

15-2. Process 관리 명령

Process 목록 보기

- ps 명령의 옵션
 - **유닉스(SVR4) 옵션:** 묶어서 사용할 수 있고, 불임표로 시작한다. (예: -ef)
 - **BSD 옵션:** 묶어서 사용할 수 있고, 불임표로 시작하지 않는다. (예: aux)
 - **GNU 옵션:** 불임표 2개로 시작한다. (예: --pid)
- ps

기능	현재 실행 중인 프로세스의 정보를 출력한다.		
형식	ps 옵션		
옵션	<p><유닉스 옵션></p> <p>-e: 시스템에서 실행 중인 모든 프로세스의 정보를 출력</p> <p>-f: 프로세스의 자세한 정보를 출력</p> <p>-u UID: 특정 사용자에게 대한 모든 프로세스의 정보를 출력</p> <p>-p PID: pid로 지정한 특정 프로세스의 정보를 출력</p> <p>-l: 프로세스의 정보를 길게 출력. 우선순위 관련 PRI와 NI 값을 확인</p> <p><BSD 옵션></p> <p>a: 터미널에서 실행한 프로세스의 정보를 출력</p> <p>u: 프로세스의 소유자 이름, CPU 사용량, 메모리 사용량 등 상세 정보를 출력</p> <p>x: 시스템에서 실행 중인 모든 프로세스의 정보를 출력</p> <p><GNU 옵션></p> <p>--pid PID목록: 목록으로 지정한 특정 PID의 정보를 출력</p>		
사용 예	ps	ps -ef	ps aux

현재 단말기의 프로세스 목록 출력하기: ps

- ps 명령을 옵션 없이 사용하면 현재 셸이나 터미널에서 실행한 사용자 프로세스의 정보를 출력한다.
 - PID: 프로세스 번호
 - TTY: 현재 터미널 번호
 - TIME: 해당 프로세스가 사용한 CPU 시간의 양
 - CMD: 프로세스가 실행 중인 명령

전체 프로세스의 목록 출력하기(유닉스 옵션): -ef 옵션

```

프로그램 위치 시스템 8월 16일 (목) 오후 07:00 root
root@itserver:~
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
[ root@itserver ~]# ps
  PID TTY          TIME CMD
21454 pts/0    00:00:00 bash
21622 pts/0    00:00:00 ps
[ root@itserver ~]# ps -ef | more
UID          PID    PPID  C STIME TTY          TIME CMD
root           1        0  0  14:06 ?        00:00:01 /sbin/init
root           2        0  0  14:06 ?        00:00:00 [kthreadd]
root           3        2  0  14:06 ?        00:00:00 [migration/0]
root           4        2  0  14:06 ?        00:00:00 [ksoftirqd/0]
root           5        2  0  14:06 ?        00:00:00 [stopper/0]
root           6        2  0  14:06 ?        00:00:00 [watchdog/0]
root           7        2  0  14:06 ?        00:00:07 [events/0]
root           8        2  0  14:06 ?        00:00:00 [events/0]
root           9        2  0  14:06 ?        00:00:00 [events_long/0]
root          10        2  0  14:06 ?        00:00:00 [events_power_ef]
root          11        2  0  14:06 ?        00:00:00 [cgroup]
root          12        2  0  14:06 ?        00:00:00 [khelper]
root          13        2  0  14:06 ?        00:00:00 [netns]
root          14        2  0  14:06 ?        00:00:00 [async/mgr]
root          15        2  0  14:06 ?        00:00:00 [pm]
root          16        2  0  14:06 ?        00:00:00 [sync_supers]
root          17        2  0  14:06 ?        00:00:00 [bdi-default]
root          18        2  0  14:06 ?        00:00:00 [kintegrityd/0]
root          19        2  0  14:06 ?        00:00:00 [kblockd/0]
root          20        2  0  14:06 ?        00:00:00 [kacpid]
root          21        2  0  14:06 ?        00:00:00 [kacpi_notify]
root          22        2  0  14:06 ?        00:00:00 [kacpi_hotplug]
root          23        2  0  14:06 ?        00:00:00 [ata_aux]
root          24        2  0  14:06 ?        00:00:02 [ata_sff/0]
root          25        2  0  14:06 ?        00:00:00 [ksuspend_usbd]
root          26        2  0  14:06 ?        00:00:00 [khubd]
root          27        2  0  14:06 ?        00:00:00 [kseriod]
root          28        2  0  14:06 ?        00:00:00 [md/0]
root          29        2  0  14:06 ?        00:00:00 [md_misc/0]
root          30        2  0  14:06 ?        00:00:00 [linkwatch]
root          33        2  0  14:06 ?        00:00:00 [khungtaskd]
root          34        2  0  14:06 ?        00:00:00 [lru-add-drain/0]
root          35        2  0  14:06 ?        00:00:00 [kswapd0]

