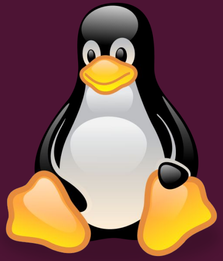
System and Network Engineering - Lecture 11

$ Time configuration + Software update management



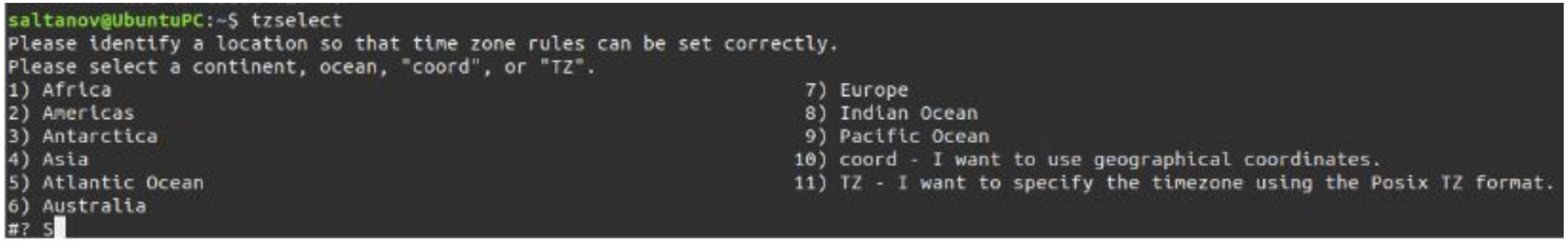
Time configuration and synchronization

Time and timezone are very important while searching through logs, synchronize with the remote distributed systems and keep everything up to date

timedatectl is a utility for controlling the system time and date.

- ❑ The timedatectl command allows you to query and change the configuration of the system clock and its settings, you can use this command to set or change the current date, time, and timezone or enable automatic system clock synchronization with a remote NTP server.
  - ❑ `$timedatectl status`
  - ❑ `$timedatectl list-timezones`
  - ❑ `$timedatectl set-timezone UTC`
  - ❑ `$timedatectl set-time '2022-11-10 16:14:50'`

Time configuration and synchronization



```
saltanov@UbuntuPC:~$ tzselect
Please identify a location so that time zone rules can be set correctly.
Please select a continent, ocean, "coord", or "TZ".
1) Africa                                                           7) Europe
2) Americas                                                         8) Indian Ocean
3) Antarctica                                                       9) Pacific Ocean
4) Asia                                                            10) coord - I want to use geographical coordinates.
5) Atlantic Ocean                                                  11) TZ - I want to specify the timezone using the Posix TZ format.
6) Australia
#? 5
```

- ❑ `RTC time` - hardware clock from motherboard, it works regardless of the state of OS (even during turn off)
- ❑ `$tzselect` - select a time zone interactively (can be used instead of timedatectl list-timezones)
- ❑ for some weather seasons there are switching time in hours, -> it's better to always use NTP
- ❑ `$timedatectl set-ntp false` OR `true` - to enable system clock synchronization

Time configuration and synchronization

The Network Time Protocol (NTP) - is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks.

chrony is a pair of programs (NTP client/server) for maintaining the accuracy of computer clocks
chronyd is a background daemon program that can be started at boot time
Configuration file: /etc/chrony.conf

Installation and configuring chronyd as an NTP client:

- ❑ `$yum install chrony`
- ❑ `$systemctl start chronyd.service`
- ❑ `$systemctl enable chronyd`
- ❑ `$systemctl daemon-reload`
- ❑ `$timedate set-ntp true # enable synchronization via chronyd`
- ❑ `$timedate set-ntp false # disable synchronization via chronyd`

chronyc is command-line interface for chronyd daemon

- ❑ `chronyc sources -v # check NTP sources with description`
- ❑ `chronyc tracking # displays parameters about the system’s clock performance`

Software update management: RPM packages

The RPM Package Manager (RPM) is a package management system used by Red Hat Linux and its derivatives such as CentOS and Fedora.

