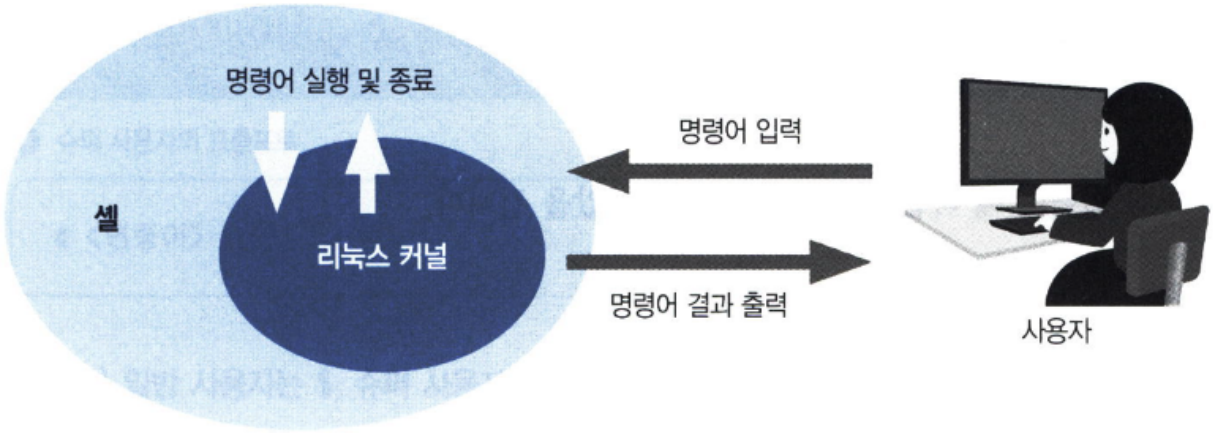
셸의 역할

- 리눅스에서는 사용자가 커널을 직접 조작할 수 없기 때문에 커널이 명령어를 받아들이고 커널의 실행 결과를 출력하는 소프트웨어가 필요
- 이 역할을 수행하는 소프트웨어가 셸임
- 셸은 사용자와 커널의 인터페이스 역할을 함

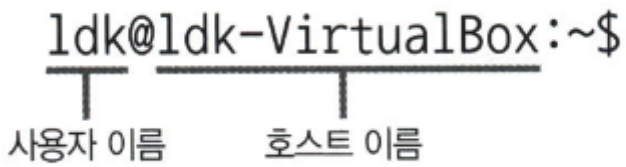


- 셸은 리눅스 커널을 감싸는 역할을 담당하기 때문에 셸이라는 이름이 붙여졌다고 볼 수 있음
- 리눅스를 다룰 때는 기본적으로 셸을 사용해야 하며, 리눅스를 잘 다루기 위해서는 셸을 잘 알아야 함

커널과 셸의 분리

- 커널과 셸은 분리되어 있음
- 커널을 운영체제 핵심 소프트웨어 이므로 교체하기 힘듬
- 셸은 사용자의 요구사항에 맞게 선택할 수 있음

프롬프트



- 셸의 프롬프트(prompt)라고 함
- 프롬프트는 사용자에게 어떤 결정을 내리도록 한다는 의미
- 즉, 셸이 사용자에게 명령어를 받아들일 준비가 되었음을 나타낸다고 보면 됨

프롬프트 기호

- 일반 사용자의 프롬프트 기호는 $ 로 표시
- 슈퍼 유저일 경우 프롬프트 기호는 # 로 표시

로그인 셸

- 사용자가 로그인할 때 리눅스가 자동으로 셸을 시작하기 때문에 리눅스에 로그인하면 셸이 사용자의 입력을 기다림
- 로그인 후 처음으로 시작되는 셸을 로그인 셸이라고 부름

셸 스크립트

- 명령어를 직접 입력하고 그 결과를 확인하는 조작 방식을 대화형(인터렉티브) 방식이라고 함
- 실행하고 싶은 명령어들을 미리 파일에 기록하고 그 파일을 셸에 넘겨주는 방식으로 명령을 수행할 수 있도록 만든 파일을 셸 스크립트라고 함
- 셸 스크립트는 명령어를 조합해 복잡한 처리를 수행한다는 리눅스의 철학이 담긴 강력한 도구
- 셸 스크립트를 익히려면 먼저 셸에 대한 기본 지식을 갖추어야 함

```
#!/bin/bash

today =$( date '+%d')
```