```

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

전체 프로세스의 목록 출력하기(BSD 옵션): aux 옵션

```

[ root@itserver ~]# ps aux | more
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.0  19364  1608 ?        Ss   14:06   0:01 /sbin/init
root           2  0.0  0.0      0     0 ?        S    14:06   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        S    14:06   0:00 [migration/0]
root           4  0.0  0.0      0     0 ?        S    14:06   0:00 [ksoftirqd/0]
root           5  0.0  0.0      0     0 ?        S    14:06   0:00 [stopper/0]
root           6  0.0  0.0      0     0 ?        S    14:06   0:00 [watchdog/0]
root           7  0.0  0.0      0     0 ?        S    14:06   0:07 [events/0]
root           8  0.0  0.0      0     0 ?        S    14:06   0:00 [events/0]
root           9  0.0  0.0      0     0 ?        S    14:06   0:00 [events_long/0]
root          10  0.0  0.0      0     0 ?        S    14:06   0:00 [events_power_ef]
root          11  0.0  0.0      0     0 ?        S    14:06   0:00 [cgrou]
root          12  0.0  0.0      0     0 ?        S    14:06   0:00 [khelper]
root          13  0.0  0.0      0     0 ?        S    14:06   0:00 [netns]
root          14  0.0  0.0      0     0 ?        S    14:06   0:00 [async/mgr]
root          15  0.0  0.0      0     0 ?        S    14:06   0:00 [pm]
root          16  0.0  0.0      0     0 ?        S    14:06   0:00 [sync_supers]
root          17  0.0  0.0      0     0 ?        S    14:06   0:00 [bdi-default]
root          18  0.0  0.0      0     0 ?        S    14:06   0:00 [kintegrityd/0]
root          19  0.0  0.0      0     0 ?        S    14:06   0:00 [kblockd/0]
root          20  0.0  0.0      0     0 ?        S    14:06   0:00 [kacpid]
root          21  0.0  0.0      0     0 ?        S    14:06   0:00 [kacpi_notify]
root          22  0.0  0.0      0     0 ?        S    14:06   0:00 [kacpi_hotplug]
root          23  0.0  0.0      0     0 ?        S    14:06   0:00 [ata_aux]
root          24  0.0  0.0      0     0 ?        S    14:06   0:02 [ata_sff/0]
root          25  0.0  0.0      0     0 ?        S    14:06   0:00 [ksuspend_usbd]
root          26  0.0  0.0      0     0 ?        S    14:06   0:00 [khubd]
root          27  0.0  0.0      0     0 ?        S    14:06   0:00 [kseriod]
root          28  0.0  0.0      0     0 ?        S    14:06   0:00 [md/0]
root          29  0.0  0.0      0     0 ?        S    14:06   0:00 [md_misc/0]
root          30  0.0  0.0      0     0 ?        S    14:06   0:00 [linkwatch]
root          33  0.0  0.0      0     0 ?        S    14:06   0:00 [khungtaskd]
root          34  0.0  0.0      0     0 ?        S    14:06   0:00 [lru-add-drain/0]
root          35  0.0  0.0      0     0 ?        S    14:06   0:00 [kswapd0]
root          36  0.0  0.0      0     0 ?        SN   14:06   0:00 [ksmd]
root          37  0.0  0.0      0     0 ?        SN   14:06   0:00 [khugepaged]
root          38  0.0  0.0      0     0 ?        S    14:06   0:00 [aio/0]
root          39  0.0  0.0      0     0 ?        S    14:06   0:00 [crypto/0]

