Odne Rindheim

DAT151 Assignment 1.

Part 1 – Filesystem

Task 1: Filesystem basics

1. Differences between Linux/Unix and Windows filesystems:
    - The Linux/Unix filesystem is hierarchical and tree-like, starting from the root (‘/’) directory. Windows uses a volume-based filesystem with drive letters (C:, D:)
    - Linux/Unix distinguishes between file types (regular, directory, link) using the first character of file permissions, Windows primarily uses file extensions.
    - Linux/Unix is case-sensitive, Windows is not.
    - Linux/Unix uses mount point for accessing different storage devices, Windows assigns drive letters.
2. What is a “Mount Point”?
    - A mount point is a directory in the Linux/Unix filesystem where additional storage is made accessible. When a storage device is mounted, its contents appear in this directory.
    - By commenting out/removing the ‘/etc/fstab’ reference and manually unmounting the partition, I’ve disconnected the automatic mounting behavior. Creating a new folder and manually mounting the partition allows me to control when and where the partition is mounted, providing flexibility in managing the storage space.
3. Seven Types of files in Unix:
    - Regular files, directories, character devices, block devices, named pipes, symbolic links, and sockets.
    - Example: ‘file /dev/urandom’ (character device), ‘file /bin/bash’ ´ls -l /bin/bash’ (regular file)
4. Experimenting with Hard and Symbolic Links:
    - ‘echo “Sample text” > testfile.txt’
    - ‘ln testfile.txt hardlink’

- ‘ln -s testfile.txt symlink
- ‘rm testfile.txt’
- ‘cat hardlink’ Shows “sample text”
- ‘cat symlink’ Shows “cat: symlink: No such file or directory”

- Observations: The hard link still works and contains the original files data. Hard links share the same inode as the original file. The symbolic link is broken. It points to the original files path, which no longer exists.

Task 2: File attributes and permissions

1. Linux/Unix filesystem bits:
    - In traditional Linux/Unix filesystem model, each file is associated with a set of 16 bits. These 16 bits are organized as follows:
      The first 4 bits represent the file type and special attributes.
      The next 3 sets of 3 bits each represent the files permissions for the file owner, group, and others.
2. Representing permissions using Octal Numbers:
    - The nine permission bits in Linux/Unix are represented using octal numbers.
      Each permission is assigned a numeric value: Read (4), Write (2), Exectue(1)
      Octal Value 745 represents:
      User: Read (4) + Write (2) + Exectue (1) = 7 (rwx)
      Group: Read (4) = (r--)
      Others: Read(4) + Exectue (1) = 5 (r-x)
      Octal value for permission bits “110 100 101”:
      User: 110 (rw-)
      Group: 100 (r--)
      Others: 101 (r-x)
      Octal value for 110 100 101 is 645.
3. Using chmod to modify file permissions

- ‘touch my_file’

  ‘chmod u+rw,go+rx my_file’

  ‘touch my_second_file’

  ‘chmod 655 my_second_file’

  Both files now have the same permissions:

  User: Read and write

  Group: Read and Execute

  Others: Read and execute.

  Octal numbers offer a straightforward way to set file permissions.

Odne Rindheim

Part 2 – Software installation and management

Task 3: Package management systems and lower-level package management

1. What is a package management system, and why do we need one? What is the package format used by the package management system on your machine?
   - A package management system is a collection of tools and libraries used to automate the process of installing, updating, configuring, and removing software packages on a computer. It provides convenient and consistent way to handle software installation and maintenance. Package managements are essential for dependency resolution, version control, security and ease of use. For AlmaLinux, based on CentOS/RHEL, the package format is RPM (.rpm)
2. Package management
   - 'rpm -qa | wc -l' 1246 packages installed
   - 'rpm -qR curl' lists a lot of libc, lubcurl, libz, rpmlib and rtld dependencies.
   - Unable to find openvpn in official AlmaLinux repository. Checked 'ls /etc/yum.repos.d' 'cat /etc/yum.repos.d/almalinux-xxxx.repo'. Also tried 'sudo dnf search openvpn' with no matches found.
   - 'sudo dnf install openvpn' Unable to find a match: openvpn. The package isn't found so unable to be installed.

Task 4: High-level package management

1. Explain how these systems can locate and install software. What is the high-level package management system on your machine?
   - High-level package management systems simplify the process of locating, installing, updating, and maintaining software packages on a system. They typically work by locating the software, where users can query this database to search for packages, and installing software, with commands like 'apt install' or 'dnf install' to install software. 'dnf' is used on my machine.
2. What is a software repo? If you are using dnf as your high-level package management system, add the 'epel' software repository to your system.

- Software repository is a centralized location where software packages, updates, and metadata are stored and maintained. Repos provide a structured way to distribute and manage software for a specific operating system or distribution.
- ‘sudo dnf install epel-release’ installed: epel-release-9-5.el9.noarch
- ‘sudo dnf install openvpn’ worked compared to in task 3, maybe it was missing epel repo.
  ‘sudo dnf remove openvpn’
- ‘sudo dnf update’ updates epel-release

Odne Rindheim

Part 3 – Systemd

Task 1 – Systemd

1) The ctrl-alt-del.target is started whenever Control+Alt+Del is pressed on console. This target will be symlinked to another target.
   a) What is the current target that ctrl-atl-del.target points to, and what are the consequences?
      i) By default, ctrl-alt-del.target points to the reboot.target. when pressed I get prompted with install updates and power off.
   b) Modify the ctrl-alt-del.target to halt the computer when ctrl+alt+del is pressed.
      i) sudo ln -s /usr/lib/systemd/system/halt.target /etc/systemd/system/ctrl-alt-del.target
2) Setting system Default Target
   a) 'sudo systemctl set-default multi-user.target'
      This will start the computer in a text-based interface.
      Writing 'sudo systemctl isolate graphical.target' will bring it back to graphical interface when running in multi-user.target.
3) Booting into emergency.target
   a) I tried doing the GRUB method with editing the load, but this didn't work, so instead I did 'sudo systemctl set-default emergency.target' to get booted into emergency.target.
      I then 'mount -o remount,rw /' to remount as read write and created a file with 'touch /root/testfile.txt'
      When I exited the emergency.target, I reset the default to graphical.target, and checked that I created the testfile 'sudo ls -l /root/testfile.txt' and confirmed I created it.
4) 'sudo cp /etc/fstab /etc/fstab.backup'
   'sudo nano /etc/fstab' added random text on the end of each line to "corrupt"
   'sudo reboot' into emergency.target
   'mount o remount,rw /'
   'sudo nano /etc/fstab' reverted changes made
   'sudo reboot' rebooted into machine normally.

5) Creating a system mount unit

   a) ‘mkdir ~/ram’

      ‘sudo nano /etc/system/system/home-odnerindheim-ram.mount’

      ```
      [Unit]
      Description = Ramdisk
      [Mount]
      What=tmpfs
      Where=home/odnerindheim/ram
      Type=tmpfs
      Options=size=128M
      [Install]
      WantedBy=default.target
      ```

      ‘sudo systemctl daemon-reload’

      ‘sudo systemctl enable home-odnerindheim-ram.mount’

      ‘sudo systemctl start home-odnerindheim-ram.mount’

      verifiying its running

      ‘df -h ~/ram’

      Output: Filesystem tmpfs, size 128m, used 0, avail 128m, use% 0%, mounted on /home/odnerindheim/ram