Introduction to Linux

Autor: Marcin Kujawski

Powłoka systemowa – Bash i Shell

Wprowadzenie

Powłoka to interpreter wiersza poleceń, który zapewnia interfejs między użytkownikiem a systemem operacyjnym Linux. Umożliwia użytkownikom wykonywanie poleceń, uruchamianie skryptów i zarządzanie zasobami systemowymi. Powłoka interpretuje dane wejściowe użytkownika, przetwarza polecenia i zwraca dane wyjściowe. Niektóre popularne typy powłok w systemie Linux obejmują:

- **Bash (Bourne Again Shell)** - domyślna powłoka w większości dystrybucji Linuksa.
- **Zsh (Z Shell)** - rozszerzona wersja Basha z ulepszonymi funkcjami.
- **Fish (Friendly Interactive Shell)** - przyjazna dla użytkownika i bogata w funkcje powłoka.
- **Tcsh (TENEX C Shell)** - oparty na C Shell (csh), powszechnie używany do tworzenia skryptów.
- **Ksh (Korn Shell)** - potężna powłoka często używana w środowiskach korporacyjnych.

Czym jest skryptowanie Bash?

Tworzenie skryptów Bash to proces pisania skryptów przy użyciu powłoki Bash w celu automatyzacji zadań. Skrypt Bash to plik zawierający serię poleceń, które są wykonywane sekwencyjnie. Skrypty są używane do automatyzacji powtarzalnych zadań, zarządzania operacjami systemowymi i wykonywania złożonych konfiguracji systemu.

Dlaczego warto używać skryptów Bash?

- **Automatyzacja** - ogranicza ręczną interwencję w przypadku powtarzalnych zadań.
- **Administracja systemem** - pomaga w zarządzaniu użytkownikami, procesami, dziennikami i aktualizacjami systemu.
- **Kopia zapasowa i konserwacja** - umożliwia zaplanowane tworzenie kopii zapasowych i czyszczenie systemu.
- **Harmonogram zadań** - może być wykonywany za pomocą zadań cron lub timerów systemd.
- **Bezpieczeństwo i kontrola dostępu** - pomaga w ograniczaniu lub przyznawaniu dostępu za pomocą skryptów.

Typowe przykłady praktyczne

1. Sprawdzenie czas pracy systemu

```
uptime
```

2. Wyświetlanie zalogowanych użytkowników

```
who
```

3. Pokaż użycie dysku

```
df -h
```

4. Monitorowanie wykorzystania pamięci

```
free -m
```

5. Lista uruchomionych procesów

```
ps aux
```

6. Znajdź konkretny proces

```
ps aux| grep apache
```

7. Zabijanie procesu według PID

```
kill -9 PID<>
```

8. Sprawdzenie łączność sieciową

```
ping -c 4 google.com
```

9. Wyświetlanie otwartych portów

```
netstat -tulnp
```

10. Sprawdź dzienniki systemowe

```
dmesg| tail -50
```

11. Tworzenie prostego skryptu Bash

```
echo "#!/bin/bash"> myscript.sh
echo "echo Hello, World!">> myscript.sh
chmod +x myscript.sh
./myscript.sh
```

12. Automatyzacja tworzenia kopii zapasowych plików

```
tar -czvf backup.tar.gz /home/user/
```

13. Znajdowanie i usuwanie starych plików

```
find /var/log -name "*.log" -type f -mtime +30 -exec rm {} \;
```

14. Pętla przez pliki w katalogu

```
for file in /path/to/directory/*; do echo "Przetwarzanie
$file"; done
```

15. Odczytywanie danych wprowadzanych przez użytkownika w skrypcie

```
read -p "Wprowadź swoje imię: " name
echo "Witaj, $name!"
```

16. Instrukcja warunkowa w Bash

```
if [ -f /etc/passwd ]; then echo "Plik istnieje"; else echo
"Plik nie został znaleziony"; fi
```

17. Uruchamianie polecenia jako inny użytkownik

```
sudo -u user <command>
```

18. Zaplanuj zadanie cron do uruchamiania codziennie o północy

```
crontab -e
0 0 * * * /path/to/script.sh
```

19. Monitorowanie wykorzystania zasobów systemowych

```
top
```

20. Zapisywanie danych wyjściowych skryptu do pliku

```
./script.sh> output.log 2>&1
```

Dodatkowe uwagi

- Zawsze używaj #!/bin/bash na początku skryptów Bash, aby zapewnić wykonanie we właściwej powłoce.
- Skrypty powinny mieć uprawnienia do wykonywania (chmod +x script.sh).
- Użyj set -e, aby zatrzymać wykonywanie skryptu w przypadku błędów.
- Użyj pułapki, aby poradzić sobie z nieoczekiwanymi przerwami.
- Używaj komentarzy (#) do dokumentowania skryptów w celu ułatwienia ich konserwacji.

Konfiguracja

Ustawienie Bash jako domyślnej powłoki

```
chsh -s /bin/bash
```

Tworzenie podstawowego profilu Bash (~/.bashrc)

```
echo 'export PS1="\u@\h:\w$ "'>> ~/.bashrc
echo 'alias ll="ls -la"'>> ~/.bashrc
source ~/.bashrc
```

Tworzenie systemowego skryptu Bash

- Utwórz skrypt:

```
echo "#!/bin/bash"> /usr/local/bin/myscript
echo "echo System Info: $(uname -a)" >>
/usr/local/bin/myscript
```

- Przypisanie praw wykonywania:

```
chmod +x /usr/local/bin/myscript
```

- Uruchom skrypt:

```
./myscript
source myscript       # ta opcja nie wymaga uprawnień +x
```

Podstawowe polecenia systemu Linux

Wprowadzenie

Linux zapewnia różnorodne narzędzia wiersza poleceń do interakcji z systemem plików, zarządzania procesami i wykonywania zadań administracyjnych systemu. Zrozumienie podstawowych poleceń jest niezbędne do sprawnego poruszania się po systemie Linux i zarządzania nim. Ta sekcja obejmuje podstawowe polecenia, w tym:

- ls (lista zawartości katalogu)
- find (wyszukiwanie plików i katalogów)
- ps (wyświetlanie informacji o procesie)
- mv (przenoszenie lub zmiana nazwy plików)
- cp (kopiowanie plików i katalogów)
- rm (usuwanie plików i katalogów)
- cd (zmiana katalogu)

Typowe przykłady praktyczne

1. Lista plików i katalogów (ls)

Wyświetlanie zawartości katalogu

```
ls
```

Pokaż ukryte pliki

```
ls -a
```

Wyświetlanie plików ze szczegółowymi informacjami

```
ls -l
```

Sortowanie plików według czasu modyfikacji (najpierw najnowsze)

```
ls -lt
```

2. Znajdowanie plików i katalogów (find)

Wyszukiwanie pliku według nazwy w bieżącym katalogu

```
find . -name "plik.txt"
```

