information technology & management

viabiliitjiit

ITMO456

INTRO TO OPEN SOURCE
OPERATING SYSTEMS

ILLINOIS INSTITUTE OF TECHNOLOGY

Compression, System Backup, & Software Installation

Sean Hughes-Durkin

ITMO/IT-O 456 Fall 2017

Information Technology & Management Programs

School of Applied Technology

Objectives

At the end of this lesson students should be able to:

- Describe common types of Linux software
- Use the graphical installer to install, manage, & remove software packages
- Use the Red Hat Package Manager (rpm) to install, manage, & remove software packages

Objectives

At the end of this lesson students should be able to:

- Use Yellow dog Updater, Modified (yum) to install, manage, & remove software packages
- Use the Advanced Packaging Tool (apt) to install, manage, & remove software packages
- Use DNF to install, manage, & remove software packages

Objectives

At the end of this lesson students should be able to:

- Compile and install software packages from source code
- Outline the features of common compression utilities
- Compress and decompress files using common compression utilities

Objectives

At the end of this lesson students should be able to:

- Perform system backups using the **`tar`**, **`cpio`**, and **`dump`** commands
- View and extract archives using the **`tar`**, **`cpio`**, and **`restore`** commands

Software Installation

- Software for Linux can consist of:
  - Binary files precompiled to run on certain hardware architectures
  - Source code, which must be compiled
    - Typically distributed in *tarball* format
      - `.tar.gz`
      - `.tgz`
      - `.tar.Z`
      - `.tar.bz2`
      - `.tbz2`

Software Packages

- A package is a collection of files to be installed on the computer
  - One application may involve a hundred files and all come together as a *package*
- Package manager
  - System that defines a standard package format
  - Used to install, query, & remove packages
  - Also manages database of available and installed packages

Understanding Linux Software Packaging

- Before RPM and DEB packaging there was… tarball packaging
- Tarball packaging:
  - A single file
    - into which multiple files are gathered
    - allowed convenient storage and distribution
  - File contained
    - Executable files
    - Documentation
    - Configuration files
    - Libraries

Tarball Packaging Weaknesses

- Tarball may not include needed additional software packages (dependent software)
- Could not see where items were installed
- No easy steps to remove software
- No way to tell if software needed updating

Complex Software Packaging

- Created to address tarball packaging weaknesses
- DEB (.deb) packaging.
  - Created by Debian GNU/Linux project
  - Used by Debian Linux distribution
  - Used by Debian-based distributions (example: Ubuntu)

Complex Software Packaging

- RPM (.rpm) packaging
  - Used by SUSE, RHEL, and Fedora Linux distros
  - Used by Red Hat-based distributions (example: CentOS)

Package Manager & Dependencies

- Many applications in Linux depend on presence of other applications and/or shared libraries
  - This creates *dependencies*, i.e. a set of files that the package you're installing depends on being present
- As well as managing the database of packages, the package manager resolves dependencies at package installation

Package Managers

- Red Hat Package Manager (`RPM`)
  - Most commonly used package manager for Linux
  - Command line program with a variety of graphical front ends available
  - Used in Red Hat and Fedora; also in Mandriva, Yellow Dog, and SUSE

Package Managers

- Debian package management system
  - **dpkg**, a low-level tool, is at the core, but is generally only used in conjunction with other tools
  - **dpkg** is normally used with the Advanced Packaging Tool (apt), which itself is used with other front-end tools such as aptitude or Synaptic
  - Debian packages have a **`.deb`** extension
  - Tools are used in Ubuntu

Managing Software with the GUI

- GNOME Software is the current GNOME graphical package for adding, removing and updating software
- Available for Fedora and RHEL
- Located in GNOME desktop: Activities → Show Applications → Software

LINUX & OPEN SOURCE

Managing Software with the GUI

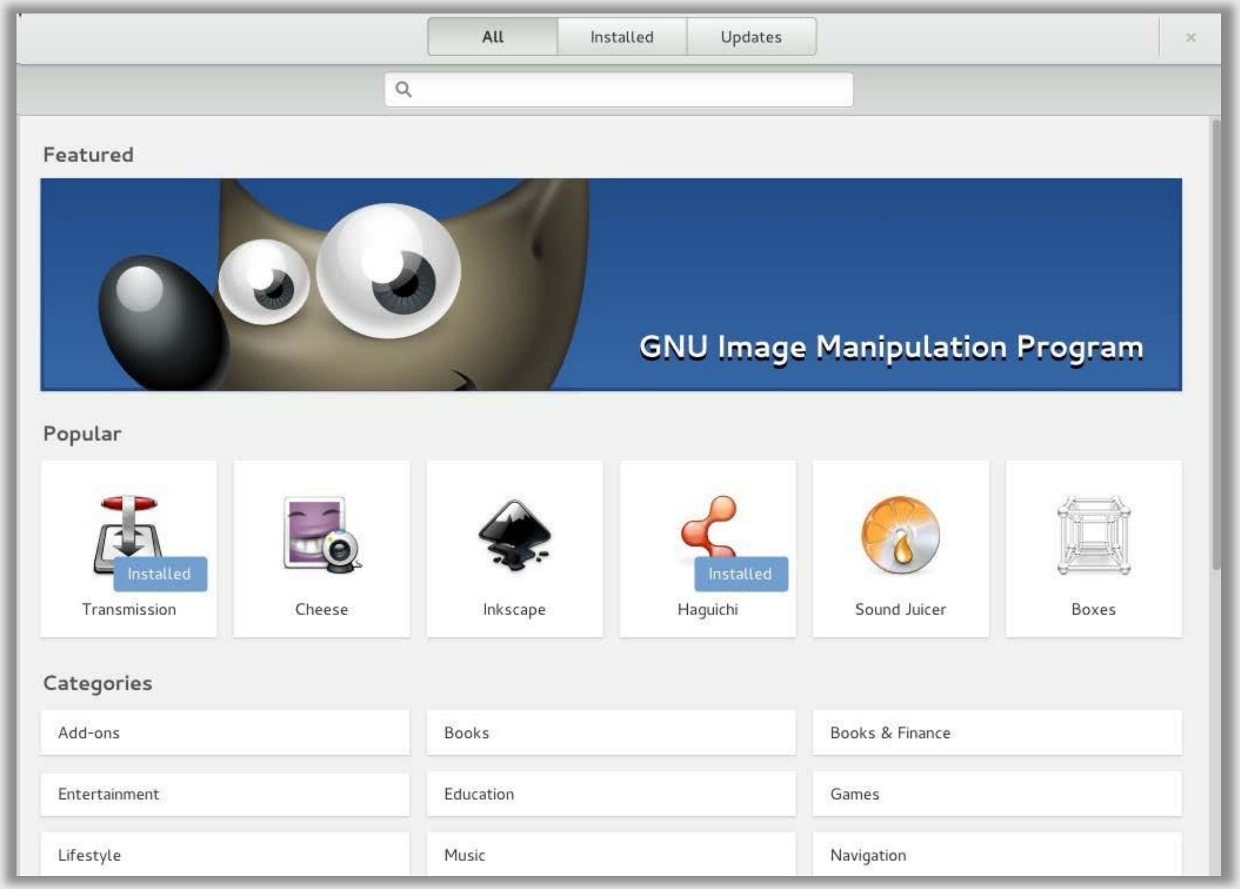


Figure 11-5: The Software utility

PackageKit and yumex

- GNOME Software replaced PackageKit
  - Software focuses on software rather than packages, as packages tended to confuse end users
    - Will not list or install command-line apps
  - PackageKit is still installed but GUI is not; it can be installed with
    `yum install gnome-packagekit-installer`
  - Also can install **yumex**: *Yum Extender*

Going beyond Software or PackageKit

- Thousands of packages available for Linux
- Why go beyond using GUI tools?
  - More repositories of software packages
  - More complex queries available at command line
  - Software validation available
  - Managing software installations on multiple systems

