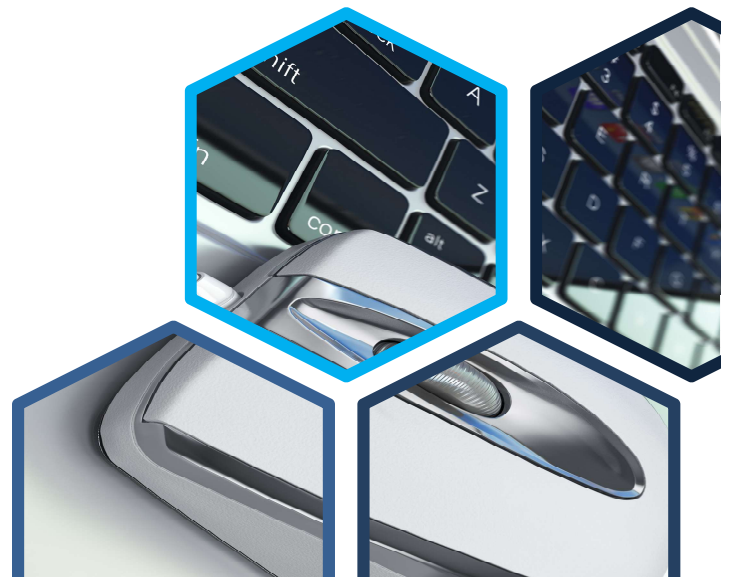


컴퓨터시스템

프로세스 관리하기





학습목표

- 프로세스 관리 명령 방법을 설명할 수 있다.
- 포그라운드·백그라운드 프로세스 및 작업 제어에 대해 설명할 수 있다.
- 작업 예약 방법에 대해 설명할 수 있다.



학습내용

- 프로세스 관리 명령
- 포그라운드·백그라운드 프로세스 및 작업 제어
- 작업 예약



프로세스 관리 명령

1 프로세스의 개요



프로세스

- 현재 시스템에서 실행 중인 프로그램

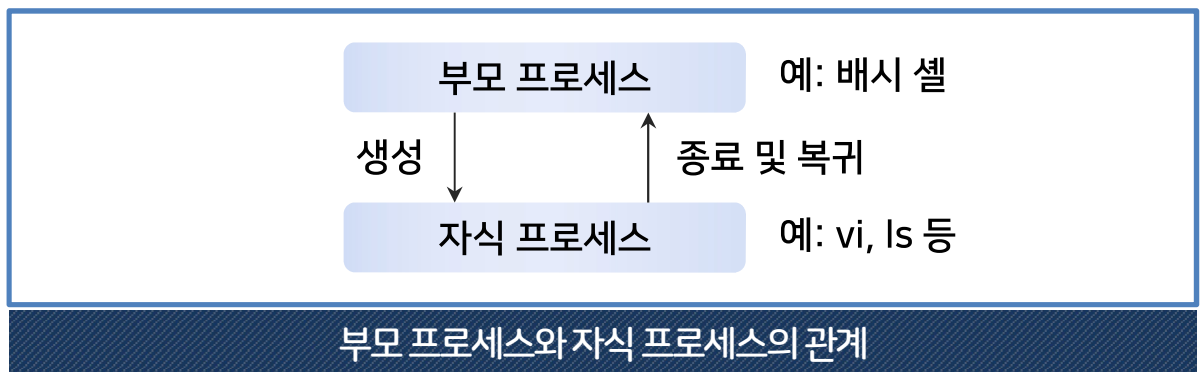


프로세스와 부모-자식 관계

- 필요에 따라 **부모 프로세스(Parent process)**는 자식 프로세스(Child process)를 **생성**하고, 자식 프로세스는 또 다른 자식 프로세스 생성 가능
- 부팅할 때 스케줄러가 실행한 프로세스인 **systemd**와 **kthreadd** 프로세스를 **제외**하면 모든 프로세스는 부모 프로세스를 가지고 있음
- 자식 프로세스는 할 일이 끝나면 **부모 프로세스에 결과를 돌려주고 종료**



부모 프로세스와 자식 프로세스의 관계



프로세스의 번호

- PID: 각 프로세스가 가지고 있는 **고유한 번호**



프로세스 관리 명령

1 프로세스의 개요



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 프로세스(systemd 프로세스)**가 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 프로세스**라고 나오기도 함
- 좀비 프로세스가 증가하면 **프로세스 테이블의 용량이 부족**해서 일반 프로세스가 실행되지 않을 수도 있음



