



우분투 리눅스

시스템 & 네트워크

Chapter 09. 소프트웨어 관리하기

목차

00. 개요

01. 우분투 패키지의 개요

02. 우분투 패키지 설치

03. 스냅 패키지 설치

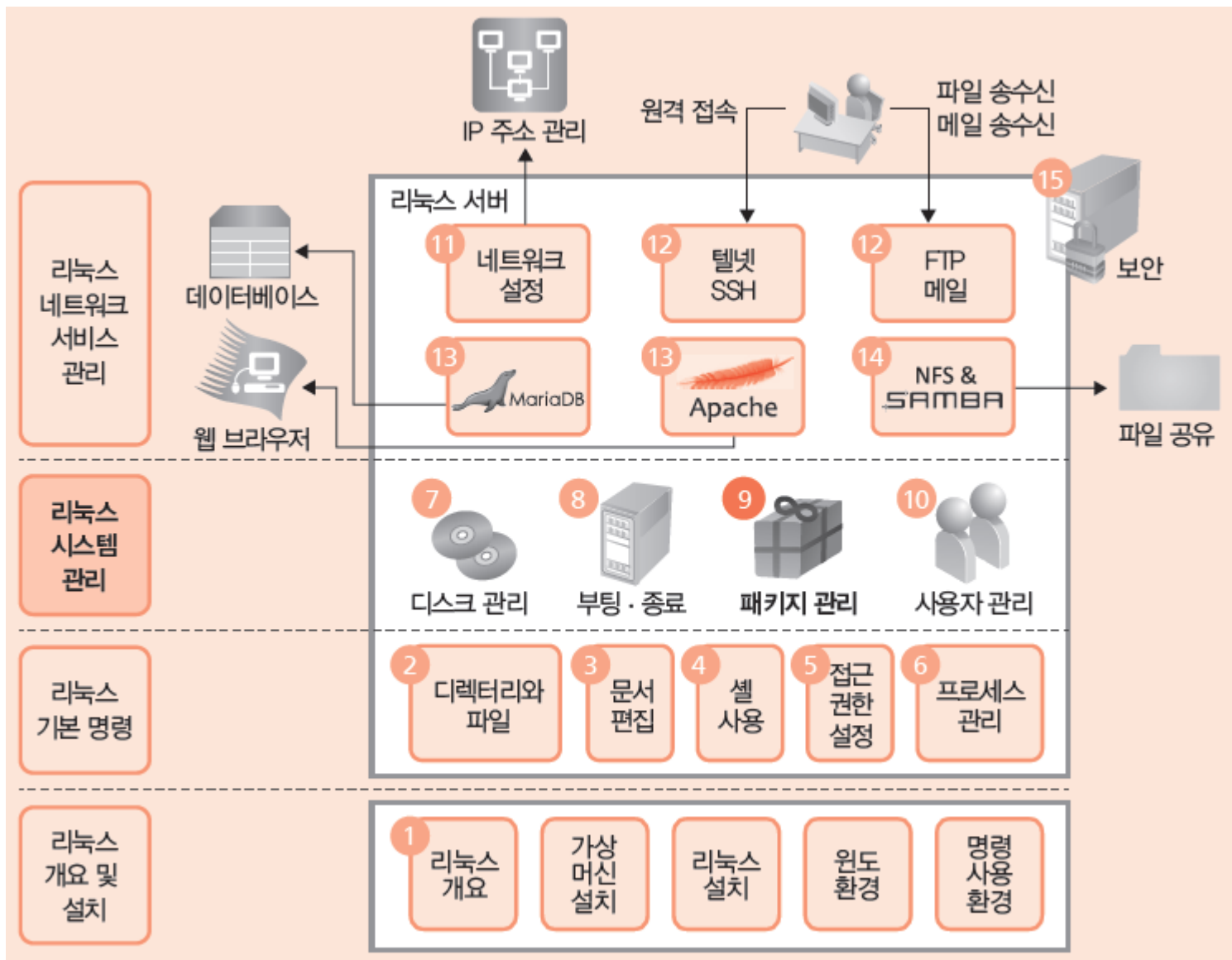
04. 파일 아카이브와 압축

05. 소프트웨어 컴파일

학습목표

- 우분투 패키지를 설치하고 업그레이드할 수 있다.
- APT 명령으로 패키지를 검색하고 상세 정보를 확인할 수 있다.
- dpkg 명령으로 패키지를 설치하고, 업그레이드하고 삭제할 수 있다.
- aptitude 명령으로 패키지를 관리할 수 있다.
- snap 패키지를 설치할 수 있다.
- 그놈 소프트웨어 센터에서 프로그램을 확인하고 설치할 수 있다.
- tar 명령으로 아카이브를 생성하고, 내용을 확인하고 풀 수 있다.
- 파일을 압축하고 압축을 풀 수 있다.
- gcc로 C 파일을 컴파일할 수 있다.
- makefile을 작성하여 make 명령으로 실행 파일을 만들 수 있다.

리눅스 실습 스터디 맵



00 개요

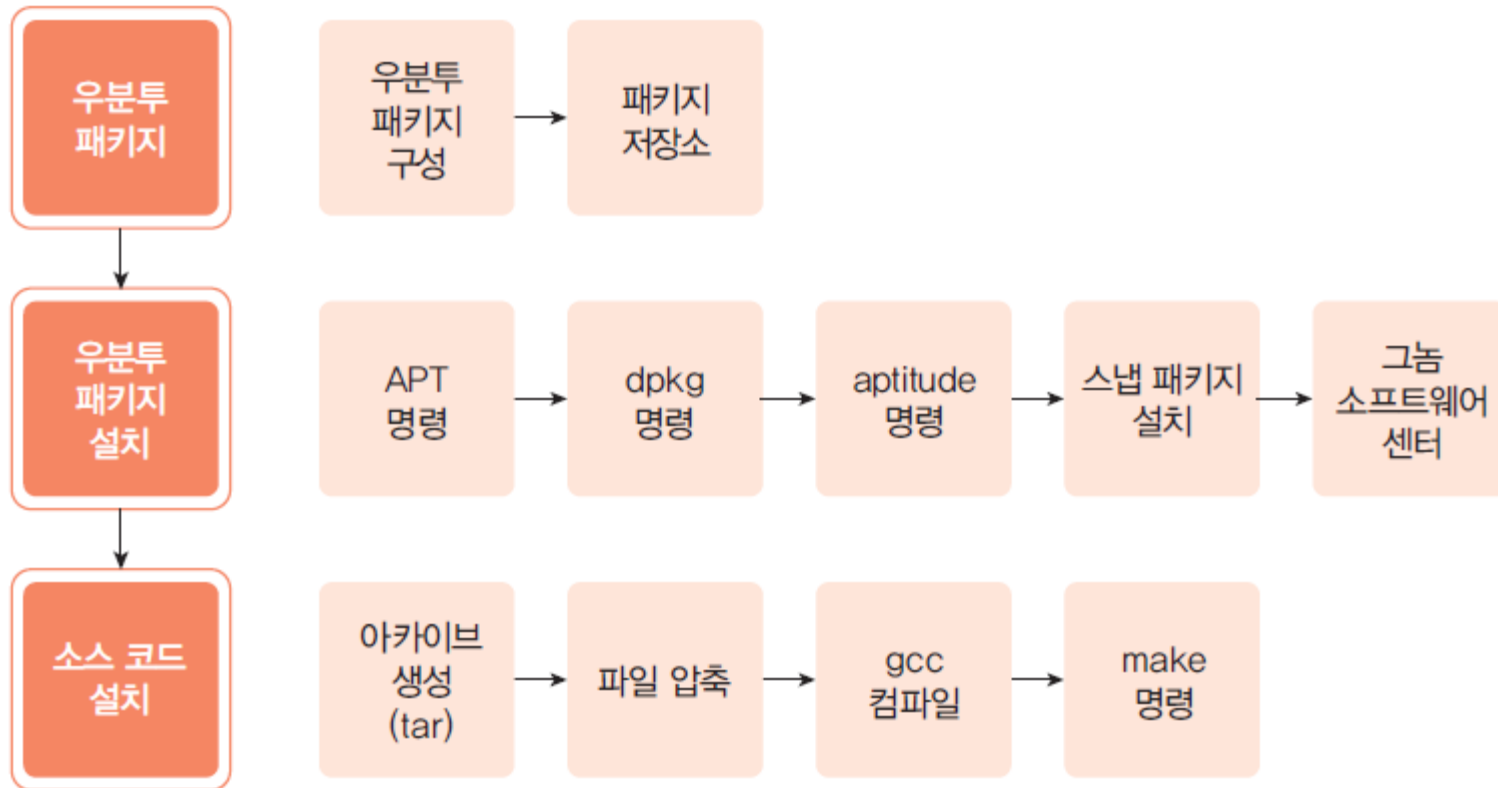


그림 9-1 9장의 내용 구성

01 우분투 패키지의 개요

■ 리눅스에서 주로 사용하는 패키지

- deb: 데비안, 우분투 계열에서 사용하는 패키지
- RPM(Redhat Package Manager): 레드햇 계열 리눅스에서 주로 사용

■ 우분투 패키지의 특징

- 바이너리 파일로 구성되어 있어 컴파일이 필요 없다.
- 패키지의 파일이 관련 디렉터리에 바로 설치된다.
- 패키지를 삭제할 때 관련된 파일을 일괄적으로 삭제할 수 있다.
- 기존에 설치한 패키지를 삭제하지 않고 바로 업그레이드할 수 있다.
- 패키지의 설치 상태를 검증할 수 있다.
- 패키지에 대한 정보를 제공한다.
- 해당 패키지와 의존성을 가지고 있는 패키지가 무엇인지 알려준다. 따라서 의존성이 있는 패키지를 미리 설치할 수도 있고, apt-get 명령을 사용하면 의존성이 있는 패키지가 자동으로 설치된다.