Compiling Source Code into Programs

- Procedure for compiling source code into binary programs standardized today among most OSS developers
- GNU C Compiler (**`gcc`**)
  - Used to compile source code into binary programs
  - Accessed by the **`make`** command
  - After compilation, must move program files to appropriate directory

Compiling Source Code into Programs

- `make` command
  - Looks for Makefile & uses it to compile source code into binary using compiler
- Makefile
  - Contains most information and commands necessary to compile the program
- `make install` command
  - Copies complied executable programs to correct location

Compiling Source Code into Programs

- Dependencies
  - Resolving dependencies is the toughest part of installation by compilation

Compiling Source Code into Programs

- Step-by-step:
  - Download and save package
    ```
    rdesktop-1.7.1.tar.gz
    ```
  - Extract the file using the **`tar`** command with the **`xvzf`** options:
    ```
    tar xvzf rdesktop-1.7.1.tar.gz
    ```
  - Change to the directory created when the file was extracted:
    ```
    cd rdesktop-1.7.1
    ```

Compiling Source Code into Programs

- Step-by-step:
  - Enter the command **./configure:**

```
./configure
```

    (this generates a compiler configuration for your system)

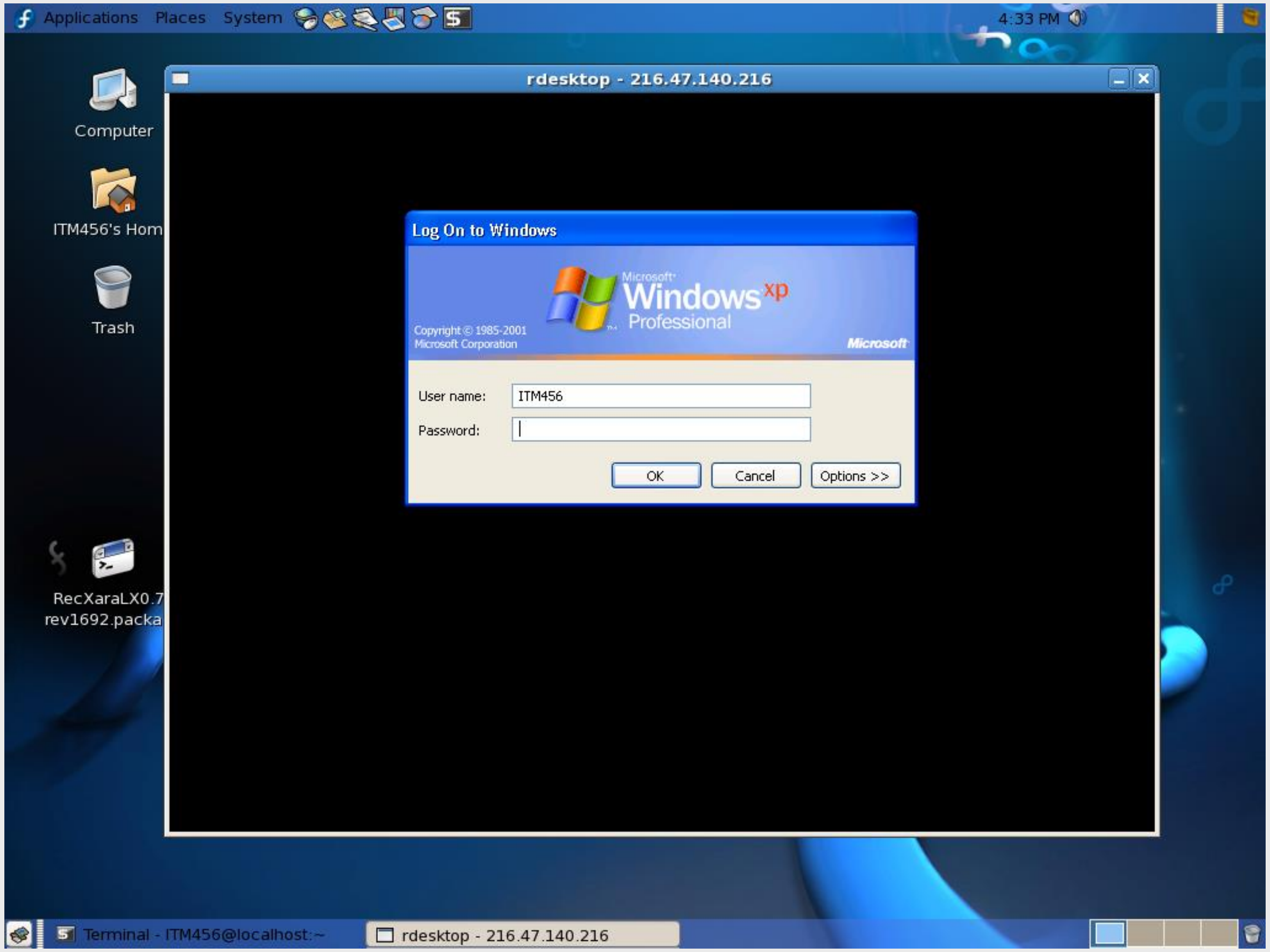
  - We may have to install some dependencies before we can compile:

```
yum install libX11-devel
yum install openssl-devel
```

Compiling Source Code into Programs

- Step-by-step:
  - Compile using the **`make`** command:
    ```
    make
    ```
  - Install the program using the **`make install`** command:
    ```
    make install
    ```

Compiling Source Code into Programs



User name: ITM456

Password:

Figure 11-3: The rdesktop program

Installing Programs Using RPM

- RPM format packages have filenames indicating the hardware architecture the software was compiled for
  - End with the **`.rpm`** extension
- **`rpm`** command
  - Command used to install, query, and remove RPM packages; **`–i`** option installs
  - For example, to install bluefish in Fedora:

```
rpm –ivh bluefish-2.2.6-3.fc20.x86_64.rpm
```