RPM also refers to the `$rpm` command and `.rpm` file format. An RPM Package consists of an archive of files and metadata. It can contain the following:

- ❑ Binary files, also known as executables (nmap, stat, xattr, ssh, sshd, etc.)
- ❑ Configuration files (sshd.conf, updatedb.conf, logrotate.conf, etc.)
- ❑ Documentation files (README, TODO, AUTHOR, etc.)

The name of an RPM package follows this format:
`<name>-<version>-<release>.<arch>.rpm`

Examples:
httpd-tools-2.4.6-7.el7.x86-64.rpm, bdsync-0.11.1-1.x86_64.rpm

- ❑ in version we can see `2.4.6 which defines: major version - minor version - patch`
- ❑ if no `arch` means no architecture specific

Software update management: RPM packages

The **`$rpm`** - is a low-level command-line tool for installing, querying, verifying, updating, and removing RPM packages

Common examples:

- ❑ `$rpm -i <package file path or URL>` - install an RPM package
- ❑ `$rpm -U <package file path or URL >` - update an RPM package (if the package is not installed it will be installed)
- ❑ `$rpm -e <package name>` - remove an RPM package

-v option tells rpm to show verbose output

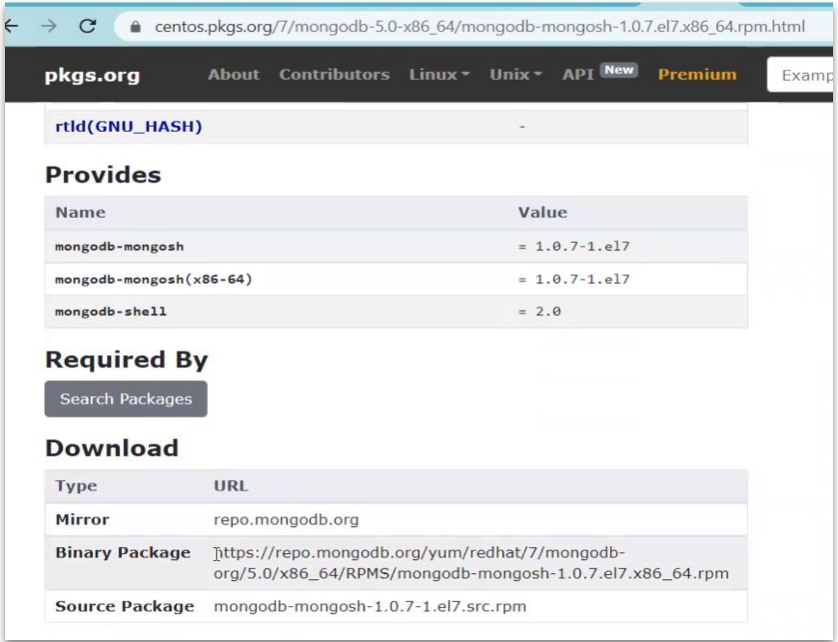
-h option to show the hash marked progress bar

--test tells rpm to run installation or removal without actually doing anything, it only shows whether the command would work or not

- ❑ `$rpm -q <package name>` - check if an RPM package installed
- ❑ `$rpm -qi <package name>` - get information about an installed RPM package
- ❑ `$rpm -ql <package name>` - get a list of all the files in an installed RPM package
- ❑ `$rpm -qa` - get a list of all installed RPM packages

!!! Be extra careful when replacing or updating important system packages, like glibc, systemd, or other services and libraries that are essential for the proper functioning of your system.

