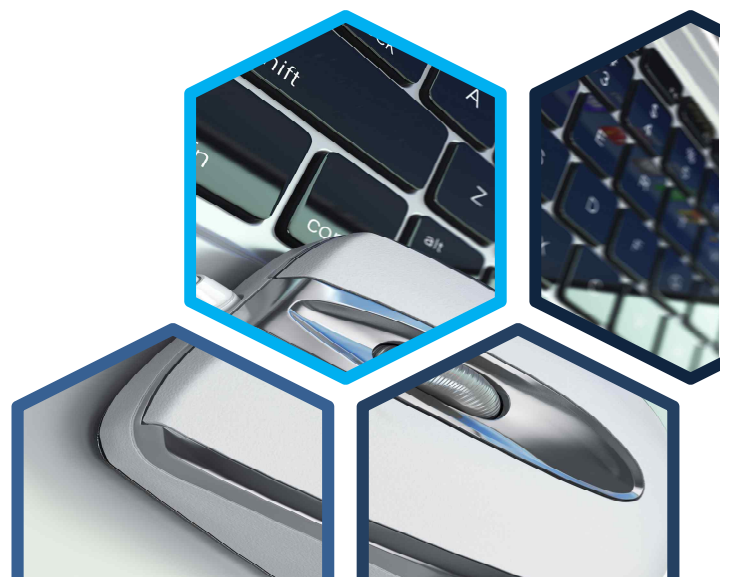


컴퓨터시스템

리눅스의 부팅





학습목표

- 리눅스 시스템의 부팅에 대해 설명할 수 있다.
- systemd 서비스에 대해 설명할 수 있다.
- 시스템 종료에 대해 설명할 수 있다.
- 데몬 프로세스에 대해 설명할 수 있다.