01 우분투 패키지의 개요

■ 우분투 패키지의 카테고리

- 공식적으로 데비안 배포판에 포함된 모든 패키지는 데비안 자유 소프트웨어 지침에 따라 자유롭게 사용하고 배포할 수 있음
- 우분투도 네 개의 카테고리로 나누어 소프트웨어를 제공
 - main: 우분투에 의해 공식적으로 지원되며 자유롭게 배포할 수 있다.
 - restricted: 우분투에 의해 지원되나 완전한 자유 라이선스 소프트웨어는 아니다.
 - universe: 리눅스에서 사용할 수 있는 거의 대부분의 소프트웨어로 자유 소프트웨어일 수도 있고 아닐 수도 있으며, 기술적 지원을 보장하지 않는다.
 - multiverse: 자유 소프트웨어가 아닌 소프트웨어가 포함되어 있으며, 개인이 직접 라이선스를 확인해야 한다.

■ 우분투 패키지의 이름 구성

파일명_버전-리비전_아키텍처.deb

그림 9-2 우분투 패키지의 이름 구성

- 파일명: 첫 번째 항목은 패키지의 성격을 나타내는 파일명이다.
- 패키지 버전: 두 번째 항목은 패키지의 버전을 의미한다.
- 패키지 리비전: 리비전은 원래 소스의 버전이 업그레이드되지는 않았지만 패키지의 보안 문제나 의존성 변화, 스크립트의 변화 등이 있음을 의미한다.
- 아키텍처: 사용하는 시스템 아키텍처로 i386은 인텔을, all은 시스템과 상관없는 문서나 스크립트 등을 뜻한다.
- 확장자: 확장자는 .deb를 사용한다.

01 우분투 패키지의 개요

■ 우분투 패키지 저장소

- 우분투는 패키지와 패키지에 대한 정보를 저장하고 있는 서버인 패키지 저장소라는 개념을 사용
- 패키지 저장소에서는 패키지의 기능 추가나 보안 패치 등 지속적인 업그레이드를 집중적으로 관리
- 사용자는 저장소에 접속하여 최신 패키지를 내려받아 설치 가능
- 패키지 저장소에 대한 정보는 `/etc/apt/sources.list` 파일에 저장
 - 패키지 유형 : `deb`는 바이너리 패키지의 저장소를, `deb-src`는 패키지의 소스 저장소를 의미한다. 보통 한 저장소에 바이너리와 소스를 함께 저장
 - 저장소 주소 : `http` 프로토콜을 사용하는 URL 주소를 사용
 - 우분투 버전 정보 : 저장소에서 관리하는 패키지에 해당하는 우분투의 버전을 표시한다. 버전은 번호가 아니라 버전의 이름을 사용
 - 카테고리 : 저장소가 가지고 있는 소프트웨어 카테고리(`main`, `restricted` 등)를 표시

```
user1@myubuntu:~$ cat /etc/apt/sources.list
#deb cdrom:[Ubuntu 17.10 _Artful Aardvark_ - Release amd64 (20171018)]/ artful
main restricted

# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://kr.archive.ubuntu.com/ubuntu/ artful main restricted
# deb-src http://kr.archive.ubuntu.com/ubuntu/ artful main restricted
(생략)
```


02 우분투 패키지 설치

- APT 명령으로 패키지 관리하기
- apt-cache 명령 : APT 캐시(패키지 데이터베이스)에서 정보를 검색하여 출력

apt-cache

- **기능** APT 캐시에 질의하여 여러 가지 정보를 검색한다.
- **형식** apt-cache [옵션] 서브 명령
- **옵션** -f: 검색 결과로 패키지에 대한 전체 기록을 출력한다.
-h: 간단한 도움말을 출력한다.
- **서브 명령** stats: 캐시에 대한 통계 정보를 출력한다.
dump: 현재 설치되어 있는 패키지를 업그레이드한다.
search 키워드: 캐시에서 키워드를 검색한다.
showpkg 패키지명: 패키지에 대한 의존성 정보와 역의존성 정보를 검색하여 출력한다.
show 패키지명: 패키지에 대한 간단한 정보를 출력한다.
pkgnames: 사용 가능한 모든 패키지의 이름을 출력한다.
- **사용 예** apt-cache stats
apt-cache show vsftpd
apt-cache search vsftpd

02 우분투 패키지 설치

■ apt-cache 명령

- APT 캐시 통계 정보 보기 : stats
 - 전체 패키지 이름 : 패키지 이름의 전체 개수
 - 일반 패키지 : 일반적으로 사용하는 패키지의 개수
 - 순수 가상 패키지(pure virtual package)
 - 가상 패키지는 패키지의 이름만 제공하며 그 이름을 가진 별도의 패키지가 실제로 있는 것은 아님
 - 단일 가상 패키지(single virtual package)
 - 한 패키지가 특정 가상 패키지의 기능을 제공
 - 혼합 가상 패키지(mixed virtual package)
 - 특정 가상 패키지를 제공하거나 가상 패키지의 이름을 패키지 이름으로 사용하는 경우
 - 빠짐(missing) : 의존성은 있지만 어떠한 패키지도 제공하지 않는 패키지
 - 개별 버전 전체(total distinct version) : 캐시에 있는 패키지 버전의 개수를 의미

```
user1@myubuntu:~$ apt-cache stats
전체 패키지 이름: 75460 (1,509 k)
전체 패키지 구조: 114547 (5,040 k)
일반 패키지: 83184
순수 가상 패키지: 1061
단일 가상 패키지: 11051
혼합 가상 패키지: 3907
빠짐: 15344
개별 버전 전체: 87753 (7,020 k)
개별 설명 전체: 152110 (3,651 k)
전체 의존성: 683784/219976 (17.2 M)
전체 버전/파일 관계: 52796 (1,267 k)
전체 설명/파일 관계: 21038 (505 k)
전체 제공 매핑: 21126 (507 k)
전체 패턴 문자열: 166318 (3,700 k)
전체 빈 용량: 20.6 k
차지하는 전체 용량: 40.8 M
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ apt-cache 명령

- 사용 가능한 패키지 이름 보기 : pkgnames

```
user1@myubuntu:~$ apt-cache pkgnames
libdatrie-doc
libfstrcmp0-dbg
libghc-monadplus-doc
librime-data-sampheng
python-pyao-dbg
fonts-georgewilliams
python3-aptdaemon.test
libcollada2glTFconvert-dev
python3-doc8
r-bioc-hypergraph
angrydd
pike8.0-pg
fonts-linuxlibertine
libslurmdb-dev
libvformat0
davical
(생략)
```

02 우분투 패키지 설치

■ apt-cache 명령

- 패키지 이름 검색하기 : search

```
user1@myubuntu:~$ apt-cache search vsftpd
resource-agents - Cluster Resource Agents
vsftpd - lightweight, efficient FTP server written for security
vsftpd-dbg - lightweight, efficient FTP server written for security (debug)
ccze - robust, modular log coloriser
ftpd - File Transfer Protocol (FTP) server
yasat - simple stupid audit tool
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ apt-cache 명령

- 패키지 정보 검색하기 : show
 - 버전, 패키지 크기, 카테고리, 체크섬 등 패키지에 관한 정보를 확인하려면 show 서브 명령을 사용

```
uuser1@myubuntu:~$ apt-cache show vsftpd
Package: vsftpd
Architecture: amd64
Version: 3.0.3-9
Priority: extra
Section: net
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Keng-Yu Lin <kengyu@lexical.tw>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 326
Provides: ftp-server
(생략)
```

02 우분투 패키지 설치

■ apt-cache 명령

- 패키지 의존성 검색하기 : showpkg

```
user1@myubuntu:~$ apt-cache showpkg vsftpd
Package: vsftpd
Versions:
3.0.3-9 (/var/lib/apt/lists/kr.archive.ubuntu.com_ubuntu_dists_artful_main_binary-
amd64_Packages)
Description Language:
File: /var/lib/apt/lists/kr.archive.ubuntu.com_ubuntu_dists_artful_
main|binary-amd64_Packages
MD5: 81386f72ac91a5ea48f8db0b023f3f9b
(생략)
Dependencies:
3.0.3-9 - debconf (18 0.5) debconf-2.0 (0 (null)) init-system-helpers (2 1.18~)
libc6 (2 2.15) libcap2 (2 1:2.10) libpam0g (2 0.99.7.1) libssl1.0.0 (2 1.0.0)
libwrap0 (2 7.6-4~) adduser (0 (null)) libpam-modules (0 (null)) lsb-base (2 3.0-
6) netbase (0 (null)) ftp-server (0 (null)) logrotate (0 (null)) ssl-cert (0
(null)) ftp-server (0 (null)) ftp-server:i386 (0 (null)) ftp-server:i386 (0 (null))
vsftpd:i386 (32 (null))
Provides:
3.0.3-9 - ftp-server (= )
Reverse Provides:
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ apt-get 명령

apt-get

- **기능** 패키지를 관리한다.
- **형식** apt-get [옵션] 서브 명령
- **옵션**
 - d: 패키지를 내려받기만 한다.
 - f: 의존성이 깨진 패키지를 수정하려고 시도한다.
 - h: 간단한 도움말을 출력한다.
- **서브 명령**
 - update: 패키지 저장소에서 새로운 패키지 정보를 가져온다.
 - upgrade: 현재 설치되어 있는 패키지를 업그레이드한다.
 - install 패키지명: 패키지를 설치한다.
 - remove 패키지명: 패키지를 삭제한다.
 - download 패키지명: 패키지를 현재 디렉터리로 내려받는다.
 - autoclean: 불완전하게 내려받았거나 오래된 패키지를 삭제한다.
 - clean: /var/cache/apt/archives에 캐시되어 있는 모든 패키지를 삭제하여 디스크 공간을 확보한다.
 - check: 의존성이 깨진 패키지를 확인한다.
- **사용 예** apt-get update apt-get install vsftpd apt-get clean

02 우분투 패키지 설치

■ apt-get 명령

- 패키지 정보 업데이트하기 : update
 - /etc/apt/sources.list에 명시한 저장소에서 패키지 정보를 읽어 동기화
 - 새로운 패키지 정보를 가져와서 APT 캐시를 수정

