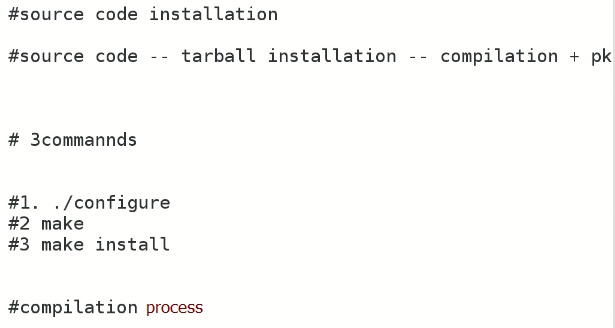
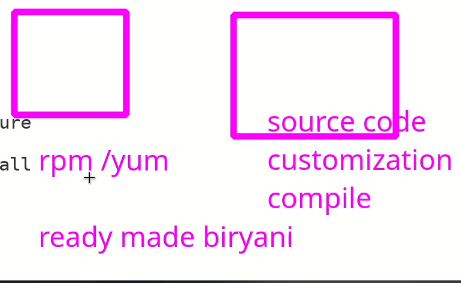
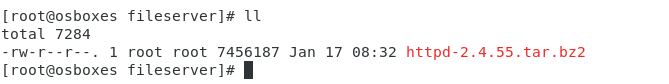
**Lecture 21**

**Source-Code-Installation**

**Source code installation**

Source code installation on Linux refers to the process of installing software from its original source code, as opposed to using a package manager or pre-built binary package. This method involves compiling the software from its source code, which requires a working development environment with the necessary build tools, libraries, and dependencies installed.

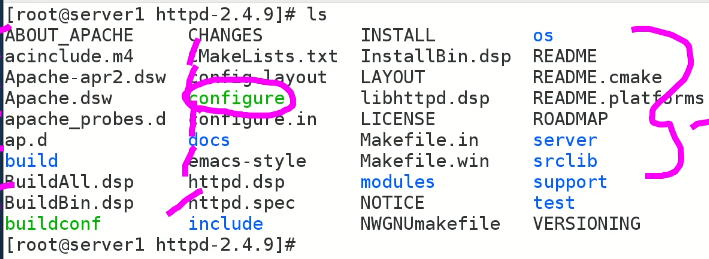
-

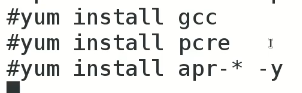
* source code compilation 🡪 for customization
* 
* Every rpm package has its source code available.
* How to install instal “apache” with source code
* 
* Multiple apache server or any service can be used by source code installation
* 
* 
* Go to internet 🡪 http download . archive 🡪 search any latest version
* Get the latest version’s link and go to /opt/fileserver
* $ wget <paste \_ copied link>
* I have downloaded latest source code 2.4.55
* 
* Extract this file.
* $ tar jxvf <file name>

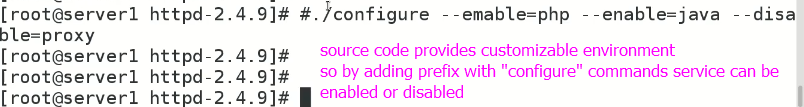
"tar jxvf" is a command used in the Unix/Linux terminal to extract the contents of a tarball archive that has been compressed using the bzip2 compression algorithm. Here's what each option means:

* "tar" is the name of the utility used for archiving and extracting files.
* "j" specifies that the archive is compressed using bzip2.
* "x" means that we want to extract the contents of the archive.
* "v" stands for "verbose" and tells tar to display the names of the files as they are extracted.
* "f" specifies the name of the archive file to work with.

So, when you run the command "tar jxvf archive.tar.bz2", it will extract the contents of the archive file "archive.tar.bz2" into the current directory.