Znajduje wszystkie pliki .log w /var/log zmodyfikowane w ciągu ostatnich 7 dni

```
find /var/log -name "*.log" -mtime -7
```

Wyszukiwanie pustych plików

```
find /home/user -type f -empty
```

Znajdź i usuń wszystkie pliki .tmp

```
find /tmp -name "*.tmp" -exec rm {} \;
```

3. Wyświetlanie uruchomionych procesów (ps)

Pokaż wszystkie uruchomione procesy

```
ps aux
```

Filtrowanie określonego procesu według nazwy

```
ps aux| grep nginx
```

Wyświetlanie procesów dla określonego użytkownika

```
ps -u username
```

Pokaż drzewo procesów

```
ps -axjf
```

4. Przenoszenie i zmiana nazwy plików (mv)

Przenoszenie pliku do innego katalogu

```
mv plik.txt /home/user/documents/
```

Zmiana nazwy pliku

```
mv oldname.txt newname.txt
```

Przenoszenie wielu plików do katalogu

```
mv plik1.txt plik2.txt /home/user/
```

5. Kopiowanie plików i katalogów (cp)

Kopiowanie pliku do innej lokalizacji

```
cp plik.txt /home/user/
```

Rekursywne kopiowanie katalogu

```
cp -r /source/dir /destination/dir
```

Kopiowanie wielu plików

```
cp plik1.txt plik2.txt /destination/
```

Zachowanie atrybutów plików podczas kopiowania

```
cp -p plik.txt /backup/
```

6. Usuwanie plików i katalogów (rm)

Usuwanie pliku

```
rm file.txt
```

Usuwanie wielu plików

```
rm plik1.txt plik2.txt
```

Usunięcie katalogu i jego zawartości

```
rm -r /home/user/temp/
```

Wymuś usunięcie pliku bez potwierdzenia

```
rm -f important.log
```

7. Zmień katalog (cd)

Przejście do katalogu domowego

```
cd ~
```

Przejście do określonego katalogu

```
cd /var/log/
```

Przejście do poprzedniego katalogu

```
cd -
```

Przejście o jeden poziom katalogu wyżej

```
cd ..
```

Dodatkowe uwagi

- Zawsze używaj rm -i, aby uniknąć przypadkowego usunięcia ważnych plików.
- Użyj find -exec, aby połączyć operacje wyszukiwania i działania.
- Podczas korzystania z ps, warto rozważyć htop dla bardziej interaktywnej przeglądarki procesów.

- ls -lh jest przydatny do przeglądania rozmiarów plików w formacie czytelnym dla człowieka.
- Użyj uzupełniania tabulatorów, aby szybko nawigować po katalogach za pomocą cd.

Wzorce poleceń systemu Linux

Wprowadzenie

W systemie Linux narzędzia wiersza poleceń, takie jak grep, awk i sed, zapewniają potężne możliwości przetwarzania tekstu. Narzędzia te pomagają efektywnie wyszukiwać, manipulować i przetwarzać tekst. Zrozumienie wzorców i parametrów w podstawowych poleceniach pozwala użytkownikom automatyzować zadania, wyodrębniać informacje i dynamicznie modyfikować pliki.

Kluczowe narzędzia:

- **grep**: Wyszukuje wzorce tekstowe w plikach.
- **awk**: Język programowania do skanowania wzorców i przetwarzania tekstu.
- **sed**: Edytor strumieni używany do modyfikowania plików i strumieni.

Typowe przykłady praktyczne

1. Wyszukiwanie wzorców za pomocą grep

Znajdowanie słowa w pliku

```
grep "error" /var/log/syslog
```

Wyszukiwanie z uwzględnieniem przypadku

```
grep -i "warning" /var/log/syslog
```

Pokaż numery linii z dopasowaniami

```
grep -n "fail" /var/log/auth.log
```

Wyszukiwanie rekurencyjne w katalogach

```
grep -r "TODO" /home/user/projects
```

Wyświetlaj tylko pasujące słowa

```
grep -o "[0-9]\{3\}-[0-9]\{3\}-[0-9]\{4\}" phonebook.txt
```

2. Przetwarzanie tekstu za pomocą awk

Drukowanie drugiej kolumny z pliku

```
awk '{print $2}' data.txt
```

Filtrowanie i wyświetlanie wierszy, w których wartość w kolumnie 3 jest większa niż 50

```
awk '$3 > 50' data.txt
```

Drukowanie wierszy zawierających określony wzorzec

```
awk '/error/ {print}' logs.txt
```

Sumowanie wartości w określonej kolumnie

```
awk '{sum+=$2} END {print sum}" sales.txt
```

Wyodrębnienie nazw użytkowników z pliku /etc/passwd

```
awk -F: '{print $1}' /etc/passwd
```

3. Modyfikowanie tekstu za pomocą sed

Zastępowanie słowa w pliku

```
sed 's/error/fixed/' logs.txt
```

Zastąpienie wszystkich wystąpień słowa

```
sed 's/error/fixed/g' logs.txt
```

Usuwanie pustych wierszy z pliku

```
sed '/^$/d' plik.txt
```

Usuwanie pierwszej kolumny z pliku CSV

```
sed 's/^[^,]*,//' data.csv
```

Wstawianie linii po określonym wzorcu

```
sed '/pattern/a\New Line Here' plik.txt
```

Dodatkowe uwagi

- grep, awk i sed mają fundamentalne znaczenie dla analizy logów, ekstrakcji danych i manipulacji tekstem.
- Wyrażenia regularne zwiększają wydajność dopasowywania wzorców.
- sed jest przydatny do modyfikacji plików w miejscu.
- awk jest potężnym narzędziem do przetwarzania danych w terenie.
- Użyj grep -E dla rozszerzonych wyrażeń regularnych.

Zdalne połączenie w systemie Linux

Wprowadzenie

Narzędzia zdalnego połączenia umożliwiają użytkownikom dostęp do maszyn z systemem Linux z różnych lokalizacji. Dwie najpopularniejsze metody zdalnego dostępu to:

- **SSH (Secure Shell):** Protokół do bezpiecznego zdalnego logowania i wykonywania poleceń.
- **VNC (Virtual Network Computing):** Graficzny system udostępniania pulpitu.

SSH jest powszechnie używany do zdalnej administracji opartej na terminalach, podczas gdy VNC jest przydatny do zdalnego dostępu do pulpitu Linux.