  - **`rpm`** normally does not resolve dependency issues

Installing Programs Using RPM

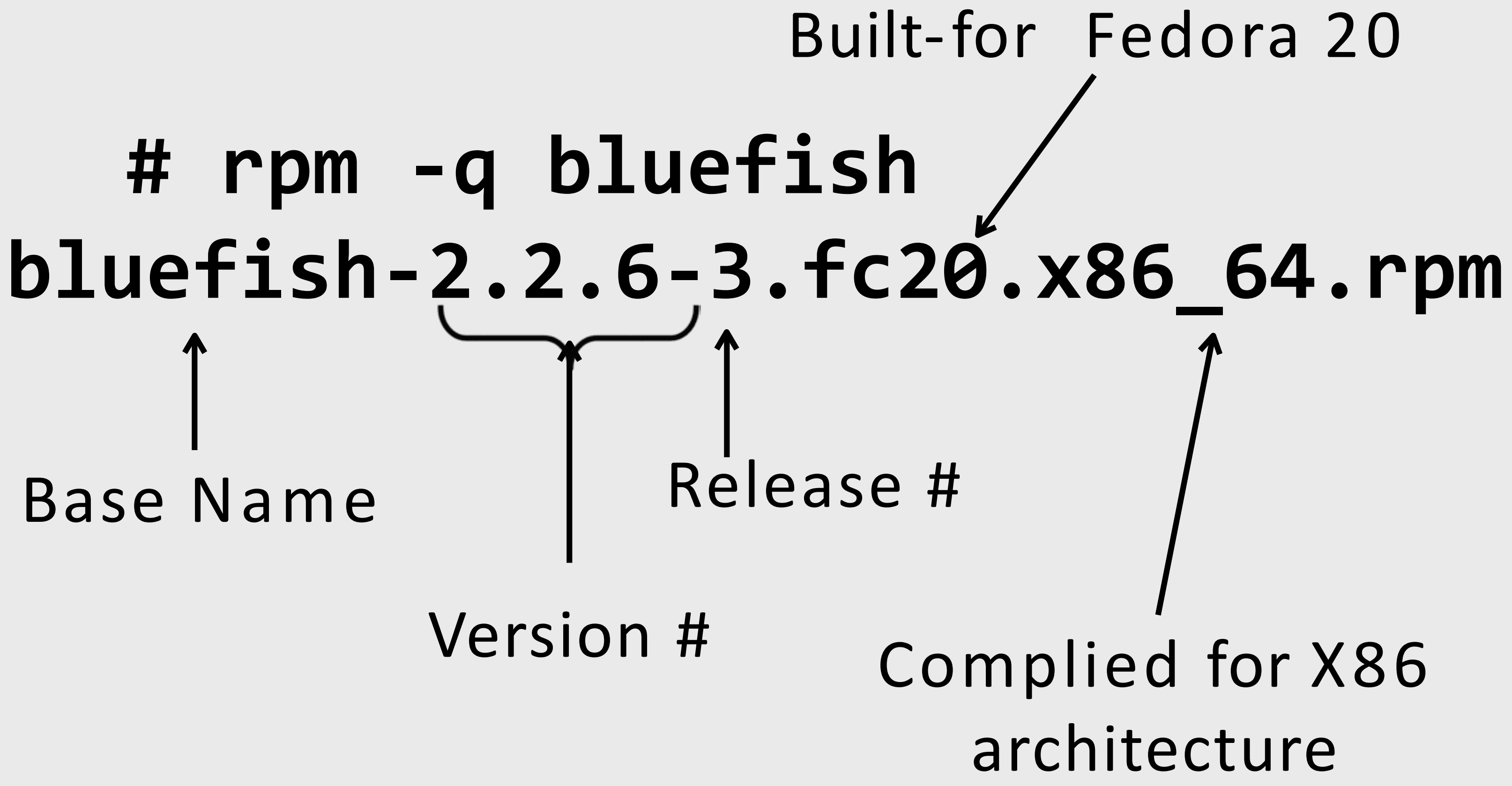
- Package dependency: some RPM packages require other RPM packages be installed on your system first
  - You receive an error message that indicates the RPM package that needs to be installed first
  - After installing the prerequisite packages, you can successfully install your desired RPM package

Installing Programs Using RPM

- After installation RPM database is updated to contain information about the installed package and files contained in it
  - **`-q`** option: query the full package name
  - **`-i`** option: together with **`–q`** used to display full package information
  - **`-f`** option: together with **`–q`** used to display the package to which a specific file belongs
- **`-e`** option: used to remove a package from the system

ILLINOIS INSTITUTE OF TECHNOLOGY
School of Applied Technology

Name of installed RPM package Example



```
# rpm -q bluefish
bluefish-2.2.6-3.fc20.x86_64.rpm
```

Installing Programs Using RPM

Option	Action
-U **-upgrade**	Updates package; actually installs it even if the package is not already installed
-F **--freshen**	Freshen; updates package only if it is already installed
-i **--install**	Installs package
-e	Removes (uninstalls) a package
-V **--verify**	Verifies that a package is present and unchanged since installation
-q **--query**	Queries a package; is it installed, what files are in it, etc.
-b	Builds a binary package
-K	Authenticates and performs integrity check on a package

Table 11-8: Most commonly used RPM operations

Installing Programs Using RPM

Option	
-a **--all**	When used with the -q option, displays all package names installed on the system
-f **--file**	When used with the -q option, displays the package that the specified file belongs to
-l **--list**	When used with the -q option, displays the files included in an installed package
-i **--info**	When used with the -q option, displays full information about the specified package
-R **--requires**	When used with the -q option, displays all packages on which this one depends, i.e. a dependency list
-pl	When used with the -q option, displays the files included in the specified uninstalled package
-pi	When used with the -q option, displays full information about the specified uninstalled package

Table 11-8: Most commonly used RPM operations

Installing Programs Using RPM

Option	Description
-h **--hash**	When used with the -i option, prints hash marks on the screen to indicate installation progress
--test	When used with the -i option, performs a test installation only
-v **--verbose**	Prints verbose information when installing or manipulating packages
--force	Forces installation even it overwrites existing files or packages
--nodeps	Installs packages with no dependency checks

Table 11-8: Most commonly used RPM operations

Where do packages come from?

- Thousands of open source projects all over the world
- Projects are referred to as “upstream software providers”
- A Linux distribution
  - Obtains source code
  - Builds code into binaries

Where do packages come from?

- A Linux distribution
  - Gathers together from upstream provider
    - Documentation
    - Configuration files
    - Additional components
  - Packages items into an RPM or DEB package
  - The package is signed
  - The package is placed into a repository

Installing RPMs

- The rpm command was first tool used to install RPM packages
- Major drawbacks of rpm command:
  - If dependent software package not installed, installation will fail
  - Must provide exact location of RPM file to perform installation

Installing Programs Using RPM

- Most RPM packages are located on Internet Servers
  - Called software repositories
- **yum** command
  - Used to search online repositories for RPM packages & install the packages
  - Treats RPM packages as part of larger software repositories, not as individual components’

Installing Programs Using YUM

- **`yum`** command
  - Start the **Yellow dog Updater, Modified**
  - Automatic updater and package installer/remover for rpm systems
  - Automatically computes dependencies and determines what should occur to install packages
  - Requires an accompanying command
  - Basic syntax: **`yum options packagename`**
  - Example: **`yum install firefox`**

Installing Programs Using YUM

- Runs neck-and-neck with **apt** as best command to install/update packages
- User only need to know package name
- YUM installation process
  - Finds latest version of package in repository
  - Downloads package to local system.
  - Install package on local system
  - Package information stored in local RPM database

Installing Programs Using YUM

- **`yum check-update`** command
  - Loads headers of all packages
    - Can be up to 30 or 40 MB
  - Necessary, but only needs to be done once
    - After this list is updated with each update
- **`yum update`** command
  - Causes yum to compare downloaded headers with those on the server and offer to update your system
  - A **y** response causes yum to fetch and install updates

Updating Programs Using YUM

- **`yum update`** run on a raw Fedora installation may find as many as 250 installed packages require updates
  - With dependencies, 450 packages may be updated
- Both YUM and apt attempt to resolve all dependencies at installation time

Managing Software with **`yum`**

- **`yum search name`**
  - Searches through repositories for a package whose name is or description contains the word “name”

Managing Software with **`yum`**

- **`yum info`** ***`packagename`***
  - Provides information about the *packagename*

Managing Software with **`yum`**

- **`yum provides`** ***`libraryname`***
  - Tells what software package provides *libraryname*, where *libraryname* can be the name of a
    - command
    - configuration file, or
    - library name

Managing Software with **`yum`**

- **`yum list installed`**
  - Displays all installed software packages
  - Example:
- **`yum list all`**
  - Displays all packages, installed and available

Managing Software with **`yum`**

- **`yum deplist`** ***`packagename`***
  - Provides what components a particular*packagename* is dependent upon

Installing Programs Using YUM

- **`yum install packagename`** command
  - yum connects to the server, finds the named package, retrieves it and installs it automatically
- **`yum update packagename`** command
  - yum connects to the server and checks for an update to the named package
  - If there is an update, yum retrieves it and installs it automatically
- **`yum`** can be run using **`cron`** to allow automatic scheduled updates of installed packages
  - **`yum install yum-cron`**

Installing Programs Using YUM

- **`yum reinstall`** ***`packagename`***
  - Reinstalls a software *packagename*
  - Useful if you mistakenly delete components of an installed package
- **`yum erase`** ***`packagename`***
  - Removes or uninstalls software packagename
- **`yum history`**
  - Displays all yum activities on the system