```
user1@myubuntu:~$ sudo apt-get update
기준:1 http://kr.archive.ubuntu.com/ubuntu artful InRelease
기준:2 http://kr.archive.ubuntu.com/ubuntu artful-updates InRelease
기준:3 http://kr.archive.ubuntu.com/ubuntu artful-backports InRelease
받기:4 http://security.ubuntu.com/ubuntu artful-security InRelease [78.6 kB]
받기:5 http://security.ubuntu.com/ubuntu artful-security/main amd64 DEP-11
Metadata [204 B]
받기:6 http://security.ubuntu.com/ubuntu artful-security/universe amd64 DEP-11
Metadata [2,184 B]
내려받기 81.0 k바이트, 소요시간 2초 (38.2 k바이트/초)
패키지 목록을 읽는 중입니다... 완료
user1@myubuntu:~$
```


02 우분투 패키지 설치

■ apt-get 명령

- 패키지 업그레이드하기 : upgrade
 - 현재 설치되어 있는 모든 패키지 중에서 새로운 버전이 있는 패키지를 모두 업그레이드

```
user1@myubuntu:~$ sudo apt-get upgrade
```

```
패키지 목록을 읽는 중입니다... 완료
```

```
의존성 트리를 만드는 중입니다
```

```
상태 정보를 읽는 중입니다... 완료
```

```
업그레이드를 계산하는 중입니다... 완료
```

```
다음 패키지를 과거 버전으로 유지합니다:
```

```
ubuntu-desktop
```

```
다음 패키지를 업그레이드할 것입니다:
```

```
accountsservice apt apt-utils distro-info-data gdm3 gir1.2-accountsservice-1.0
```

```
gir1.2-gdm-1.0 gir1.2-gtk-3.0 gir1.2-mutter-1 gnome-control-center
```

```
gnome-control-center-data gnome-control-center-faces gnome-session-bin
```

```
(생략)
```

```
44개 업그레이드, 0개 새로 설치, 0개 제거 및 1개 업그레이드 안 함.
```

```
0 바이트/13.4 M바이트 아카이브를 받아야 합니다.
```

```
이 작업 후 37.9 k바이트의 디스크 공간이 비워집니다.
```

```
계속 하시겠습니까? [Y/n] n
```

```
중단.
```

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

02 우분투 패키지 설치

■ apt-get 명령

- 특정 패키지 설치 또는 업그레이드하기 : install
 - 하나 이상의 패키지를 설치하거나 업그레이드할 때는 install 서브 명령을 사용

```
user1@myubuntu:~$ sudo apt-get install netcat
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
    netcat-traditional
다음 새 패키지를 설치할 것입니다:
    netcat netcat-traditional
0개 업그레이드, 2개 새로 설치, 0개 제거 및 45개 업그레이드 안 함.
65.1 k바이트 아카이브를 받아야 합니다.
이 작업 후 157 k바이트의 디스크 공간을 더 사용하게 됩니다.
계속 하시겠습니까? [Y/n]
(생략)
netcat-traditional (1.10-41.1) 설정하는 중입니다 ...
netcat (1.10-41.1) 설정하는 중입니다 ...
Processing triggers for man-db (2.7.6.1-2) ...
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ apt-get 명령

- 특정 패키지 설치 또는 업그레이드하기 : install
 - 여러 패키지를 한 번에 설치하려면 다음과 같이 패키지 이름을 나열

```
sudo apt-get install nethogs goaccess
```

- 패키지를 설치할 때 업그레이드를 하지 않으려면 '--no-upgrade' 옵션을 사용

```
sudo apt-get install netcat --no-upgrade
```

- 새로운 패키지를 설치하지 않고 업그레이드만 할 때는 '--only-upgrade' 옵션을 사용

```
sudo apt-get install netcat --only-upgrade
```

02 우분투 패키지 설치

■ apt-get 명령

- 패키지 삭제하기 : remove

```
user1@myubuntu:~$ sudo apt-get remove netcat
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 패키지가 자동으로 설치되었지만 더 이상 필요하지 않습니다:
  netcat-traditional
Use 'sudo apt autoremove' to remove it.
다음 패키지를 지울 것입니다:
  netcat
0개 업그레이드, 0개 새로 설치, 1개 제거 및 45개 업그레이드 안 함.
이 작업 후 13.3 k바이트의 디스크 공간이 비워집니다.
계속 하시겠습니까? [Y/n]
(데이터베이스 읽는 중 ...현재 130505개의 파일과 디렉터리가 설치되어 있습니다.)
Removing netcat (1.10-41.1) ...
user1@myubuntu:~$
```

- 설정 파일을 포함하여 패키지를 삭제하려면 purge 서브 명령을 사용

```
sudo apt-get purge netcat
```

```
sudo apt-get remove --purge netcat
```

02 우분투 패키지 설치

■ apt-get 명령

- 패키지 자동 정리 및 삭제하기 : autoremove
 - 자동으로 설치되었으나 필요 없는 패키지는 autoremove 서브 명령으로 정리

```
user1@myubuntu:~$ sudo apt-get autoremove
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 패키지를 지울 것입니다:
  netcat-traditional
0개 업그레이드, 0개 새로 설치, 1개 제거 및 45개 업그레이드 안 함.
이 작업 후 143 k바이트의 디스크 공간이 비워집니다.
계속 하시겠습니까? [Y/n]
(데이터베이스 읽는 중 ...현재 130502개의 파일과 디렉터리가 설치되어 있습니다.)
Removing netcat-traditional (1.10-41.1) ...
Processing triggers for man-db (2.7.6.1-2) ...
user1@myubuntu:~$
```

- 디스크 공간 정리하기 : clean
 - 검색했거나 내려받은 패키지 파일들을 삭제하고 디스크 공간을 정리

```
sudo apt-get clean
```

02 우분투 패키지 설치

■ apt-get 명령

- 패키지 내려받기 : download
 - 패키지를 설치하지 않고 내려받기만 하려면 download 서브 명령을 사용

```
user1@myubuntu:~$ sudo apt-get download netcat
받기:1 http://kr.archive.ubuntu.com/ubuntu artful/universe amd64 netcat all 1.10-41.1 [3,436 B]
내려받기 3,436 바이트, 소요시간 0초 (170 k바이트/초)
W: Download is performed unsandboxed as root as file '/home/user1/netcat_1.10-41.1_all.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: 허가 거부)
user1@myubuntu:~$ ls net*
netcat_1.10-41.1_all.deb
user1@myubuntu:~$
```

- 패키지의 소스 관련 서브 명령 : source
 - 특정 패키지의 소스코드를 내려받기만 하는 경우

```
sudo apt-get --download-only source 패키지명
```

- 특정 패키지의 소스코드를 내려받고 압축을 푸는 경우

```
sudo apt-get source 패키지명
```

- 특정 패키지의 소스코드를 내려받아 압축을 풀고 컴파일하는 경우

```
sudo apt-get --compile source 패키지명
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

dpkg

- **기능** 데비안의 패키지 관리 명령이다.
- **형식** `dpkg [옵션] 파일명 또는 패키지명`
- **옵션**
 - l: 설치된 패키지의 목록을 출력한다.
 - l 패키지명: 패키지의 설치 상태를 출력한다.
 - s 패키지명: 패키지의 상세 정보를 출력한다.
 - S 경로명: 경로명이 포함된 패키지를 검색한다.
 - L 패키지명: 패키지에서 설치된 파일의 목록을 출력한다.
 - c .deb 파일: 지정한 .deb 파일의 내용을 출력한다.
 - i .deb 파일: 해당 파일을 설치한다(sudo).
 - r 패키지명: 해당 패키지를 삭제한다(sudo).
 - P 패키지명: 해당 패키지와 설정 정보를 모두 삭제한다(sudo).
 - x .deb 파일 디렉터리: 해당 파일을 지정한 디렉터리에 풀어놓는다.
- **사용 예**
 - `dpkg -l` `dpkg -s netcat` `dpkg -S /bin/ls`
 - `sudo dpkg -i netcat_1.10-40_all.deb`

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지 목록 보기 : -l 옵션
 - 출력 결과에서 첫 글자는 상단의 희망 상태를 나타내고, 두 번째 글자는 상태를 표시

```
user1@myubuntu:~$ dpkg -l
희망상태=알수없음(U)/설치(I)/지우기(R)/깨끗이(P)/고정(H)
! 상태=아님(N)/설치(I)/설정(C)/풀림(U)/절반설정(F)/일부설치(H)/트리거대기(W)/
! /      트리거밀림(T)
!/? 오류?=(없음)/다시설치필요(R) (상태, 오류가 대문자=불량)
!!/ 이름                버전                Architecture 설명
++=====
ii  accountsservic 0.6.42-0ubun amd64      query and manipulate user account
ii  acl              2.2.52-3buil amd64      Access control list utilities
ii  acpi-support    0.142         amd64      scripts for handling many ACPI eve
ii  acpid            1:2.0.28-1ub amd64      Advanced Configuration and Power I
ii  adduser          3.113+nmu3ub all         add and remove users and groups
(생략)
user1@myubuntu:~$
```


02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지 목록 보기 : -l 옵션
 - -l 다음에 특정 패키지의 이름을 지정하면 해당 패키지에 관한 정보만 출력

