Tasks for preliminary preparation.

**The material was prepared by student Andrew Kryvenko (@AndrewKryvenko)**

**4. Based on the material reviewed, answer the following questions:**

**What is the purpose of the tar, xz, zip, bzip, gzip commands? Make a brief description of each command and highlight their main parameters.**

tar: The tar command is used to create and manipulate archives (also known as tarballs) of files and directories. It can be used to create compressed or uncompressed archives, add or extract files from an existing archive, and more. Its main parameters are:

c: create a new archive

x: extract files from an archive

v: verbose mode (display progress and list of files being processed)

f: specify the name of the archive file

xz: The xz command is used to compress and decompress files using the LZMA algorithm. Its main parameters are:

c: create a compressed file

d: decompress a compressed file

k: keep the original file after compression/decompression

zip: The zip command is used to create, modify and extract archives in the ZIP format. Its main parameters are:

c: create a new archive

x: extract files from an archive

v: verbose mode (display progress and list of files being processed)

r: add files recursively

bzip: The bzip command is used to compress and decompress files using the Burrows-Wheeler algorithm. Its main parameters are:

c: create a compressed file

d: decompress a compressed file

k: keep the original file after compression/decompression

gzip: The gzip command is used to compress and decompress files using the Lempel-Ziv algorithm. Its main parameters are:

c: create a compressed file

d: decompress a compressed file

k: keep the original file after compression/decompression

**Give three examples of how to implement archiving and data compression using different commands.**

Three examples of how to implement archiving and data compression using different commands:

To create a compressed tarball of a directory named "mydir", use: tar -czvf mydir.tar.gz mydir

To compress a file named "myfile.txt" using bzip, use: bzip -c myfile.txt > myfile.txt.bz2

To extract a zip archive named "myarchive.zip", use: unzip myarchive.zip

**What is the purpose of the cat, less, more, head and tail commands? Make a brief description of each command and highlight their main parameters. How can they be set?**

cat: The cat command is used to display the contents of one or more files on the terminal. Its main parameters are:

n: display line numbers

b: display line numbers only for non-blank lines

E: display an '$' symbol at the end of each line

less: The less command is used to view the contents of a file one page at a time, allowing you to scroll through the file. Its main parameters are:

N: display line numbers

F: keep the file open and display new content as it's added

more: The more command is similar to less, but with fewer features. It's used to display the contents of a file one page at a time. Its main parameter is:

N: display line numbers

head: The head command is used to display the first few lines of a file. Its main parameter is:

n: specify the number of lines to display (default is 10)

tail: The tail command is used to display the last few lines of a file. Its main parameters are:

n: specify the number of lines to display (default is 10)

f: keep the file open and display new content as it's added

These commands are typically built into most Unix-based systems and do not require installation. To set parameters or options for these commands, you can include them in the command line after the command name. For example, to display the contents of a file with line numbers using the cat command, you can run:

cat -n myfile.txt

Similarly, to display the first 20 lines of a file using the head command, you can run:

head -n 20 myfile.txt

And to display the last 5 lines of a file, with new content added as it's added, using the tail command, you can run:

tail -f -n 5 myfile.txt

**Explain how the command shell works with channels, streams, and filters**

The command shell is a program that provides an interface for users to interact with their operating system. It accepts commands from the user, interprets them, and executes them. Channels, streams, and filters are important concepts in the command shell that help to manipulate and redirect input and output.

Channels: A channel is a connection between two processes, typically used for passing data between them. In the command shell, channels are used to redirect the input and output of commands. There are three types of channels:

STDIN (standard input): The channel through which commands receive input, typically from the keyboard or from a file using redirection.

STDOUT (standard output): The channel through which commands send their output, typically to the terminal or to a file using redirection.

STDERR (standard error): The channel through which commands send error messages, typically to the terminal or to a file using redirection.

Streams: A stream is a sequence of data, typically used for input or output. In the command shell, streams are used to redirect and manipulate data being sent between commands. There are two types of streams:

Input stream: A stream of data being sent to a command, typically from a file or from the output of another command using a pipe.

Output stream: A stream of data being sent from a command, typically to the terminal or to a file using redirection or to another command using a pipe.