Installing Programs Using YUM

- Yum repositories
  - Can be in **`/etc/yum.conf`** but preferred is in files named **`repositoryname.repo`** in **`/etc/yum.repos.d`**
    - Each refers to a different yum repository
  - Most popular Fedora repository is RPM Fusion
    - Merger of FreshRPMs and Livna.org
    - Promise: will never replace packages already in Fedora

Installing GPG Keys to Use YUM

- Using additional Yum repositories requires installation of GPG keys
  - Determine the URL of the GPG key from the repository's Web site
  - At the command line run
    `rpm --import GPGkeyURL`
    (substituting the URL of the GPG key for **`GPGkeyURL`**)

Update Package Groups with **`yum`**

- YUM supports package groups
  - Package group is an entire set of software packages
  - Example: The Virtualization group contains software packages needed to set up a computer as a virtual host
- Maintaining a package group is easier than dealing with the individual software packages which make up that group

Update Package Groups with **yum**

- **`yum grouplist`**
  - Shows a complete list of software package groups
- **`yum groupinfo`** ***`packagegroupname`***
  - Displays detailed information about a particular software package group
- **`yum groupinstall`** ***`packagegroupname`***
  - Install a particular software package group
- **`yum groupremove`** ***`packagegroupname`***
  - Uninstalls a particular software package group

Maintain RPM Package DB & Cache

- Maintenance tasks may involve:
  - RPM database problem checks and fixes
  - Clear out metadata cache
  - Removing unneeded downloaded package files
- **`yum clean packages`**
  - Remove unneeded package files from system

Maintain RPM Package DB & Cache

- **`yum clean metadata`**
  - Deletes unneeded metadata from
  - **`/var/cache/yum`**
- **`yum check`**
  - Reviews RPM database for errors
- **`yum clean rpmdb`**
  - Cleans out and rebuilds the RPM database

Download RPMs from a repository

- **`yumdownloader`** ***`packagename`***
- Downloads RPM package from a YUM repository, but does ***not*** install it

YUM Replacement: DNF

- Fork of YUM
- Replaced YUM in Fedora 22
- Commands same as YUM
- Fully compatible with current version of Python (v3)
- https://fedoraproject.org/wiki/Yum_to_DNF_Cheatsheet

YUM Replacement: DNF

- Yum was forked into DNF for these main reasons:
  - An undocumented API—this meant more work for developers
    - In order for developers to do what they needed, it was often necessary to browse through the Yum code base just to be able to write a call
    - This meant development was very slow
  - Python 3—Fedora was about to make the shift to Python 3 and Yum wouldn't survive this change, whereas DNF can run using either Python 2 or 3
  - Broken dependency solving algorithm—this has been an Achilles heel of the Fedora package manager for a long time
    - DNF uses a state-of-the-art satisfiability (SAT)-based dependency solver
    - Same type of dependency solver used in SUSE's and openSUSE's Zypper

Installing Programs Using APT

- Advanced Packaging Tool: **`apt`**
  - Debian package management tool that can be used with either **`rpm`** or **`dpkg`**
  - Default package tool in Ubuntu
  - From the command line uses **`apt-get`** which can download and install software and even compile source code
  - Package installation with **`apt-get`**:

```
apt-get install packagename
```

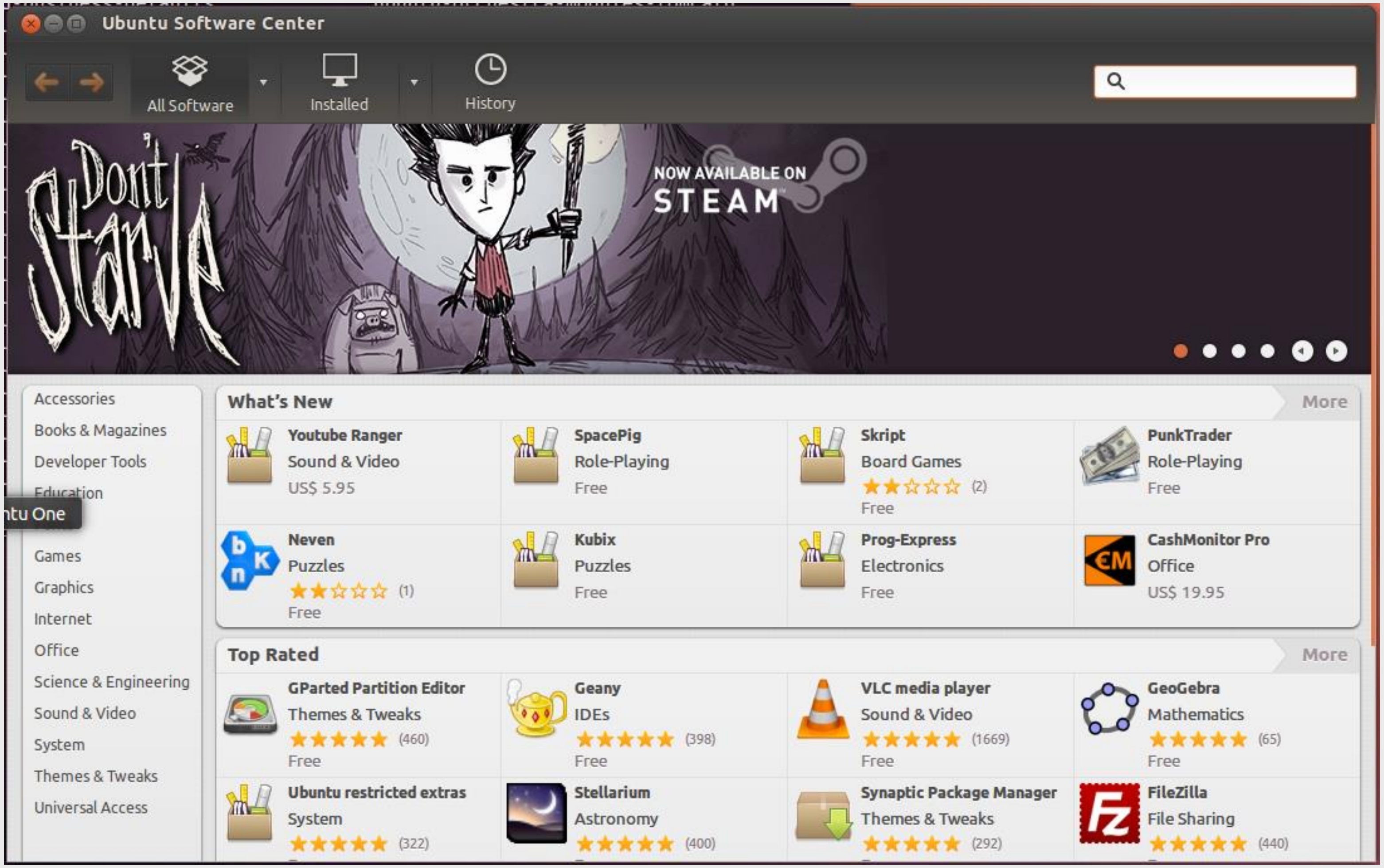
Installing Programs Using APT

- Advanced Packaging Tool: **`apt`**
  - Upgrade all packages with **`apt-get`**:
    `apt-get –u upgrade`
    - **`–u`** will list packages as they are upgraded
  - Upgrade to a new release of the distribution with **`apt-get`**:
    `apt-get dist-upgrade`
  - Uninstall a package with **`apt-get`**:
    `apt-get remove packagename`

Installing Programs in Ubuntu

- Add new repositories using
- **`add-apt-repository`**
  - Or search available repository information using apt-cache
- Ubuntu also has an **`apt-url`** function that allows you to install apps from Web pages; try **`http://appnr.com/`**

Installing Programs in Ubuntu



What's New

Youtube Ranger Sound & Video US$ 5.95	SpacePig Role-Playing Free	Skript Board Games ★★☆☆☆ (2) Free	PunkTrader Role-Playing Free
Neven Puzzles ★★☆☆☆ (1) Free	Kubix Puzzles Free	Prog-Express Electronics Free	CashMonitor Pro Office US$ 19.95