```
user1@myubuntu:~$ dpkg -l zip
희망상태=알수없음(U)/설치(I)/지우기(R)/깨끗이(P)/고정(H)
! 상태=아님(N)/설치(I)/설정(C)/풀림(U)/절반설정(F)/일부설치(H)/트리거대기(W)/
! /      트리거밀림(T)
!/ 오류?=(없음)/다시설치필요(R) (상태, 오류가 대문자=불량)
||/ 이름                버전                Architecture  설명
+-+=====
ii  zip                  3.0-11build1  amd64          Archiver for .zip files
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지 상세 정보 보기 : -s 옵션

```
user1@myubuntu:~$ dpkg -s zip
Package: zip
Status: install ok installed
Priority: optional
Section: utils
Installed-Size: 623
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: foreign
Version: 3.0-11build1
Depends: libbz2-1.0, libc6 (>= 2.14)
Recommends: unzip
Description: Archiver for .zip files
 This is InfoZIP's zip program. It produces files that are fully
 compatible with the popular PKZIP program; however, the command line
 options are not identical. In other words, the end result is the same,
 but the methods differ. :-)
.
 This version supports encryption.
Original-Maintainer: Santiago Vila <sanvila@debian.org>
Homepage: http://www.info-zip.org/Zip.html
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 특정 파일이 포함된 패키지 검색하기 : -S 옵션

```
user1@myubuntu:~$ dpkg -S /bin/ls
coreutils: /bin/ls
user1@myubuntu:~$
```

- 패키지가 설치한 파일 목록 검색하기 : -L 옵션

```
user1@myubuntu:~$ dpkg -L zip
/.
/usr
/usr/bin
/usr/bin/zip
/usr/bin/zipcloak
/usr/bin/zipnote
/usr/bin/zipsplit
(생략)
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지의 파일 목록 검색하기 : -c 옵션

```
user1@myubuntu:~$ dpkg -c netcat_1.10-41.1_all.deb
drwxr-xr-x root/root      0  2017-04-14  03:31 ./
drwxr-xr-x root/root      0  2017-04-14  03:31 ./usr/
drwxr-xr-x root/root      0  2017-04-14  03:31 ./usr/share/
drwxr-xr-x root/root      0  2017-04-14  03:31 ./usr/share/doc/
drwxr-xr-x root/root      0  2017-04-14  03:31 ./usr/share/doc/netcat/
-rw-r--r-- root/root    1906  2017-04-14  03:31 ./usr/share/doc/netcat/changelog.
Debian.gz
-rw-r--r-- root/root      735  2011-02-11  14:21 ./usr/share/doc/netcat/copyright
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지 설치하기 : -i 옵션

① netcat_1.10-40_all.deb 패키지를 설치 시도

```
user1@myubuntu:~$ sudo dpkg -i netcat_1.10-41.1_all.deb
[sudo] user1의 암호:
Selecting previously unselected package netcat.
(데이터베이스 읽는 중 ...현재 130468개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack netcat_1.10-41.1_all.deb ...
Unpacking netcat (1.10-41.1) ...
dpkg: dependency problems prevent configuration of netcat:
 netcat 패키지는 다음 패키지에 의존: netcat-traditional (>= 1.10-39): 하지만:
 netcat-traditional 패키지는 설치하지 않았습니다.

dpkg: error processing package netcat (--install):
 의존성 문제 - 설정하지 않고 남겨둠
처리하는데 오류가 발생했습니다:
 netcat
user1@myubuntu:~$
```

- APT 명령과 달리 dpkg 명령은 의존성이 있는 패키지를 자동으로 설치하지 않으므로 사용자가 일일이 설치해야함
- 따라서 netcat-traditional 패키지를 내려받아 먼저 설치한 다음 netcat 패키지를 설치해야함

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

② netcat-traditional 패키지를 다운로드

```
user1@myubuntu:~$ apt-get download netcat-traditional
받기:1 http://kr.archive.ubuntu.com/ubuntu artful/universe amd64 netcat-
traditional amd64 1.10-41.1 [61.7 kB]
내려받기 61.7 k바이트, 소요시간 0초 (395 k바이트/초)
user1@myubuntu:~$ ls netcat*
netcat-traditional_1.10-41.1_amd64.deb  netcat_1.10-41.1_all.deb
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

③ 차례로 설치

```
user1@myubuntu:~$ sudo dpkg -i netcat-traditional_1.10-41.1_amd64.deb
Selecting previously unselected package netcat-traditional.
(데이터베이스 읽는 중 ...현재 130471개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack netcat-traditional_1.10-41.1_amd64.deb ...
Unpacking netcat-traditional (1.10-41.1) ...
netcat-traditional (1.10-41.1) 설정하는 중입니다 ...
Processing triggers for man-db (2.7.6.1-2) ...
user1@myubuntu:~$ sudo dpkg -i netcat_1.10-41.1_all.deb
(데이터베이스 읽는 중 ...현재 130507개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack netcat_1.10-41.1_all.deb ...
Unpacking netcat (1.10-41.1) over (1.10-41.1) ...
netcat (1.10-41.1) 설정하는 중입니다 ...
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- 패키지 삭제하기 : -r, -P 옵션
 - -r 옵션은 설치된 패키지만 삭제하고, -P 옵션은 패키지와 설정 정보를 모두 삭제

```
user1@myubuntu:~$ sudo dpkg -r netcat
(데이터베이스 읽는 중 ...현재 130505개의 파일과 디렉터리가 설치되어 있습니다.)
Removing netcat (1.10-41.1) ...
user1@myubuntu:~$
```

```
user1@myubuntu:~$ sudo dpkg -P netcat-traditional
(데이터베이스 읽는 중 ...현재 130502개의 파일과 디렉터리가 설치되어 있습니다.)
Removing netcat-traditional (1.10-41.1) ...
Processing triggers for man-db (2.7.6.1-2) ...
user1@myubuntu:~$
```


02 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

- .deb 파일 풀기 : -x 옵션

```
user1@myubuntu:~$ mkdir netcat
user1@myubuntu:~$ sudo dpkg -x netcat_1.10-41.1_all.deb netcat
user1@myubuntu:~$ ls -R netcat
netcat:
usr

netcat/usr:
share

netcat/usr/share:
doc

netcat/usr/share/doc:
netcat

netcat/usr/share/doc/netcat:
changelog.Debian.gz  copyright
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ aptitude 명령으로 패키지 관리하기

- aptitude는 APT 명령처럼 패키지 관리를 자동화하여 쉽게 작업할 수 있도록 해줌
- 옵션이나 서브 명령 없이 실행할 경우 curses를 이용한 비주얼 모드로 동작

aptitude

- **기능** 우분투에서 패키지를 관리한다.
- **형식** aptitude [서브 명령]
- **서브 명령** 단독 실행: curses 프로그램이 나타난다.
 - search 키워드: 키워드를 검색하여 일치하는 패키지 목록을 출력한다.
 - update: 패키지 저장소를 업데이트한다.
 - upgrade: 모든 패키지를 최신 버전으로 업그레이드한다.
 - show 패키지명: 패키지에 대한 자세한 정보를 보여준다.
 - download 패키지명: 패키지를 내려받는다.
 - clean: 패키지 캐시 디렉터리에서 모든 패키지 파일을 삭제한다.
 - install: 패키지를 설치한다.
 - remove: 패키지를 삭제한다.
 - purge: 패키지와 설정 파일을 모두 삭제한다.

02 우분투 패키지 설치

■ aptitude 명령 단독으로 실행하기

- aptitude 단독으로 실행하기

```
user1@myubuntu:~$ sudo aptitude
sudo: aptitude: 명령이 없습니다
user1@myubuntu:~$
```

- aptitude 명령이 설치되어 있지 않으면 apt-get 명령으로 aptitude를 설치

```
user1@myubuntu:~$ sudo apt-get install aptitude
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  aptitude-common libcwid3v5
제안하는 패키지:
  aptitude-doc-en | aptitude-doc apt-xapian-index debtags tasksel libcwid-dev
다음 새 패키지를 설치할 것입니다:
  aptitude aptitude-common libcwid3v5
0개 업그레이드, 3개 새로 설치, 0개 제거 및 43개 업그레이드 안 함.
2,571 k바이트 아카이브를 받아야 합니다.
이 작업 후 10.9 M바이트의 디스크 공간을 더 사용하게 됩니다.
계속 하시겠습니까? [Y/n] y
(생략)
user1@myubuntu:~$
```