Filters: A filter is a command that reads input data from STDIN, processes it, and then writes output data to STDOUT. Filters are often used in combination with pipes to create complex command chains.

**What is the purpose of the grep command?**

The grep command is a powerful tool used to search for a specific pattern or string of characters within one or more files.

**The material was prepared by student Andrew Kryvenko (@AndrewKryvenko)**

1. 1. Work through all of the sample commands presented in the NDG Linux Essentials labs - Lab 9: Archiving and Compression and Lab 10: Working With Text. Create a table to describe these commands\*\*\*.

|  |  |
| --- | --- |
| Name of command | Its purpose and functionality |
| mkdir mybackups | Create a new mybackups directory in the user's home directory |
| tar -cvf mybackups/udev.tar /etc/udev | The tar command is used to combine several files into a single file. In this case, the contents of the /etc/udev directory will be saved to the udev.tar archive in the mybackups directory. The -c option tells the tar command to create a tar file. The -v option stands for "verbose," which tells the tar command to show what it is doing. The -f option is used to specify the name of the tar file. |
| Tar -tvf mybackups/udev.tar | The command "tar -tvf mybackups/udev.tar" is used to list the contents of a tar archive named "udev.tar" that is located in the "mybackups" directory. |
| ls -lh mybackups/ | The command "ls -lh mybackups/" is used to list the contents of the "mybackups" directory with detailed information about each file. |
| ls etc/udev | The command "ls etc/udev" is used to list the contents of the "udev" directory located in the "/etc" directory. |
| ls etc/udev/rules.d | The command "ls etc/udev/rules.d" is used to list the contents of the "rules.d" directory located within the "udev" directory in the "/etc" directory. |
| tar -rvf udev.tar /etc/hosts | The command "tar -rvf udev.tar /etc/hosts" is used to add the "/etc/hosts" file to an existing tar archive called "udev.tar" |
| zip -r udev.zip /etc/udev | The command "zip -r udev.zip /etc/udev" is used to create a zip archive called "udev.zip" that contains the entire "udev" directory located in the "/etc" directory. |
| find ~ -name "\*bash\*" | The command "find ~ -name "bash"" is used to search for files and directories within the home directory that contain the string "bash" in their name. |
| find /etc -name hosts | The command "find /etc -name hosts" is used to search for files and directories within the "/etc" directory that have the exact name "hosts". |
| find /etc -name hosts 2> err.txt | The command "find /etc -name hosts 2> err.txt" is used to search for files and directories within the "/etc" directory that have the exact name "hosts", and any error messages produced by the "find" command will be redirected to a file called "err.txt" |
| find /etc -name hosts > std.out 2> std.err | The command "find /etc -name hosts > std.out 2> std.err" is used to search for files and directories within the "/etc" directory that have the exact name "hosts", and any standard output and error messages produced by the "find" command will be redirected to separate files. |
| ls /etc | tail -5 | The command "ls /etc | tail -5" is used to list the contents of the "/etc" directory and display only the last 5 lines of the output. |
| head -n -20 /etc/passwd | The command "head -n -20 /etc/passwd" is used to display all lines of the "/etc/passwd" file except for the last 20 lines |
| /etc$ grep 'sync$' passwd | The command "grep 'sync$' passwd" searches for all lines in the "passwd" file located in the "/etc" directory that end with the word "sync". |
| /etc$ grep '.y' passwd | The command "grep '.y' passwd" searches for all lines in the "passwd" file located in the "/etc" directory that contain the letter "y" preceded by any other character. |
| /etc$ grep 'sshd|root|operator' passwd | The command "grep 'sshd|root|operator' passwd" searches for all lines in the "passwd" file located in the "/etc" directory that contain any of the three strings "sshd", "root", or "operator". |
| /etc$ grep -E 'sshd|root|operator' passwd | The command "grep -E 'sshd|root|operator' passwd" is similar to the previous command you asked about, but it uses the "-E" option to enable extended regular expressions |
| /etc$ head passwd | grep '[0-9]' | The command "head passwd | grep '[0-9]'" will print the first 10 lines of the "passwd" file (since we didn't specify a line count with the "head" command), and then search those lines for any digits using the regular expression "[0-9]". |
|  |  |
|  |  |
|  |  |

**The material was prepared by student Kanavets Kateryna (@kanavetsk)**

3. Ознайомтесь з командою tar та за її допомогою виконати у терміналі наступні дії:

- створити файл з розширенням .tar;

Команда tar (або Tape Archive) - це утиліта для створення та роботи з архівними файлами, що зберігає багато файлів в одному файлі, а також може стискувати дані.