Typowe przykłady praktyczne

1. Łączenie się ze zdalnym serwerem przez SSH

Podstawowe połączenie SSH

```
ssh user@remote-server
```

Połączenie z określonym portem

```
ssh -p 2222 user@remote-server
```

Włączenie trybu pełnego debugowania

```
ssh -v user@remote-server
```

Używanie klucza prywatnego do uwierzytelniania

```
ssh -i ~/.ssh/id_rsa user@remote-server
```

Uruchom polecenie na zdalnym serwerze

```
ssh user@remote-server "ls -l /var/log"
```

Kopiowanie plików z serwera lokalnego na zdalny

```
scp plik.txt user@remote-server:/home/user/
```

Kopiowanie plików ze zdalnego serwera na komputer lokalny

```
scp user@remote-server:/home/user/file.txt .
```

2. Konfiguracja serwera VNC

Instalacja serwera VNC

```
sudo apt update
sudo apt install xfce4 xfce4-goodies -y
sudo apt install tightvncserver -y
vncserver
```

Ustawianie hasła VNC

```
vncpasswd
```

Uruchom serwer VNC na wyświetlaczu :1

```
vncserver :1
```

Zatrzymanie serwera VNC

```
vncserver -kill :1
```

Konfiguracja VNC do uruchamiania z określoną rozdzielczością

```
#!/bin/sh

xrdb "$HOME/.Xresources"
xsetroot -solid grey
#x-terminal-emulator -geometry 80x24+10+10 -ls -title
"$VNCDESKTOP Desktop" &
#x-window-manager &
# Poprawka umożliwiająca działanie GNOME
```

```
export XKL_XMODMAP_DISABLE 1=
/etc/X11/Xsession
gnome-panel &
gnome-settings-daemon &
metacity &
nautilus &
startxfce4 &
```

Utwórz plik wykonywalny i zrestartuj serwer

chmod +x ~/.vnc/xstartup
vncserver -localhost

3. Łączenie z serwerem VNC

Bezpieczny dostęp przez tunelowanie (SSH lub PowerShell)

```
ssh -L 59000:localhost:5901 -C -N -l student 192.168.1.100
```

Połączenie z Linuksa za pomocą vncviewer

```
sudo apt install tigervnc-viewer
vncviewer -via student@192.168.1.100 localhost:1
```

Połączenie przy użyciu tunelu SSH dla bezpieczeństwa

```
ssh -L 5901:localhost:5901 user@remote-server
```

Następnie otwórz klienta VNC i połącz się z localhost:1.

Połączenie z Windows przy użyciu VNC Viewer

- Zainstaluj **przeglądarkę RealVNC Viewer** lub **TightVNC Viewer**.
- Otwórz aplikację i wprowadź remote-server:1.
- Wprowadź hasło VNC i połącz się.

Dodatkowe uwagi

- SSH zapewnia bezpieczny, szyfrowany dostęp zdalny.
- VNC jest przydatny do zdalnego dostępu opartego na GUI, ale powinien być używany z tunelowaniem SSH dla bezpieczeństwa.
- Zawsze wyłączaj logowanie roota przez SSH dla bezpieczeństwa (PermitRootLogin no w /etc/ssh/sshd_config).
- Aby poprawić bezpieczeństwo SSH, należy używać **uwierzytelniania opartego na kluczach** zamiast haseł.
- Użyj ufw lub iptables, aby ograniczyć zdalny dostęp do portów SSH i VNC.

Konfiguracja

Konfiguracja SSH dla uwierzytelniania opartego na kluczach

- Wygeneruj parę kluczy SSH na komputerze klienckim:

```
ssh-keygen -t rsa -b 4096
```

- Skopiuj klucz publiczny na zdalny serwer:

```
ssh-copy-id user@remote-server
```

- Wyłącz uwierzytelnianie hasłem (opcjonalne dla bezpieczeństwa): Edytuj /etc/ssh/sshd_config i ustaw:

```
PasswordAuthentication nie
```

- Uruchom ponownie usługę SSH:

```
sudo systemctl restart ssh
```

Konfiguracja VNC dla trwałych sesji

- Edytuj plik konfiguracyjny VNC (~ /.vnc/xstartup):

```
#!/bin/bash
xrdb $HOME/.Xresources
startxfce4 &
```

- Ustaw prawidłowe uprawnienia:

```
chmod +x ~/.vnc/xstartup
```

- Uruchom ponownie serwer VNC:

```
vncserver -kill :1
vncserver :1
```

Edycja plików tekstowych przy użyciu podstawowych edytorów (vi i nano)

Wprowadzenie

Linux zapewnia potężne edytory tekstowe do modyfikowania plików konfiguracyjnych i skryptów. Dwa najczęściej używane edytory to: - **vi/vim:** Wysoce wydajny, modalny edytor tekstu dostępny w prawie wszystkich systemach uniksopodobnych. - **nano**: przyjazny dla użytkownika, prosty edytor tekstu do szybkich modyfikacji.

Typowe przykłady praktyczne

Korzystanie z vi/vim

Otwieranie pliku w vi

```
vi nazwa_pliku.txt
```

Przejście do trybu wstawiania (rozpoczęcie edycji)

```
i
```

Zapisz zmiany i wyjdź

```
:wq
```

Wyjście bez zapisywania

```
:q!
```

Wyszukiwanie słowa w pliku

```
/słowo
```

Kopiuj i wklej wiersz

```
yy    # Kopiuj linię
p    # Wklej poniżej
```

Usuń linię

```
dd
```

Korzystanie z nano

Otwórz plik za pomocą nano

```
nano nazwa_pliku.txt
```

Zapisz zmiany

```
CTRL + O, a następnie naciśnij ENTER
```

Wyjście nano

```
CTRL + X
```

Wyszukiwanie słowa

```
CTRL + W, a następnie wpisz słowo i naciśnij ENTER
```

Wytnij i wklej wiersz

```
CTRL + K    # Wytnij
CTRL + U    # Wklej
```

Dodatkowe uwagi

- **vi/vim** jest preferowany przez zaawansowanych użytkowników ze względu na jego zaawansowane funkcje.
- **nano** jest łatwiejsze w użyciu dla początkujących i szybkich edycji.
- Przed edycją plików konfiguracyjnych należy zawsze tworzyć ich kopie zapasowe.

Ten przewodnik obejmuje teraz edycję tekstu za pomocą vi i nano. Daj mi znać, jeśli potrzebujesz dodatkowych szczegółów!

Procesy

Wprowadzenie

W systemie Linux **proces** jest instancją uruchomionego programu. System operacyjny zarządza procesami w celu zapewnienia wydajnej alokacji zasobów i wielozadaniowości. Procesy można wymieniać, monitorować i kontrolować za pomocą różnych poleceń.

