TUGAS UAS

COMMAD LINE LINUX

Disusun guna untuk memenuhi tugas mata kuliah

SISTEM OPERASI

Dosen Pengampu :

EDY HARYANTO, S.Kom.M.Eng

Disusun oleh:

Nama Dan NIM Kelompok:

1. **Martiano Alesandri Jedeot(23241070)**
2. **Sulfan Aditya Prasetyo(23241052)**
3. **M.Isra(23241063)**

KELAS II/B

PROGRAM STUDI PENDIDIKAN TEKNOLOGI INFORMASI

FAKULTAS SAINS, TEKNIK DAN TERAPAN

UNIVERSITAS PENDIDIKAN MANDALIKA MATARAM

TAHUN 2023/2024

ABSTRAK

Makalah ini membahas penggunaan command line interface (CLI) di sistem operasi Linux, yang merupakan salah satu alat penting bagi administrator sistem, pengembang, dan pengguna tingkat lanjut. Meskipun antarmuka grafis pengguna (GUI) lebih populer di kalangan pengguna biasa, CLI menawarkan fleksibilitas dan kontrol yang lebih besar dalam manajemen sistem dan otomatisasi tugas. Penelitian ini mencakup sejarah singkat CLI di Linux, perbandingan antara CLI dan GUI, serta struktur dasar perintah di Linux. Selain itu, makalah ini juga menjelaskan berbagai perintah dasar yang sering digunakan untuk navigasi sistem file, manajemen file dan direktori, pengelolaan proses, dan pemantauan sistem. Aspek scripting dan otomatisasi melalui shell script dan cron jobs juga dibahas untuk menunjukkan bagaimana tugas-tugas rutin dapat diotomatisasi dengan efisiensi tinggi. Kesimpulannya, meskipun CLI memiliki kurva pembelajaran yang curam, pemahaman yang baik tentang perintah dasar dan kemampuan scripting dapat memaksimalkan penggunaan Linux untuk berbagai kebutuhan teknis dan operasional. Penelitian ini didasarkan pada studi literatur dari berbagai sumber yang kredibel, termasuk buku, artikel, dan dokumentasi resmi.

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat, hidayah, dan karunia-Nya sehingga kami dapat menyelesaikan makalah ini dengan baik. Tak lupa pula shalawat serta salam senantiasa tercurah kepada baginda Nabi Muhammad SAW, yang telah menjadi rahmat bagi semesta alam.

Makalah ini kami susun sebagai bagian dari TUGAS UAS akademis untuk memenuhi salah satu syarat dalam menyelesaikan studi kami **di UNIVERSITAS PENDIDIKAN MANDALIKA**. Makalah ini membahas tentang **COMMAD LINE LINUX** Yang kami harapkan dapat memberikan pemahaman yang lebih mendalam tentang **COMMAD LINE LINUX**

Kami ingin mengucapkan terima kasih yang sebesar-besarnya kepada bapak **EDY HARYANTO, S.Kom.M.Eng** yang telah memberikan bimbingan, arahan, serta masukan yang sangat berharga dalam penyusunan makalah ini. Kami juga mengucapkan terima kasih kepada teman-teman sejawat yang telah memberikan dukungan dan inspirasi selama proses pembuatan makalah ini.

Akhir kata, semoga makalah ini dapat memberikan manfaat dan Terima kasih.

DAFTAR ISI

BAB I

PENDAHULUAN

A.Latar Belakang

Penggunaan komputer dan sistem operasi telah menjadi bagian integral dari kehidupan sehari-hari, baik dalam konteks pribadi maupun profesional. Dalam ekosistem sistem operasi, Linux menonjol sebagai salah satu yang paling kuat dan fleksibel, digunakan oleh jutaan server di seluruh dunia, komputer desktop, serta perangkat tertanam. Salah satu aspek paling menonjol dari Linux adalah penggunaan command line interface (CLI). command line interface (CLI) adalah metode interaksi dengan sistem operasi melalui perintah teks. Pengguna mengetikkan perintah ke dalam terminal atau konsol, yang kemudian dieksekusi oleh sistem. Meskipun antarmuka grafis pengguna (GUI) menyediakan cara yang lebih intuitif dan visual untuk berinteraksi dengan komputer, CLI tetap menjadi alat yang sangat penting, terutama bagi pengguna tingkat lanjut seperti administrator sistem, pengembang, dan profesional IT.

CLI menawarkan sejumlah keuntungan dibandingkan GUI, termasuk:

1. **Efisiensi dan Kecepatan**: Banyak tugas dapat diselesaikan lebih cepat melalui perintah teks dibandingkan dengan navigasi melalui GUI.
2. **Kontrol dan Fleksibilitas**: CLI menyediakan kontrol penuh atas sistem dan memungkinkan pengguna untuk melakukan konfigurasi yang sangat khusus dan kompleks.
3. **Otomatisasi**: CLI memungkinkan otomatisasi tugas-tugas berulang melalui scripting, yang tidak hanya menghemat waktu tetapi juga mengurangi kesalahan manusia.
4. **Sumber Daya Ringan**: CLI menggunakan lebih sedikit sumber daya sistem dibandingkan dengan GUI, yang penting dalam lingkungan server atau sistem dengan spesifikasi perangkat keras terbatas.