프로세스 관리 명령

2 프로세스 정보 출력



프로세스 목록 보기: ps

- 기능: 현재 실행 중인 프로세스의 목록(정보)를 출력함
- 형식: ps [옵션]
- 옵션

유닉스(SVR4) 옵션	BSD 옵션	GNU 옵션
묶어서 사용할 수 있고, 붙임표로 시작함	묶어서 사용할 수 있고, 붙임표로 시작하지 않음	붙임표 두 개로 시작함
예 ▶ -ef	예 ▶ aux	예 ▶ --pid

유닉스 옵션	-e	시스템에서 실행 중인 모든 프로세스의 정보를 출력함
	-f	프로세스의 자세한 정보를 출력함
	-u uid	특정 사용자에게 대한 모든 프로세스의 정보를 출력함
	-p pid	pid로 지정한 특정 프로세스의 정보를 출력함
BSD 옵션	a	터미널에서 실행한 프로세스의 정보를 출력함
	u	프로세스 소유자 이름, CPU 사용량, 메모리 사용량 등 상세 정보를 출력함
	x	시스템에서 실행 중인 모든 프로세스의 정보를 출력함
GNU 옵션	--pid PID 목록	목록으로 지정한 특정 PID의 정보를 출력함



사용 예:

ps

ps -ef

ps aux



프로세스 관리 명령

2 프로세스 정보 출력



현재 터미널의 프로세스 목록 출력하기: ps

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



프로세스의 상세 정보 출력하기: -f 옵션

- 프로세스의 상세한 정보를 출력
- PPID와 터미널 번호, 시작 시간 등
- ps -f의 출력 정보

항목	의미
UID	프로세스를 실행한 사용자 ID
PID	프로세스 번호
PPID	부모 프로세스 번호
C	CPU 사용량(% 값)

항목	의미
STIME	프로세스의 시작 날짜나 시간
TTY	프로세스가 실행된 터미널의 종류와 번호
TIME	프로세스 실행 시간
CMD	실행되고 있는 프로그램 이름(명령)



프로세스 관리 명령

2 프로세스 정보 출력



터미널에서 실행한 프로세스의 정보 출력하기: a 옵션

- 터미널에서 실행한 프로세스의 정보를 출력
- STAT에 사용되는 문자의 의미

문자	의미	문자	의미	
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)	멀티스레드	



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**

- ps au의 출력 정보

항목	의미	항목	의미
USER	사용자 계정 이름	VSZ	사용 중인 가상 메모리의 크기 (KB)
%CPU	퍼센트로 표시한 CPU 사용량	RSS	사용 중인 물리적 메모리의 크기(KB)
%MEM	퍼센트로 표시한 물리적 메모리 사용량	START	프로세스 시작 시간



프로세스 관리 명령

2 프로세스 정보 출력



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



-ef 옵션 사용

- ☁ 전체 프로세스의 더 자세한 정보 출력



전체 프로세스 목록 출력하기(BSD 옵션): ax 옵션

- ☁ 시스템에서 실행 중인 모든 프로세스를 출력



aux 옵션 사용

- ☁ -ef처럼 시스템에서 실행 중인 모든 프로세스에 대한 자세한 정보를 출력



특정 사용자의 프로세스 목록 출력하기: -u 옵션



더 상세한 정보를 보고 싶으면 -f 옵션을 함께 사용



프로세스 관리 명령

3 프로세스 정보 검색



ps 명령을 이용해 특정 프로세스 정보 검색하기

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



pgrep 명령을 이용해 특정 프로세스 정보 검색하기

- ☁ 기능: 지정한 패턴과 일치하는 프로세스의 정보를 출력함
- ☁ 형식: pgrep [옵션] [패턴]
- ☁ 옵션

-x	패턴과 정확히 일치하는 프로세스의 정보를 출력함
-n	패턴을 포함하고 있는 가장 최근 프로세스의 정보를 출력함
-u 사용자명	특정 사용자에게 대한 모든 프로세스를 출력함
-l	PID와 프로세스 이름을 출력함
-t term	특정 터미널과 관련된 프로세스의 정보를 출력함

- ☁ 사용 예

```
pgrep bash
```

- 🔍 bash 패턴을 지정하여 검색한 예

```
[user1@]localhost ~]$ pgrep -x bash
2115
[user1@]localhost ~]$
```

- ☁ pgrep의 경우 -l 옵션을 지정해도 단지 PID와 명령 이름만 출력
- ☁ 더 자세한 정보를 검색하려면 pgrep 명령을 ps 명령과 연결하여 사용
- ☁ -u 옵션으로 사용자명을 지정하여 검색