Software update management: RPM packages


centos.pkgs.org/7/mongodb-5.0-x86_64/mongodb-mongosh-1.0.7.el7.x86_64.rpm.html
pkgs.org
About
Contributors
Linux
Unix
API New
Premium
Examp
Search Packages

rtld(GNU_HASH)	-

Provides

Name	Value
mongodb-mongosh	= 1.0.7-1.el7
mongodb-mongosh(x86-64)	= 1.0.7-1.el7
mongodb-shell	= 2.0

Required By

Download

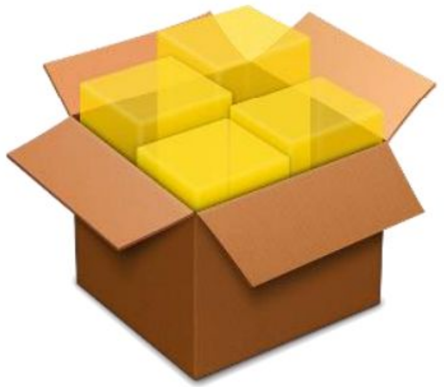
Type	URL
Mirror	repo.mongodb.org
Binary Package	https://repo.mongodb.org/yum/redhat/7/mongodb-org/5.0/x86_64/RPMS/mongodb-mongosh-1.0.7.el7.x86_64.rpm
Source Package	mongodb-mongosh-1.0.7-1.el7.src.rpm

```
[root@localhost ~]# rpm -ivh https://repo.mongodb.org/yum/redhat/7/mongodb-org/5.0/x86_64/RPMS/mongodb-mongosh-1.0.7.el7.x86_64.rpm
```

```
[root@localhost ~]# rpm -qa | grep "mongosh"
mongodb-mongosh-1.0.7-1.el7.x86_64
```

Software update management: RPM packages

Sometimes you might have access to an open-source application source code but might not have the RPM file to install it on your system. In that situation, you can either compile the source code and install the application from source code or build an RPM file from source code by yourself and use the RPM file to install the application. There might also be a situation where you want to build a custom RPM package for the application that you developed.



RPM Packaging Guide

Software management with yum

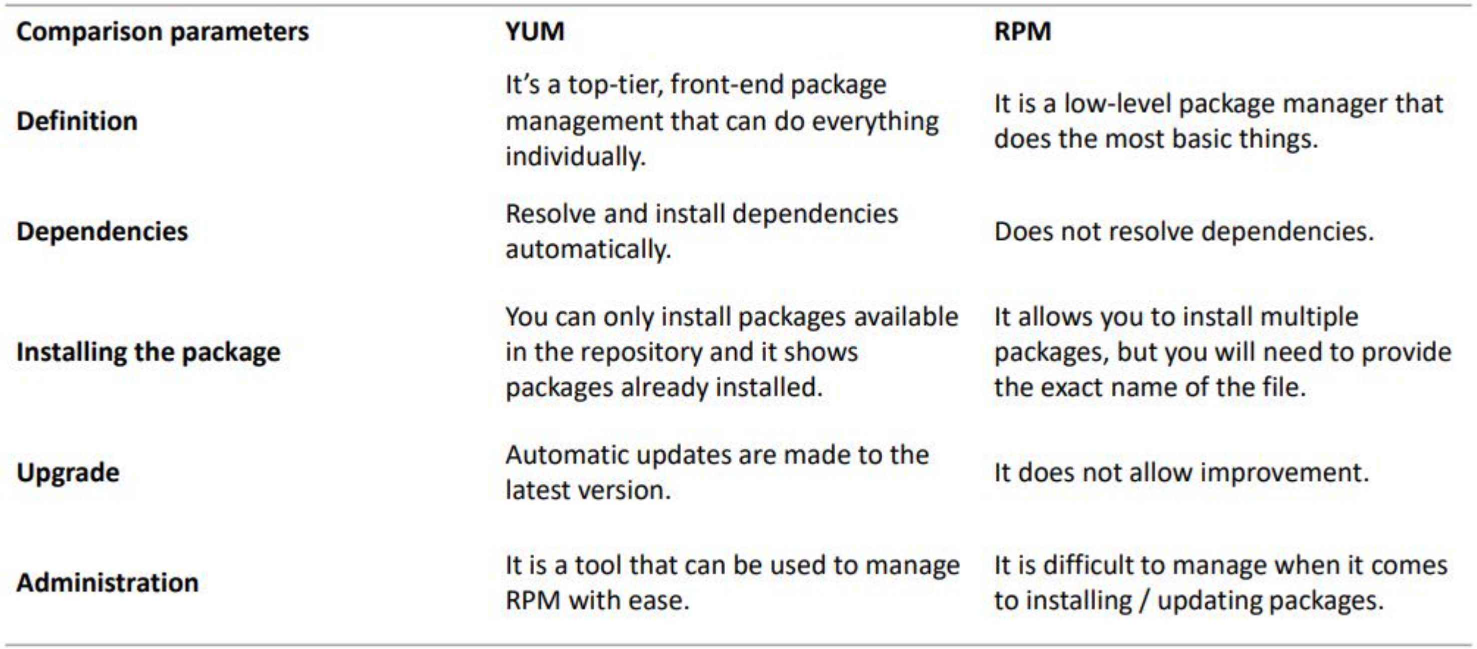
Yum - is the Red Hat **package manager** that can query for information about available packages, fetch packages from repositories, install and uninstall them, and update an entire system to the latest available version.