종류

- 리눅스에서 기본적으로 사용하는 셸은 배시(bash)를 사용
- 리눅스에서는 배시 외에도 다양한 셸을 사용할 수있음

sh

- AT&T 벨 연구소의 스티븐 본(Steven Bourne)이 만들어서 본 셸이라고 불림
- 아주 오래전에 만들어진 셸이며 리눅스 뿐만 아니라 유닉스 계열에서도 사용할 수 있음
- sh는 긴 역사를 통해 표준 셸의 지위를 가지고 있으며, 현재도 셸 스크립트를 작성할 때는 sh를 사용하는 것일 일반적임
- 하지만 오래된 셸이라 기능이 적고 특히 대화형에서 사용하기는 불편하기 때문에 일반 사용자가 사용하는 경우는 거의 없음

csh

- csh도 무척 오래된 셸 중 하나도 C셸로 불림
- sh보다 대화형 조작에 편리한 기능을 갖고 있어 인기가 많았지만 셸 문법이 sh와 달라 셸 스크립트 작성에는 적합하지 않음

- 현재는 csh의 뒤를 잇는 tcsh가 나와 많이 사용하지 않음

bash

- sh를 바탕으로 기능이 추가된 셸
- sh와 호환성이 있어 sh를 대체할 수 있고, 대화형 조작에 필요한 기능을 갖추고 있어 많은 리눅스에서 기본 로그인 셸로 사용하고 있음
- 셸 스크립트를 작성하는 데도 적합함

tsch

- csh에 이어 개발된 C셸 계열의 셸
- 대화형 조작에 편리한 기능을 많이 갖추고 있지만, csh와 마찬가지로 셸 스크립트에 적합하지 않음
- tcsh등 C셸 계열에서는 일반 사용자 프롬프트가 $이 아닌 %를 사용
- 맥OS도 10.2전까지는 기본 로그인 셸로 tcsh를 사용함
  44

zsh

- 비교적 최근에 개발된 셸로, bash와 tcsh의 기능에 독자적인 기능이 추가됨
- 무척 다양한 기능을 갖추고 있어 매뉴얼만 17개의 섹션에 달함
- 모든 기능을 익히는 데 시간이 걸리지만, 익숙해지면 작업 효율을 크게 높일 수 있음
- 다만 초보자가 바로 시작하기에는 다소 어려움이 있음

셸의 변경

- 셸도 하나의 명령어에 불과하므로 사용하고 싶은 셸 이름을 입력하고 실행만 하면 됨

```
hadoop@hadoop:~$ sh
$ bash
hadoop@hadoop:~$
```

터미널

- 터미널이란 컴퓨터의 입출력만을 담당하는 전용 하드웨어를 말함
- 입력 장치인 키보드와 출력 장치인 모니터로 구성
- 아주 오래전에 데이터 센터에는 간혹 입출력 기능만 갖춘 간이 단말기(dumb terminal)이 있었으나 현재는 거의 없음

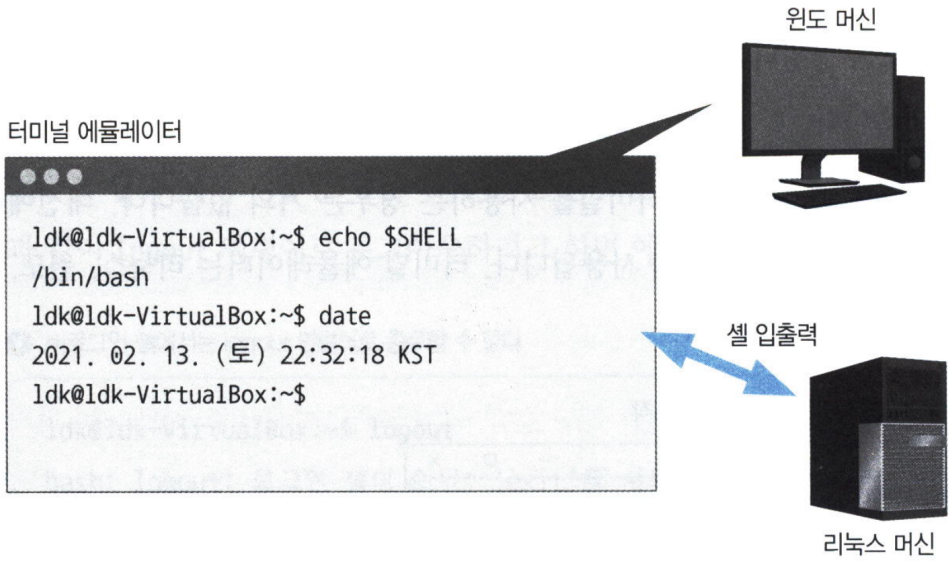
- 대신에 소프트웨어로 구현한 터미널 에뮬레이터가 사용되고 리눅스, 윈도, 맥 등에서 애플리케이션으로 동작