프로세스 관리 명령

4 프로세스 종료 및 관리도구



시그널

- 프로세스에 **무언가 발생했음**을 알리는 메시지
- 리눅스에서 지원하는 시그널 목록: **kill -l 명령**으로 알 수 있음
- 주로 사용하는 시그널

시그널	번호	기본 처리	의미
SIGHUP	1	종료	터미널과의 연결이 끊어졌을 때 발생함
SIGINT	2	종료	인터럽트로 사용자가 Ctrl +c를 입력하면 발생함
SIGQUIT	3	종료, 코어덤프	종료 신호로 사용자가 Ctrl +\를 입력하면 발생함
SIGKILL	9	종료	이 시그널을 받은 프로세스는 무시할 수 없으며 강제로 종료됨
SIGALRM	14	종료	알람에 의해 발생함
SIGTERM	15	종료	kill 명령이 보내는 기본 시그널임



프로세스 관리 명령

4 프로세스 종료 및 관리도구



kill 명령을 이용해 프로세스 종료하기

- 기능: 지정한 시그널을 프로세스로 보냄
- 형식: kill [-시그널] PID
- 시그널

2	인터럽트 시그널을 보냄(Ctrl +c)
9	프로세스를 강제로 종료함
15	프로세스와 관련된 파일을 정리한 후 종료하나 종료되지 않는 프로세스가 있을 수 있음

- 사용 예:

```
Kill 1001      Kill -15 1001   Kill -9 1001
```



pkill 명령을 이용해 프로세스 종료하기

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



프로세스 관리 명령

4 프로세스 종료 및 관리도구



프로세스 관리 도구

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

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

- top 명령의 내부 명령

내부 명령	기능
Enter Space Bar	화면을 즉시 다시 출력함
h,?	도움말 화면을 출력함
k	프로세스를 종료함, 종료할 프로세스의 PID를 물어봄
n	출력하는 프로세스 개수를 바꿈
q	top 명령을 종료함



프로세스 관리 명령 실습 영상은
학습 콘텐츠에서 확인하실 수 있습니다.



포그라운드·백그라운드 프로세스 및 작업 제어

1 포그라운드 작업



포그라운드 프로세스

- 사용자가 입력한 명령이 실행되어 **결과가 출력될 때까지 기다려야 하는** 포그라운드 방식으로 처리되는 프로세스



포그라운드 작업

- 포그라운드 프로세스를 작업 제어에서는 포그라운드 작업이라고 함

```
[user1@]localhost ~]$ sleep 100 → 포그라운드 작업  
→ sleep 명령이 끝날 때까지 기다려야 함
```

2 백그라운드 작업



백그라운드 프로세스

- 명령을 실행하면 명령의 처리가 끝나는 것과 관계없이 **곧바로 프롬프트가 출력**되어 사용자가 다른 작업을 계속할 수 있음



백그라운드 작업

- 백그라운드 프로세스를 작업 제어에서는 백그라운드 작업이라고 함

```
[user1@]localhost ~]$ sleep 100 & → 백그라운드 작업  
[user1@]localhost ~]$ → 프롬프트가 바로 나와 다른 명령을  
실행할 수 있음
```



포그라운드·백그라운드 프로세스 및 작업 제어

3 작업 제어



작업 전환, 작업 일시 중지, 작업 종료

작업 전환	작업 일시 중지	작업 종료
포그라운드 작업 ↓ 백그라운드 작업, 백그라운드 작업 ↓ 포그라운드 작업으로 전환	작업을 잠시 중단	프로세스를 종료하는 것처럼 작업을 종료



작업 목록 보기: jobs

• jobs 명령의 출력 정보

항목	출력 예	의미
상태	실행 중	작업 상태를 표시함 • Running: 현재 실행 중 • Done: 작업이 정상적으로 종료됨 • Terminated: 작업이 비정상적으로 종료됨 • Stopped: 작업이 잠시 중단됨
명령	sleep 80 &	백그라운드로 실행 중인 명령



포그라운드·백그라운드 프로세스 및 작업 제어

3 작업 제어



작업 전환하기



작업 전환 명령

Ctrl +z 또는
stop %작업 번호



포그라운드 작업을 중지함
(종료하는 것이 아니라 잠시 중단하는 것)

bg %작업 번호



작업 번호가 지시하는 작업을
백그라운드 작업으로 전환함

fg %작업 번호



작업 번호가 지시하는 작업을
포그라운드 작업으로 전환함



작업전환 예: 포그라운드 -> 백그라운드

```
[user1@]localhost ~]$ jobs
[user1@]localhost ~]$ sleep 80
^Z
[1]+  Stopped                  sleep 80
[user1@]localhost ~]$ bg %1
[1]+  sleep 80 &
[user1@]localhost ~]$ jobs
[1]+  Running                  sleep 80 &
[user1@]localhost ~]$
```