학습내용

- 리눅스 시스템의 부팅
- systemd 서비스
- 시스템 종료
- 데몬 프로세스

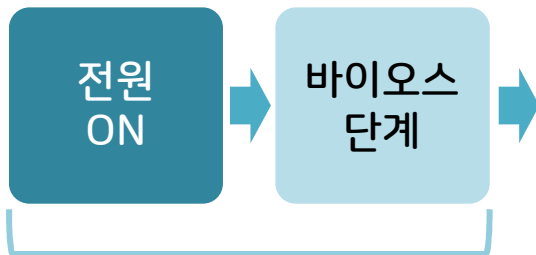


리눅스 시스템의 부팅

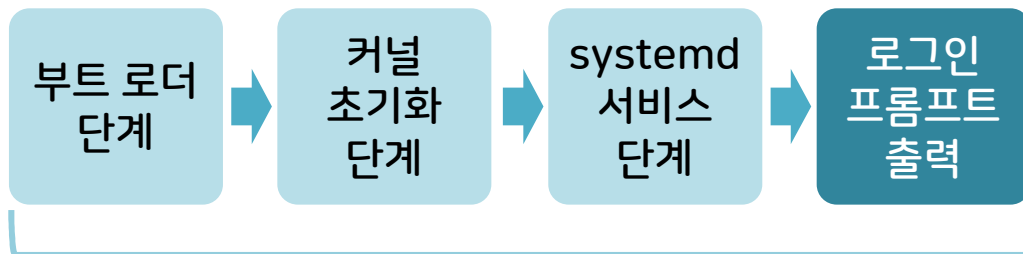
1 바이오스 단계와 부트 로더 단계



리눅스 시스템의 부팅 과정



PC 부팅 리눅스의 부팅 과정



리눅스 부팅

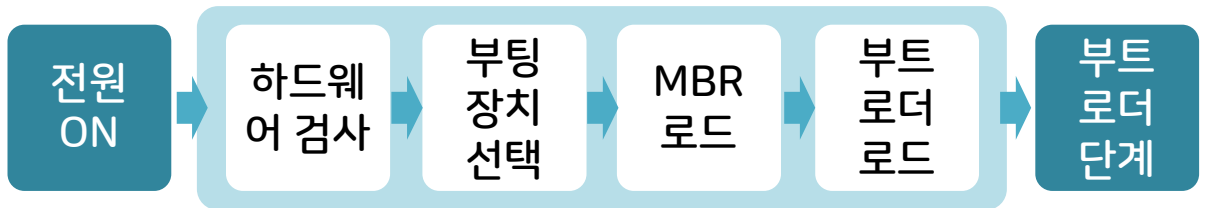


리눅스 시스템의 부팅

1 바이오스 단계와 부트 로더 단계



바이오스 단계



- 부팅 시 **바이오스**(BIOS, Basic Input/Output System) 동작
- PC에 장착된 기본적인 하드웨어(키보드, 디스크 등)의 상태를 확인
- 부팅 장치 선택 후 **부팅 디스크의 첫 섹터에서 512바이트를 로딩**
- 마스터 부트 레코드(Master Boot Record, MBR):
2차 부팅 프로그램(부트 로더)의 위치 저장



부트 로더 단계

- 바이오스 단계에서 **MBR**은 부트 로더를 찾아 메모리에 로딩
- 여러 운영체제 중에서 부팅할 운영체제를 선택할 수 있도록 **메뉴를 제공**
- 리눅스 커널을 **메모리에 로딩**
- 리눅스 커널은 /boot 디렉터리 아래에 '**vmlinuz-버전명**'의 형태로 제공
- 리눅스의 대표적인 부트 로더
 - GRUB와 LILO**
 - 우분투에서는 GRUB 사용



리눅스 시스템의 부팅

2 커널 초기화 및 systemd 서비스 단계



커널 초기화 단계

- 가장 먼저 시스템에 연결된 메모리, 디스크, 키보드, 마우스 등 **장치 검사**
- 장치 검사 등 기본적인 초기화 과정 후
fork를 사용하지 않고 생성되는 프로세스와 스레드 생성
- 프로세스들은 메모리 관리 같은 커널의 여러 가지 동작을 수행
- 프로세스는 일반적인 프로세스와 구분되도록 **대괄호([])**로 표시하며,
주로 **PID 번호가 낮게 배정**



systemd 서비스 단계

- 기존의 init 스크립트를 대체한 것으로** 다양한 서비스를 동작시킴
- 각 서비스가 시작하는 과정은 화면에 메시지로 출력됨
- 부트 스플래시 화면이 바로 종료되고 **서비스가 정상적으로 시작하는지(Ok)**
또는 실패인지(Fail)를 알려주는 메시지가 출력되는 화면으로 전환됨



리눅스 시스템의 부팅

3 부팅 후 과정

01 부팅 후 메시지 확인

- `dmesg` 명령 또는 `more /var/log/boot.log` 명령으로 확인

02 1번 프로세스 실행

- 유닉스에서는 `init` 프로세스가 처음 생성된 프로세스로서 PID가 1번이나
우분투 15.04버전부터 `systemd`를 사용함

03 로그인 프롬프트 출력

- 그래픽 로그인 시스템인 `GDM(GNOME display manager)`을 동작 후
로그인 프롬프트 출력



systemd 서비스

1 init 프로세스와 런레벨(Runlevel)



init 프로세스

- 전통적인 유닉스 시스템에서 부팅 시 가장 먼저 생성되는 초기 프로세스
- 현재 **systemd**로 대체되어 사용하지 않음

upstart와 관련된 스크립트 파일

etc/init 디렉터리에
'작업명.conf' 파일로 구성

init와 관련된 스크립트 파일

/etc/init.d 디렉터리에 있으며
아직 일부 서비스의
스크립트 파일이 남아 있음



런레벨(Runlevel)

- 시스템 관리의 용이함을 위해 각 부팅 환경을 레벨 단위로 나누어 놓은 것
- 시스템의 단계를 0~6까지의 런레벨로 구성
- 각 단계에 따라 셸 스크립트를 실행함



systemd 서비스

1 init 프로세스와 런레벨(Runlevel)