Typowe przykłady praktyczne

1. Procesy notowania

Wyświetlanie wszystkich uruchomionych procesów

```
ps aux
```

Wyświetlanie procesów dla określonego użytkownika

```
ps -u username
```

Wyświetlanie procesów w formacie drzewa hierarchicznego

```
ps axjf
```

Wyświetlanie monitorowania procesów w czasie rzeczywistym (podobne do Menedżera zadań)

```
top
```

Interaktywne monitorowanie procesów w czasie rzeczywistym z ulepszonym interfejsem użytkownika

```
htop
```

2. Zarządzanie procesami

Zabij proces według PID

```
kill <PID>
```

Zabij proces według nazwy

```
pkill nazwa_procesu
```

Wymuszone zakończenie procesu

```
kill -9 PID
```

Zakończenie wszystkich instancji procesu

```
killall nazwa_procesu
```

Zawieszenie procesu (wysłanie go w tło)

```
CTRL + Z
```

Wznowienie zawieszonego procesu w tle

```
bg
```

Przeniesienie procesu działającego w tle na pierwszy plan

```
fg
```

3. Monitorowanie i debugowanie procesów

Wyświetlanie szczegółowych informacji o procesie

```
ps -p PID -o pid,ppid,%cpu,%mem,cmd
```

Wyświetlanie otwartych plików przez proces

```
lsof -p PID
```

Wyświetlanie procesów nasłuchujących na portach sieciowych

```
netstat -tulnp
```

Monitorowanie wykorzystania zasobów systemowych na proces

```
sar -u 5 10
```

Znajdź najbardziej pamięciożerne procesy

```
ps aux --sort -%mem=| head -10
```

Dodatkowe uwagi

- **SIGTERM (kill -15)** jest domyślnym sygnałem używanym do łagodnego zakończenia procesu.
- **SIGKILL (kill -9**) przymusowo kończy proces, nie pozwalając na jego wyczyszczenie.
- **jobs** wyświetla listę procesów działających w tle i zawieszonych.
- Polecenia **nice i renice** dostosowują poziomy priorytetów procesów.

Użycie dysku

Wprowadzenie

Monitorowanie wykorzystania przestrzeni dyskowej jest kluczowe dla administracji systemem. Linux zapewnia narzędzia takie jak **du** (użycie dysku) i **df** (wolny dysk) do sprawdzania dostępnej przestrzeni, analizowania zużycia pamięci i zapobiegania wyczerpaniu dysku.

Typowe przykłady praktyczne

1. Sprawdzanie dostępnego miejsca na dysku

Wyświetlanie użycia dysków zamontowanych systemów plików

```
df -h
```

Wyświetlanie miejsca na dysku w bajtach

```
df -B1
```

Wyświetlanie użycia i-węzłów zamiast miejsca na dysku

```
df -i
```

Sprawdzanie użycia dysku dla określonego systemu plików

```
df -T /dev/sda1
```

2. Sprawdzanie wykorzystania przestrzeni katalogów i plików

Wyświetlanie użycia dysku w katalogu

```
du -sh /var/log
```

Pokaż użycie dla wszystkich podkatalogów

```
du -h --max-depth= 1 /home
```

Znajdowanie największych plików w katalogu

```
du -ah /home| sort -rh| head -10
```

Podsumowanie całkowitego użycia katalogu

```
du -csh /var/www
```

3. Monitorowanie przestrzeni dyskowej w czasie rzeczywistym

Obserwuj zmiany miejsca na dysku co 10 sekund

```
watch -n 10 df -h
```

Określenie, który katalog zajmuje najwięcej miejsca.

```
ncdu /
```

Dodatkowe uwagi

- **df** jest najlepiej używany do sprawdzania wolnego miejsca w całych systemach plików.
- **du** pomaga w identyfikacji dużych katalogów i plików.
- **ncdu** (jeśli jest zainstalowany) zapewnia interaktywny sposób nawigacji po wykorzystaniu dysku.
- Regularne monitorowanie zapobiega nagłym niedoborom przestrzeni dyskowej, które mogą wpływać na wydajność systemu.

Praca z plikami i katalogami

Wprowadzenie

Zarządzanie plikami i katalogami jest podstawową umiejętnością w systemie Linux. Użytkownicy muszą efektywnie tworzyć, kopiować, przenosić i usuwać pliki i katalogi, aby utrzymać zorganizowany system plików. Linux udostępnia różne polecenia, takie jak **touch, mkdir, cp, mv, rm** itp. do wykonywania tych zadań.

Typowe przykłady praktyczne

1. Tworzenie plików i katalogów

Utwórz pusty plik

```
touch myfile.txt
```

Tworzenie wielu plików

```
touch plik1.txt plik2.txt plik3.txt
```

Utwórz katalog

```
mkdir mydirectory
```

Tworzenie zagnieżdżonych katalogów

```
mkdir -p parent/child/grandchild
```

2. Kopiowanie plików i katalogów

Kopiowanie pliku do innej lokalizacji

```
cp myfile.txt /home/user/
```

Kopiowanie wielu plików

```
cp plik1.txt plik2.txt /home/user/
```

Kopiowanie katalogu i jego zawartości

```
cp -r mydirectory /home/user/
```

Kopiowanie plików z zachowaniem atrybutów

```
cp -p myfile.txt /home/user/
```

3. Przenoszenie i zmiana nazw plików i katalogów

Przenoszenie pliku do innej lokalizacji

```
mv myfile.txt /home/user/
```

Zmiana nazwy pliku

```
mv oldname.txt newname.txt
```

Przenoszenie wielu plików do katalogu

```
mv plik1.txt plik2.txt /home/user/
```

Przenoszenie katalogu

```
mv mydirectory /home/user/
```

4. Usuwanie plików i katalogów

Usuwanie pliku

```
rm myfile.txt
```

Usuwanie wielu plików

```
rm plik1.txt plik2.txt
```

Usuń katalog (tylko jeśli jest pusty)

```
rmdir mydirectory
```

Usuwanie katalogu i jego zawartości rekurencyjnie

```
rm -r mydirectory
```

Wymuś usunięcie plików (używaj ostrożnie)

```
rm -f myfile.txt
```

Dodatkowe uwagi

- touch służy do tworzenia pustych plików lub aktualizacji znaczników czasu plików.
- **mkdir -p** zapewnia istnienie katalogów nadrzędnych przed utworzeniem podkatalogów.
- **cp -r** jest wymagane podczas kopiowania katalogów.
- **rm -r** jest niezbędne do usuwania niepustych katalogów.
- Zawsze używaj **rm -f** ostrożnie, aby zapobiec przypadkowemu usunięciu.

Wyszukiwanie plików

Wprowadzenie

Efektywne wyszukiwanie plików jest kluczowe w administracji systemem Linux. Niezależnie od tego, czy szukasz plików na podstawie nazwy, rozmiaru, czasu modyfikacji lub innych atrybutów, Linux zapewnia potężne narzędzia, takie jak **find, locate, du** i **ls**, aby usprawnić ten proces.