Linux, sebagai sistem operasi yang berakar dari Unix, mengadopsi banyak perintah dan konsep dari Unix, menjadikannya sangat kuat dan fleksibel. Sejarah Linux dimulai pada awal 1990-an ketika Linus Torvalds, seorang mahasiswa dari Finlandia, menciptakan kernel Linux sebagai proyek hobi. Sejak itu, Linux berkembang menjadi sistem operasi yang digunakan secara luas dan didukung oleh komunitas besar yang terus meningkatkan dan memperluas kemampuannya.

B. Rumusan Masalah

1. Apa itu command line interface (CLI) di Linux dan bagaimana sejarah perkembangannya?
2. Bagaimana struktur dasar perintah di CLI Linux?
3. Apa saja perintah-perintah dasar yang sering digunakan di Linux?
4. pa kelebihan CLI dibandingkan antarmuka grafis pengguna (GUI)?
5. Bagaimana cara menggunakan scripting untuk otomatisasi tugas-tugas di Linux?

C. Tujuan

1. Menyediakan Pengetahuan Dasar tentang CLI di Linux
2. Menguraikan Struktur Dasar dan Fungsi Perintah CLI
3. Memperkenalkan Perintah-Perintah Dasar di Linux
4. Menjelaskan Manfaat dan Kelebihan CLI
5. Membahas Teknik Scripting dan Otomatisasi di Linux
6. Membahas Teknik Scripting dan Otomatisasi di Linux

BAB II

PEMBAHASAN

A.Pengertian Command Line di Linux

Command line interface (CLI) di Linux adalah metode interaksi dengan sistem operasi melalui teks. Pengguna mengetikkan perintah ke dalam terminal atau konsol, yang kemudian dieksekusi oleh sistem. CLI memungkinkan pengguna untuk menjalankan berbagai tugas, seperti manajemen file, pengelolaan proses, dan pemantauan sistem, tanpa menggunakan antarmuka grafis (GUI).

Komponen Utama CLI di Linux

1. **Shell**
    - Shell adalah program yang menyediakan antarmuka pengguna untuk CLI. Shell menerima input perintah dari pengguna, menafsirkan perintah tersebut, dan mengarahkan sistem operasi untuk menjalankan tugas yang diminta. Beberapa jenis shell yang populer di Linux adalah Bash (Bourne Again Shell), Zsh (Z Shell), dan Fish (Friendly Interactive Shell).
2. **Terminal Emulator**
    - Terminal emulator adalah aplikasi yang menyediakan lingkungan CLI di dalam GUI. Contoh terminal emulator yang umum digunakan adalah GNOME Terminal, Konsole, dan xterm. Terminal emulator memungkinkan pengguna untuk mengakses shell di dalam lingkungan desktop grafis.

Fungsi dan Manfaat CLI di Linux

- **Efisiensi dan Kecepatan**
    - Banyak tugas dapat diselesaikan lebih cepat melalui perintah teks dibandingkan dengan navigasi melalui GUI. CLI memungkinkan pengguna untuk menjalankan perintah kompleks dengan cepat melalui satu baris perintah atau script.
- **Kontrol dan Fleksibilitas**
    - CLI menyediakan kontrol penuh atas sistem dan memungkinkan pengguna untuk melakukan konfigurasi yang sangat spesifik dan kompleks. Pengguna dapat mengakses fitur-fitur sistem yang mungkin tidak tersedia melalui GUI.
- **Otomatisasi Tugas**
    - CLI memungkinkan otomatisasi tugas-tugas berulang melalui scripting. Pengguna dapat menulis script untuk mengotomatisasi pekerjaan rutin, yang menghemat waktu dan mengurangi kesalahan manusia.

- **Penggunaan Sumber Daya Sistem yang Efisien**
- CLI menggunakan lebih sedikit sumber daya sistem dibandingkan dengan GUI. Ini penting dalam lingkungan server atau sistem dengan spesifikasi perangkat keras terbatas.

Contoh Perintah Dasar di CLI Linux

1. **Navigasi Sistem File**

    - pwd: Menampilkan direktori saat ini.
    - cd [direktori]: Mengubah direktori kerja.
    - ls: Menampilkan isi direktori.

2. **Manajemen File dan Direktori**

    - cp [sumber] [tujuan]: Menyalin file atau direktori.
    - mv [sumber] [tujuan]: Memindahkan atau mengganti nama file atau direktori.
    - rm [file]: Menghapus file.
    - mkdir [direktori]: Membuat direktori baru.

3. **Pengelolaan Proses**

    - ps: Menampilkan informasi tentang proses yang sedang berjalan.
    - top: Menampilkan penggunaan sumber daya sistem secara real-time.
    - kill [PID]: Menghentikan proses berdasarkan ID.