우분투의 런레벨

런레벨	의미	관련 스크립트의 위치
0	시스템 종료	/etc/rc0.d
1, S	응급 복구 모드(단일 사용자 모드)	/etc/rc1.d, /etc/rcS.d
2	다중 사용자 모드	/etc/rc2.d
3		/etc/rc3.d
4		/etc/rc4.d
5	그래피컬 다중 사용자 모드	/etc/rc5.d
6	재시작	/etc/rc6

디렉터리의
내용이 모두
같음

- 런레벨별로 실행하는 스크립트 파일은 **/etc/init.d** 디렉터리에 있는
파일에 대한 심벌릭 링크임



systemd 서비스

2 systemd 서비스



systemd의 장점

- init 방식에 비해 다음과 같은 **장점**을 가지고 있음
 - ① 소켓 기반으로 동작하여 inetd와 호환성을 유지함
 - ① 셸과 독립적으로 부팅이 가능함
 - ① 마운트 제어가 가능함
 - ① fsck 제어가 가능함
 - ① 시스템 상태에 대한 스냅샷을 유지함
 - ① SELinux와 통합이 가능함
 - ① 서비스에 시그널을 전달 가능함
 - ① shutdown 전에 사용자 세션의 안전한 종료가 가능함



systemd 유닛

- **전체 시스템을 시작하고 관리**하는 데 유닛(unit)이라 부르는 구성 요소를 사용
- systemd는 관리 대상의 이름을 '**서비스명.유닛 종류**'의 형태로 관리
- 각 유닛은 같은 이름과 종류로 구성된 설정 파일과 동일한 이름을 사용
- 유닛과 관련한 보다 자세한 내용 확인 방법
 - ① **man systemd.유닛명**



systemd 서비스

2 systemd 서비스



systemd 유닛의 종류

유닛	기능	예
service	가장 명백한 유닛으로 데몬을 시작·종료·재시작·로딩함	atd.service
socket	소켓을 관리하는 유닛으로 AF_INET, AF_INET6, AF_UNIX 소켓 스트림과 데이터그램, FIFO를 지원함	abus.socket
device	리눅스 장치 트리에 있는 장치를 관리함	sys-module-fuse.device
mount	디렉터리 계층 구조의 마운트 포인트를 관리함	boot.mount
automount	디렉터리 계층 구조에서 자동 마운트 포인트를 관리함	proc-sys-fs-binfmt_misc.automount
target	유닛을 그룹핑함 (예: multi-user.target → runlevel 5에 해당하는 유닛)	basic.target multi-user.target
swap	스왑 장치를 관리함	dev-mapper-fedora\x2dswap.swap
path	경로를 관리함	cups.path
timer	타이머와 관련된 기능을 관리함	dnf-makecache.timer
slice	프로세스 그룹의 자원을 계층적으로 관리함	system-getty.slice
scope	외부에서 생성된 프로세스를 관리함	init.scope



systemd 서비스

2 systemd 서비스



systemd 관련 명령: **systemctl**

- systemd 기반으로 서비스를 시작하거나 종료할 때 사용하는 명령
- 기능: systemd를 제어함
- 형식: systemctl [옵션] [명령] [유닛명]
- 옵션

-a	상태와 관계 없이 유닛 전체를 출력함
-t 유닛 종류	지정한 종류의 유닛만 출력함

명령

start	유닛을 시작함
stop	유닛을 정지함
reload	유닛의 설정 파일을 다시 읽어옴
restart	유닛을 재시작함
enable	부팅 시 유닛이 시작하도록 설정함
disable	부팅 시 유닛이 시작하지 않도록 설정함
is-active	유닛이 동작하고 있는지 확인함
is-enabled	유닛이 시작되었는지 확인함
isolate	지정한 유닛 및 이와 관련된 유닛만 시작하고 나머지는 정지함
kill	유닛에 시그널을 전송함