Typowe przykłady praktyczne

1. Wyszukiwanie plików według nazwy

Znajduje plik według dokładnej nazwy w bieżącym katalogu i podkatalogach

```
find . -name "myfile.txt"
```

Znajduje plik według nazwy, ignorując wielkość liter

```
find /home -iname "myfile.txt"
```

Znajdowanie wielu plików z różnymi rozszerzeniami

```
find /var/log -type f \( -name "*.log" -o -name "*.txt")
```

2. Wyszukiwanie plików według rozmiaru

Znajdowanie plików większych niż 100 MB

```
find / -type f -size +100M
```

Znajdowanie plików mniejszych niż 10 KB

```
find /home/user -type f -size -10k
```

Znajdowanie plików o rozmiarze od 50 MB do 200 MB

```
find /var -type f -size +50M -size -200M
```

3. Wyszukiwanie plików według czasu modyfikacji

Znajdowanie plików zmodyfikowanych w ciągu ostatnich 7 dni

```
find /etc -type f -mtime -7
```

Znajdowanie plików, które nie zostały zmodyfikowane w ciągu ostatnich 30 dni

```
find /home -type f -mtime +30
```

Znajdowanie plików, do których uzyskano dostęp w ciągu ostatnich 24 godzin

```
find /var -type f -atime -1
```

4. Wyświetlanie plików i sprawdzanie rozmiaru

Lista wszystkich plików w katalogu z rozmiarami czytelnymi dla człowieka

```
ls -lh /home/user
```

Lista tylko katalogów

```
ls -d */
```

Znajduje 10 największych plików w katalogu

```
du -ah /var/log| sort -rh| head -10
```

Sprawdź całkowite użycie dysku w katalogu

```
du -sh /home/user
```

5. Korzystanie z polecenia Locate w celu szybszego wyszukiwania

Zaktualizuj zlokalizowaną bazę danych (uruchom jako root)

```
updatedb
```

Znajdowanie pliku przy użyciu funkcji locate (szybszej niż find)

```
locate myfile.txt
```

Znajdowanie plików zawierających określone słowo

```
locate --ignore-case "config"
```

Dodatkowe uwagi

- **find** jest najpotężniejszym narzędziem wyszukiwania, ale może być wolniejsze.
- **Lokalizacja** jest znacznie szybsza, ale wymaga zaktualizowanej bazy danych.
- **du** pomaga w identyfikacji plików zajmujących dużo miejsca.
- **ls -lh** zapewnia szybki wgląd w rozmiary plików i szczegóły.

Kompresowanie plików

Wprowadzenie

Kompresja plików jest niezbędna w systemie Linux do zmniejszenia przestrzeni dyskowej i optymalizacji transferu plików. Różne narzędzia, takie jak **tar, gzip, bzip2 i xz**, zapewniają potężne metody kompresji i archiwizacji plików przy jednoczesnym zachowaniu integralności danych.

Kompresja plików pozwala zaoszczędzić miejsce na dysku i/lub czas transmisji sieciowej. Zawsze zwiększamy wydajność kompresji kosztem dłuższego czasu kompresji:

- **tar**: zaprojektowany do tworzenia pojedynczego archiwum z wielu plików lub katalogów, archiwum może być zwykłym plikiem lub urządzeniem

- **cpio**: odczytuje i zapisuje pliki w formacie binarnym lub ASCII.

- **gzip**: wykorzystuje kodowanie Lempel-Ziv (LZ77) i tworzy pliki .gz

- **bzip2**: wykorzystuje algorytm kompresji tekstu z sortowaniem blokowym Burrowsa-Wheelera i kodowaniem Huffmana i tworzy pliki .bz2

- **xz**: tworzy pliki .xz. i obsługuje również starszy format .lzma

TAR

Jest to domyślny program do zarządzania archiwami dostępny w systemie Linux. Jest przeznaczony do tworzenia pojedynczego archiwum z wielu plików lub katalogów i manipulowania nimi. Archiwum może być zwykłym plikiem lub urządzeniem (takim jak napęd taśmowy, stąd nazwa programu - archiwizator taśm), które może znajdować się na przykład na komputerze lokalnym lub zdalnym.

```
tar -cvfv archive.tar /usr/bin/.
```

Flagi:

-c, --create: tworzenie nowego archiwum

-x, --extract,

-f, --file=ARCHIWUM: użycie pliku lub urządzenia ARCHIWUM

-v, --verbose: wypisuje szczegóły o przetwarzanych plikach

-j, --bzip2: filtruje archiwum przez bzip2

-j, --xz: filtruje archiwum przez xz

-z, --gzip, --gunzip, --ungzip: filtruje archiwum przez gzip

-p, --preserve-permissions: przywraca informacje o uprawnieniach plików

-w, --verify: próbuje zweryfikować archiwum po zapisie

Przykłady tworzenia i kompresji archiwów:

```
tar zcvf archive.tar.gz /usr/bin/
tar jcvf archive.tar.bz2 /usr/bin/
tar jcvf archive.tar.xz /usr/bin/
```

Rozpakowywanie archiwum:

```
tar xzvf source.tar.gz
tar xjvf source.tar.bz2
tar xJvf source.tar.xz
```

GZIP

Poleceniegzip kompresuje pliki przy użyciu kodowania Lempel-Ziv (LZ77). Dowiązania symboliczne są domyślnie ignorowane.

Flagi:

-k, --keep: zachowuje (nie usuwa) plik wejściowy podczas wykonywania kompresji lub dekompresji

-d, --decompress: dekompresuje

-v, --verbose: tryb gadatliwy

-r, --recursive: kompresuje rekursywnie (katalog i podkatalogi)

-c, --stdout: zapisuje dane wyjściowe, zachowując oryginalne pliki bez zmian

-l, --list: wyświetla szczegóły archiwum (zawartość i dane kompresji)

-q, --quiet: nie wyświetla ostrzeżeń

gzip pozwala określić zakres poziomów kompresji, od 1 do 9: - -1 lub --fast: oznacza najszybszą prędkość kompresji z minimalnym współczynnikiem kompresji - -9 lub --best: oznacza najwolniejszą prędkość kompresji z maksymalnym współczynnikiem kompresji

Kompresja:

```
gzip plik
gzip -k plik
gzip -rk /tmp/folder <- utworzy archiwum każdego pliku w
katalogu /tmp/folder/.
gzip -l archive.gz
```

Dekompresja:

```
gzip -d archive.gz
gzip -dk archive.gz
```

Domyślny poziom kompresji to -6.