4. **Pemantauan Sistem**

    - df: Menampilkan penggunaan ruang disk.
    - du [direktori]: Menampilkan penggunaan ruang disk oleh file atau direktori.
    - free: Menampilkan penggunaan memori sistem.

B. Sejarah Singkat Command Line di Linux

Command line interface (CLI) di Linux memiliki akar yang dalam dalam sejarah komputasi, dimulai dari pengembangan sistem operasi Unix dan evolusinya hingga Linux. Berikut adalah garis besar perkembangan sejarah CLI di Linux:

1. Awal Mula: Unix

- **1969-1970**: Unix dikembangkan oleh Ken Thompson, Dennis Ritchie, dan rekan-rekan di Bell Labs. Unix dirancang untuk menjadi sistem operasi yang portabel, multi-tugas, dan multi-pengguna.
- **Shell Unix Awal**: Shell Unix pertama yang signifikan adalah Thompson Shell, yang kemudian diikuti oleh Bourne Shell (sh) yang dikembangkan oleh Stephen Bourne pada tahun 1977. Bourne Shell menjadi dasar dari banyak shell modern dan memperkenalkan banyak fitur scripting yang masih digunakan hingga sekarang.

2. Perkembangan Shell di Unix

- **C Shell (csh)**: Diperkenalkan oleh Bill Joy pada akhir 1970-an di University of California, Berkeley. C Shell menawarkan fitur-fitur scripting tambahan dan sintaks yang mirip dengan bahasa pemrograman C.
- **Korn Shell (ksh)**: Dikembangkan oleh David Korn di Bell Labs pada awal 1980-an. Korn Shell menggabungkan fitur-fitur terbaik dari Bourne Shell dan C Shell serta menambahkan fitur-fitur baru untuk peningkatan kinerja dan fungsionalitas scripting.

3. Kelahiran Linux

- **1991**: Linus Torvalds, seorang mahasiswa di University of Helsinki, Finlandia, merilis kernel Linux sebagai proyek hobi. Kernel ini terinspirasi oleh Minix, sistem operasi Unix kecil yang digunakan untuk tujuan pendidikan.
- **Integrasi dengan GNU**: Kernel Linux digabungkan dengan komponen-komponen dari proyek GNU (GNU's Not Unix) yang dimulai oleh Richard Stallman pada tahun 1983. GNU menyediakan banyak utilitas dan tool yang penting untuk sistem operasi Unix-like, termasuk Bash (Bourne Again Shell).

4. Perkembangan Shell di Linux

- **Bash (Bourne Again Shell)**: Dikembangkan oleh Brian Fox untuk Proyek GNU sebagai pengganti Bourne Shell. Bash menjadi shell default di banyak distribusi Linux karena kompatibilitasnya yang luas dan fitur-fiturnya yang kuat.
- **Shell Modern Lainnya**: Seiring waktu, berbagai shell lain juga dikembangkan dan digunakan di Linux, termasuk:
    - **Zsh (Z Shell)**: Dikenal karena fleksibilitas dan fitur-fiturnya yang canggih, seperti penyelesaian otomatis yang lebih baik dan integrasi dengan berbagai plugin.
    - **Fish (Friendly Interactive Shell)**: Dirancang untuk menjadi user-friendly dengan fitur-fitur seperti syntax highlighting dan auto-suggestions.

5. CLI dalam Ekosistem Linux Saat Ini

- **Dominasi CLI**: Meskipun GUI telah menjadi lebih umum dan user-friendly, CLI tetap dominan dalam banyak tugas administrasi sistem, pengembangan, dan operasi server. CLI menawarkan efisiensi dan kontrol yang tidak dapat dicapai dengan GUI.
- **Perkembangan dan Standarisasi**: Komunitas open-source terus mengembangkan dan memperbaiki shell dan utilitas CLI di Linux, menjadikannya lebih kuat dan mudah digunakan.

C. Perbandingan Command Line dengan Antarmuka Grafis (GUI)

Command line interface (CLI) dan antarmuka grafis pengguna (GUI) adalah dua cara utama pengguna berinteraksi dengan sistem operasi. Masing-masing memiliki kelebihan dan kekurangan yang membuatnya cocok untuk situasi dan pengguna yang berbeda. Berikut adalah perbandingan mendetail antara CLI dan GUI di Linux:

1. Efisiensi dan Kecepatan

- **CLI**: Pengguna dapat menjalankan banyak perintah dengan cepat dan efisien, terutama melalui penggunaan skrip dan perintah batch. CLI memungkinkan akses langsung ke fungsi-fungsi sistem dan eksekusi perintah yang lebih cepat, tanpa perlu navigasi melalui menu atau ikon.
    - *Contoh*: Menggunakan cp untuk menyalin banyak file sekaligus dengan satu perintah.
- **GUI**: Lebih intuitif untuk pengguna umum dan pemula, karena berbasis visual dan menggunakan klik mouse untuk navigasi. Namun, beberapa tugas mungkin memerlukan lebih banyak waktu karena pengguna harus menavigasi melalui berbagai menu dan jendela.
    - *Contoh*: Menggunakan drag-and-drop untuk menyalin file, yang bisa memakan waktu lebih lama dibandingkan dengan perintah cp di CLI.