사용 예

```
systemctl
systemctl -a
systemctl start atd.service
```



systemd 서비스

2 systemd 서비스



동작 중인 유닛 출력하기

- 옵션이나 명령 없이 systemctl 명령만 사용하면 **현재 동작 중인 유닛이 출력됨**
- systemctl 명령에서 사용하는 옵션

-a	전체 유닛이 출력됨
-t	특정 종류의 유닛만 출력됨



유닛 서비스 시작하기: start 명령

- 유닛 서비스를 시작하려면 start 명령을 사용
- cron 유닛을 시작한 후 **is-active** 명령으로 동작 여부를 확인해보면 **active 상태임**
- 유닛의 상태를 확인하려면 **status** 명령을 사용



유닛 서비스 정지하기: stop 명령

- 유닛 서비스를 정지하려면 stop 명령을 사용
- cron 유닛을 정지한 후 다시 **status** 명령으로 상태를 확인해보면 **inactive(dead) 상태임**



systemd 서비스

2 systemd 서비스



systemd와 런레벨

- 런레벨은 현재 시스템의 상태를 나타내는 한 자리 숫자(문자 S, s 포함)
- 런레벨에 대응하는 systemd의 target 유닛 제공
 - 이 파일들은 /lib/systemd/system 디렉터리에 있음
- 현재 런레벨 확인하기: **runlevel** 명령
- 런레벨과 target 유닛의 관계

런레벨	target 파일(심벌릭 링크)	target 원본 파일
0	runlevel0.target	power off.target
1	runlevel1.target	rescue.target
2	runlevel2.target	multi-user.target
3	runlevel3.target	
4	runlevel4.target	
5	runlevel5.target	graphical.target
6	runlevel6.target	reboot.target



systemd 서비스

2 systemd 서비스



기본 target 지정하기

- 부팅할 때 동작할 기본 runlevel은 기본 target으로 바뀌었고, 다음과 같은 형식으로 지정

```
systemctl set-default <name or target>.target
```

- /etc/systemd/system 디렉터리 아래에 심벌릭 링크인 default.target이 가리키는 target 파일을 변경
- 현재 target인 graphical.target에서 multi-user.target으로 바꾸는 예

```
user1@myubuntu:~$ sudo systemctl set-default multi-user.target
```



target 변경하기

- systemd에서 runlevel을 변경하는 것도 isolate 명령으로 간단히 해결
- multi-user.target(runlevel 3)으로 변경하려면 다음 명령 중 하나를 입력

```
systemctl isolate multi-user(확장자 .target은 생략 가능)
```

```
systemctl isolate runlevel3(확장자 .target은 생략 가능)
```

- graphical.target(runlevel 5)으로 변경하려면 다음 명령 중 하나를 사용

```
systemctl isolate graphical(확장자 .target은 생략 가능)
```

```
systemctl isolate runlevel5(확장자 .target은 생략 가능)
```



systemd 서비스

2 systemd 서비스



telinit, init 명령으로 런레벨 변경하기

- init은 init 프로세스의 runlevel을 바꿀 때 사용하기도 함
- 현재 init는 **systemd에 대한 심벌릭 링크**

```
user1@myubuntu:~$ ls -l /sbin/init
```

- init 명령만 입력하면 다음과 같이 출력

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

- runlevel을 바꾸는 명령으로 **telinit**

```
user1@myubuntu:~$ ls -l /sbin/telinit
```




systemd 서비스

2 systemd 서비스



단일 사용자 모드로 전환하기: **rescue.target(runlevel 1)**

- 시스템에 문제가 있을 경우 시스템을 **rescue.target** 유닛(runlevel 1, runlevel S)으로 변경하여 점검
- 윈도우의 안전 모드 같은 것으로 다중 사용자 모드에서 시스템 관리자만 사용할 수 있는 단일 사용자 모드로 전환하는 것

```
systemctl isolate rescue  
(확장자 .target은 생략 가능)
```

```
init 1
```

```
systemctl isolate runlevel1  
(확장자 .target은 생략 가능)
```

```
telinit S
```

→ 단일 사용자 모드로 전환 시 그래픽 환경이 **텍스트 모드**로 바뀜



systemd 서비스 실습 영상은
학습 콘텐츠에서 확인하실 수 있습니다.