Щоб створити архівний файл з розширенням .tar:



- створити файл з розширенням .tar, що складається з декількох файлів і каталогів одночасно;

Щоб створити архівний файл з розширенням .tar, що складається з декількох файлів та каталогів одночасно:



- перегляду вмісту файлу;

Щоб переглянути вміст архівного файлу:



- витягти вміст файлу tar;

Щоб витягти вміст архівного файлу:



- створити архівний файл tar, стиснений за допомогою bzip;

Щоб створити архівний файл tar, стиснений за допомогою bzip:



- витягти вміст файлу tar bzip;

Щоб витягти вміст архівного файлу tar зі стисненням bzip:



- створити архівний tar файл, стисненого за допомогою gzip;

Щоб створити архівний tar файл, стиснений за допомогою gzip:



- витягти вміст файлу tar gzip.

Щоб витягти вміст архівного файлу tar зі стисненням gzip:



5.Розгляньте наведені нижче приклади та поясніть, що виконують дані команди та який тип перенаправлення потоків вони використовують:

|  |  |  |
| --- | --- | --- |
| **Команда**  **(контейнер команд)** | **Що виконує команда?** | **Який потік перенаправлення?** |
| $echo "It is a new story." > story | ця команда записує рядок тексту "It is a new story." в файл з назвою story. | Операція перенаправлення > вказує на те, що вміст виводу команди echo буде записано в файл story. Якщо файл існує, то вміст файлу буде перезаписаний. |
| $ date > date.txt | ця команда записує поточну дату та час у файл з назвою date.txt | Операція перенаправлення > вказує на те, що вміст виводу команди date буде записано в файл date.txt. Якщо файл існує, то вміст файлу буде перезаписаний. |
| $ cat file1 file2 file3 > bigfile | ця команда зчитує вміст файлів file1, file2 та file3 і записує їх у файл з назвою bigfile. | Операція перенаправлення > вказує на те, що вміст цих файлів буде записаний у файл bigfile. Якщо файл bigfile існує, то вміст файлу буде перезаписаний. |
| $ls -l >> directory | ця команда виконує команду ls -l для поточного каталогу та додає результат у кінець файлу з назвою directory. | Операція перенаправлення >> вказує на те, що вміст виводу команди ls -l буде доданий до кінця файлу directory. Якщо файл існує, то вміст файлу не буде перезаписаний, а додано до кінця файлу. |
| $ sort < file1\_unsorted > file2\_sorted | ця команда зчитує вміст файлу file1\_unsorted та сортує його рядки, записуючи результат у файл з назвою file2\_sorted | Операція перенаправлення < вказує на те, що вхідний потік даних для команди sort буде взятий з файлу file1\_unsorted. Операція перенаправлення > вказує на те, що вміст виводу команди sort буде записаний у файл file2\_sorted. |
| $ find -name '\*.txt' > file.txt 2> /dev/null | виконує пошук у поточному каталозі всіх файлів з розширенням .txt | перенаправляє список знайдених файлів до файлу file.txt. |
| $ cat file1\_unsorted | sort > file2\_sorted | бере вміст файлу file1\_unsorted, сортує його та перенаправляє результат в файл file2\_sorted. | У команді cat file1\_unsorted | sort > file2\_sorted, стандартний вивід команди sort перенаправляється до файлу file2\_sorted. |
| $ cat myfile | grep student | wc -l | зчитує вміст файлу myfile, фільтрує рядки, які містять слово "student" за допомогою команди grep, та виконує підрахунок кількості таких рядків за допомогою команди wc -l | У команді cat myfile | grep student | wc -l, стандартний вивід команди wc -l (кількість рядків, що містять шуканий текст) перенаправляється до стандартного виводу терміналу. |