2. Kontrol dan Fleksibilitas

- **CLI**: Menawarkan kontrol yang lebih besar dan fleksibilitas tinggi. Pengguna dapat melakukan operasi kompleks dan detail yang mungkin tidak tersedia di GUI. CLI memungkinkan scripting dan otomatisasi yang efisien.
    - *Contoh*: Menulis skrip bash untuk backup otomatis pada waktu tertentu.
- **GUI**: Terbatas pada fitur-fitur yang disediakan oleh antarmuka. Meskipun GUI dapat disesuaikan, itu tidak sefleksibel CLI dalam hal akses langsung ke semua fungsi sistem.
    - *Contoh*: GUI backup tools mungkin tidak menyediakan semua opsi yang bisa diakses melalui skrip CLI.

3. Kurva Pembelajaran

- **CLI**: Memiliki kurva pembelajaran yang lebih curam. Pengguna harus belajar dan menghafal berbagai perintah, sintaks, dan opsi. Meskipun dokumentasi tersedia, pengguna baru mungkin merasa CLI menakutkan.
    - *Contoh*: Menggunakan grep dengan berbagai opsi untuk pencarian teks yang kompleks membutuhkan pemahaman mendalam tentang perintah tersebut.
- **GUI**: Lebih mudah digunakan oleh pemula. Antarmuka yang intuitif dan berbasis ikon membuatnya mudah untuk dipahami tanpa banyak pembelajaran awal. Pengguna dapat mulai menggunakan sistem operasi segera tanpa memerlukan pengetahuan mendalam.
    - *Contoh*: Menemukan dan membuka aplikasi dengan klik ikon aplikasi di desktop.

4. Otomatisasi

- **CLI**: Sangat kuat dalam hal otomatisasi tugas. Melalui scripting, pengguna dapat membuat skrip untuk melakukan tugas berulang secara otomatis, menghemat waktu dan mengurangi kemungkinan kesalahan manusia.
    - *Contoh*: Menggunakan cron jobs untuk menjalankan skrip backup setiap malam.
- **GUI**: Biasanya tidak mendukung otomatisasi tingkat tinggi seperti CLI. Meskipun beberapa aplikasi GUI menawarkan fitur otomatisasi, mereka sering kali kurang fleksibel dibandingkan dengan skrip CLI.

    - *Contoh*: Menggunakan aplikasi GUI untuk backup mungkin memerlukan interaksi manual setiap kali proses dijalankan.

5. Penggunaan Sumber Daya Sistem

- **CLI**: Menggunakan lebih sedikit sumber daya sistem (CPU dan RAM) dibandingkan GUI. Ini membuat CLI ideal untuk sistem dengan spesifikasi rendah atau dalam situasi di mana kinerja adalah prioritas utama.
    - *Contoh*: CLI sering digunakan pada server karena ringan dan tidak memerlukan banyak sumber daya.
- **GUI**: Memerlukan lebih banyak sumber daya sistem untuk rendering grafis dan operasi antarmuka. Ini bisa menjadi masalah pada sistem dengan spesifikasi rendah atau dalam situasi di mana efisiensi sumber daya adalah kritis.
    - *Contoh*: GUI desktop membutuhkan lebih banyak RAM dan CPU untuk berjalan dibandingkan dengan terminal CLI.

6. Pemantauan dan Pengelolaan Sistem

- **CLI**: Lebih efektif untuk pemantauan dan pengelolaan sistem secara real-time. Alat seperti top, htop, ps, dan netstat memberikan informasi detail yang dapat dianalisis dengan cepat oleh administrator sistem.
    - *Contoh*: Menggunakan top untuk memantau penggunaan CPU dan memori oleh proses secara real-time.
- **GUI**: Meskipun menyediakan alat pemantauan yang ramah pengguna, mereka mungkin tidak sekomprehensif alat CLI. GUI sering kali memberikan informasi yang lebih mudah dipahami tetapi kurang detail dibandingkan dengan alat CLI.
    - *Contoh*: Pengelola tugas GUI menunjukkan proses dan penggunaan sumber daya tetapi mungkin tidak sekompleks output dari htop.