Test kompresji:

```
gzip -rck --best /usr/bin/ > archive-best.gz <- przekierowuje
skompresowane pliki do określonego pliku archiwum
gzip -rck --fast /usr/bin/ > archive-fast.gz <- porównanie
plików
```

Typowe przykłady praktyczne

1. Kompresowanie plików za pomocą tar

Tworzenie archiwum tar bez kompresji

```
tar -cvf archive.tar /path/to/files
```

Wyodrębnianie archiwum tar

```
tar -xvf archive.tar
```

Lista zawartości archiwum tar

```
tar -tvf archive.tar
```

2. Kompresja za pomocą Gzip

Utwórz archiwum tar i skompresuj za pomocą gzip

```
tar -cvzf archive.tar.gz /path/to/files
```

Wyodrębnij archiwum tar.gz

```
tar -xvzf archive.tar.gz
```

Kompresja pojedynczego pliku za pomocą gzip

```
gzip nazwa_pliku.txt
```

Dekompresja pliku gzip

```
gunzip filename.txt.gz
```

3. Kompresowanie za pomocą Bzip2

Utwórz archiwum tar i skompresuj za pomocą bzip2

```
tar -cvjf archive.tar.bz2 /path/to/files
```

Wyodrębnij archiwum tar.bz2

```
tar -xvjf archive.tar.bz2
```

Kompresja pojedynczego pliku za pomocą bzip2

```
bzip2 nazwa_pliku.txt
```

Dekompresja pliku bzip2

```
bunzip2 nazwa_pliku.txt.bz2
```

4. Kompresja za pomocą XZ

Utwórz archiwum tar i skompresuj za pomocą xz

```
tar -cvJf archive.tar.xz /path/to/files
```

Wyodrębnij archiwum tar.xz

```
tar -xvJf archive.tar.xz
```

Kompresja pojedynczego pliku za pomocą xz

```
xz nazwa_pliku.txt
```

Dekompresja pliku xz

```
unxz nazwa_pliku.txt.xz
```

5. Porównywanie współczynników kompresji

Sprawdź rozmiar pliku przed i po kompresji

```
ls -lh filename.txt filename.txt.gz
```

Porównanie wydajności kompresji różnych algorytmów

```
du -sh archive.tar archive.tar.gz archive.tar.bz2
archive.tar.xz
```

Dodatkowe uwagi

- **gzip** jest szybszy, ale zapewnia umiarkowaną kompresję.
- **bzip2** ma lepszą kompresję, ale jest wolniejszy.
- **xz** oferuje najlepszą kompresję, ale jest najwolniejsza.
- **tar** jest przydatny do grupowania wielu plików w jedno archiwum przed kompresją.

Pliki i uprawnienia

Wprowadzenie

Uprawnienia są kombinacją trzech uprawnień:

- r- odczyt (4)
- w- pisanie (2)
- x- wykonać (1)

odpowiednio dla właściciela, grupy i pozostałych użytkowników, np:

-rw-r-r- 1 root users 0 Oct 24 18:04 file

Plik **ma uprawnienia 644**, tj:

- odczyt i zapis (rw-: 4+2=6 w notacji dziesiętnej) dla właściciela, w tym przypadku dla użytkownika "root".
- odczyt w prawo dla grupy (r- : 4 notacja dziesiętna), w tym przypadku dla grupy "users".
- odczyt w prawo dla reszty (r- : 4 zapis dziesiętny)

Polecenie służące do modyfikowania i wyświetlania uprawnień:

```
ls -la                          # listowanie
chmod 755                       # zmiana uprawnień
chown owner:group               # zmiana właściciela
umask                           # wyświetla domyślne
                                  przypisanie uprawnień
```

Domyślny umask ustawiony w /etc/profile

SetUID - pozwala określić, pod jakim identyfikatorem użytkownika program jest wykonywany, w przypadku plików oznacza to, że proces będzie miał prawa użytkownika, który jest właścicielem pliku wykonywalnego, a nie użytkownika uruchamiającego ten plik. Należy być bardzo ostrożnym przy ustawianiu tego bitu i nie nadawać uprawnień administratora (root) przez przypadek. Takim przykładem jest "passwd" lub "ping", ustawiony przyjmuje wartość 4 co oznacza binarne 100. chmod 4755 file

SetGID - pozwala określić, pod jakim identyfikatorem grupy program jest wykonywany, w przypadku użycia z katalogiem wszystkie podkatalogi i pliki w nim mają grupę katalogu nadrzędnego, jest ona ustawiona jako drugi bit cyfry ósemkowej (binarnie 010).

```
chmod 2755 plik
chmod 2644 file
```

Sticky bit - pozwala programowi pozostać w pamięci po zakończeniu działania. Jest to pozostałość z przeszłości, ponieważ nie ma potrzeby, aby programy pozostawały w pamięci po zakończeniu. Jest najczęściej używany z katalogiem. Gdy jest używany z katalogiem, użytkownicy mogą usuwać tylko pliki, dla których mają wyraźne uprawnienia do zapisu, nawet jeśli mają uprawnienia do zapisu do katalogu, takiego jak katalog /tmp, jest ustawiony jako dodatkowa 1, która jest umieszczona przed standardowymi 3 bitami.

```
chmod 1755 plik
```

Ustawienie na katalogu grupy, nadaj chmod g+w * dla katalogów.

chmod g+s * - niebezpieczne bo nie zabezpiecza przed usunięciem

chmod +t * - zabezpiecza przed usunięciem tylko /tmp to ma

Uprawnienia nadane a efektywne

Special Bit	User	Group	Others
	r w x	r w x	r w **t** r w **T**
0 0 1	1 1 1	1 1 1	1 1 1 1 1 0
1	7	7	7 6

Jeśli ten bit NIE jest ustawiony, jest oznaczony jako "T", jeśli jest ustawiony, jest to "t".

t = T+x

s = S+x

ACL

- ACL - (Access Control List), które zapewniają możliwość ustawienia rozszerzonej grupy uprawnień do plików i katalogów.

- Aby móc korzystać z ACL, system plików musi być zamontowany z włączoną obsługą ACL, opcja acl musi być dodana do zamontowanego systemu plików w pliku /etc/fstab. W systemie plików XFS taka obsługa jest domyślnie włączona, w systemach plików EXT obsługa ACL musi być włączona

- Listy ACL dzielą się na dwie kategorie:

- access ACLs - listy ACL dostępu, ustawione na plikach i/lub katalogach
- default ACLs - domyślne listy ACL, ustawiane tylko na poziomie katalogu. Pliki i podkatalogi zawarte w katalogu z ustawionymi domyślnymi ACL dziedziczą domyślne ACL katalogu nadrzędnego, tzn. domyślne ACL ustawione dla katalogu spowoduje, że wszystkie pliki w nim utworzone będą miały określone domyślne uprawnienia

- WAŻNE!!! Przypisanie tych uprawnień nie wpłynie na uprawnienia istniejących już w katalogu plików, a jedynie na te nowo utworzone!

```
mount -o remount,acl /home
```

getfacl - wyświetla ustawienia ACL dla pliku i katalogu. setfacl - ustawia, modyfikuje i usuwa ACL dla pliku lub katalogu. chacl - zmienia ustawienia ACL dla pliku lub katalogu. Polecenie systemowe IRIX UNIX.