02 우분투 패키지 설치

■ aptitude 명령 단독으로 실행하기

- aptitude 명령을 다시 실행

```
user1@myubuntu:~$ sudo aptitude
```

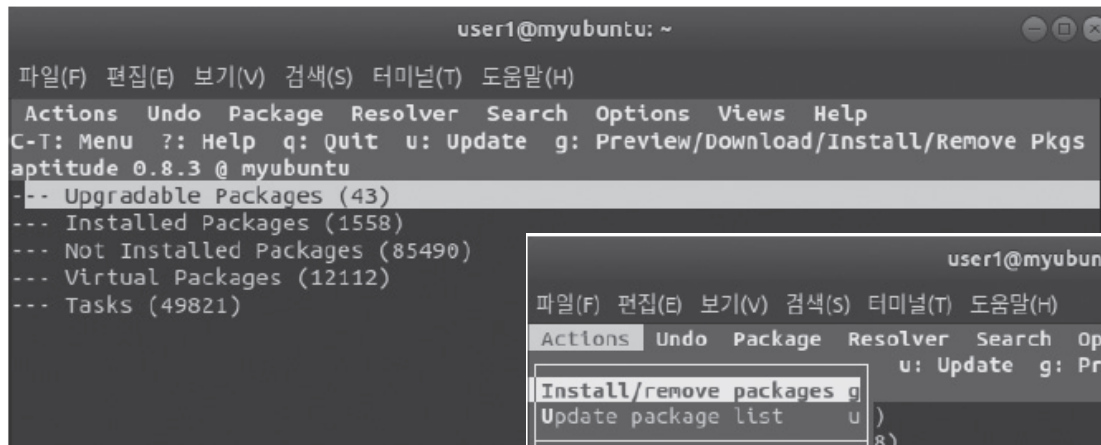


그림 9-5 aptitude 실행 화면

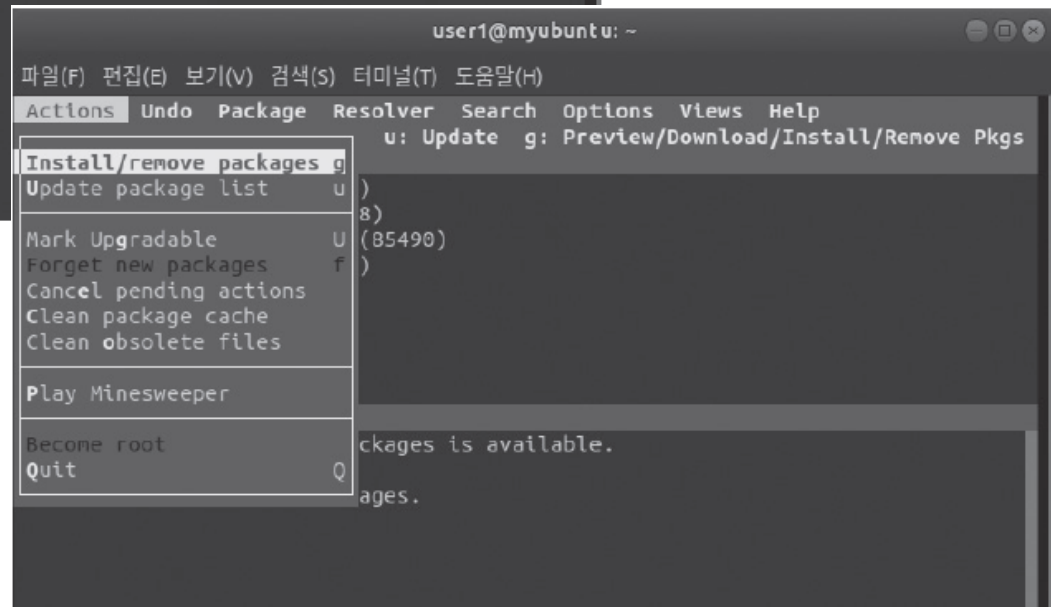
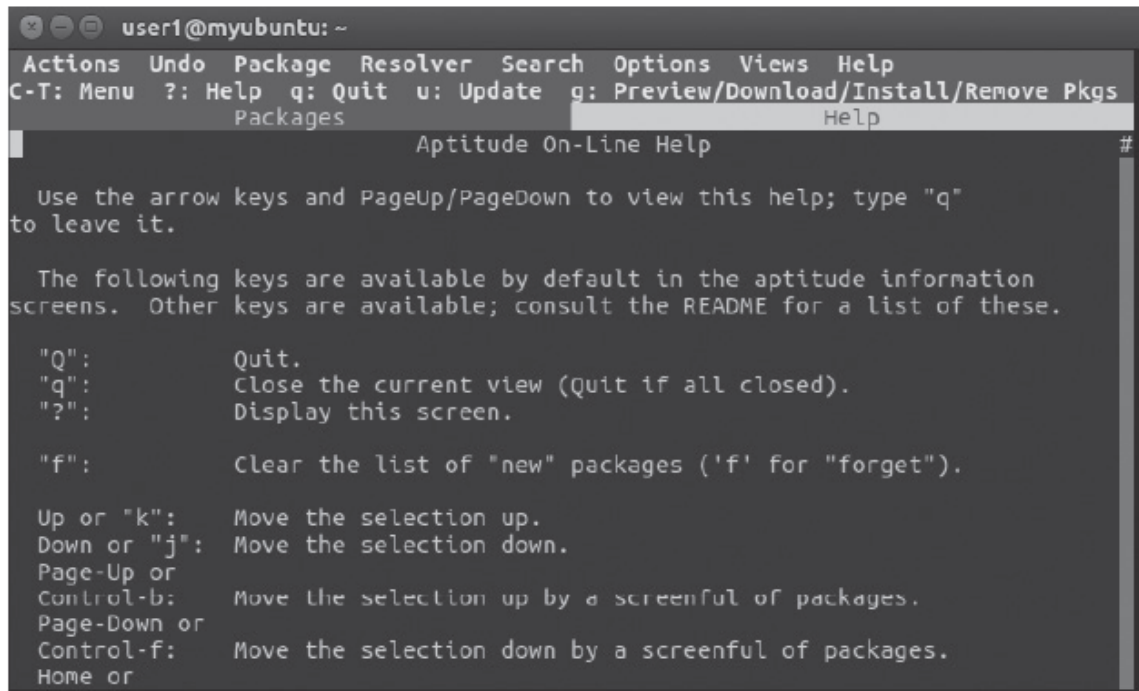


그림 9-6 aptitude 메뉴 선택

02 우분투 패키지 설치

■ aptitude 명령 단독으로 실행하기

- ?를 입력하면 도움말이 출력
- q로 aptitude를 종료
- j, k와 화살표로 이동



```
user1@myubuntu: ~  
Actions Undo Package Resolver Search Options Views Help  
C-T: Menu ?: Help q: Quit u: Update g: Preview/Download/Install/Remove Pkgs  
Packages Help  
Aptitude On-Line Help #  
  
Use the arrow keys and PageUp/PageDown to view this help; type "q"  
to leave it.  
  
The following keys are available by default in the aptitude information  
screens. Other keys are available; consult the README for a list of these.  
  
"Q": Quit.  
"q": Close the current view (Quit if all closed).  
"?": Display this screen.  
  
"f": Clear the list of "new" packages ('f' for "forget").  
  
Up or "k": Move the selection up.  
Down or "j": Move the selection down.  
Page-Up or  
Control-b: Move the selection up by a screenful of packages.  
Page-Down or  
Control-f: Move the selection down by a screenful of packages.  
Home or
```

그림 9-7 aptitude 도움말 화면

02 우분투 패키지 설치

■ aptitude를 명령으로 사용하기

- 패키지 정보 업데이트하기 : update

```
user1@myubuntu:~$ sudo aptitude update
Hit http://kr.archive.ubuntu.com/ubuntu artful InRelease
Hit http://kr.archive.ubuntu.com/ubuntu artful-updates InRelease
Get: 1 http://kr.archive.ubuntu.com/ubuntu artful-backports InRelease [65.5 kB]
Get: 2 http://security.ubuntu.com/ubuntu artful-security InRelease [78.6 kB]
Fetched 144 kB in 1초 (102 kB/s)

user1@myubuntu:~$
```

- 패키지 검색하기 : search

```
user1@myubuntu:~$ aptitude search gnome
p  amule-gnome-support          - ed2k links handling support for GNOME web b
p  backintime-gnome             - GNOME front-end for backintime (transitiona
p  chrome-gnome-shell           - GNOME Shell extensions integration for web
p  clamtk-gnome                 - GNOME (Nautilus) MenuProvider extension for
p  compiz-gnome                 - OpenGL window and compositing manager - GNO
p  compiz-gnome:i386            - OpenGL window and compositing manager - GNO
p  desktopnova-module-gnome     - GNOME module for DesktopNova
p  desktopnova-module-gnome:i386 - GNOME module for DesktopNova
(생략)
```

02 우분투 패키지 설치

■ aptitude를 명령으로 사용하기

- 패키지 상세 정보 확인하기 : show

```
user1@myubuntu:~$ aptitude show gnome-clocks
Package: gnome-clocks
Version: 3.26.1-1
State: not installed
Priority: 옵션
Section: universe/gnome
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Uncompressed Size: 1,787 k
Depends: libc6 (>= 2.4), libcairo2 (>= 1.2.4), libgdk-pixbuf2.0-0 (>= 2.22.0),
        libgeoclue-2-0 (>= 2.4.0), libgeocode-glib0 (>= 1.0), libglib2.0-0 (>=
        2.45.7), libgnome-desktop-3-12 (>= 3.17.92), libgsound0 (>= 1.0.1),
        libgtk-3-0 (>= 3.20.0), libgweather-3-6 (>= 3.13.91),
        dconf-gsettings-backend | gsettings-backend, geoclue-2.0
Conflicts: gnome-clocks:i386
Description: Simple GNOME app with stopwatch, timer, and world clock support
 GNOME Clocks is a simple application to show the time and date in multiple
 locations and set alarms or timers. A stopwatch is also included.
Homepage: https://wiki.gnome.org/Apps/Clocks

user1@myubuntu:~$
```

02 우분투 패키지 설치

■ aptitude를 명령으로 사용하기

- 패키지 설치하기 : install

```
user1@myubuntu:~$ sudo aptitude install gnome-clocks
The following NEW packages will be installed:
  gnome-clocks libgsound0{a}
0 packages upgraded, 2 newly installed, 0 to remove and 43 not upgraded.
Need to get 338 kB of archives. After unpacking 1,822 kB will be used.
Do you want to continue? [Y/n/?]
Get: 1 http://kr.archive.ubuntu.com/ubuntu artful/universe amd64 libgsound0 amd64
1.0.2-1ubuntu1 [7,894 B]
Get: 2 http://kr.archive.ubuntu.com/ubuntu artful/universe amd64 gnome-clocks
amd64 3.26.1-1 [330 kB]
Fetched 338 kB in 0초 (858 kB/s)
(생략)
gnome-clocks (3.26.1-1) 설정하는 중입니다 ...
Processing triggers for libc-bin (2.26-0ubuntu2) ...

user1@myubuntu:~$
```