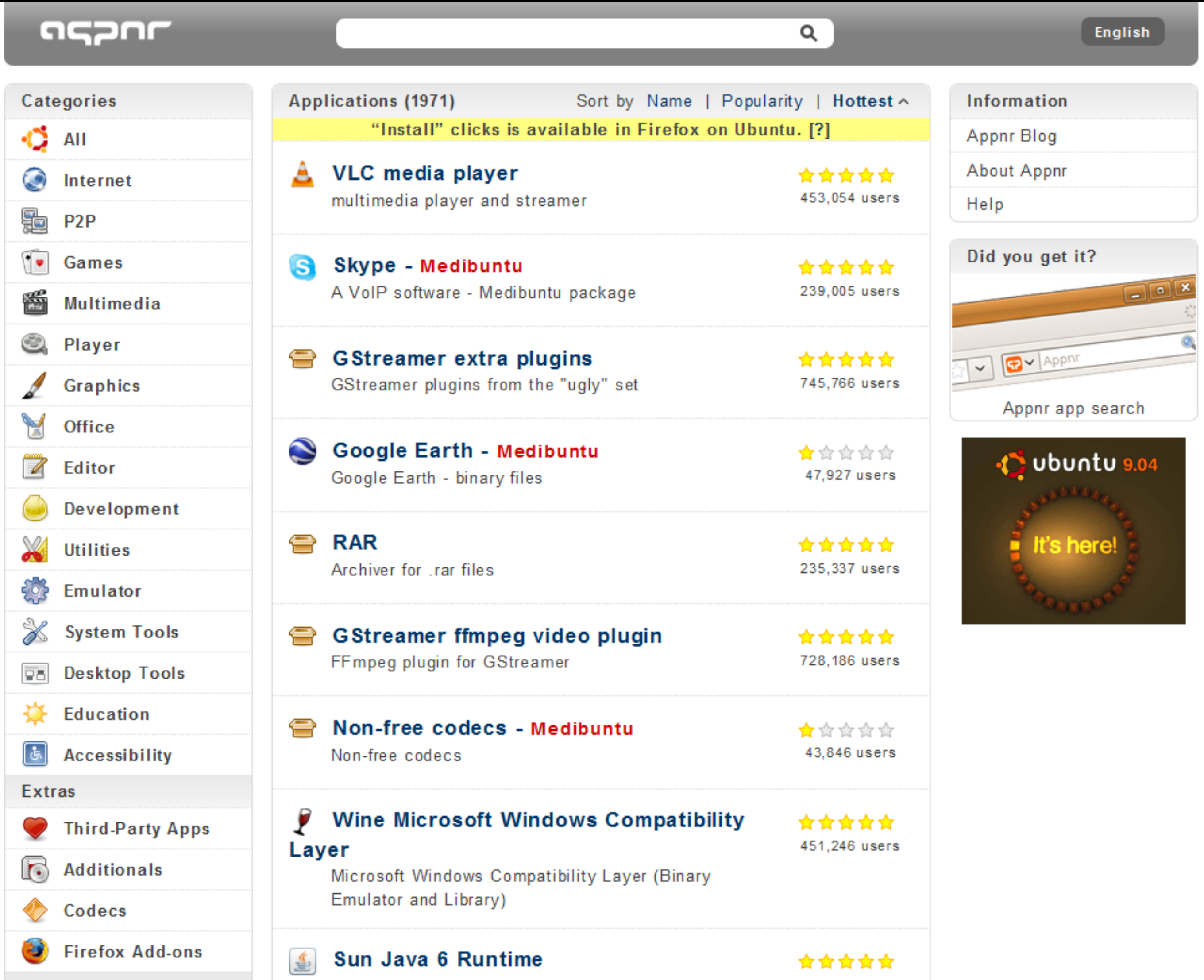
Top Rated

GParted Partition Editor Themes & Tweaks ★★★★★ (460) Free	Geany IDEs ★★★★★ (398) Free	VLC media player Sound & Video ★★★★★ (1669) Free	GeoGebra Mathematics ★★★★★ (65) Free
Ubuntu restricted extras System ★★★★★ (322)	Stellarium Astronomy ★★★★★ (400)	Synaptic Package Manager Themes & Tweaks ★★★★★ (292)	FileZilla File Sharing ★★★★★ (440)

Ubuntu Software Center

Installing Programs in Ubuntu

appnr.com
web-based
repository
for Ubuntu



Applications (1971) Sort by Name | Popularity | Hottest

"Install" clicks is available in Firefox on Ubuntu. [?]

VLC media player	multimedia player and streamer	453,054 users
Skype - Medibuntu	A VoIP software - Medibuntu package	239,005 users
GStreamer extra plugins	GStreamer plugins from the "ugly" set	745,766 users
Google Earth - Medibuntu	Google Earth - binary files	47,927 users
RAR	Archiver for .rar files	235,337 users
GStreamer ffmpeg video plugin	FFmpeg plugin for GStreamer	728,186 users
Non-free codecs - Medibuntu	Non-free codecs	43,846 users
Wine Microsoft Windows Compatibility Layer	Microsoft Windows Compatibility Layer (Binary Emulator and Library)	451,246 users
Sun Java 6 Runtime		

ITMO456

Installing Programs in Ubuntu

- Software repositories in Ubuntu are in **`/etc/apt/sources.list`**
  - Uncommenting the **`partners`** repository entry will expand available proprietary packages
- Adding additional (non-Ubuntu) reposi-tories requires installing GPG keys
  - See **`http://wiki.debian.org/SecureApt`** for details on adding keys

Compression

- Compression
  - Process in which files are reduced in size by a compression algorithm
- Compression algorithm
  - Set of instruction used to reduce the contents of a file systematically
- Compression ratio
  - Amount of compression that occurred during compression

Compression

- The three most common compression utilities available to Linux users:
  - `compress`
  - `gzip`
  - `bzip2`
  - `zip`

The `compress` Utility

- **`compress`** command
  - Compress files using Lempel-Ziv-Welch compression algorithm (was patented until 2003)
  - Yields 40-50% compression ratio
  - Files named with a **`.Z`** extension
  - Specify files to compress as arguments
- **`zcat`** command
  - View contents of an archive created with **`compress`** or **`gzip`** to Standard Output

The **`compress`** Utility

- **`zmore`** and **`zless`** commands
  - View the contents of an archive created with **`compress`** or **`gzip`** to Standard Output in a page-by-page fashion
- **`uncompress`** command
  - Decompress files compressed by the **`compress`** command
- More: **`http://en.wikipedia.org/wiki/Compress`**

The **`compress`** Utility

Option	Description
-c	When used with the uncompress command, displays the contents of the compress file to Standard Output (same function as the zcat command)
-f	When used with the compress command, can be used to compress symbolic links; when used with the uncompress command, overwrites any existing files without prompting the user
-r	Specifies whether to compress or decompress all files recursively within a specified directory
-V	Displays verbose output (compression ratio and filenames) during compression and decompression

Table 11-1: Common options used with the **`compress`** utility

The **`gzip`** Utility

- GNU zip (**`gzip`**)
  - Compress files using an unpatented Lempel-Ziv compression algorithm
    - Called DEFLATE; varies slightly from algorithm used by **`compress`**
  - Uses **`.gz`** filename extension by default
  - Can control level of compression
  - Typically, yields better compression than **`compress`** (60-70%)

The **gzip** Utility

- **zcat**, **zmore** & **zless** commands
  - View contents of an archive created with **compress** or **gzip** to Standard Output in complete or page-by-page display
- **gunzip** command
  - decompress **.gz** or **.Z** files
- Advantage of compress: control level of compression via numeric option
- More: **http://www.gzip.org/**