- 위의 그림은 윈도에서 터미널 에뮬레이터를 기동해 리눅스를 다루는 모습
- 터미널 에뮬레이터는 물리적인 터미널처럼 입출력을 위한 인터페이스를 제공
- 요즘에는 터미널이라고 하면 이러한 터미널 에뮬레이터를 말하는 경우가 많음

운영체제	터미널 에뮬레이터
Windows	Putty, Tera Term
macOS	terminal, iTerm2
Linux	GNOME Terminal, konsole

터미널과 셸

- 터미널 에뮬레이터와 셸은 완전히 다른 소프트웨어임



  - 터미널 에뮬레이터는 입출력 화면을 제공만 하는 소프트웨어
  - 위의 그림에서 리눅스 머신에서 돌아가는 셸의 입출력을 제공하는 소프트웨어가 바로 터미널 에뮬레이터

터미널을 편리하게 사용하는 팁

이전 명령어

- 리눅스를 사용하다 보면 이전에 실행한 커맨드 라인을 다시 실행하고 싶은 경우가 자주 생김
- 혹은 이전에 실행한 명령을 조금 수정해서 실행하고 싶은 경우도 있음
- bash는 한 번 입력한 커맨드 라인을 기록함
- 이러한 명령 이력 기능을 활용하면 이전에 입력한 명령들을 다시 불러올 수 있음

단축키	내용
Ctrl + p or 방향키 위	바로 전 명령으로 이동
Ctrl + n or 방향키 아래	다음 명령으로 이동
Ctrl + r	이력을 검색

- p는 previous
- n는 next

명령 검색

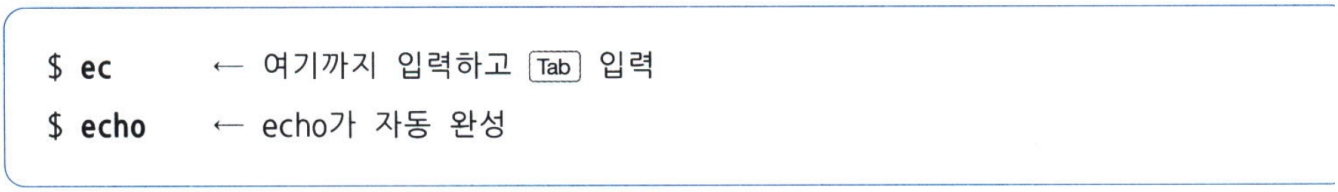
- 명령 이력을 하나씩 찾는 것은 비효율적
- Ctrl + r를 입력하면 명령 이력을 검색할 수 있음

```
(reverse-i-search)`':
```

- 증분 검색은 문자를 하나 입력할 때마다 이력을 검색
- 즉, 검색할 문자를 전부 입력한 뒤 Enter를 누르지 않아도 문자를 입력할 때마다 자동으로 검색 결과가 바뀜

tab 키

- 탭 키를 통한 자동 완성을 지원
- 명령을 실행하려면 명령어를 키보드로 직접 입력해야 하지만 자동 완성 기능을 사용하면 처음 몇 글자만 입력하면 자동으로 완성



- 명령어뿐만 아니라 경로에도 사용할 수 있음