Opcje setfacl:

- -m: Ustawia lub modyfikuje ACL
- -x: Usuwa określoną listę ACL
- -d: Ustawia domyślne listy ACL
- -k: Usuwa wszystkie domyślne listy ACL
- -b: Usuwa wszystkie listy ACL
- -R: Ustawia ACL rekurencyjnie na wszystkich plikach i podkatalogach.
- -c: Pomija wyświetlanie nagłówka uprawnień

Zakres:

- u[ser]:UID:perms: Uprawnienia przypisane do określonego użytkownika (nazwa użytkownika lub UID). Użytkownik musi znajdować się w pliku /etc/passwd.
- g[roup]:GID:perms: Uprawnienia przypisane do określonej grupy (nazwa użytkownika lub GID). Użytkownik musi znajdować się w pliku /etc/group.
- m[ask]:perms: Maksymalne uprawnienia, jakie dany użytkownik lub grupa może mieć do pliku lub katalogu. Np. rw- oznacza, że żaden użytkownik lub grupa nie będzie miała więcej uprawnień niż do odczytu i zapisu.
- o[ther]:perms: Uprawnienia przypisane do użytkowników, którzy nie są członkami grupy właściciela.

Typowe przykłady praktyczne

Nietypowe listy ACL (dostępu).

Jeśli przyznamy użytkownikowi student prawa do zapisu i odczytu i jednocześnie zmienimy maskę na tylko do odczytu, efektywne uprawnienia dla studenta będą tylko do odczytu:

```
setfacl -m u:student:rw,m:r plik1
getfacl -c file1
```

Oznacza to, że użytkownik user1 nie będzie mógł modyfikować tego pliku, nawet jeśli wydaje się, że ma uprawnienia do zapisu.

Po zmianie maski jak poniżej, użytkownik student będzie mógł modyfikować plik1.

```
setfacl -m m:rw plik1
getfacl -c plik1
getfacl /home/student/plik1
setfacl -m g:finances:rx marketing/ <- przypisanie ACL dla
grupy
setfacl -d -m g:finance:rx marketing/ <- przypisanie
domyślnego ACL dla grupy
setfacl -d -m g:marketing:rx finance/
setfacl -m u:student:rx /home/student/plik1 <- przypisanie
ACL dla użytkownika
setfacl -x u:student /home/student/plik1 <- usuń ACL dla
użytkownika
```

Usunięcie wszystkich list ACL:

```
setfacl -b /path/to/file
```

** Domyślne listy ACL**.

Zaloguj się jako użytkownik user1 i utwórz katalog projects. Użyj polecenia getfacl, aby sprawdzić domyślne uprawnienia:

```
pwd
mkdir projects
getfacl projects
```

Przyznajmy domyślny ACL do odczytu i zapisu dla użytkowników user2 i user3 do katalogu projektów:

```
setfacl -m d:u:user2:6,d:u:user3:6 projects
getfacl -c projects
```

W katalogu projektów utwórzmy podkatalog dev i sprawdźmy, czy dziedziczy on ustawienia ACL z katalogu nadrzędnego.

```
cd projects
mkdir dev
getfacl -c dev
```

Utwórzmy plik file1 w katalogu projektów i sprawdźmy, czy dziedziczy on ACL z katalogu nadrzędnego.

```
touch file1
getfacl -c file1
```

Powinniśmy zobaczyć, że maksymalne uprawnienia dla członków grupy to odczyt i zapis. Prawo do wykonywania jest nieskuteczne ze względu na ustawienia maski.

Filesystem Hierarchy System

Wprowadzenie

Standard hierarchii systemu plików (FHS) definiuje strukturę katalogów i ich zawartość w systemach operacyjnych Unix i Linux. Pomaga zachować spójność między dystrybucjami i zapewnia, że użytkownicy i aplikacje mogą przewidywalnie znaleźć potrzebne pliki.

FHS organizuje pliki w predefiniowane katalogi, z których każdy służy określonemu celowi. Zrozumienie tej hierarchii jest kluczowe dla administrowania systemem, rozwiązywania problemów i instalacji oprogramowania.

- Hierarchiczny
- Nazwa pliku jest tylko jedną z właściwości jego i-węzła, który jest najbardziej podstawowym i fundamentalnym obiektem.
- I-węzły są przechowywane w systemie plików i zawierają informacje takie jak prawa, rozmiar znacznika czasu itp.
- Listę aktualnie obsługiwanych systemów plików można znaleźć pod adresem /proc/filesystems
- system plików - urządzenie blokowe - możemy je skopiować do pliku i używać jak systemu plików

Wspólne katalogi FHS

Poniżej znajduje się tabela podsumowująca standardowe punkty montażowe FHS i ich przeznaczenie:

Katalog	Cel
/	Katalog główny, katalog najwyższego poziomu systemu plików.
/bin	Niezbędne pliki binarne poleceń potrzebne do uruchomienia systemu i trybu pojedynczego użytkownika.
/boot	Zawiera pliki bootloadera, w tym obrazy jądra i initramfs.
/dev	Pliki urządzeń reprezentujące komponenty sprzętowe (np. /dev/sda).
/etc	Ogólnosystemowe pliki konfiguracyjne i skrypty.
/home	Katalogi domowe użytkowników (np. /home/user).
/lib	Biblioteki współdzielone wymagane do podstawowego działania systemu.
/lib64	64-bitowe biblioteki współdzielone (dla systemów 64-bitowych).
/media	Punkt montowania nośników wymiennych, takich jak dyski USB i płyty CD.
/mnt	Tymczasowy punkt montowania dla ręcznie montowanych systemów plików.

/opt	Opcjonalne pakiety oprogramowania instalowane ręcznie.
/proc	Wirtualny system plików zawierający informacje o procesach i systemie.
/root	Katalog domowy użytkownika root.
/run	Przechowuje zmienne dane runtime (np. identyfikatory procesów, gniazda).
/sbin	Pliki binarne do administrowania systemem, zwykle do użytku przez superużytkowników.
/srv	Dane dla usług takich jak serwery WWW (/srv/www).
/sys	Wirtualny system plików zapewniający interakcję jądra i sprzętu.
/tmp	Pliki tymczasowe, które są usuwane po ponownym uruchomieniu komputera.
/usr	Binaria użytkownika, biblioteki, dokumentacja i kod źródłowy.
/var	Zmienne pliki, takie jak dzienniki, pamięci podręczne i kolejki poczty.