Yum performs automatic dependency resolution on packages you are updating, installing, or removing, and thus it can automatically determine, fetch, and install all available dependent packages

<u>Common examples:</u>

- ❑ `$yum check-update` - check which system packages can be updated
- ❑ `$yum install <package name>` - install new package with all dependencies
- ❑ `$yum localinstall <package file path>` - install new local RPM package
- ❑ `$yum update` - update all the presently installed packages to their latest versions
- ❑ `$yum upgrade` - the same as “yum update”, but once finished it also removes all the obsolete packages from the system
- ❑ `$yum update <package name>` - update a particular package
- ❑ `$yum remove <package name>` - remove a package
- ❑ `$yum search <package name or keyword>` - search a package
- ❑ `$yum info <package name>` - get detailed information about a package, including the disk space needed for installation

Software management with yum



Comparison parameters	YUM	RPM
Definition	It's a top-tier, front-end package management that can do everything individually.	It is a low-level package manager that does the most basic things.
Dependencies	Resolve and install dependencies automatically.	Does not resolve dependencies.
Installing the package	You can only install packages available in the repository and it shows packages already installed.	It allows you to install multiple packages, but you will need to provide the exact name of the file.
Upgrade	Automatic updates are made to the latest version.	It does not allow improvement.
Administration	It is a tool that can be used to manage RPM with ease.	It is difficult to manage when it comes to installing / updating packages.

Software management with yum

The official CentOS 7 repository has a huge list of packages, and it covers almost all bases in terms of software for servers, but sometimes we need some additional packages which are not available in the official repositories. In that case, we can simply add new repositories to further expand the catalogue of packages available to us.

Install the yum-utils package which includes yum-config-manager:

- ❏ `$yum install yum-utils`

Install the EPEL (Extra Packages for Enterprise Linux) rpm:

- ❏ `$rpm -Uvh https://download-ib01.fedoraproject.org/pub/epel/7/aarch64/Packages/e/epel-release-7-12.noarch.rpm`

List all packages available in the EPEL repository:

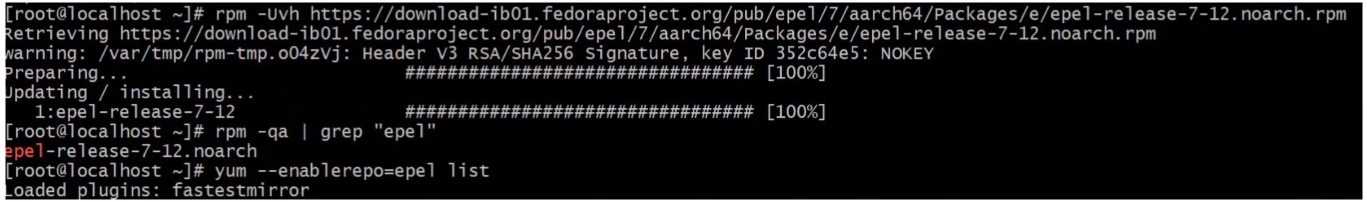
- ❏ `$yum --enablerepo=epel list`

List all enabled and disabled repositories :