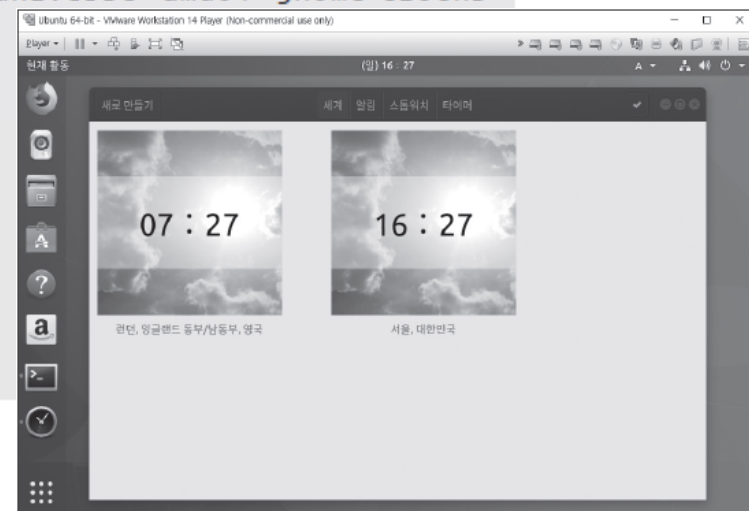


그림 9-8 gnome-clocks 실행 화면

02 우분투 패키지 설치

■ 우분투 소프트웨어 센터

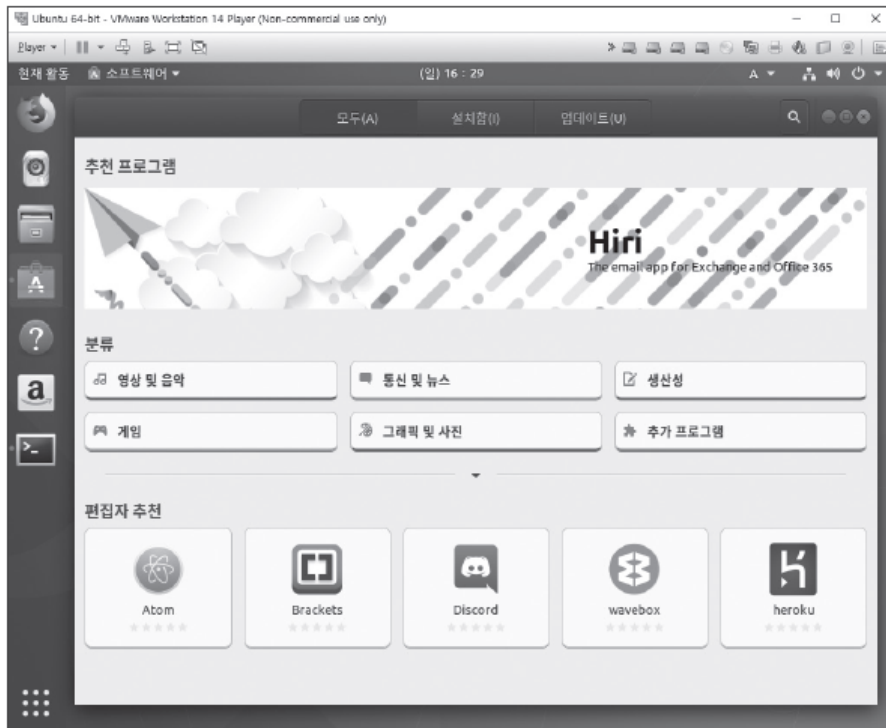


그림 9-9 우분투 소프트웨어 센터 실행 화면

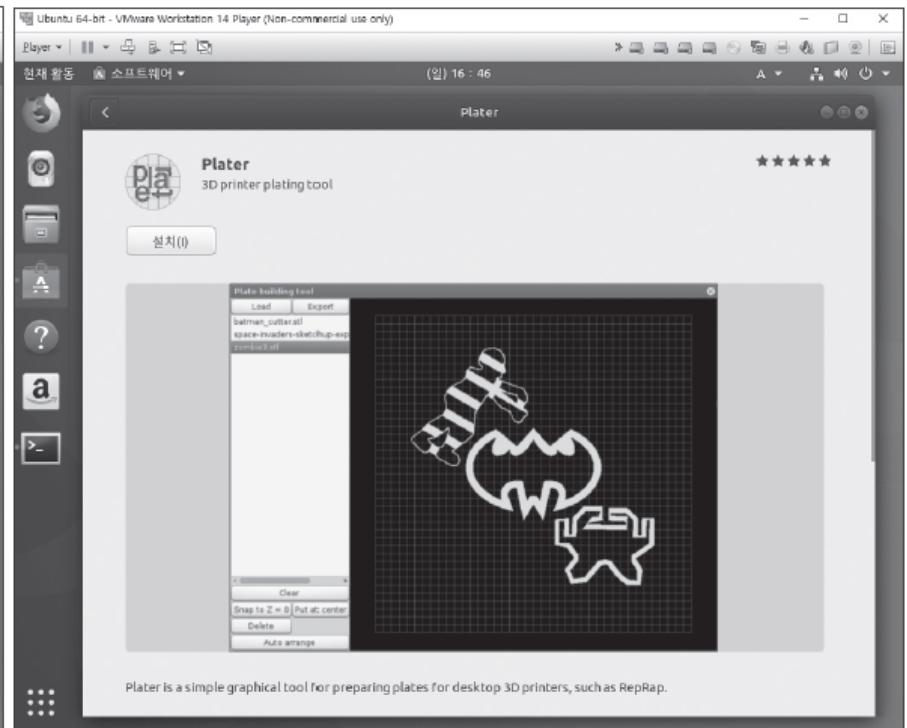


그림 9-10 우분투 소프트웨어 센터: 프로그램 선택 화면

02 우분투 패키지 설치

■ 우분투 소프트웨어 센터

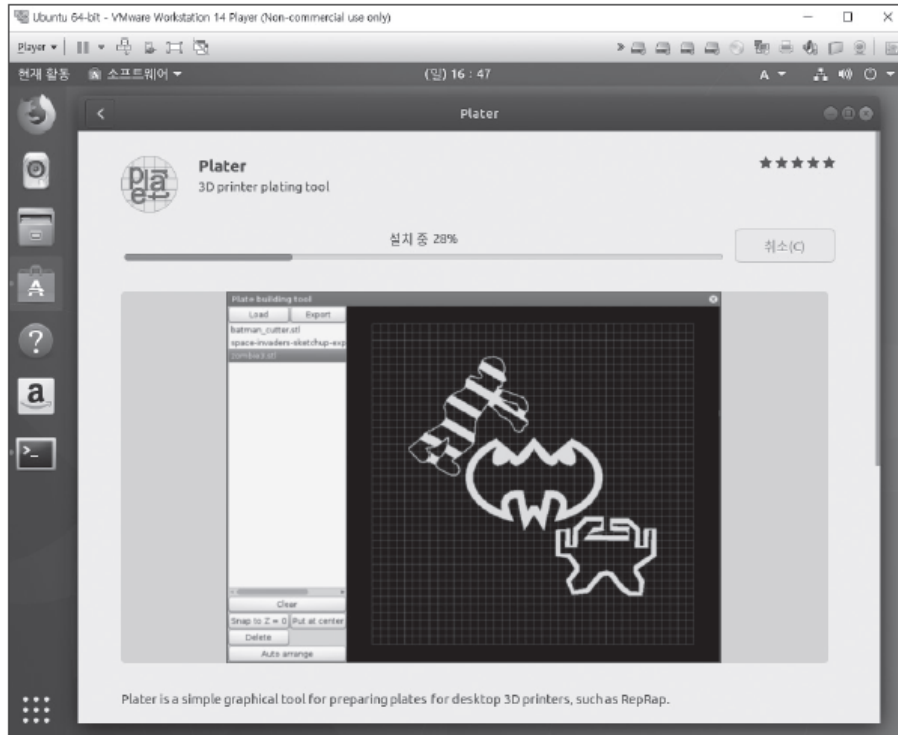


그림 9-11 우분투 소프트웨어 센터: 프로그램 설치 중 화면

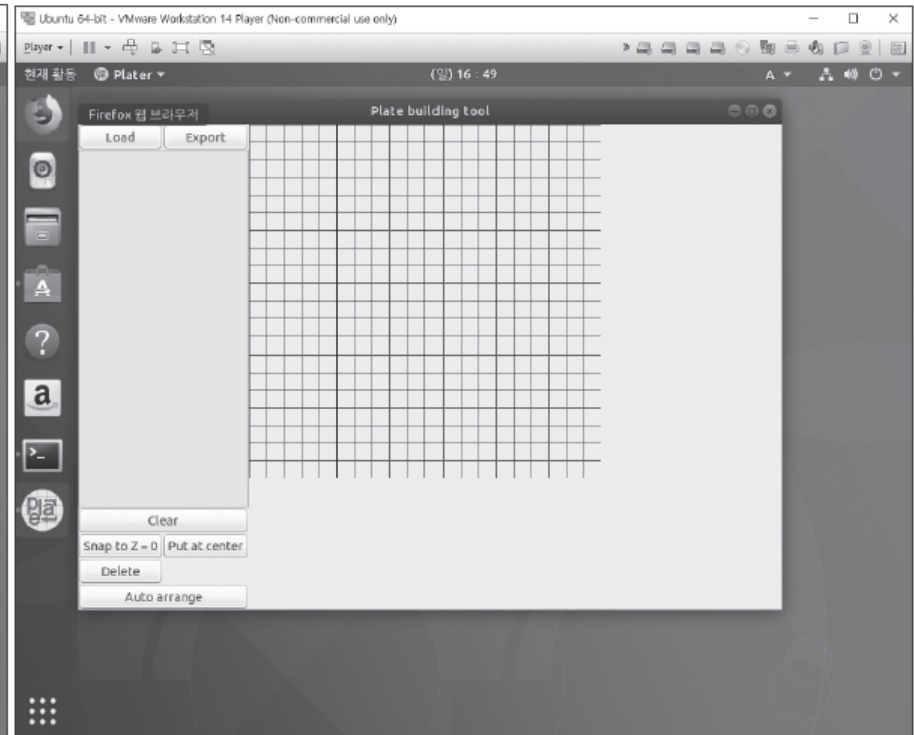


그림 9-12 Plater 실행 화면

03 스냅 패키지 설치

■ 스냅의 개념

- 우분투가 새로 도입한 패키지 형식으로 샌드박스 형태의 패키지
- 패키지를 만들 때 프로그램이 사용하는 모든 라이브러리를 패키지 안에 포함
- 패키지 개념인 샌드박스 형식을 스냅이 사용함으로써 얻게 되는 장점
 - 개발자가 다른 패키지나 라이브러리와 의존성을 신경 쓰지 않아도 된다.
 - 기존 시스템과 격리되어 실행하는 샌드박스 형식이므로 보안이 강화된다.
- 단점: 패키지의 용량이 커진다는 것
- 스냅은 이제 시작 단계이며 앞으로 어떻게 발전할지 관심을 가지고 지켜볼 필요가 있음