```

USER	사용자 계정 이름	VSZ	사용하고 있는 가상 메모리의 크기(KB)
%CPU	CPU 사용량	RSS	사용하고 있는 물리적 메모리 크기(KB)
%MEM	물리적 메모리 사용량	START	프로세스의 시작 시간

특정 Process 정보 검색하기

ps 명령을 이용해 특정 프로세스 정보 검색하기

- ps -ef | grep 명령
 - ps 명령과 grep 명령을 |(파이프)로 연결하여 특정 프로세스를 검색할 수 있다.
- [user1@itserver ~]\$ ps -ef | grep bash

pgrep 명령을 이용해 특정 프로세스 정보 검색하기

- pgrep
 - = ps + grep

기능	지정한 패턴과 일치하는 프로세스의 정보를 출력한다.
형식	pgrep 옵션 패턴

옵션	-x: 패턴과 정확히 일치하는 프로세스의 정보를 출력 -n: 패턴을 포함하고 있는 가장 최근의 프로세스 정보를 출력 -u 사용자명: 특정 사용자에게 대한 모든 프로세스를 출력 -l: PID와 프로세스의 이름을 출력 -t term: 특정 단말기와 관련된 프로세스의 정보를 출력
사용 예	pgrep bash

```

[ root@itserver ~]# ps -ef | grep bash
root      21454 21451  0 18:49 pts/0    00:00:00 bash
root      21504 21140  0 18:53 tty2      00:00:00 -bash
root      21684 21454  0 19:11 pts/0    00:00:00 grep bash
[ root@itserver ~]# pgrep -x bash
21454
21504
[ root@itserver ~]# pgrep -l bash
21454 bash
21504 bash
[ root@itserver ~]#
[ root@itserver ~]#

```

- 더 자세한 정보를 검색하려면 pgrep 명령과 ps 명령을 연결해서 사용한다.
 - **ps -fp \$(pgrep -x bash)**
 - pgrep으로 검색하려는 프로세스의 PID를 찾아 ps 명령으로 자세한 정보를 확인하는 것
 - **ps -fp \$(pgrep -u user1 bash)**
 - -u 옵션으로 사용자명을 지정하여 그 사용자의 프로세스 정보만 자세하게 검색하는 것

Process 종료하기

시그널

- 어떤 이벤트가 발생했다는 것을 프로세스에게 알리기 위해 전달하는 메시지이다. = 소프트웨어 인터럽트
- 사용자가 특정 키를 입력했을 때나, kill 같은 시스템 호출, 알람 같은 소프트웨어 조건, 하드웨어의 예외 상황에서 발생
- 시그널 종류

- 2. SIGINT: Ctrl + c
- 3. SIGQUIT: Ctrl + w
- 9. SIGKILL: 프로세스 종료
- 10. SIGUSR1: 사용자 정의 시그널
- 15. SIGTERM: kill 시스템 호출

kill 명령을 이용해 프로세스 종료하기

- **kill**

- kill 명령은 인자로 지정한 프로세스에 시그널을 전달한다.
- 시그널은 프로세스에 보내는 간단한 숫자 메시지이다.

기능	지정한 시그널을 프로세스에 보낸다.		
형식	kill 시그널 PID ...		
시그널	-(소문자 L): 시그널 목록 출력 -2: 인터럽트 시그널 -9: 프로세스 강제 종료 -15: 프로세스가 관련된 파일을 정리하고 프로세스를 종료. 종료되지 않는 프로세스가 있을 수도(기본값)		
사용 예	kill 1001	kill -15 1001	kill -9 1001

pkill 명령을 이용해 프로세스 종료하기

- pkill은 kill 명령과 같이 시그널을 보내는데, PID가 아니라 프로세스의 명령 이름으로 프로세스를 찾아 종료한다.
- kill 명령과의 차이점은 명령 이름으로 찾아 종료하므로 같은 명령이 여러 개 검색될 경우 한번에 모두 종료한다는 점이다.
- 물론 자신이 소유한 프로세스만 종료할 수 있다.
- **pkill -x 명령어**