시스템 종료

1 shutdown 명령



리눅스 종료하기: shutdown

- 기능: 리눅스를 종료함(리눅스 시스템을 가장 정상적으로 종료하는 방법)
- 형식: shutdown [옵션] [시간] [메시지]
- 옵션

-k	실제로 시스템을 종료하는 것이 아니라 사용자들에게 메시지만 전달함
-r	종료 후 재시작함
-h	종료하며 halt나 power-off 상태로 이동함
-c	이전에 했던 shutdown 명령을 취소함
시간	종료할 시간임(hh:mm, +m, now)
메시지	모든 사용자에게 보낼 메시지임

- 사용 예

```
shutdown -h now    shutdown -c  
shutdown -r +3 "System is going down"
```



시스템 종료

1 shutdown 명령



shutdown 명령으로 시스템 즉시 종료하기

- ☁ -h 옵션과 함께 현재 시간으로 지정

```
user1@myubuntu:~$ sudo shutdown -h now
```



shutdown 메시지 보내고 종료하기

- ☁ 시스템을 종료할 때 shutdown 명령으로 메시지를 보낼 수 있음
- ☁ 사용자들이 메시지를 받고 정리할 시간이 필요하므로 시간을 now로 지정하면 안 되고 특정 시간을 지정해야 함
- ☁ 2분 후에 종료한다는 메시지를 발송할 경우

예

```
▶ user1@myubuntu:~$ sudo shutdown -h +2 "System is going down in 2 min."
```

- ☁ 사용자 터미널에 아래의 메시지가 출력됨

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

```
Broadcast message from root@myubuntu on pts/0 (Sat 2019-11-18 22:04:49 KST):
```

```
System is going down in 2 min.
```

```
The system is going down for poweroff at Sat 2020-11-18 22:06:49 KST!
```



시스템 종료

2 런레벨 변경이나 systemd를 이용한 종료



런레벨 변경하여 종료하기

- 시스템을 종료: 런레벨을 0으로 변경

```
user1@myubuntu:~$ sudo init 0
```

- 시스템 재시작: 런레벨을 6으로 변경

```
user1@myubuntu:~$ sudo init 6
```



systemd로 종료하기

- systemd에서 target 유닛을 변경하면 시스템을 종료 또는 재시작 가능
- 시스템 종료하기

```
systemctl isolate poweroff.target
```

```
systemctl isolate runlevel0.target
```

- 시스템 재시작하기

```
systemctl isolate reboot.target
```

```
systemctl isolate runlevel6.target
```



시스템 종료

3 기타 시스템 종료



기타 시스템 종료 명령: **reboot, halt, poweroff**

- 시스템을 종료하거나 재시작하기 위해 사용할 수 있는 명령
- 모두 systemctl의 심벌릭 링크
- /var/log/wtmp** 파일에 시스템 종료 기록을 남기고 시스템을 종료하거나 재시작함
- 사용할 수 있는 옵션

-n	재시작이나 종료 전에 sync를 호출하지 않음
-w	실제로 재시작하거나 종료하지는 않지만 wtmp 파일에 기록을 남김
-d	wtmp 파일에 기록을 남기지 않음(-n 옵션은 -d 옵션을 포함함)
-f	강제로 명령을 실행하며 shutdown을 호출하지 않음
-p	시스템의 전원을 끄



데몬 프로세스

1 개요



데몬(daemon)

- 리눅스의 백그라운드에서 동작하면서 특정한 서비스를 제공하는 프로세스를 의미함



슈퍼 데몬

- 유닉스에서 슈퍼 데몬의 이름은 `inetd`임
- 우분투에서는 보안 기능이 포함된 `xinetd`를 사용



데몬의 동작 방식

- 독자형(standalone)
 - 시스템의 백그라운드에서 서비스별로 항상 동작함
 - 자주 호출되는 데몬이 아니라면 시스템의 자원을 낭비할 우려가 있음
- 슈퍼 데몬에 의한 동작 방식
 - 평소에는 슈퍼 데몬만 동작하다가 서비스 요청이 오면 슈퍼 데몬이 해당 데몬을 동작시킴
 - 독자형보다는 서비스에 응답하는 데 시간이 약간 더 걸릴 수 있지만 자원을 효율적으로 사용한다는 것이 장점임



