



## 8장. 프로세스와 사용자 명령 익히기[1]

## ❖ 학습목표

- 프로세스의 개념을 이해한다
- 프로세스 관련 유닉스 명령의 사용 방법을 익힌다
- 포그라운드 처리와 백그라운드 처리의 차이를 이해한다
- 사용자 정보를 보는 유닉스 명령의 사용 방법을 익힌다

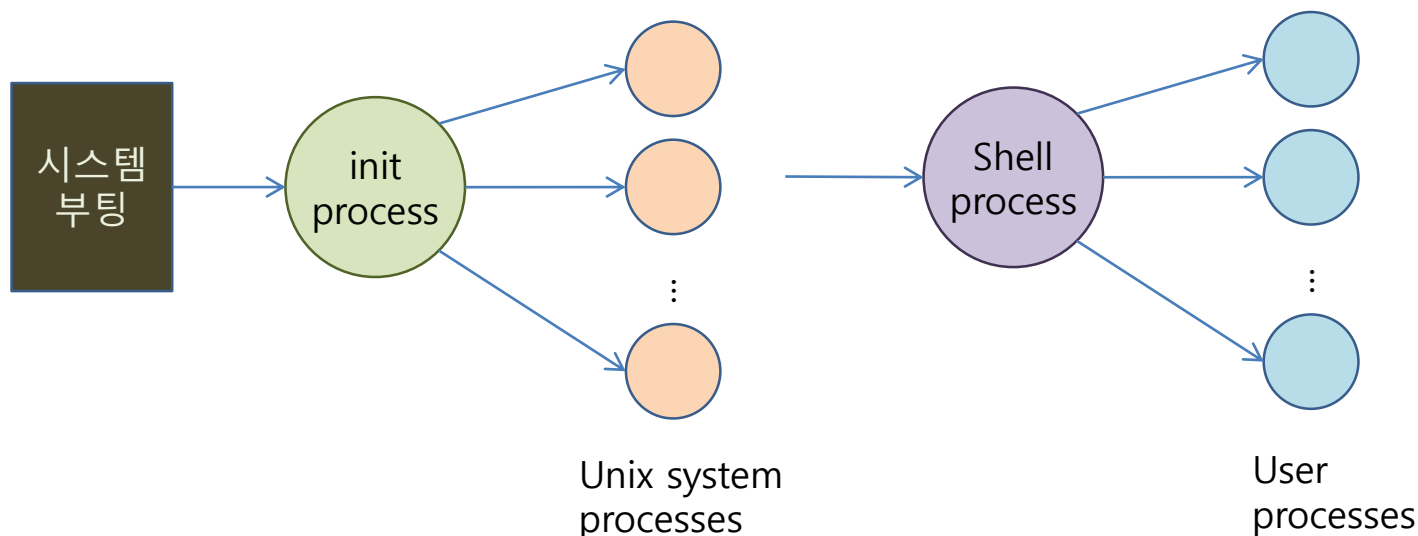
## ❖ 내용

- 프로세스의 개념과 종류
- 프로세스 관리 명령
- 포그라운드와 백그라운드 프로세스
- 사용자 정보보기

# 01. 프로세스의 개념과 종류

## ❖ 프로세스

- 실행중인 프로그램
- 종류
  - 시스템 프로세스 : 유닉스 운영에 필요한 기능 수행
  - 사용자 프로세스 : 사용자들이 실행시킨 프로세스



# 프로세스의 종류

| 프로세스       | 설명   |
|------------|--|
| 데몬(daemon) | 유닉스 커널에 의해 실행되는 프로세스로 특정 서비스 제공                              |
| 부모(parent) | 자식 프로세스를 만드는 프로세스  |
| 자식(child)  | 부모에 의해 생성된 프로세스<br>실행이 끝나면 부모 프로세스에 결과를 돌려주고 종료              |
| 고아(orphan) | 실행도중에 부모 프로세스가 종료된 프로세스<br>고아프로세스는 1번 프로세스를 새로운 부모로 가진다      |
| 좀비(zombie) | 부모프로세스가 종료처리를 하지 않은 프로세스<br>종료가 되었음에도 할당된 시스템 자원이 회수되지 않은 상태 |

## 02. 프로세스 관리

### ❖ 프로세스 목록 보기

- ps
- pgrep

### ❖ 프로세스 종료 시키기

- kill
- pkill

### ❖ 포그라운드(전위)와 백그라운드(후위) 작업 제어

- fg
- bg
- jobs

# 프로세스 목록 보기 - ps

## ps [ 옵션 ]

### ❖ process status

### ❖ 프로세스 정보를 출력

- PID, 터미널, CPU 시간, 명령어

### ❖ 옵션

- -e : 시스템에 있는 모든 프로세스 정보 출력
- -f : 프로세스에 대한 자세한 정보 출력
- -u UID : 특정 사용자에게 대한 모든 프로세스 출력