D. Struktur Dasar Command Line

1. Perintah Dasar

1. Navigasi Sistem File

- **`pwd` (Print Working Directory)**
    - **Deskripsi**: Menampilkan direktori saat ini.
    - **Contoh**:`Pwd`
    - Output: `arduino/home/user`
- **`cd` (Change Directory)**
    - **Deskripsi**: Mengubah direktori kerja saat ini.
    - **Contoh**:`cd /etc`
- **`ls` (List)**
    - **Deskripsi**: Menampilkan isi direktori.
    - **Contoh**:`ls`

2. Manajemen File dan Direktori

- **`cp` (Copy)**
  - **Deskripsi**: Menyalin file atau direktori.
  - **Contoh**: `cp file1.txt file2.txt`
- **`mv` (Move)**
  - **Deskripsi**: Memindahkan atau mengganti nama file atau direktori.
  - **Contoh**: `mv file1.txt /home/user/documents/`
- **`rm` (Remove)**
  - **Deskripsi**: Menghapus file.
  - **Contoh**: `rm file1.txt`
- **`mkdir` (Make Directory)**
  - **Deskripsi**: Membuat direktori baru.
  - **Contoh**:`mkdir new_directory`
- **`rmdir` (Remove Directory)**
  - **Deskripsi**: Menghapus direktori kosong.
  - **Contoh**: `rmdir empty_directory`

3. Pengelolaan Proses

- **`ps` (Process Status)**
  - **Deskripsi**: Menampilkan informasi tentang proses yang sedang berjalan.
  - **Contoh**:`ps aux`
- **`top`**
  - **Deskripsi**: Menampilkan penggunaan sumber daya sistem secara real-time.
  - **Contoh**: `top`
- **`kill`**
  - **Deskripsi**: Menghentikan proses berdasarkan ID.
  - **Contoh**: `kill 1234`
- **`htop`**
  - **Deskripsi**: Menampilkan pemantauan proses interaktif dengan informasi yang lebih detail (harus diinstal terlebih dahulu).
  - **Contoh**: `htop`

4. Pemantauan Sistem

- **`df` (Disk Free)**
  - **Deskripsi**: Menampilkan penggunaan ruang disk.
  - **Contoh**: `df -h`
- **`du` (Disk Usage)**
  - **Deskripsi**: Menampilkan penggunaan ruang disk oleh file atau direktori.
  - **Contoh**: `du -sh /home/user`
- **`free`**
  - **Deskripsi**: Menampilkan penggunaan memori sistem.
  - **Contoh**: `free -m`

5. Manajemen Paket

- **`apt-get`** (untuk distribusi berbasis Debian seperti Ubuntu)
  - **Deskripsi**: Mengelola paket perangkat lunak (memerlukan hak akses root).

    - **Contoh**: `sudo apt-get update`,`sudo apt-get install package_name`,`sudo apt-get upgrade`
- **`yum`** (untuk distribusi berbasis Red Hat seperti CentOS)
    - **Deskripsi**: Mengelola paket perangkat lunak (memerlukan hak akses root).
    - **Contoh**: `sudo yum update`,`sudo yum install package_name`
- **`dnf`** (pengganti yum untuk Fedora dan CentOS 8)
    - **Deskripsi**: Mengelola paket perangkat lunak (memerlukan hak akses root).
    - **Contoh**:`sudo dnf update`,`sudo dnf install package_name`

6. Manipulasi Teks

- **`cat` (Concatenate)**
    - **Deskripsi**: Menampilkan isi file.
    - **Contoh**: `cat file1.txt`
- **`grep` (Global Regular Expression Print)**
    - **Deskripsi**: Mencari teks dalam file.
    - **Contoh**:`grep "search_term" file1.txt`
- **`echo`**
    - **Deskripsi**: Menampilkan pesan atau teks.
    - **Contoh**:`echo "Hello, World!"`
- **`nano`, `vim`, `emacs`**
    - **Deskripsi**: Editor teks yang digunakan untuk mengedit file.
    - **Contoh**:`nano file1.txt`,`vim file1.txt`,`emacs file1.txt`

7. Kompresi dan Ekstraksi

- **`tar`**
    - **Deskripsi**: Membuat dan mengekstrak arsip tar.
    - **Contoh**:`tar -cvf archive.tar directory_name`,`tar -xvf archive.tar`
- **`zip` dan `unzip`**
    - **Deskripsi**: Mengompres dan mengekstrak file zip.
    - **Contoh**:`zip -r archive.zip directory_name` atau `unzip archive.zip`

2. Opsi dan Argumen

Di command line Linux, perintah sering kali disertai dengan opsi (flags) dan argumen untuk mengontrol perilaku dan target dari perintah tersebut. Memahami cara menggunakan opsi dan argumen dengan benar adalah kunci untuk memanfaatkan kekuatan penuh dari CLI.

1. Opsi (Flags)

Opsi adalah modifikasi yang diterapkan pada perintah untuk mengubah cara perintah tersebut dieksekusi. Opsi biasanya dimulai dengan satu atau dua tanda hubung (- atau --).

- **Single-dash Options**: Opsi singkat yang menggunakan satu tanda hubung dan biasanya satu huruf.
    - *Contoh*: -l pada perintah ls -l untuk menampilkan daftar isi direktori dalam format panjang.
- **Double-dash Options**: Opsi panjang yang menggunakan dua tanda hubung dan biasanya lebih deskriptif.
    - *Contoh*: --help pada perintah ls --help untuk menampilkan panduan penggunaan perintah ls.

2. Argumen

Argumen adalah input yang diberikan kepada perintah untuk menentukan objek atau file yang akan diproses. Argumen biasanya muncul setelah opsi.