데몬 프로세스

2 데몬의 조상

systemd 데몬

커널 스레드 데몬



데몬 프로세스

2 데몬의 조상



systemd 데몬

- 대부분 프로세스의 조상 프로세스
- ps tree 명령으로 프로세스의 실행구조를 확인할 수 있음

```
user1@myubuntu:~$ pstree
systemd───ModemManager───2*[{ModemManager}]
          │
          └─NetworkManager───dhclient
                        │
                        └─NetworkManager}]
          │
          └─-daemon───2*[{accounts-daemon}]
          │
          └─acpid
          │
          └─atd
```

(생략)



데몬 프로세스

2 데몬의 조상



커널 스레드 데몬

- ☁ 커널의 일부분을 프로세스처럼 관리하는 데몬
- ☁ ps 명령으로 확인했을 때 **대괄호([])**로 둘러싸여 있는 프로세스들임
- ☁ 예전에는 대부분 k로 시작했으나 요즘은 이를 반드시 준수하지는 않음
- ☁ 커널 데몬은 대부분 **입출력이나 메모리 관리, 디스크 동기화** 등을 수행하며 대체로 PID가 낮은 번호로 할당
- ☁ 커널 데몬을 동작시키는 조상 데몬은 커널 스레드 데몬(kthreadd): **PID 2번**

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	21:56	?	00:00:01	/sbin/init splash
root	2	0	0	21:56	?	00:00:00	[kthreadd]
root	3	2	0	21:56	?	00:00:00	[kworker/0:0]

(생략)



데몬 프로세스

2 데몬의 조상



리눅스의 주요 데몬

데몬	기능
atd	특정 시간에 실행하도록 예약한 명령을 실행함(at 명령으로 예약)
cron	주기적으로 실행하도록 예약한 명령을 실행함
dhcpcd	동적으로 IP 주소를 부여하는 서비스를 제공함
httpd	웹 서비스를 제공함
lpd	프린트 서비스를 제공함
nfs	네트워크 파일 시스템 서비스를 제공함
named	DNS 서비스를 제공함
sendmail	이메일 서비스를 제공함
smtpd	메일 전송 데몬
popd	기본 편지함 서비스를 제공함
routed	자동 IP 라우터 테이블 서비스를 제공함
smb	삼바 서비스를 제공함
syslogd	로그 기록 서비스를 제공함
sshd	원격 보안 접속 서비스를 제공함
in.telnetd	원격 접속 서비스를 제공함
ftpd	파일 송수신 서비스를 제공함
ntpd	시간 동기화 서비스를 제공함



📌 핵심요약

1 리눅스 시스템의 부팅

- 📌 리눅스 시스템의 부팅과정은 PC 부팅과 리눅스 부팅으로 구분되며 PC 부팅은 전원 ON, 바이오스 단계로 구성됨
- 📌 PC 부팅이 끝나면 리눅스 부팅이 행해지며 부트로더단계, 커널 초기화단계, systemd 서비스 단계로 이루어짐
- 📌 systemd 서비스 단계가 끝나면 로그인 프롬프트를 출력함으로써 리눅스 부팅이 성공적으로 이루어짐



2 systemd 서비스

- 📌 init 프로세스와 runlevel란 파일 시스템을 디렉터리 계층 구조에 마운트하는 것을 의미
- 📌 systemd 서비스은 독립적으로 구성된 디스크 파티션을 하나로 연결하여 한 파티션처럼 사용할 수 있도록 해줌






핵심요약

3 시스템 종료

-  리눅스 시스템 종료하기 명령: shutdown
-  기타 시스템 종료 명령: reboot, halt, poweroff

4 데몬 프로세스

-  리눅스의 백그라운드에서 동작하면서 특정한 서비스를 제공하는 프로세스
-  systemd 데몬: 대부분 프로세스의 조상 프로세스
-  커널 스레드 데몬: 커널의 일부분을 프로세스 처럼 관리하는 데몬