Process 관리 도구

top 명령

- **top**

- 현재 실행 중인 프로세스에 대한 정보를 주기적으로 출력하는데, 프로세스의 자세한 요약 정보를 상단에 출력하고 각 프로세스의 정보를 하단에 출력한다.
- top 명령의 출력 정보

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		

```

프로그램 위치 시스템
root@itserver:~
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
top - 19:27:07 up 5:20, 3 users, load average: 0.00, 0.00, 0.00
Tasks: 151 total, 1 running, 150 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.7%us, 0.7%sy, 0.0%ni, 98.7%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 1923104k total, 1403680k used, 519424k free, 56604k buffers
Swap: 4128764k total, 628k used, 4128136k free, 893676k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 7701 root        20   0   205m   40m   12m  S   0.3   2.2   0:21.96 Xorg
21451 root        20   0   317m   15m   10m  S   0.3   0.8   0:01.55 gnome-terminal
21723 root        20   0 15028  1264   948  R   0.3   0.1   0:00.08 top
   1 root        20   0 19364  1608  1308  S   0.0   0.1   0:01.20 init
   2 root        20   0     0     0     0  S   0.0   0.0   0:00.01 kthreadd
   3 root        RT   0     0     0     0  S   0.0   0.0   0:00.00 migration/0
   4 root        20   0     0     0     0  S   0.0   0.0   0:00.03 ksoftirqd/0
   5 root        RT   0     0     0     0  S   0.0   0.0   0:00.00 stopper/0
   6 root        RT   0     0     0     0  S   0.0   0.0   0:00.04 watchdog/0
   7 root        20   0     0     0     0  S   0.0   0.0   0:08.59 events/0
   8 root        20   0     0     0     0  S   0.0   0.0   0:00.00 events/0
   9 root        20   0     0     0     0  S   0.0   0.0   0:00.00 events_long/0
  10 root        20   0     0     0     0  S   0.0   0.0   0:00.00 events_power_ef
  11 root        20   0     0     0     0  S   0.0   0.0   0:00.00 cgroup
  12 root        20   0     0     0     0  S   0.0   0.0   0:00.01 khelper
  13 root        20   0     0     0     0  S   0.0   0.0   0:00.00 netns
  14 root        20   0     0     0     0  S   0.0   0.0   0:00.00 async/mgr
  15 root        20   0     0     0     0  S   0.0   0.0   0:00.00 pm
  16 root        20   0     0     0     0  S   0.0   0.0   0:00.09 sync_supers
  17 root        20   0     0     0     0  S   0.0   0.0   0:00.00 bdi-default
  18 root        20   0     0     0     0  S   0.0   0.0   0:00.00 kintegrityd/0

```

현재시간
시스템 가동시간
로그인된 사용자 수
시스템 평균부하 [1분, 5분, 15분]