The **`gzip`** Utility

Option	Description
-#	Specifies how thorough the compression will be, where # may be the number 1-9 (the option -1 represents fast compression, which takes less time to compress but results in a lower compression ratio; the option -9 represents thorough compression, which takes more time but results in a higher compression ratio)
-best	Same as the -9 option; results in a higher compression ratio
-c **--stdout** **--to-stdout**	When used with the gunzip command, displays the contents of the compressed file to Standard Output (same function as the zcat command)
-d **--decompress** **--uncompress**	When used with the gzip command, decompresses the files specified (same as the gunzip command)
-f **--force**	When used with the gzip command, can be used to compress symbolic links; when used with the gunzip command, overwrites any existing files without prompting the user
-fast	Same as the -1 option; results in a lower compression ratio

Table 11-2: Common options used with the **`gzip`** utility

The `gzip` Utility

Option	Description
-h **--help**	Displays the syntax and available options for the gzip and gunzip commands
-l **--list**	Lists the compression ratio for files that have been compressed with gzip
-n **--no-name**	Does not allow gzip and gunzip to preserve the original modification and access time for files
-q **--quiet**	Suppresses all warning messages
-r **--recursive**	Specifies to compress or decompress all files recursively within a specified directory
-S .suffix **--suffix .suffix**	Specifies a file suffix other than .gz when compressing or decompressing files
-t **--test**	When used with the gunzip command, performs a test decompression such that a user may view any error messages before decompression; does not decompress files
-v **--verbose**	Displays verbose output (compression ratio and filenames) during compression and decompression

Table 11-2: Common options used with the **`gzip`** utility

The **`bzip2`** Utility

- **`bzip2`** command
  - Compress files using a Burrows-Wheeler Block Sorting Huffman Coding compression algorithm
  - Cannot compress directory full of files
  - Cannot use **`zcat`** and **`zmore`** commands to view files zipped with **`bzip2`**
  - Files typically have a **`.bz2`** extension
  - Compression ratio averages 50-75%

The **`bzip2`** Utility

- **`bzcat`** command
  - View the contents of an archive created with **`bzip2`** to Standard Output
- **`bunzip2`** command
  - Decompress files compressed by **`bzip2`**
- More: **`http://www.bzip.org/`**

The **`bzip2`** Utility

Option	Description
-#	Specifies the block size used during compression; -1 indicates a block size of 100KB whereas -9 indicates a block size of 900KB
-c **--stdout**	When used with the bunzip2 command, displays the contents of the compressed file to Standard Output
-d **--decompress**	When used with the bzip2 command, decompresses the files specified (same as the bunzip2 command)
-f **--force**	When used with the bzip2 command, can be used to compress symbolic links; when used with the bunzip2 command, overwrites any existing files without prompting the user
-k **--keep**	Keeps the original file during compression; a new file will be created with the extension .bz2
-q **--quiet**	Suppresses all warning messages
-s **--small**	Minimizes memory usage during compression
-t **--test**	When used with the bunzip2 command, performs a test decompression such that a user may view any error messages before decompression; does not decompress files
-V **--verbose**	Displays verbose output (compression ratio) during compression and decompression

Table 11-3: Common options used with the **`bzip2`** utility

`.zip` Files in Linux

- `.zip` is a very common file compression format primarily used in DOS/Windows
  - Supported in Linux GUIs by both the Gnome and KDE file managers
  - Generally uses the DEFLATE algorithm but can also use bzip2
  - Linux includes command-line zip utilities

.**zip** Command Line Utilities

- **zip** command
  - Compress a file using the .zip method
- **unzip** command
  - Decompress files compressed by **zip**
- More: **http://en.wikipedia.org/wiki/ZIP_file_format**

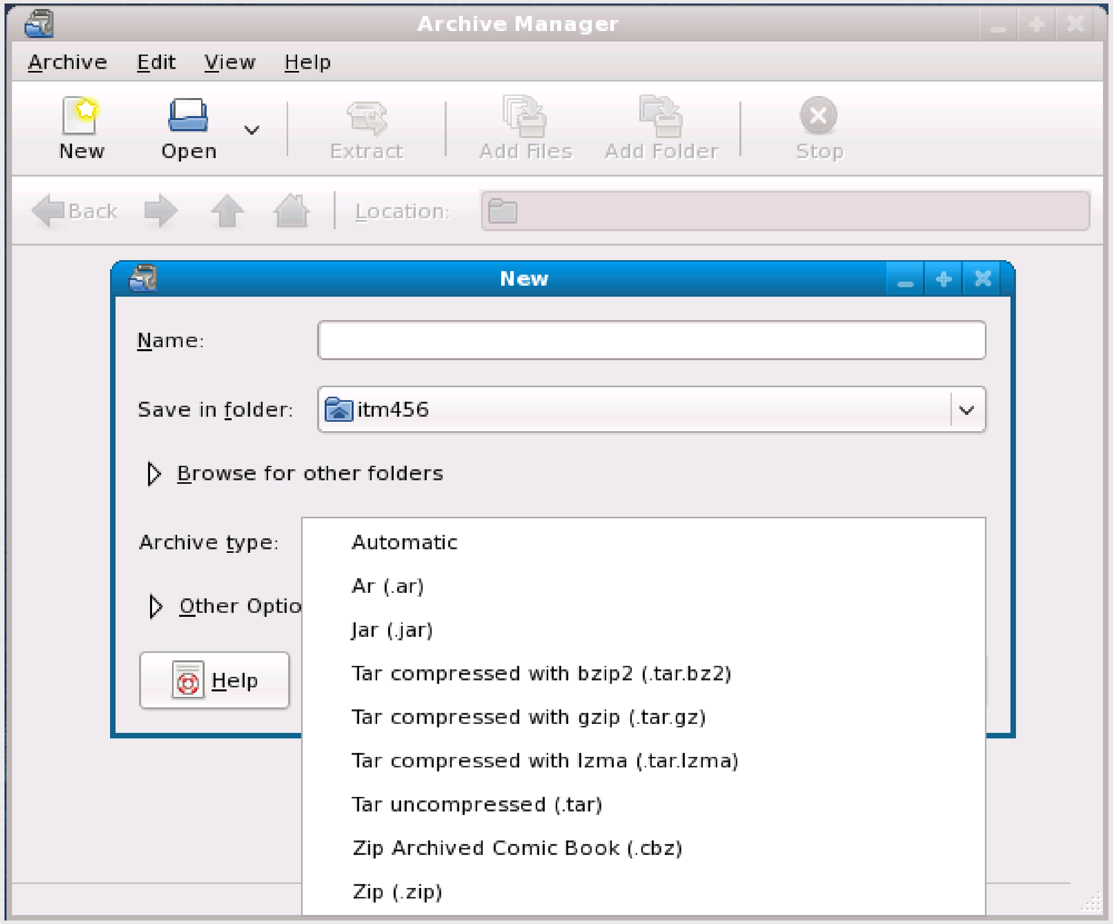
`.7z` Files in Linux

- 7zip is a compression format gaining popularity due to very high compression ratios & an outstanding free Windows application
  - Files normally have **`.7z`** file extension
- Requires installation of **`p7zip`** to unzip; see **`http://sourceforge.net/projects/p7zip/`**
- Binaries by distro at **`http://www.7-zip.org/download.html`**

Archive Manager GUI Tool

- The GUI tool Archive Manager can be used to create, extract from, and manage archive files
  - Uses a variety of file formats
  - Can create 8 formats and extract from most archive formats
  - Package name is **`Archive Manager`**
  - Can use **`p7zip`** if installed