■ 스냅 사용하기

- apt-get 명령으로 snap을 설치하고 시작

```
user1@myubuntu:~$ sudo apt-get install snap
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 새 패키지를 설치할 것입니다:
  snap
0개 업그레이드, 1개 새로 설치, 0개 제거 및 43개 업그레이드 안 함.
(생략)
snap (2013-11-29-6) 설정하는 중입니다 ...
Processing triggers for man-db (2.7.6.1-2) ...
user1@myubuntu:~$
```

03 스냅 패키지 설치

■ 스냅 사용하기

- snap 명령의 기본 형식

snap

- **기능** 스냅 패키지를 설치하고, 설정하고, 삭제한다.
- **형식** snap [옵션] 명령
- **옵션** -h: 도움말을 출력한다.
- **명령**
 - disable: 스냅 서비스와 실행 파일의 사용을 중지한다.
 - download 스냅명: 지정한 스냅 패키지를 내려받는다.
 - enable: 스냅 서비스와 실행 파일의 사용을 시작한다.
 - find 스냅명: 지정한 스냅을 검색한다.
 - info 스냅명: 지정한 스냅의 상세 정보를 출력한다.
 - install 스냅명: 지정한 스냅을 설치한다.
 - list: 설치한 스냅의 목록을 출력한다.
 - remove 스냅명: 지정한 스냅을 삭제한다.
- **사용 예**
 - snap list
 - snap install hello-world

03 스냅 패키지 설치

■ 스냅 목록 출력하기: list 명령

```
user1@myubuntu:~$ snap list
No snaps are installed yet. Try "snap install hello-world".
user1@myubuntu:~$
```

■ 스냅 찾기: find 명령

- 앞의 예에서 출력된 'hello-world' 스냅 찾기

```
user1@myubuntu:~$ snap find hello-world
Name                Version  Developer  Notes  Summary
hello-world          6.3     canonical  -      The 'hello-world' of snaps
hello-world-om26er   0.2     om26er     -      A great snap
hello-lhc             1.0     cprov      -      Hello world application for LHC
user1@myubuntu:~$
```

03 스냅 패키지 설치

■ 스냅 설치하기: install 명령

- find로 검색한 hello-world 스냅을 설치

```
user1@myubuntu:~$ sudo snap install hello-world
core 14.62 MB / 83.73 MB [====>-----] 17.47% 647.28 KB/s 1m49s
```

```
user1@myubuntu:~$ sudo snap install hello-world
hello-world 6.3 from 'canonical' installed
user1@myubuntu:~$
```

- hello-world 스냅이 설치한 명령을 실행해보면 'Hello World!'가 출력

```
user1@myubuntu:~$ hello-world
Hello World!
user1@myubuntu:~$
```

03 스냅 패키지 설치

■ 스냅의 상세 정보 확인하기: info 명령

- 설치된 스냅의 상세 정보를 확인하는 명령
- info 명령을 실행하면 스냅의 이름과 요약 정보, 제공자, 연락처, 간단한 설명, 관련 명령, 설치된 버전 정보 등이 출력

```
user1@myubuntu:~$ snap info hello-world
name:      hello-world
summary:    The 'hello-world' of snaps
publisher: canonical
contact:    snappy-devel@lists.ubuntu.com
description: |
  This is a simple hello world example.
snap-id:    buPKUD3TKqC0gLEjjHx5kSiCpIs5cMuQ
commands:
  - hello-world.env
  - hello-world.evil
  - hello-world
  - hello-world.sh
tracking:    stable
installed:   6.3 (27) 20kB -
refreshed:   2016-07-12 06:20:44 +0900 KST
channels:
  stable:    6.3 (27) 20kB -
  candidate: 6.3 (27) 20kB -
  beta:      6.3 (27) 20kB -
  edge:      6.3 (27) 20kB -
user1@myubuntu:~$
```

03 스냅 패키지 설치

■ 스냅 삭제하기: remove 명령

- hello-world 스냅을 삭제하기

```
user1@myubuntu:~$ sudo snap remove hello-world
hello-world removed
user1@myubuntu:~$ ls /snap
bin  core
user1@myubuntu:~$
```


04 파일 아카이브와 압축

■ 파일 아카이브

- 파일을 묶어서 하나로 만든 것
- tar(tape archive) 명령은 원래 여러 파일이나 디렉터리를 묶어서 마그네틱 테이프와 같은 이동식 저장 장치에 보관하기 위해 사용하는 명령
- 현재는 다른 시스템과 파일을 주고받거나, 백업을 하기 위해 여러 파일이나 디렉터리를 하나의 아카이브 파일로 생성하거나, 기존 아카이브에서 파일을 추출하기 위해 사용

tar

- **기능** 파일과 디렉터리를 묶어 하나의 아카이브 파일을 생성한다.
- **형식** tar 기능[옵션] [아카이브 파일] 파일명
- **기능** c: 새로운 tar 파일을 생성한다.
t: tar 파일의 내용을 출력한다.
x: tar 파일에서 원본 파일을 추출한다.
r: 새로운 파일을 추가한다.
u: 수정된 파일을 업데이트한다.
- **옵션** f: 아카이브 파일이나 테이프 장치를 지정한다. 파일명을 '-'로 지정하면 tar 파일 대신 표준 입력에서 읽어들인다.
v: 처리하고 있는 파일의 정보를 출력한다.
h: 심벌릭 링크의 원본 파일을 포함한다.
p: 파일 복구 시 원래의 접근 권한을 유지한다.
j: bzip2로 압축하거나 해제한다.
z: gzip으로 압축하거나 해제한다.
- **사용 예** tar cvf unix.tar Unix tar xvf unix.tar

04 파일 아카이브와 압축

■ 아카이브 생성하기 : cvf

```
user1@myubuntu:~/linux_ex$ tar cvf ch2.tar ch2
ch2/
ch2/temp/
ch2/temp/hosts
ch2/temp/text1
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.sl
ch2/data
ch2/test
ch2/data1.ln
user1@myubuntu:~/linux_ex$ ls
ch2  ch2.tar  ch3  ch4  ch5  ch6
user1@myubuntu:~/linux_ex$
```

- tar 명령으로 파일을 묶어서 아카이브 파일을 만들어도 원본 파일은 그대로 있음

04 파일 아카이브와 압축

■ 아카이브 내용 확인하기 : tvf

```
user1@myubuntu:~/linux_ex$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/temp/
-rw-r--r-- user1/user1    223 2017-11-11 22:34 ch2/temp/hosts
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/temp/text1
-rw-r--r-- user1/user1    223 2017-11-11 22:30 ch2/temp/text2
-rw-r--r-- user1/user1    223 2017-11-12 11:39 ch2/temp/data1.cp
lrwxrwxrwx user1/user1      0 2017-11-12 11:26 ch2/data1.sl -> data1
-rw-r--r-- user1/user1  19183 2017-11-12 11:42 ch2/data
-rw-rw-r-- user1/user1      0 2017-12-31 12:00 ch2/test
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/data1.ln
user1@myubuntu:~/linux_ex$
```

04 파일 아카이브와 압축

■ 아카이브 풀기 : xvf

```
user1@myubuntu:~/linux_ex$ mkdir ch9
user1@myubuntu:~/linux_ex$ mv ch2.tar ch9
user1@myubuntu:~/linux_ex$ cd ch9
user1@myubuntu:~/linux_ex/ch9$ tar xvf ch2.tar
ch2/
ch2/temp/
ch2/temp/hosts
ch2/temp/text1
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.sl
ch2/data
ch2/test
tar: ch2/test: time stamp 2017-12-31 12:00:00 is 3603721.147729212 s in the
future
ch2/data1.ln
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 아카이브 업데이트하기 : uvf

- u 기능은 지정한 파일이 아카이브에 없는 파일이거나, 아카이브에 있는 파일이지만 수정된 파일일 경우 아카이브의 마지막에 추가 -> ch2/data 파일의 수정시간을 touch 명령으로 수정후 아카이브 업데이트

```
user1@myubuntu:~/linux_ex/ch9$ touch ch2/data
user1@myubuntu:~/linux_ex/ch9$ tar uvf ch2.tar ch2
ch2/data
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/temp/
-rw-r--r-- user1/user1    223 2017-11-11 22:34 ch2/temp/hosts
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/temp/text1
-rw-r--r-- user1/user1    223 2017-11-11 22:30 ch2/temp/text2
-rw-r--r-- user1/user1    223 2017-11-12 11:39 ch2/temp/data1.cp
lrwxrwxrwx user1/user1      0 2017-11-12 11:26 ch2/data1.sl -> data1
-rw-r--r-- user1/user1 19183 2017-11-12 11:42 ch2/data
-rw-rw-r-- user1/user1      0 2017-12-31 12:00 ch2/test
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/data1.ln
-rw-r--r-- user1/user1 19183 2017-11-19 19:04 ch2/data
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 아카이브에 파일 추가하기 : rvf