→ 백그라운드 작업이 없음
→ 포그라운드로 실행함
→ ctrl+Z로 일시 중지함
→ 일시 중지된 상태임
→ 백그라운드로 전환함

→ 백그라운드로 실행 중임



작업전환 예: 백그라운드 -> 포그라운드

```
[user1@]localhost ~]$ jobs
[1]+  Running                  sleep 80 &
[user1@]localhost ~]$ fg
sleep 80
```

→ 포그라운드로 전환함
→ 포그라운드로 실행 중임



포그라운드·백그라운드 프로세스 및 작업 제어

3 작업 제어



작업 종료하기: Ctrl+c

- 포그라운드 작업은 Ctrl+c를 입력하면 **대부분 종료됨**
- 백그라운드 작업은 kill 명령으로 강제 종료: **PID 또는 '%작업 번호'**



로그아웃 후에도 백그라운드 작업 계속 실행하기: nohup

- 로그아웃한 다음에도 작업이 완료될 때까지 백그라운드 작업을 실행해야 할 경우 nohup 명령을 사용
- **형식: nohup 명령&**



포그라운드·백그라운드 프로세스 및 작업 제어 실습 영상은 학습 콘텐츠에서 확인하실 수 있습니다.



작업 예약

특정한 시간에 작업을 수행하도록
예약할 수 있는 두 가지 **방법**

정해진 시간에
한 번 수행

정해진 시간에
반복 수행



작업 예약

1 at 명령



정해진 시간에 한 번 실행: at

- 기능: 예약한 명령을 정해진 시간에 실행함
- 형식: at [옵션] [시간]
- 옵션

-l	현재 실행 대기 중인 명령의 전체 목록을 출력함(atq 명령과 동일)
-r 작업 번호	현재 실행 대기 중인 명령 중 해당 작업 번호를 삭제함(atrm과 동일)
-m	출력 결과가 없더라도 작업이 완료되면 사용자에게 메일로 알려줌
-f 파일	표준 입력 대신 실행할 명령을 파일로 지정함

사용 예

```
at 11:00 pm
at 2:45 am May 11
```



작업 예약

1 at 명령



at 명령 설정하기

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

```
[user1@]localhost ch6]$ at 02:30 pm  
at>
```

- 시간을 지정하는 형식

at 5pm + 4 days	지금부터 4일 후 오후 5시에 작업을 수행함
at 11am Jul 15	7월 15일 오전 11시에 작업을 수행함
at 3am tomorrow	내일 오전 3시에 작업을 수행함
at 11:30am today	오늘 오전 11시 30분에 작업을 수행함

- at로 실행할 명령은 기본적으로 표준 입력으로 지정
 - 명령의 입력을 마치려면 ctrl+d 입력**



작업 예약

1 at 명령



at 작업 파일 확인하기

- at로 생성된 작업 파일은 **/var/spool/at** 디렉터리에 저장
- root 사용자만** /var/spool/at 디렉터리 내용 확인 가능



at 작업 목록 확인하기: -l 옵션, atq

- at 명령으로 설정된 작업의 목록은 **-l** 옵션으로 확인
- atq** 명령으로도 확인 가능



at 작업 삭제하기: -d 옵션

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



at 작업 삭제하기: atrm

- atrm 명령은 **at -d**와 같은 기능 수행
- 기능: 지정된 작업 번호의 작업을 삭제함
- 형식: atrm 작업 번호



작업 예약

1 at 명령



at 명령 사용 제한하기: /etc/at.allow와 /etc/at.deny

- ☁ /etc/at.allow 파일과 /etc/at.deny 파일에는 한 줄에 사용자 이름을 하나씩만 기록

/etc/at.allow 파일이 있는 경우	/etc/at.allow 파일이 없는 경우	두 파일이 모두 없는 경우
이 파일에 있는 사용자만 at 명령을 사용 가능	/etc/at.deny 파일에 지정된 사용자를 제외한 모든 사용자가 at 명령을 사용 가능	root만 at 명령을 사용 가능