- *Contoh*: Pada perintah cp file1.txt /home/user/Documents/, file1.txt dan /home/user/Documents/ adalah argumen yang menentukan file sumber dan tujuan.

Contoh Penggunaan Opsi dan Argumen

Berikut adalah beberapa contoh perintah dengan opsi dan argumen untuk memperjelas konsep ini:

3. Contoh Perintah Dasar dengan Opsi dan Argumen

- **ls (List)**
    - **Deskripsi**: Menampilkan isi direktori.
    - **Opsi**:
        - -l: Menampilkan dalam format panjang.
        - -a: Menampilkan semua file, termasuk file tersembunyi.
    - **Contoh**: ls -la /home/user

Menampilkan isi direktori /home/user dalam format panjang, termasuk file tersembunyi.

- **cp (Copy)**
    - **Deskripsi**: Menyalin file atau direktori.
    - **Opsi**:
        - -r: Menyalin direktori secara rekursif.
        - -v: Menampilkan file yang sedang disalin.
    - **Contoh**: cp -rv /home/user/Documents /backup/

Menyalin direktori /home/user/Documents ke /backup/ secara rekursif dan menampilkan file yang sedang disalin.

- **mv (Move)**
    - **Deskripsi**: Memindahkan atau mengganti nama file atau direktori.
    - **Opsi**:
        - -i: Meminta konfirmasi sebelum menimpa file.
        - -v: Menampilkan file yang sedang dipindahkan atau diganti nama.
    - **Contoh**: mv -iv file1.txt /home/user/Documents/

Memindahkan file1.txt ke direktori /home/user/Documents/ dan meminta konfirmasi jika ada file yang ditimpa.

- **rm (Remove)**
  - **Deskripsi**: Menghapus file.
  - **Opsi**:
    - -r: Menghapus direktori beserta isinya secara rekursif.
    - -f: Menghapus tanpa meminta konfirmasi.
  - **Contoh**:rm -rf /home/user/old_files

Menghapus direktori /home/user/old_files beserta isinya tanpa meminta konfirmasi.

- **grep (Global Regular Expression Print)**
  - **Deskripsi**: Mencari teks dalam file.
  - **Opsi**:
    - -i: Mengabaikan perbedaan antara huruf besar dan kecil.
    - -r: Mencari secara rekursif dalam direktori.
    - -n: Menampilkan nomor baris.
  - **Contoh**: grep -rin "search_term" /home/user/Documents

Mencari search_term dalam semua file di direktori /home/user/Documents secara rekursif, mengabaikan huruf besar dan kecil, dan menampilkan nomor baris.

4. Menampilkan Bantuan dan Dokumentasi

Banyak perintah di Linux memiliki opsi --help atau -h yang menampilkan bantuan dan dokumentasi singkat tentang cara menggunakan perintah tersebut.

- **Contoh**:ls --help

Selain itu, perintah man (manual) dapat digunakan untuk melihat dokumentasi yang lebih lengkap.

- **Contoh**:man ls

E. Perintah-Perintah Dasar Linux

1. Perintah Navigasi dan Manajemen Direktori

- **pwd** (Print Working Directory)
  - **Deskripsi**: Menampilkan direktori kerja saat ini.
  - **Contoh**: pwd

Output: /home/user

- **ls** (List)

    - **Deskripsi**: Menampilkan isi direktori.
    - **Opsi**:
        - -l: Menampilkan dalam format panjang.
        - -a: Menampilkan semua file, termasuk yang tersembunyi.
    - **Contoh**: ls -la

      Output:

      sql

      total 24

      drwxr-xr-x 3 user user 4096 Jul 2 10:00 .

      drwxr-xr-x 90 user user 4096 Jul 2 09:55 ..

      -rw-r--r-- 1 user user 675 Jul 2 10:00 file1.txt

      -rw-r--r-- 1 user user 345 Jul 1 15:30 file2.txt

- **cd** (Change Directory)
    - **Deskripsi**: Mengubah direktori kerja saat ini.
    - **Contoh**:cd /home/user/Documents
- **mkdir** (Make Directory)
    - **Deskripsi**: Membuat direktori baru.
    - **Contoh**:mkdir new_directory
- **rmdir** (Remove Directory)
    - **Deskripsi**: Menghapus direktori kosong.
    - **Contoh**:rmdir empty_directory

2. Perintah Manajemen File

- **cp** (Copy)
    - **Deskripsi**: Menyalin file atau direktori.
    - **Contoh**:cp file1.txt /home/user/Documents
- **mv** (Move)
    - **Deskripsi**: Memindahkan atau mengganti nama file atau direktori.
    - **Contoh**:mv file1.txt new_location/
- **rm** (Remove)
    - **Deskripsi**: Menghapus file.
    - **Opsi**:
        - -r: Menghapus direktori beserta isinya secara rekursif.
        - -f: Menghapus tanpa konfirmasi.
    - **Contoh**:rm file1.txt
- **touch**
    - **Deskripsi**: Membuat file kosong atau mengubah waktu modifikasi file.
    - **Contoh**:touch new_file.txt