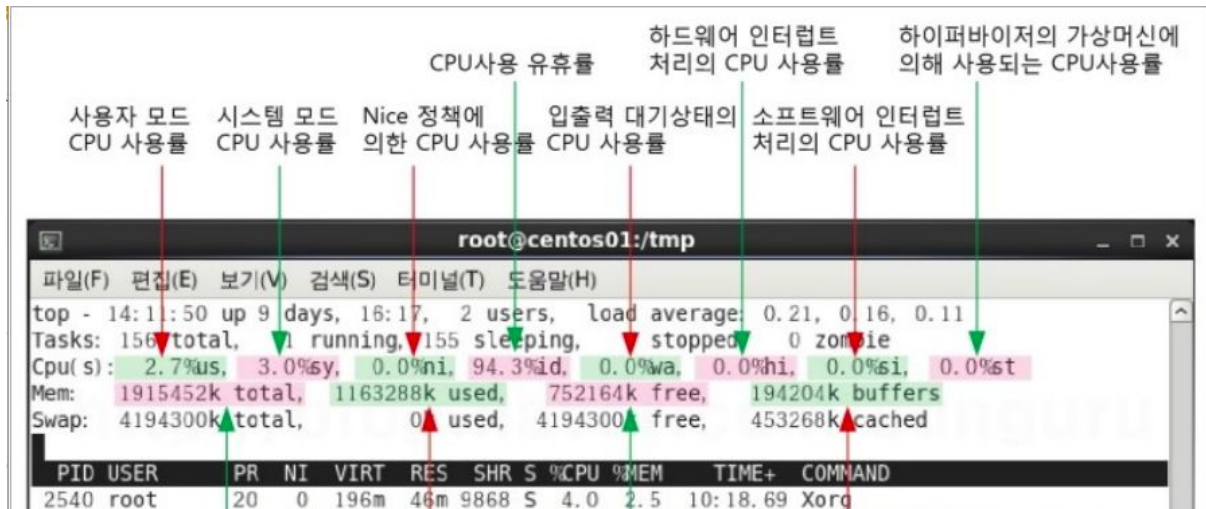
```

root@centos01:/tmp
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
top - 14:11:50 up 9 days, 18:17, 2 users, load average: 0.21, 0.16, 0.11
Tasks: 156 total, 1 running, 155 sleeping, 0 stopped, 0 zombie
Cpu(s): 2.7%us, 3.0%sy, 0.0%ni, 94.3%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 1915452k total, 1163288k used, 752164k free, 194204k buffers
Swap: 4194300k total, 0k used, 4194300k free, 453268k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
2540 root        20   0   196m   46m  9868  S   4.0   2.5  10:18.69 Xorg

```

전체 프로세스 수
현재 실행 중인 프로세스 수
유휴 상태인 프로세스 수
정지 상태인 프로세스 수
좀비 프로세스 수



nice 명령

• nice

- 프로세스의 우선순위를 변경하는 명령으로 NI값을 설정할 때 사용한다.
- NI의 기본값은 0이고, 지정 가능한 값의 범위는 -20~19까지이다.
- 값이 작을수록 우선순위가 높다.
- nice 명령으로 지정하는 값은 기존 설정된 값에 증감하는 형식이다.
- 일반 사용자는 NI 값을 증가만 가능하고, root 사용자만이 NI 값을 감소시켜 우선순위를 높일 수 있다.

기능	프로세스의 우선순위를 변경한다.
형식	nice 옵션 프로세스명
옵션	-값: 프로세스에 설정된 NI 값을 지정한 NI 값 증감. 미지정시 기본적으로 10 지정
사용예	nice nice --10 bash nice bash

- ps -l
 - NI 값을 미리 확인한다.
- nice --10 bash
 - bash의 NI 값을 -10만큼 감소시켜서 우선순위를 높인다.

renice 명령

• renice

- 실행 중인 프로세스의 우선순위를 변경할 때 사용하는 명령으로 PID, 사용자 이름, 프로세스의 그룹ID를 이용한다.
- nice는 프로세스명으로 우선순위를 조정하고, 명령을 실행하면 새로운 프로세스가 추가로 발생되지만, renice는 주로 PID를 사용하고 기존의 프로세스를 교체하여 조정한다.
- renice는 기존의 NI 값에 상관없이 지정한 NI 값이 바로 설정된다.

- nice와 동일하게 root만이 NI 값을 낮춤으로써 우선순위를 높일 수 있고, NI 값의 범위는 -20부터 19이다.

기능	프로세스의 우선순위를 변경한다.
형식	renice 옵션 NI값 PID
옵션	-p: PID를 지정(기본값) -u: 사용자 이름을 지정 -g: GID를 지정
사용예	renice 10 2868

- renice 10 2827
 - PID가 2827에 NI 값을 10으로 지정한다.
- renice 1 987 -u daemon root -p 1222
 - PID가 987, 1222인 프로세스와 사용자가 daemon, root인 모든 프로세스의 NI 값을 1로 지정한다.

시스템 도구

- CentOS 6에서 기본으로 제공하는 도구 중에 시스템 정보가 있다.
- 프로그램 >> 시스템 도구 >> 시스템 정보 선택 후 자원 탭을 클릭하면 된다.