- ☁ 한 사용자가 두 파일 모두에 속해 있다면
→ 그 사용자는 /etc/at.allow 파일이 적용되기 때문 at 명령을 사용 가능
- ☁ /etc/at.deny를 빈 파일로 두면(초기 설정)
→ 모든 사용자가 at 명령을 사용 가능
- ☁ at.deny 파일에 user1 사용자가 기록되어 있다면
→ at 명령을 실행했을 때 사용 권한이 없다는 메시지가 출력됨



작업 예약

2 crontab 명령



정해진 시간에 반복 실행: crontab

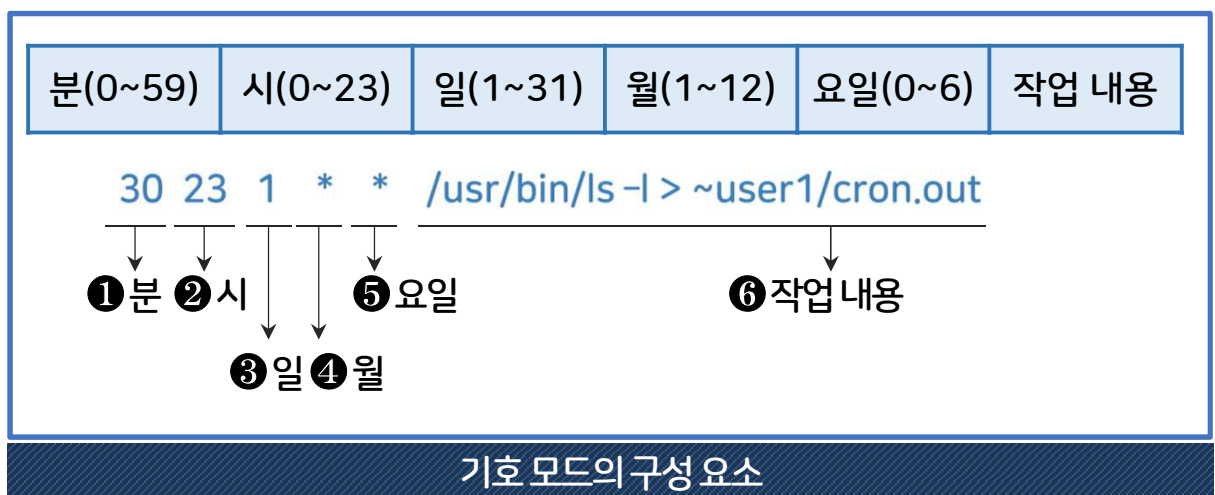
- 기능: 사용자의 crontab 파일을 관리함
- 형식: crontab [-u 사용자 ID] [옵션] [파일명]
- 옵션

-e	사용자의 crontab 파일을 편집함
-l	crontab 파일의 목록을 출력함
-r	crontab 파일을 삭제함

- 사용 예

```
crontab -l      crontab -u user2 -e      crontab -r
```

- 파일 형식





작업 예약

2 crontab 명령



crontab 파일 생성하고 편집하기: `crontab -e`



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

```
[user1@]localhost ch6]$ EDITOR=vi ;export EDITOR
```



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



crontab 파일 내용 확인하기: `crontab -l`



crontab 파일 삭제하기: `crontab -r`



작업 예약

2 crontab 명령



crontab 명령 사용 제한하기: /etc/cron.allow, /etc/cron.deny






cron.deny 파일은 기본적으로 있지만 **cron.allow** 파일은 관리자가 만들어야 함

/etc/at.allow 파일이 있는 경우	/etc/cron.allow 파일이 없고 /etc/cron.deny 파일이 있는 경우	두 파일이 모두 없는 경우
이 파일에 있는 사용자만 crontab 명령을 사용 가능	이 파일에 사용자 계정이 없어야 crontab 명령을 사용 가능	시스템 관리자만 crontab 명령을 사용 가능





핵심요약

1 프로세스 관리 명령

-  프로세스는 현재 시스템에서 실행 중인 프로그램을 의미함
-  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



2 포그라운드·백그라운드 프로세스와 작업 제어

-  포그라운드 프로세스: 사용자가 입력한 명령이 실행되어 결과가 출력될 때까지 기다려야 하는 포그라운드 방식으로 처리되는 프로세스
-  백그라운드 프로세스: 명령을 실행하면 명령의 처리가 끝나는 것과 관계없이 곧바로 프롬프트가 출력되어 사용자가 다른 작업을 계속할 수 있음



핵심요약

3 **작업 예약**

-  at 명령을 사용하여 정해진 시간에 명령을 실행하도록 예약하려면 at 명령 뒤에 시간을 명시함
-  crontab 명령: 정해진 시간에 반복 실행