- **nano** atau **vim**
    - **Deskripsi**: Mengedit file teks menggunakan editor nano atau vim.
    - **Contoh**:nano file1.txt atau vim file1.txt

3. Perintah Pemantauan dan Proses

- **ps** (Process Status)
    - **Deskripsi**: Menampilkan proses yang sedang berjalan.
    - **Contoh**:ps aux
- **top**
    - **Deskripsi**: Menampilkan daftar proses yang sedang berjalan secara real-time.
    - **Contoh**:top
- **kill**
    - **Deskripsi**: Menghentikan proses dengan nomor ID proses (PID).
    - **Contoh**:kill 1234

4. Perintah Pengelolaan Paket (untuk distribusi berbasis Debian, seperti Ubuntu)

- **apt-get**
    - **Deskripsi**: Mengelola paket perangkat lunak.
    - **Contoh**:sudo apt-get update,sudo apt-get install package_name

5. Perintah Pencarian dan Manipulasi Teks

- **grep** (Global Regular Expression Print)
    - **Deskripsi**: Mencocokkan pola teks dalam file.
    - **Contoh**: grep "pattern" file.txt
- **sed** (Stream Editor)
    - **Deskripsi**: Mengedit atau mengubah teks dalam file.
    - **Contoh**: sed 's/old_word/new_word/g' file.txt

6. Perintah Jaringan

- **ping**
    - **Deskripsi**: Memeriksa koneksi jaringan ke host tertentu menggunakan ICMP.
    - **Contoh**:ping google.com
- **ifconfig**
    - **Deskripsi**: Menampilkan informasi konfigurasi jaringan.
    - **Contoh**:ifconfig

7. Perintah Sistem

- **uname**
    - **Deskripsi**: Menampilkan informasi kernel sistem.
    - **Contoh**:uname -a

- **date**
  - **Deskripsi**: Menampilkan tanggal dan waktu saat ini.
  - **Contoh**:date

8. Perintah Pemantauan Sistem

- **df** (Disk Free)
  - **Deskripsi**: Menampilkan penggunaan ruang disk.
  - **Contoh**:df -h
- **free**
  - **Deskripsi**: Menampilkan penggunaan memori sistem.
  - **Contoh**:free -m

F.Scripting dan Otomatisasi di Linux

Scripting dan otomatisasi merupakan kekuatan utama dari command line Linux yang memungkinkan pengguna untuk mengotomatiskan tugas-tugas rutin, memanipulasi data, dan mengelola sistem dengan efisien. Berikut ini adalah konsep dasar serta beberapa contoh penggunaan scripting di Linux:

1. Bash Scripting

Bash (Bourne Again Shell) adalah shell default yang digunakan di banyak distribusi Linux dan merupakan lingkungan scripting yang kuat untuk otomatisasi tugas-tugas sistem. Berikut adalah langkah-langkah dasar dalam membuat dan menjalankan skrip bash:

a. Membuat Skrip Bash

1. **Buka editor teks favorit Anda** seperti `nano`, `vim`, atau `gedit`.
2. **Buat file baru** dengan ekstensi `.sh` (misalnya `myscript.sh`).
3. **Tambahkan shebang line** di baris pertama untuk menunjukkan shell yang akan digunakan. Contoh:

   ```
   #!/bin/bash
   ```

4. **Tambahkan perintah-perintah bash** yang ingin Anda jalankan, seperti perintah `ls`, `cp`, `mv`, `echo`, `if`, `for`, `while`, dll.

b. Contoh Skrip Bash Sederhana

Berikut adalah contoh skrip bash sederhana untuk memahami dasar-dasarnya:

```
#!/bin/bash

# Menampilkan pesan sederhana
echo "Halo, selamat datang!"

# Menampilkan daftar file dalam direktori saat ini
```

```
echo "Daftar file:"
ls

# Menyalin file1.txt ke direktori baru
cp file1.txt /home/user/Documents

# Looping untuk menampilkan angka 1 sampai 5
echo "Angka dari 1 sampai 5:"
for i in {1..5}
do
    echo $i
done
```

c. Menjalankan Skrip Bash

Setelah membuat skrip bash, Anda perlu memberikan izin eksekusi kepada skrip tersebut sebelum dapat dijalankan:

```
chmod +x myscript.sh
```

Kemudian, jalankan skrip dengan cara berikut:

```
./myscript.sh
```

2. Otomatisasi Tugas

Bash scripting sering digunakan untuk otomatisasi tugas-tugas seperti:

- **Backup Data**: Menyalin file-file penting ke tempat penyimpanan jarak jauh atau lokal secara otomatis pada jadwal tertentu.
- **Monitoring dan Logging**: Memantau penggunaan sumber daya sistem dan mencatatnya ke dalam file log untuk analisis lebih lanjut.
- **Konfigurasi Sistem**: Mengonfigurasi pengaturan sistem, menginstal paket, dan menyesuaikan lingkungan kerja sesuai kebutuhan.

