ex/ch6$ sudo ls -l /var/spool/cron/atjobs
합계 4
-rwx----- 1 user1 daemon 2417  2월 24 14:25 a0000601625542
user1@myubuntu:~/linux_ex/ch6$
```

- root 사용자만 /var/spool/at 디렉터리 내용 확인 가능

## 04 작업 예약

### ■ at 작업 목록 확인하기 : -l 옵션, atq

- at 명령으로 설정된 작업의 목록은 -l 옵션으로 확인

```
user1@myubuntu:~/linux_ex/ch6$ at -l
6          Tue Feb 25 11:10:00 2014 a user1
user1@myubuntu:~/linux_ex/ch6$
```

- atq 명령으로도 확인 가능

```
user1@myubuntu:~/linux_ex/ch6$ atq
6          Tue Feb 25 11:10:00 2014 a user1
user1@myubuntu:~/linux_ex/ch6$
```

## 04 작업 예약

### ■ at 작업 삭제하기 : -d 옵션, atrm

- at 명령으로 설정한 작업이 실행되기 전에 삭제하려면 -d 옵션을 사용하고 삭제할 작업 번호를 지정

#### ① 작업예약

```
user1@myubuntu:~/linux_ex/ch6$ at 1am tomorrow
warning: commands will be executed using /bin/sh
at> ls
at> <EOT>
job 7 at Tue Feb 25 01:00:00 2014
user1@myubuntu:~/linux_ex/ch6$
```

#### ② 설정된 작업 확인

```
user1@myubuntu:~/linux_ex/ch6$ atq
7          Tue Feb 25 01:00:00 2014 a user1
6          Tue Feb 25 11:10:00 2014 a user1
user1@myubuntu:~/linux_ex/ch6$
```

#### ③ 작업 삭제

```
user1@myubuntu:~/linux_ex/ch6$ at -d 7
user1@myubuntu:~/linux_ex/ch6$ atrm 6
user1@myubuntu:~/linux_ex/ch6$ atq
user1@myubuntu:~/linux_ex/ch6$
```

## 04 작업 예약

### ■ at 명령 사용 제한하기

- 관련된 파일: /etc/at.allow와 /etc/at.deny
  - /etc/at.allow 파일과 /etc/at.deny 파일에는 한 줄에 사용자 이름을 하나씩만 기록
  - /etc/at.allow 파일이 있으면 이 파일에 있는 사용자만 at 명령을 사용할 수 있다. 이 경우에 /etc/at.deny 파일은 무시된다.
  - /etc/at.allow 파일이 없으면 /etc/at.deny 파일에 지정된 사용자를 제외한 모든 사용자가 at 명령을 사용할 수 있다.
  - 만약 두 파일이 모두 없다면 root만 at 명령을 사용할 수 있다.
  - 한 사용자가 두 파일 모두에 속해 있다면 그 사용자는 at 명령을 사용할 수 있다. /etc/at.allow 파일이 적용되기 때문이다.
  - /etc/at.deny를 빈 파일로 두면 모든 사용자가 at 명령을 사용할 수 있는데, 이것이 초기 설정이다.
- 
- at.deny 파일에 user1 사용자가 기록되어 있다면 at 명령을 실행했을 때 사용 권한이 없다는 메시지가 출력

```
user1@myubuntu:~/linux_ex/ch6$ at
You do not have permission to use at.
user1@myubuntu:~/linux_ex/ch6$
```

## 04 작업 예약

### ■ 정해진 시간에 반복 실행

#### crontab

**기능** 사용자의 crontab 파일을 관리한다.

**형식** crontab [-u 사용자 ID] [옵션] [파일 이름]

**옵션** -e : 사용자의 crontab 파일을 편집한다.

-l : crontab 파일의 목록을 출력한다.

-r : crontab 파일을 삭제한다.

**사용 예** crontab -l                  crontab -u user1 -e                  crontab -r

#### ■ crontab 파일 형식

분(0~59)	시(0~23)	일(1~31)	월(1~12)	요일(0~6)	작업 내용
---------	---------	---------	---------	---------	-------

30 23 1 \* \* /usr/bin/ls -l ~user1 > ~user1/cron.out

↓ ↓ ↓ ↓ ↓

1 분 2 시 3 일 4 월 5 요일 6 작업 내용

[그림 6-5] crontab 파일의 형식

## 04 작업 예약

### ■ crontab 파일 생성하고 편집하기 : crontab -e

- crontab 편집기는 기본적으로 VISUAL 또는 EDITOR 환경 변수에 지정된 편집기를 사용

```
user1@myubuntu:~/linux_ex/ch6$ EDITOR=vi;export EDITOR
user1@myubuntu:~/linux_ex/ch6$
```

- crontab -e 명령으로 편집한 파일을 저장하면 자동적으로 /var/spool/cron/crontabs 디렉터리에 사용자 이름으로 생성

```
user1@myubuntu:~/linux_ex/ch6$ sudo ls -l /var/spool/cron/crontabs
[sudo] password for user1:
합계 4
-rw----- 1 user1 crontab 1137  2월 24 14:55 user1
user1@myubuntu:~/linux_ex/ch6$
```

### ■ crontab 파일 내용 확인하기 : crontab -l

```
user1@myubuntu:~/linux_ex/ch6$ crontab -l
30 23 1 * * /bin/ls -l ~user1 > ~user1/cron.out
user1@myubuntu:~/linux_ex/ch6$
```

### ■ crontab 파일 삭제하기 : crontab -r

```
user1@myubuntu:~/linux_ex/ch6$ crontab -r
user1@myubuntu:~/linux_ex/ch6$ crontab -l
no crontab for user1
user1@myubuntu:~/linux_ex/ch6$
```

## 04 작업 예약

### ■ crontab 명령 사용 제한하기

- /etc/cron.allow, /etc/cron.deny 파일
- cron.deny 파일은 기본적으로 있지만 cron.allow 파일은 관리자가 만들어야 함
- 두 파일이 적용되는 기준
  - /etc/cron.allow 파일이 있으면 이 파일 안에 있는 사용자만 crontab 명령을 사용할 수 있다.
  - /etc/cron.allow 파일이 없고 /etc/cron.deny 파일이 있으면 이 파일에 사용자 계정이 없어야 crontab 명령을 사용할 수 있다.
  - /etc/cron.allow 파일과 /etc/cron.deny 파일이 모두 없다면 시스템 관리자만 crontab 명령을 사용할 수 있다.
- 두 파일이 모두 없는데 일반 사용자가 crontab 명령을 사용하려고 하면 다음과 같은 메시지가 출력

```
user1@myubuntu:~/linux_ex/ch6$ crontab -e
You (user1) are not allowed to use this program (crontab)
See crontab(1) for more information
user1@myubuntu:~/linux_ex/ch6$
```



# 우분투 리눅스

시스템 & 네트워크