- r 기능은 지정한 파일을 무조건 아카이브의 마지막에 추가

```
user1@myubuntu:~/linux_ex/ch9$ cp /etc/hosts .
user1@myubuntu:~/linux_ex/ch9$ tar rvf ch2.tar hosts
hosts
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/temp/
-rw-r--r-- user1/user1    223 2017-11-11 22:34 ch2/temp/hosts
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/temp/text1
-rw-r--r-- user1/user1    223 2017-11-11 22:30 ch2/temp/text2
-rw-r--r-- user1/user1    223 2017-11-12 11:39 ch2/temp/data1.cp
lrwxrwxrwx user1/user1      0 2017-11-12 11:26 ch2/data1.sl -> data1
-rw-r--r-- user1/user1 19183 2017-11-12 11:42 ch2/data
-rw-rw-r-- user1/user1      0 2017-12-31 12:00 ch2/test
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/data1.ln
-rw-r--r-- user1/user1 19183 2017-11-19 19:04 ch2/data
-rw-r--r-- user1/user1    223 2017-11-19 19:05 hosts
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 파일 압축과 아카이브

- 아카이브를 생성하면서 동시에 압축 수행
- 예: gzip으로 압축

```
user1@myubuntu:~/linux_ex/ch9$ tar cvzf ch2.tar.gz ch2
ch2/
ch2/temp/
ch2/temp/hosts
ch2/temp/text1
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.sl
ch2/data
ch2/test
ch2/data1.ln
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.gz  hosts
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 파일 압축과 아카이브

- 아카이브를 생성하면서 동시에 압축 실행
- 예: bzip2로 압축 실행: bzip2로 압축할 경우 j 옵션을 사용

```
user1@myubuntu:~/linux_ex/ch9$ tar cvjf ch2.tar.bz2 ch2
ch2/
ch2/temp/
ch2/temp/hosts
(생략)
ch2/data1.ln
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.bz2  ch2.tar.gz  hosts
user1@myubuntu:~/linux_ex/ch9$
```

- 압축한 아카이브 파일의 내용은 tvf로 확인이 가능하며 xvf로 추출 가능

```
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar.gz
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/
drwxrwxr-x user1/user1      0 2017-11-12 12:03 ch2/temp/
-rw-r--r-- user1/user1    223 2017-11-11 22:34 ch2/temp/hosts
-rw-r--r-- user1/user1    223 2017-11-11 22:29 ch2/temp/text1
-rw-r--r-- user1/user1    223 2017-11-11 22:30 ch2/temp/text2
(생략)
```


04 파일 아카이브와 압축

■ 파일 압축하기: gzip/gunzip - .gz 파일

gzip

- **기능** 파일을 압축한다.
- **형식** gzip [옵션] 파일명
- **옵션**
 - d: 파일 압축을 해제한다.
 - l: 압축 파일의 정보를 보여준다.
 - r: 하위 디렉터리를 이동하여 파일을 압축한다.
 - t: 압축 파일을 검사한다.
 - v: 압축 정보를 화면에 출력한다.
 - 9: 최대한 압축한다.
- **사용 예**
 - gzip a.txt
 - gzip -v b.txt c.txt

```
user1@myubuntu:~/linux_ex/ch9$ rm ch2.tar.gz
rm: 일반 파일 'ch2.tar.gz'를 제거할까요? y
user1@myubuntu:~/linux_ex/ch9$ gzip ch2.tar
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar.bz2  ch2.tar.gz  hosts
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 압축 파일의 내용 보기 : zcat

zcat

- **기능** gzip으로 압축된 파일의 내용을 출력한다.
- **형식** zcat 파일명
- **사용 예** zcat abc.gz
zcat abc

```
user1@myubuntu:~/linux_ex/ch9$ zcat ch2.tar.gz | more
ch2/

0
r1
127.0.1.1      myubuntu

# The following lines are desirable for IPv6 capable hosts
::1          ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
(생략)
```

04 파일 아카이브와 압축

■ 압축 풀기 : gunzip

gunzip

- **기능** gzip으로 압축된 파일의 압축을 푼다.
- **형식** gunzip 파일명
- **사용 예** gunzip abc.gz
gunzip abc

```
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar.bz2  ch2.tar.gz  hosts
user1@myubuntu:~/linux_ex/ch9$ gunzip ch2.tar.gz
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.bz2  hosts
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ bzip2/bunzip2 : .bz2 파일

bzip2

- **기능** 파일을 압축한다.
- **형식** bzip2 [옵션] 파일명
- **옵션**
 - d: 파일 압축을 해제한다.
 - l: 압축 파일의 정보를 보여준다.
 - t: 압축 파일을 검사한다.
 - v: 압축 정보를 화면에 출력한다.
 - best: 최대한 압축한다.
- **사용 예**
bzip2 abc.txt
bzip2 -v a.txt b.txt

```
user1@myubuntu:~/linux_ex/ch9$ rm ch2.tar.bz2
rm: 일반 파일 'ch2.tar.bz2'를 제거할까요? y
user1@myubuntu:~/linux_ex/ch9$ bzip2 ch2.tar
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar.bz2  hosts
user1@myubuntu:~/linux_ex/ch9$
```

04 파일 아카이브와 압축

■ 압축 파일의 내용 보기 : bzip2

bzcat

- **기능** bzip2로 압축된 파일의 내용을 출력한다.
- **형식** bzcat 파일명
- **사용 예** bzcat abc.bz2
bzcat abc

■ 압축 풀기 : bunzip2

bunzip2

- **기능** bzip2로 압축된 파일의 압축을 푼다.
- **형식** bunzip2 파일명
- **사용 예** bunzip2 1.c.bz2
bunzip2 1.c

```
user1@myubuntu:~/linux_ex/ch9$ bunzip2 ch2.tar.bz2
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  hosts
user1@myubuntu:~/linux_ex/ch9$
```

05 소프트웨어 컴파일

■ 컴파일러 설치하기

- C 언어로 작성한 프로그램을 컴파일하기 위해서는 C 컴파일러가 필요
- 리눅스에서 사용하는 C 컴파일러는 GNU C 컴파일러로 패키지 이름이 gcc
- gcc 설치

```
user1@myubuntu:~/linux_ex/ch9$ aptitude show gcc
Package: gcc
Version: 4:7.2.0-1ubuntu1
State: not installed
Priority: 옵션
Section: devel
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
(생략)
gcc-7 (7.2.0-8ubuntu3) 설정하는 중입니다 ...
gcc (4:7.2.0-1ubuntu1) 설정하는 중입니다 ...
Processing triggers for libc-bin (2.26-0ubuntu2) ...
user1@myubuntu:~/linux_ex/ch9$
```

05 소프트웨어 컴파일

■ C 프로그램 작성하기

```
user1@myubuntu:~/linux_ex/ch9$ vi hello.c

#include <stdio.h>

int main() {
    printf("Hello, World.\n");
}

:wq
```

■ C 프로그램 컴파일하기: 실행파일명은 a.out

```
user1@myubuntu:~/linux_ex/ch9$ gcc hello.c
user1@myubuntu:~/linux_ex/ch9$ ls
a.out  ch2  ch2.tar  hello.c  hosts
user1@myubuntu:~/linux_ex/ch9$
```

■ C 프로그램 실행하기: 경로 지정 확인

```
user1@myubuntu:~/linux_ex/ch9$ ./a.out
Hello, World.
user1@myubuntu:~/linux_ex/ch9$
```

05 소프트웨어 컴파일

■ 실행 파일명 변경하기

- gcc로 생성한 기본 실행 파일은 a.out
- 사용자가 원하는 이름으로 지정하려면 -o 옵션 사용

```
user1@myubuntu:~/linux_ex/ch9$ gcc -o hello hello.c
user1@myubuntu:~/linux_ex/ch9$ ./hello
Hello, World.
user1@myubuntu:~/linux_ex/ch9$
```


05 소프트웨어 컴파일

■ make 명령 사용하기

- make 명령은 makefile(또는 Makefile)에 설정된 정보를 읽어서 여러 소스 파일을 컴파일하고 링크하여 최종 실행 파일을 생성
- 소스파일 준비

```
user1@myubuntu:~/linux_ex/ch9$ vi one.c
#include <stdio.h>

extern int two();

int main() {
    printf("Go to Module Two--\n");
    two();
    printf("End of Module One.\n");
}
:wq
```

- 컴파일 하면 오류 발생: two()가 무엇인지 모르겠다는 메시지

```
user1@myubuntu:~/linux_ex/ch9$ gcc one.c
/tmp/ccc00qoc.o: In function `main':
one.c:(.text+0x16): undefined reference to `two'
collect2: error: ld returned 1 exit status
user1@myubuntu:~/linux_ex/ch9$
```

05 소프트웨어 컴파일

■ make 명령 사용하기

- 두 번째 파일 생성: two() 함수 정의

```
user1@myubuntu:~/linux_ex/ch9$ vi two.c
#include <stdio.h>

int two() {
    printf("In Module Two--\n");
    printf("--- This is a Moudule Two.\n");
    printf("End of Module Two.\n");
}

:wq
```

05 소프트웨어 컴파일

■ makefile 작성하기

```
user1@myubuntu:~/linux_ex/ch9$ vi makefile
TARGET=one
OBJECTS=one.o two.o

${TARGET} : ${OBJECTS}
    gcc -o ${TARGET} ${OBJECTS}

one.o : one.c
    gcc -c one.c
two.o : two.c
    gcc -c two.c

:wq
```

05 소프트웨어 컴파일

■ make 파일 실행

```
user1@myubuntu:~/linux_ex/ch9$ make
gcc -c two.c
gcc -o one one.o two.o
user1@myubuntu:~/linux_ex/ch9$
```

- 실행 파일의 이름을 one으로 했으므로 다음과 같이 실행

```
user1@myubuntu:~/linux_ex/ch9$ ./one
Go to Module Two--
In Module Two--
--- This is a Moudule Two.
End of Module Two.
End of Module One.
user1@myubuntu:~/linux_ex/ch9$
```