3. Penggunaan Variabel dan Pengontrol Alur

Dalam scripting bash, penggunaan variabel, pengontrol alur (seperti `if`, `for`, `while`), serta kemampuan untuk mengambil argumen dari baris perintah (`$1`, `$2`, dst.) memungkinkan skrip untuk menangani berbagai tugas secara dinamis dan beradaptasi dengan input yang berbeda.

Contoh Penggunaan Variabel:

```
#!/bin/bash

# Mendefinisikan variabel
name="John"
age=30

# Menggunakan variabel
echo "Nama: $name"
echo "Umur: $age tahun"
```

Contoh Penggunaan Pengontrol Alur (If Statement):

```
#!/bin/bash

# Memeriksa apakah sebuah file ada
if [ -f file.txt ]; then
    echo "File file.txt ditemukan."
else
    echo "File file.txt tidak ditemukan."
fi
```

4. Eksekusi Perintah Eksternal

Skrip bash dapat memanggil dan mengeksekusi perintah-perintah eksternal dan bahasa pemrograman lainnya. Ini memungkinkan integrasi dengan alat-alat eksternal untuk tugas-tugas yang lebih kompleks.

Contoh Eksekusi Perintah Eksternal:

```
#!/bin/bash

# Menjalankan perintah eksternal (misalnya Python)
python3 myscript.py

# Menjalankan perintah dengan output piped ke perintah lain
ls -l | grep ".txt"
```

BAB III

PENUTUP

A.Kesimpulan

Command line di Linux adalah alat yang sangat kuat dan fleksibel, memberikan kontrol penuh kepada pengguna atas sistem operasi mereka. Memahami perintah-perintah dasar, opsi, dan argumen, serta kemampuan scripting dan otomatisasi adalah kunci untuk memaksimalkan produktivitas dan efisiensi dalam bekerja dengan Linux Menguasai command line Linux tidak hanya meningkatkan efisiensi dan produktivitas tetapi juga memberikan pemahaman yang lebih dalam tentang bagaimana sistem operasi bekerja. Pengguna yang terampil dalam menggunakan command line dapat mengatasi berbagai tantangan teknis, memecahkan masalah dengan cepat, dan mengotomatisasi tugas-tugas yang berulang, membuat mereka lebih kompeten dan efektif dalam mengelola lingkungan Linux.

Poin-poin Utama:

1. **Perintah Dasar**: Memahami perintah-perintah dasar seperti pwd, ls, cd, cp, mv, dan rm memungkinkan pengguna untuk menavigasi dan mengelola sistem file dengan mudah.
2. **Opsi dan Argumen**: Penggunaan opsi (flags) dan argumen pada perintah memungkinkan penyesuaian eksekusi perintah untuk kebutuhan spesifik, memberikan fleksibilitas yang lebih besar dalam pengelolaan sistem.
3. **Pemantauan Sistem dan Proses**: Perintah seperti ps, top, dan kill membantu dalam pemantauan dan pengelolaan proses yang berjalan, sementara df dan free memberikan informasi tentang penggunaan sumber daya sistem.
4. **Pengelolaan Paket**: Perintah pengelolaan paket seperti apt-get (untuk distribusi berbasis Debian) memungkinkan instalasi, pembaruan, dan penghapusan perangkat lunak dengan mudah.
5. **Scripting dan Otomatisasi**: Bash scripting membuka peluang besar untuk otomatisasi tugas-tugas rutin, backup, pemantauan, dan konfigurasi sistem. Dengan menggunakan variabel, pengontrol alur, dan eksekusi perintah eksternal, pengguna dapat membuat skrip yang kompleks dan efisien.

DAFTAR PUSTAKA

Sobell, Mark G.. *A Practical Guide to Linux Commands, Editors, and Shell Programming*. Prentice Hall, 2017.

Shotts, William E.. *The Linux Command Line: A Complete Introduction*. No Starch Press, 2019.

Nemeth, Evi, Snyder, Garth, Hein, Trent R., Whaley, Ben, Mackin, Dan. *UNIX and Linux System Administration Handbook*. Addison-Wesley Professional, 2017.

Cooper, Mendel. *Advanced Bash Scripting Guide*. Linux Documentation Project, 2014.

Kerrisk, Michael. *The Linux Programming Interface: A Linux and UNIX System Programming Handbook*. No Starch Press, 2010.

Linux Manual Pages. *The Linux Command Line Manual Pages*. https://man7.org/linux/man-pages/

Arch Wiki. *Command-line Shell*. https://wiki.archlinux.org/index.php/Command-line_shell

Ubuntu Documentation. *Basic Commands*. https://help.ubuntu.com/community/BasicCommands

TLDP (The Linux Documentation Project). *Bash Guide for Beginners*. https://tldp.org/LDP/Bash-Beginners-Guide/html/

Stack Overflow. *Linux Command Line and Scripting*. https://stackoverflow.com/questions/tagged/linux