Archive Manager GUI Tool



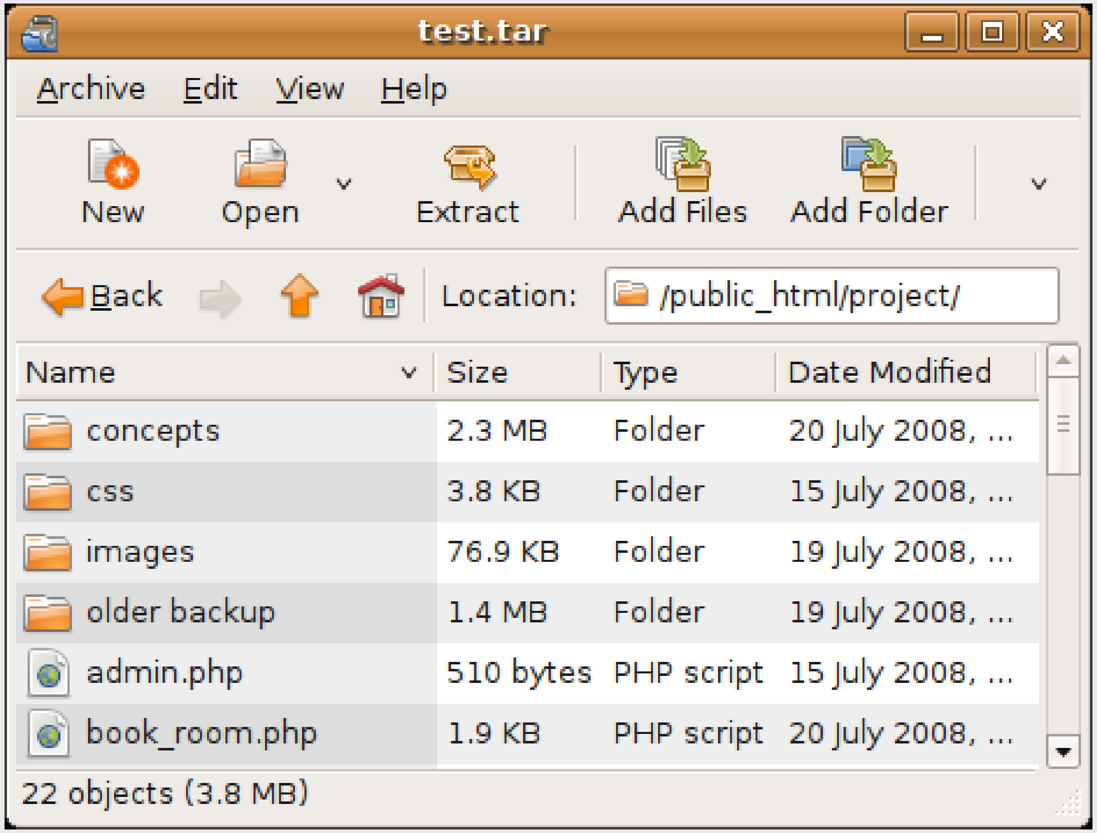
Name:

Save in folder: itm456

Archive type:
- Automatic
- Ar (.ar)
- Jar (.jar)
- Tar compressed with bzip2 (.tar.bz2)
- Tar compressed with gzip (.tar.gz)
- Tar compressed with lzma (.tar.lzma)
- Tar uncompressed (.tar)
- Zip Archived Comic Book (.cbz)
- Zip (.zip)

Archive Manager

Archive Manager GUI Tool



/public_html/project/

Name	Size	Type	Date Modified
concepts	2.3 MB	Folder	20 July 2008, ...
css	3.8 KB	Folder	15 July 2008, ...
images	76.9 KB	Folder	19 July 2008, ...
older backup	1.4 MB	Folder	19 July 2008, ...
admin.php	510 bytes	PHP script	15 July 2008, ...
book_room.php	1.9 KB	PHP script	20 July 2008, ...

22 objects (3.8 MB)

Archive Manager in Ubuntu

Archive Manager GUI Tool

Format	Filename Extension
ARJ archive	**.arj**
RAR / Zip archived comic book	**.cbr, .cbz**
Debian archives (Read-only mode)	**.deb**
ISO files (Read-only mode)	**.iso**
Java archives	**.jar, .ear, .war**
LHA archive	**.lzh**
Resource Adapter archive	**.rar**
RPM archives (Read-only mode)	**.rpm**
Uncompressed tar archive	**.tar**
Tar archive compressed with bzip	**.tar.bz or .tbz**
Tar archive compressed with bzip2	**.tar.bz2 or .tbz2**
Tar archive compressed with gzip	**.tar.gz or .tgz**
Tar archive compressed with lzop	**.tar.lzo or .tzo**
Tar archive compressed with compress	**.tar.Z or .taz**
PKZIP or WinZip archive	**.zip**
7-Zip archive	**.7z**
Zoo archive	**.zoo**

Archive Manager file formats

System Back-Up

- System back-up
  - Process whereby files are copied to an archive
- Archive
  - Location (file or device) that contains a copy of files
  - Typically created by a back-up utility
- Archives should be stored at an alternate location

System Back-Up

- Many types of media can be used to create archives
  - Tapes, CDs, DVDs, or hard disks
- Should backup user files from home directories and any important system configuration files
  - Files used by system services, as well
- Several backup utilities available
  - **`tar`**, **`cpio`**, **`dump/restore`**

System Back-Up

Device File	Description
/dev/st0	First SCSI tape device (rewinding)
/dev/st1	Second SCSI tape device (rewinding)
/dev/st2	Third SCSI tape device (rewinding)
/dev/nst0	First SCSI tape device (non-rewinding)
/dev/ht0	First ATAPI IDE tape device (rewinding)
/dev/nht0	First ATAPI IDE tape device (non-rewinding)
/dev/ftape	First floppy tape device

Table 11-4: Common tape device files

System Back-Up

- The most common back-up utilities:
  - `tar`
  - `cpio`
  - `dump/restore`

The **`tar`** Utility

- Tape archive (**`tar`**) utility
  - One of the oldest and most common back-up utilities
  - Can create an archive in a file on a filesystem or directly on a device
  - Arguments list the files to place in the archive
  - Accepts options to determine the location of the archive and the action to perform on the archive

The `tar` Utility

Option	Description
-A **--catenate** **--concatenate**	Appends whole archives to another archive
-c **--create**	Creates a new archive
-exclude *FILENAME*	Excludes *FILENAME* when creating an archive
-f *FILENAME* **--file *FILENAME***	Specifies the location of the archive *(FILENAME);* may be a file on a filesystem or a device file
-h **--dereference**	Will prevent tar from backing up symbolic links; instead, tar will back up the target files of symbolic links
-j **--bzip**	Compresses/decompresses the archive using the bzip2 utility
-P **--absolute-paths**	Stores filenames in an archive using absolute pathnames
-r **--append**	Appends files to an existing archive
--remove-files	Removes files after adding them to an archive
-t **--list**	Lists the filename contents (table of contents) of an existing archive

Table 11-5: Common options used with the `tar` utility

The `tar` Utility

Option	Description
-u **--update**	Appends files to an existing archive only if they are newer than the same filename inside the archive.
-v **--verbose**	Displays verbose output (file & directory information) when manipulating archives
-w **--interactive** **--confirmation**	Prompts the user for confirmation of each action
-W **--verify**	Verifies the contents of each archive after creation
-x **--extract** **--get**	Extracts the contents of an archive
-z **--gzip** **--ungzip**	Compresses/decompresses the archive using the gzip utility
-Z **--compress** **--uncompress**	Compresses/decompresses the archive using the compress utility

Table 11-5: Common options used with the `tar` utility

The **`tar`** Utility

- tar utility does not compress files inside archive
  - Time needed to transfer archive across a network is high
  - Can compress archive
- Backing up files to compressed archive on a filesystem is useful when transferring data across a network
  - Use –z option with tar

The **`tar`** Utility

- After creating an archive, you can view its contents by specifying the **`-t`** (table of contents) option to the tar command and the archive to view
- Use the **`-x`** option with tar to extract a specified archive