* A directory is present after extracting tar.gz2 file
* This directory has source code and necessary files
  + - ./concat insty
* Configure 🡪 it’s a script.
* Read install file to read instruction.
* Text

  Description automatically generated
* Remember remain in same directory
* 
* Steps (error may occur 🡪 CC compiler not found 🡪 install gcc (yum install gcc), in CentOS 8 $ dnf install gcc, 🡪 other errors may occur 🡪 solution 
* "APR" stands for "Apache Portable Runtime", and it is a software library that provides a platform-independent set of APIs for building applications that can run on different operating systems and hardware architectures.
* "PCRE" stands for "Perl Compatible Regular Expressions". It is a widely-used regular expression library that provides a set of functions and syntax for pattern matching with regular expressions.
* "GCC" stands for "GNU Compiler Collection". It is a suite of compilers and tools that is widely used to develop software in many programming languages, including C, C++, Objective-C, Fortran, Ada, and others.
  1. **./configure** 🡪
     + 
     + The “configure” command also creates “binaries”
     + What is a binary file?



A binary file is a file that contains machine-readable code or data, as opposed to a text file that contains human-readable text. Binary files are used for a variety of purposes in software development and other computer-related tasks. Here are some common reasons for creating binary files:

1. Executable programs: Binary files are used to store the compiled machine code of executable programs. When you run a program, the binary file is loaded into memory and executed by the processor.
2. Libraries: Binary files can be used to store pre-compiled code that can be linked into other programs as a library. This can speed up the development process and reduce the amount of code that needs to be compiled.
3. Data storage: Binary files can be used to store non-textual data, such as images, audio, video, and other types of binary data. This data can be efficiently stored and accessed using binary file formats designed for the specific data type.
4. Serialization: Binary files can be used for serializing and deserializing complex data structures in a way that can be efficiently stored and transferred between systems. This is often used in network communication, file storage, and other data exchange scenarios.

Overall, the purpose of creating a binary file is to provide a compact, efficient, and machine-readable representation of code or data that can be used for a variety of purposes in software development and other computer-related tasks.

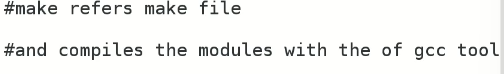
* + - To check and verify if configure command worker perfect.
    - $ echo $? 🡪 if the value is 0 it means command executed successfully, output other then 0 means command do not run successfully.

In a Unix/Linux shell environment, "echo $?" is a command that displays the exit status of the last executed command. The exit status is a numeric value that represents the success or failure of a command. In general, an exit status of 0 indicates success, while a non-zero exit status indicates some kind of error or failure.

For example, if you run a command like "ls /path/to/nonexistent/directory", which attempts to list the contents of a directory that does not exist, the command will fail and the shell will return a non-zero exit status. If you then run the command "echo $?", it will display the value of the exit status of the previous command, which will be a non-zero value indicating the failure of the "ls" command. The specific value of the exit status can vary depending on the command and the reason for the failure.

* 1. **Make** command 🡪 actual compilation process. “configure” checks dependencies and prepare thing for “make (command)” for compilation.
     + $ make 🡪 main process, depends upon RAM to execute fast.

|  |
| --- |
| The "make" command is a widely used utility in Unix and Linux systems that is used to build executable programs and other software artifacts from source code. The make utility uses a file called a "Makefile" that specifies the dependencies between the different source files and other resources required to build the final executable.  Here are the basic steps involved in using the "make" command:   1. Create a Makefile: This file should specify the source files, object files, and other resources required to build the final executable. It should also specify the build rules for each source file, such as the compiler and linker options. 2. Run the "make" command: When you run the "make" command, it reads the Makefile and determines which files need to be built based on their dependencies. It then runs the appropriate build rules to compile each source file and link them together to produce the final executable. 3. Repeat as necessary: If you make changes to the source code or other resources, you can re-run the "make" command to rebuild only the files that have changed or their dependencies.   The "make" command is a powerful and flexible tool that is widely used in software development for automating the build process and ensuring that the final executable is up-to-date and free of errors. It can also be used for other tasks, such as creating archives, generating documentation, and running tests. |