# 프로세스 목록 보기 - ps 사용예

```
$ ps
  PID TTY          TIME CMD
 1283 pts/1        0:00 bash
 1289 pts/1        0:00 ps
$
```

```
$ ps -e
  PID TTY          TIME CMD
    0 ?            0:04 sched
    5 ?            0:01 zpool-rp
    6 ?            0:00 kmem_tas
...
$
```

```
$ ps -f
  UID    PID  PPID    C   STIME TTY          TIME CMD
   user1 1283  1278    0  19:25:11 pts/1        0:00 -bash
   user1 1293  1283    0  19:26:51 pts/1        0:00 ps -f
$ ps -u user1
  PID TTY          TIME CMD
 1283 pts/1        0:00 bash
 1313 pts/1        0:00 ps
$
```

# 프로세스 목록 보기 - ps 사용예

```
$ ps -ef | more
```

| UID  | PID | PPID | C | STIME    | TTY | TIME | CMD         |
|------|-----|------|---|----------|-----|------|-------------|
| root | 0   | 0    | 0 | 19:23:21 | ?   | 0:04 | sched       |
| root | 5   | 0    | 0 | 19:23:20 | ?   | 0:01 | zpool-rpool |
| root | 6   | 0    | 0 | 19:23:22 | ?   | 0:00 | kmem_task   |

```
...
```

```
--More--
```

| 구분   | 설명          | 구분    | 설명              |
|------|-------------|-------|-----------------|
| UID  | 소유자의 사용자 ID | STIME | 프로세스 시작시간       |
| PID  | 프로세스 번호     | TTY   | 터미널 번호( ? : 데몬) |
| PPID | 부모 프로세스 번호  | TIME  | CPU 사용시간        |
| C    | 프로세스 우선순위   | CMD   | 명령어 이름          |



# 특정 프로세스 정보검색하기 - pgrep

## pgrep [ 옵션 ] 패턴

### ❖ 프로세스 이름으로 찾아 정보를 출력

- 솔라리스에만 있는 특별한 명령
- = ps [옵션] | grep 패턴

### ❖ 옵션

- -x : 패턴과 정확히 일치하는 프로세스 정보 출력
- -n : 패턴을 포함하고 있는 가장 최근의 프로세스 정보 출력
- -u uid : 특정 사용자의 모든 프로세스 출력
- -l : PID와 프로세스 이름 출력
- -t term : 특정 단말기와 관련된 프로세스 정보 출력

### ❖ 패턴

- 찾으려는 정보

# 특정 프로세스 정보검색하기 - pgrep

```
$ pgrep bash
1283
$ ps | grep bash
 1283 pts/1          0:00 bash
$ pgrep -l bash
1283 bash
$
```

```
$ pgrep -l telnet
1276 in.telnetd
$ pgrep -lt pts/1
1283 bash
1278 login
$
```

```
$ pgrep -l -u user1
1283 bash
$ pgrep -lu user1
1283 bash
$
```

옵션의 중복  
사용 방법

# 프로세스 종료 시키기

## ❖ 프로세스의 종료

- ps 명령으로 찾은 프로세스 중 불필요한 프로세스를 강제로 종료시킨다.
- 프로세스를 종료시키면 그 자식 프로세스들도 같이 종료된다.
- 프로세스를 종료시킬 때 PID나 프로세스 이름을 알아야 한다.

## ❖ 프로세스 종료 시키기

- kill
- pkill

# 프로세스 종료 - kill

**kill [ 시그널 ] pid**

## ❖ PID로 지정한 프로세스에게 시그널을 보냄

- 시그널을 받은 프로세스는 지정된 동작을 수행
- 시그널을 지정하지 않으면 프로세스를 종료시키는 15 번 시그널 (SIGTERM) 을 보냄
- 사용자의 프로세스만 종료시킬 수 있음
- 관리자(root)는 모든 프로세스를 종료시킬 수 있음

## ❖ 시그널

- 프로세스에게 보내는 신호
- 프로세스는 시그널을 수신하면 지정된 동작을 수행한다.
  - 예 : 신호 무시, 프로세스 종료, 일시 정지 등
- 종류

| 시그널<br>번호 | 시그널<br>이름 | 기능   | 기본<br>응답 |
|-----------|-----------|--|----------|
| 1         | SIGHUP    | • 터미널 연결이 끊어진 경우에 발생   | 종료       |
| 2         | SIGINT    | • 보통 Ctrl-C에 의해 발생   | 종료       |
| 9         | SIGKILL   | • 프로세스를 강제종료 시킨다.<br>• 이 시그널은 무시할 수 없다.                        | 종료       |
| 15        | SIGTERM   | • 프로세스를 종료시킨다.<br>• 이 시그널은 무시할 수도 있다.<br>• kill 명령이 보내는 기본 시그널 | 종료       |