The **`tar`** Utility

- Can use the **`tar`** to back up data directly to a device, such as a tape
  - Use the **`-f`** option to specify the pathname to the appropriate device file
- To add to a **`tar`** archive that already exists on a tape device, use the -rvf option with the tar command

The **`tar`** Utility

- Tarballs
  - A **`gzip`**-compressed **`tar`** archive
  - Often used for source code configured for program installation
  - Most common compressed **`tar`** option
- **`tar`** is ill-suited to backing up large amounts of data for system recovery

The **`cpio`** Utility

- Copy in/out (**`cpio`**)
  - Common back-up utility
  - Includes options similar to the **`tar`** utility
  - Has added features
    - Ability to back up device files
    - Long filenames
  - Uses absolute pathnames by default when archiving

The `cpio` Utility

Option	Description
-A **--append**	Appends files to an existing archive
-B	Changes the default block size from 512 bytes to 5 kilobytes, speeding up the transfer of information
-c	Uses a storage format (SVR4) that is widely recog-nized by different versions of cpio for UNIX and Linux
-d **--make-directories**	Creates directories as needed during extraction
-i **--extract**	Reads files from an archive
-I *FILENAME*	Represents the input archive; *Filename* is the file or device file of the archive used when viewing or extracting files

Table 11-6: Common options used with the **`cpio`** utility

The `cpio` Utility

Option	Description
--no-absolute-filenames	Stores filenames in an archive using relative pathnames
-o **--create**	Creates a new archive
-O *FILENAME*	Represents the output archive; *Filename* is the file or device of the target archive when backing up files
-t **--list**	Lists the filename contents (table of contents) of an existing archive
-u **--unconditional**	Overwrites existing files during extraction without user confirmation
-v **--verbose**	Displays verbose output (file & directory information) when manipulating archives

Table 11-6: Common options used with the **`cpio`** utility

The **dump/restore** Utility

- **dump/restore**
  - Used to back up files and directories to a device or to a file on the filesystem
  - Updated to work with ext4
  - Designed to backup entire filesystems to an archive
- **/etc/dumpdates**
  - File used to store information about incremental and full back-ups for use by the **dump/restore** utility

The `dump/restore` Utility

- Full back-up
  - An archive of an entire filesystem
- Incremental back-up
  - Archive of a filesystem that contains only files that were modified since the last archive was created
  - Can perform up to nine different incremental backups
- **`restore`** command
  - Extract archives created with dump

The `dump/restore` Utility

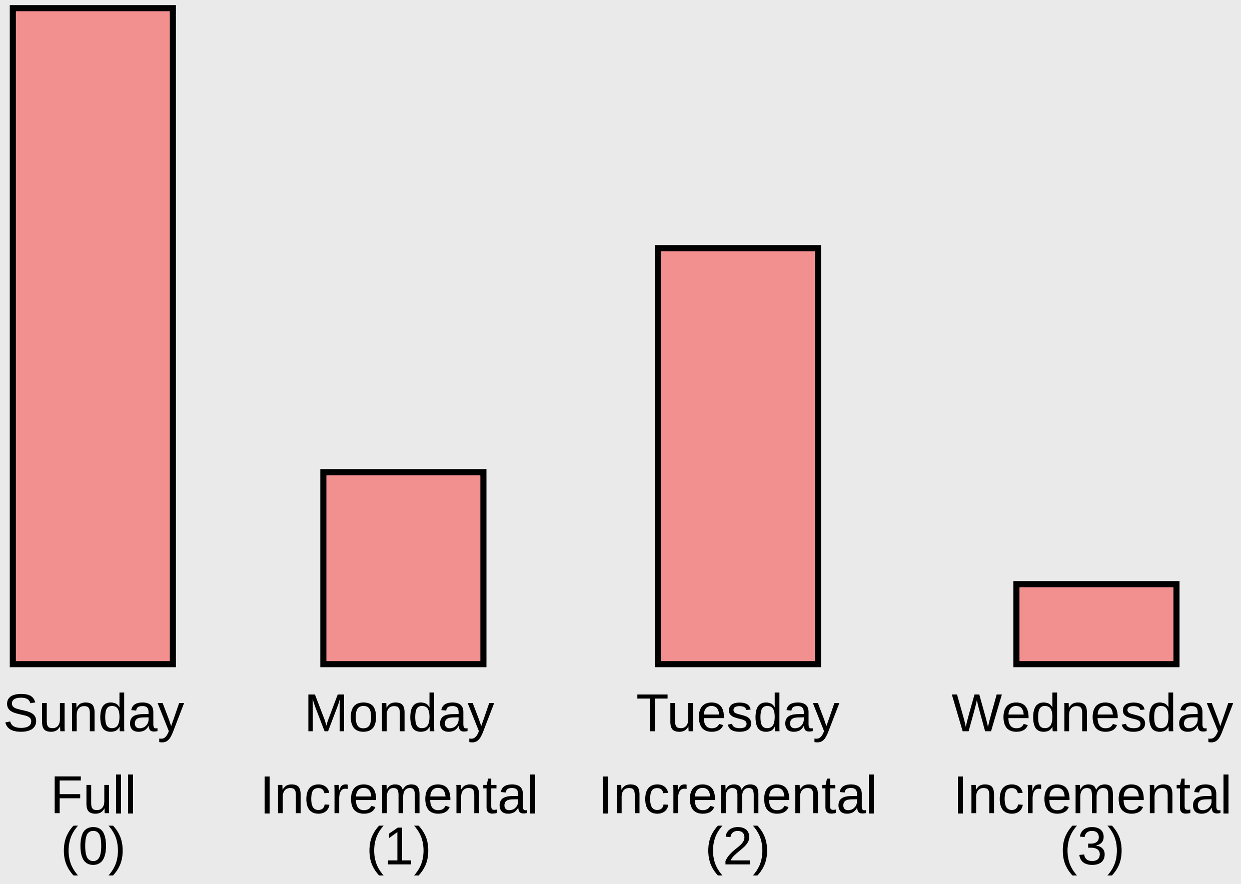


Figure 11-1: A sample back-up strategy

The **dump/restore** Utility

Option	Description
-#	Specifies the type of back-up when used with the dump command (if # is 0, a full back-up is performed; if # is 1 through 9, then the appropriate incremental back-up is performed)
-b *NUM*	Specifies a certain block size to use in kilobytes; the default block size is 10 kilobytes
-f *FILENAME*	Specifies the pathname to the archive; *FILENAME may* be a file on a filesystem or a device file
-u	Specifies to update the /etc/dumpdates file after a successful back-up
-n	Notifies the user if any errors occur and when the back-up has completed
-r	When used with the restore command, extracts an entire archive
-x *FILENAME*	When used with the restore command, extracts a certain file or files represented by *FILENAME*
-i	When used with the restore command, restores files interactively, prompting the user for confirmation for all actions
-t	When used with the restore command, lists the filename contents (table of contents) of an existing archive
-v	Displays verbose output (file & directory information) when manipulating archives

Table 11-7: Common options used with the **dump/restore** utility

Summary

- Package managers install and manage compiled software of the same format
- Red Hat Package Manager is the most common package manager available for Linux systems today
- Source code for Linux software may be obtained and compiled afterwards using the GNU C Compiler
- Most source code available in tarball format via the Internet

Summary

- Yum is an alternative command-line tool for package installation and update
  - **`yum`** command obtains RPM packages from software repositories on the Internet
- Yum is being replaced in RedHat and Fedora by DNF

Summary

- **apt** and the graphical front end for **apt**, Synaptic, can be used to install both Debian and RPM packages
- compress (**.Z**), GNU zip (**.gz**) and bzip2 (**.bz2**) are commonly used for file compression under Linux

Summary

- `tar` utility is the most common back-up utility used today
  - Typically used to create compressed archives called tarballs
- Files may be backed up to an archive using a back-up utility

The End…

- Questions?