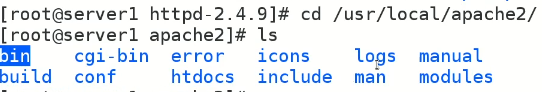
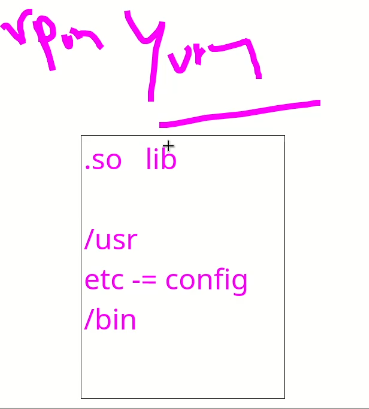
* + - 
  1. $ make install 🡪 similar to “$ rpm -rvh”

In Unix and Linux systems, the "make install" command is typically used to install a software package that has been built from source code using the "make" command. The "make install" command usually performs the following tasks:

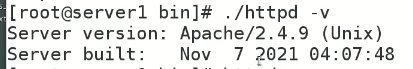
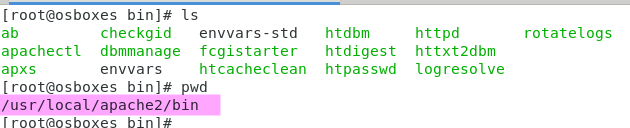
1. Copies the compiled binaries and other required files to their final installation locations on the system.
2. Sets the appropriate file permissions on the installed files.
3. Updates the system's package database with information about the newly installed software.
4. May perform additional configuration steps, such as setting environment variables, creating symbolic links, or installing documentation.

It's important to note that the "make install" command requires root privileges to perform the installation, since it typically writes files to system directories that are not writable by regular users.

The specific steps performed by the "make install" command can vary depending on the software being installed and the installation options chosen by the user. It's important to review the installation instructions provided by the software's developer to ensure that the installation is performed correctly and with the desired options.

* + - rpm, yum VS source code
      * files are copied in different location.
      * Source code keeps the file in single directory 🡪 in this case /usr/local/apache can be changes in “./configure” command.
      * 
      * 
      * Check version 🡪 in /usr/local/apache/bin 🡪 $ ./httpd -v



* + - * 
* While apache installed via yum or rpm version can be viewed 🡪 $ http -v
* To start apache 🡪
* In bin directory $ httpd -k start 🡪 starts from the binary file inside “bin” directory
* For “apache” installed through “rpm” or “yum” command,
* See all files
* $ rpm -qa httpd
* For source code installation all files are in “bin” directory
* 
* How to remove a package installed by yum or rpm
* 
* How to remove a package installed by source code
  1. Go to the directory where tar file is extracted
  2. Stop the apache service
  3. In bin directory
  4. 

Removing a package that was installed from source code typically requires a few manual steps, depending on how the package was installed. Here is a general outline of the steps you can follow:

1. Go to the directory where you originally compiled and installed the package from source.
2. Run the **make uninstall** command. This should remove the package files and undo any changes made to your system during installation. However, not all packages come with an uninstall target, so this step may not always work.
3. If **make uninstall** does not work or is not available, you can manually remove the files installed by the package. You can typically find a list of files installed by the package in the installation log or in the **Makefile**. You should remove these files manually.
4. If the package installed any configuration files, you may need to remove them manually as well. These files are typically located in the **/etc** directory.
5. If the package installed any shared libraries, you may need to update the system's library cache by running **sudo ldconfig**.

It's important to note that removing a package installed from source may not be as simple as using your system's package manager to uninstall a pre-built package. When you install a package from source, it can make changes to your system that are not tracked by the package manager. So, you should be careful when removing packages installed from source to avoid causing problems with other software on your system.