- ❏ `$yum repolist all`

Enable and disable repositories:

- ❏ `$yum-config-manager --enable repository <repository name>`
- ❏ `$yum-config-manager --disable repository <repository name>`



```
[root@localhost ~]# rpm -Uvh https://download-ib01.fedoraproject.org/pub/epel/7/aarch64/Packages/e/epel-release-7-12.noarch.rpm
Retrieving https://download-ib01.fedoraproject.org/pub/epel/7/aarch64/Packages/e/epel-release-7-12.noarch.rpm
warning: /var/tmp/rpm-tmp.o04zVj: Header V3 RSA/SHA256 Signature, key ID 352c64e5: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
   1:epel-release-7-12                ################################# [100%]
[root@localhost ~]# rpm -qa | grep "epel"
epel-release-7-12.noarch
[root@localhost ~]# yum --enablerepo=epel list
Loaded plugins: fastestmirror
```

Debian Packages

A **Debian "package"**, or a Debian archive file is an analog of RPM package, but for Debian based system, like Ubuntu. These packages contain the executable files, libraries, and documentation associated with a particular suite of program or set of related programs. Normally, a Debian archive file has a filename that ends in **.deb**

The name of a DEB package follows this format:

- `<name>_<version>-<release>_<arch>.deb`

Examples:

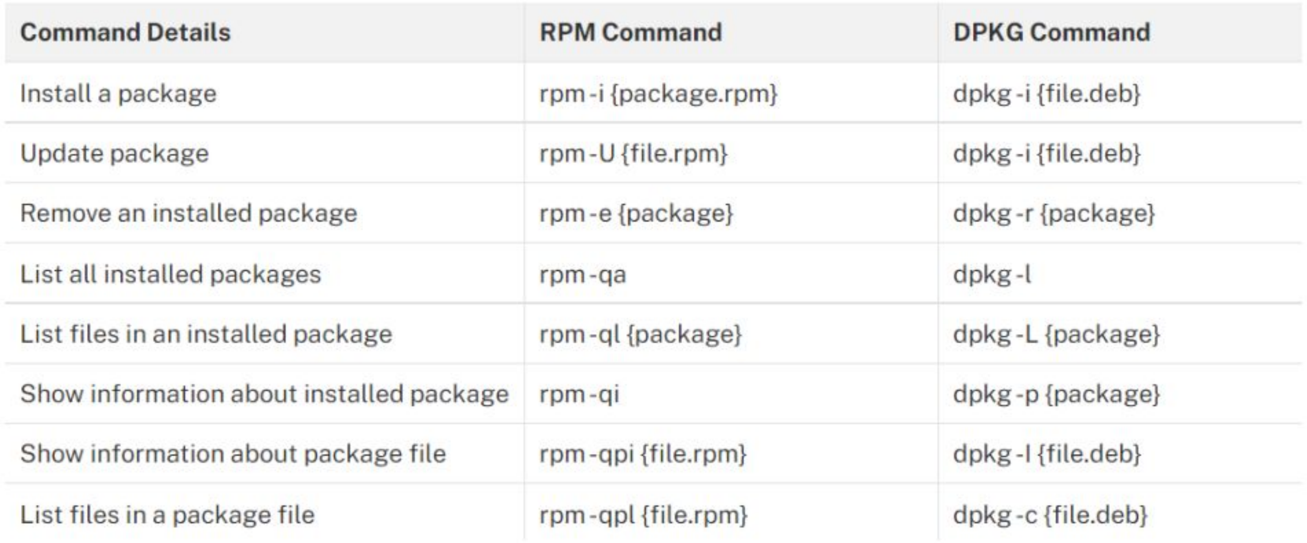
- nginx-core_1.14.0-0ubuntu1.10_amd64.deb, apache2_2.4.29-1ubuntu4.22_arm64.deb

Package management tools for Debian packages:

- Low-level tools:
  - `$dpkg`
- High-level tools:
  - `$apt`
  - `$aptitude`
  - `$synaptic`