# 프로세스 종료 - kill 사용예

```
$ kill 15759  
$
```

soft kill

15759 프로세스에게 SIGTERM 시그널을 보낸다. 프로세스는 하던 작업을 마무리하고 종료됨. 정상 종료

```
$ kill -9 15759  
$
```

sure kill

15759 프로세스에게 SIGKILL 시그널을 보낸다. SIGKILL 을 받으면 프로세스가 즉시 종료된다. 어떤 시그널도 무시하는 프로세스를 종료시킬 때 유용하다. 비정상 종료

# 프로세스 종료 - pkill

## pkill [ 시그널 ] 프로세스명

### ❖ 프로세스의 명령이름으로 프로세스를 찾아 지정한 시그널 보냄

- 솔라리스에서만 제공
- 사용자가 소유한 프로세스만 종료 가능

```
$ pkill sleep  
$
```

```
$ pkill -9 sleep  
$
```

# 프로세스 관리 도구 - top

## top [ 옵션 ]

- ❖ 주기적으로 현재 실행중인 프로세스에 대한 정보 출력
  - 솔라리스 11부터 기본 패키지로 탑재
- ❖ 자세한 요약 정보 출력
- ❖ 종료 : q



# 프로세스 관리 도구 - top 사용예

```
i20181234@localhost:~  
[i20181234@localhost ~]$ top  
top - 16:53:22 up 99 days, 23:13, 2 users, load average: 0.00, 0.01, 0.05  
Tasks: 281 total, 1 running, 280 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
KiB Mem : 8162524 total, 2001064 free, 1216908 used, 4944552 buff/cache  
KiB Swap: 8257532 total, 8257532 free, 0 used, 6479544 avail Mem  
  
  PID USER      PR  NI   VIRT    RES    SHR  S  %CPU  %MEM    TIME+  COMMAND  
  6889 i201812+  20   0  159952    4588   3748  R   0.3   0.1   0:00.05  top  
16005 i201525+  20   0  668656   29264  24220  S   0.3   0.4   3:15.31  goa-daemon  
    1 root      20   0  194436    9036   5668  S   0.0   0.1   2:42.24  systemd  
    2 root      20   0         0         0        0  S   0.0   0.0   0:05.53  kthreadd  
    3 root      20   0         0         0        0  S   0.0   0.0   0:14.91  ksoftirqd/0  
    5 root       0 -20         0         0        0  S   0.0   0.0   0:00.00  kworker/0:0H  
    8 root      20   0         0         0        0  S   0.0   0.0  24:59.91  rcu_sched  
    9 root      20   0         0         0        0  S   0.0   0.0   0:00.00  rcu_bh  
   10 root      20   0         0         0        0  S   0.0   0.0   7:13.66  rcuos/0  
   11 root      20   0         0         0        0  S   0.0   0.0   0:00.00  rcuob/0  
   12 root      rt    0         0         0        0  S   0.0   0.0   0:02.74  migration/0  
   13 root      rt    0         0         0        0  S   0.0   0.0   1:55.17  watchdog/0  
   14 root      rt    0         0         0        0  S   0.0   0.0   0:19.90  watchdog/1  
   15 root      rt    0         0         0        0  S   0.0   0.0   0:02.68  migration/1  
   16 root      20   0         0         0        0  S   0.0   0.0   0:00.69  ksoftirqd/1  
   18 root       0 -20         0         0        0  S   0.0   0.0   0:00.00  kworker/1:0H  
   19 root      20   0         0         0        0  S   0.0   0.0   1:10.04  rcuos/1  
   20 root      20   0         0         0        0  S   0.0   0.0   0:00.00  rcuob/1  
   22 root      rt    0         0         0        0  S   0.0   0.0   0:18.53  watchdog/2  
   23 root      rt    0         0         0        0  S   0.0   0.0   0:02.75  migration/2  
   24 root      20   0         0         0        0  S   0.0   0.0   0:02.46  ksoftirqd/2  
   26 root       0 -20         0         0        0  S   0.0   0.0   0:00.00  kworker/2:0H  
   27 root      20   0         0         0        0  S   0.0   0.0   0:32.94  rcuos/2  
   28 root      20   0         0         0        0  S   0.0   0.0   0:00.00  rcuob/2
```



---

*다음 강좌에 계속*