Debian Packages

dpkg is the software at the base of the package management system in the free operating system Debian and its derivatives.
dpkg is used to install, remove, and provide information about .deb packages. dpkg itself is a low level tool



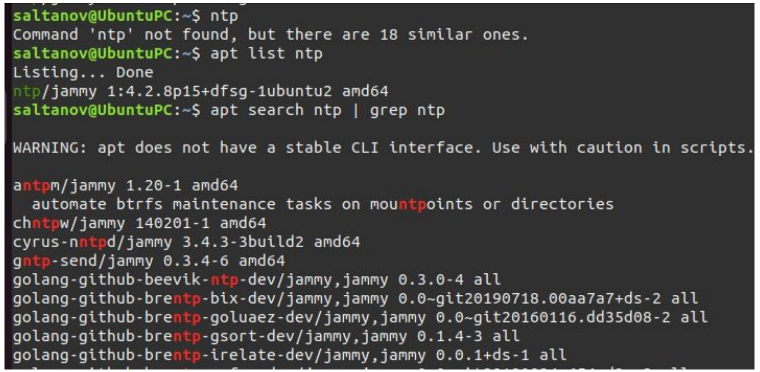
Command Details	RPM Command	DPKG Command
Install a package	rpm -i {package.rpm}	dpkg -i {file.deb}
Update package	rpm -U {file.rpm}	dpkg -i {file.deb}
Remove an installed package	rpm -e {package}	dpkg -r {package}
List all installed packages	rpm -qa	dpkg -l
List files in an installed package	rpm -ql {package}	dpkg -L {package}
Show information about installed package	rpm -qi	dpkg -p {package}
Show information about package file	rpm -qpi {file.rpm}	dpkg -I {file.deb}
List files in a package file	rpm -qpl {file.rpm}	dpkg -c {file.deb}

Software management with apt

Advanced Package Tool, more commonly known as APT. It is a collection of tools used to install, update, remove, and otherwise manage software packages on Debian and its derivative operating systems, including Ubuntu.

<u>Common examples:</u>

- ❑ `$apt list <package name>` - list packages based on package names
- ❑ `$apt search <package name>` - search in package descriptions
- ❑ `$apt show <package name>` - show package details
- ❑ `$apt install <package name>` - install packages
- ❑ `$apt remove <package name>` - remove packages
- ❑ `$apt autoremove` - remove automatically all unused packages
- ❑ `$apt update` - update list of available packages in repos
- ❑ `$apt upgrade` - upgrade the system by installing/upgrading packages



```
saltanov@UbuntuPC:~$ ntp
Command 'ntp' not found, but there are 18 similar ones.
saltanov@UbuntuPC:~$ apt list ntp
Listing... Done
ntp/jammy 1:4.2.8p15+dfsg-1ubuntu2 amd64
saltanov@UbuntuPC:~$ apt search ntp | grep ntp

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

antpm/jammy 1.20-1 amd64
  automate btrfs maintenance tasks on mountpoints or directories
chntpw/jammy 140201-1 amd64
cyrus-nntpd/jammy 3.4.3-3build2 amd64
gntp-send/jammy 0.3.4-6 amd64
golang-github-beevik-ntp-dev/jammy,jammy 0.3.0-4 all
golang-github-brentp-bix-dev/jammy,jammy 0.0~git20190718.00aa7a7+ds-2 all
golang-github-brentp-goluaez-dev/jammy,jammy 0.0~git20160116.dd35d08-2 all
golang-github-brentp-gsort-dev/jammy,jammy 0.1.4-3 all
golang-github-brentp-irelate-dev/jammy,jammy 0.0.1+ds-1 all
```

e.g. $apt list vs $apt search

The main apt sources configuration file: `/etc/apt/sources.list`

- ❑ `By default - the source for official ubuntu repos`
- ❑ `$apt edit-sources` - edit the source information file
- ❑ `$/etc/apt/sources.list.d/` - placement for the `*.list` 3rd party repos sources