Typowe przykłady praktyczne

1. Sprawdzanie wykorzystania dysków przez katalogi FHS

```
du -sh /var /home /tmp
```

2. Wyświetlanie plików w /etc w celu wyświetlenia plików konfiguracyjnych

```
ls -l /etc
```

3. Sprawdzanie zamontowanych systemów plików i wykorzystania pamięci masowej

```
df -h
```

4. Wyświetlanie uruchomionych procesów w /proc

```
ls /proc| head -10
```

5. Sprawdzanie wersji jądra z /proc

```
cat /proc/version
```

Miękkie i twarde linki

Miękkie linki

Soft link - skrót, po usunięciu pliku link nadal istnieje, ale nie wskazuje na nic (ponieważ usunęliśmy plik). Liczba i-węzłów (i-nodes) = 1.

```
ln -s plik link                 # miękki link
ls -li                          # lista plików z liczbą węzłów
```

Twarde łącze

Hard link - duplikat, wszystkie zmiany dokonane na linku lub na oryginalnym pliku są widoczne wszędzie, po usunięciu pliku link pozostaje i nie jest usuwany, jest nadal dostępny i użyteczny, prawa nadane plikowi lub linkowi są duplikowane na duplikacie. Liczba i-węzłów (i-węzłów) = 2.

```
ln file link                    # twardy link
ls -li                          # lista plików z liczbą węzłów
```

Aby znaleźć wszystkie hardlinki, można uruchomić polecenie find:

```
find . -inum NUM
```

Dodatkowe uwagi

- Struktura FHS ma kluczowe znaczenie dla kompatybilności oprogramowania i integralności systemu.
- /usr/local/ jest używane dla ręcznie zainstalowanego oprogramowania, oddzielając je od pakietów zarządzanych przez system.
- Administratorzy systemu powinni unikać modyfikowania katalogów systemowych, chyba że jest to konieczne, aby zapobiec uszkodzeniu systemu operacyjnego.

Manager Pakietów

Wprowadzenie

Menedżer pakietów oprogramowania to narzędzie, które automatyzuje proces instalacji, aktualizacji, konfiguracji i usuwania oprogramowania w systemie Linux. Upraszcza zarządzanie zależnościami i zapewnia, że oprogramowanie jest poprawnie zainstalowane ze wszystkimi wymaganymi komponentami.

W dystrybucjach opartych na Debianie, takich jak Ubuntu, **APT (Advanced Package Tool)** jest domyślnym menedżerem pakietów. Współpracuje on z repozytoriami pakietów systemu w celu wydajnej instalacji i aktualizacji oprogramowania.

Typowe przykłady praktyczne

Korzystanie z APT do zarządzania pakietami

1. Instalowanie pakietów

Aby zainstalować pakiet, użyj:

```
sudo apt install< package-name>
```

Przykład:

```
sudo apt install vim
```

Spowoduje to zainstalowanie edytora tekstu Vim.

2. Wyszukiwanie pakietów

Aby wyszukać pakiet przed jego instalacją:

```
apt search< package-name>
```

Przykład:

```
apt search nginx
```

3. Wyświetlanie informacji o pakiecie

Aby uzyskać szczegółowe informacje o pakiecie:

```
apt show< package-name>
```

Przykład:

```
apt show curl
```

4. Aktualizacja list pakietów

Przed instalacją lub aktualizacją pakietów należy zaktualizować lokalną bazę danych pakietów:

```
sudo apt update
```

Powoduje to odświeżenie informacji o pakietach z repozytoriów.

5. Aktualizacja zainstalowanych pakietów

Aby zaktualizować wszystkie zainstalowane pakiety:

```
sudo apt upgrade
```

Do pełnej aktualizacji systemu należy użyć

```
sudo apt full-upgrade
```

6. Usuwanie pakietów

Aby odinstalować pakiet, ale zachować jego pliki konfiguracyjne:

```
sudo apt remove< nazwa-pakietu>
```

Przykład:

```
sudo apt remove nano
```

Aby usunąć pakiet i jego pliki konfiguracyjne:

```
sudo apt purge< package-name>
```

Przykład:

```
sudo apt purge apache2
```

7. Czyszczenie nieużywanych pakietów

Aby usunąć nieużywane zależności:

```
sudo apt autoremove
```

Aby wyczyścić lokalną pamięć podręczną pakietów:

```
sudo apt clean
```

Dodatkowe uwagi

- Przed instalacją lub aktualizacją oprogramowania należy zawsze zaktualizować listę pakietów.
- Regularnie używaj sudo apt upgrade, aby zapewnić bezpieczeństwo systemu.
- Usuwanie niepotrzebnych pakietów za pomocą autoremove pomaga zwolnić miejsce na dysku.

Informacje Sprzętowe

Wprowadzenie

Aby efektywnie zarządzać systemem Linux, administratorzy często muszą zbierać informacje o komponentach sprzętowych. Kilka narzędzi wiersza poleceń pomaga wyświetlić szczegóły dotyczące sprzętu systemu, w tym podłączonych urządzeń USB, procesora, pamięci, pamięci masowej i innych.

Typowe przykłady praktyczne

Narzędzia do wyświetlania informacji o sprzęcie

1. Wyświetlanie urządzeń USB

Aby wyświetlić listę wszystkich podłączonych urządzeń USB:

```
lsusb
```

To polecenie dostarcza szczegółowych informacji o podłączonych urządzeniach USB, takich jak dyski flash, klawiatury i zewnętrzne dyski twarde.

Aby uzyskać szczegółowe informacje o określonym urządzeniu USB:

```
lsusb -v
```

2. Wyświetlanie urządzeń PCI

Aby wyświetlić listę urządzeń PCI, takich jak karty sieciowe i procesory graficzne:

```
lspci
```

Więcej szczegółowych informacji:

```
lspci -v
```

3. Sprawdzanie informacji o procesorze

Aby wyświetlić szczegółowe specyfikacje procesora:

```
lscpu
```

Aby uzyskać nieprzetworzone szczegóły z systemu plików /proc:

```
cat /proc/cpuinfo
```

4. Sprawdzanie informacji o pamięci

Aby wyświetlić użycie pamięci systemowej:

```
free -h
```

Szczegółowe specyfikacje pamięci:

```
cat /proc/meminfo
```

5. Sprawdzanie urządzeń pamięci masowej

Aby wyświetlić listę dostępnych urządzeń blokowych (dysków i partycji):

```
lsblk
```

Aby wyświetlić szczegółowe informacje o partycjach dysku:

```
fdisk -l
```

6. Sprawdzanie stanu dysku twardego (status SMART)

Aby sprawdzić stan dysku twardego:

```
sudo smartctl -a /dev/sda
```

(Wymaga pakietu smartmontools).

7. Wyświetlanie interfejsów sieciowych

Aby wyświetlić listę wszystkich interfejsów sieciowych:

```
ip link show
```

Aby wyświetlić szczegóły interfejsu sieciowego:

```
ifconfig -a   # nie używać, zamiast tego użyj polecenia 'ip'
ip addr show
```

8. Sprawdzanie stanu baterii (laptopy)

Aby sprawdzić stan baterii w laptopie:

```
upower -i /org/freedesktop/UPower/devices/battery_BAT0
```

9. Lista zamontowanych systemów plików

Aby wyświetlić zamontowane systemy plików:

```
mount| column -t
```

Aby wyświetlić użycie systemu plików:

```
df -h
```

10. Wyświetlanie informacji o jądrze i systemie

Aby uzyskać wersję jądra Linux:

```
uname -r
```

Aby uzyskać szczegółowe informacje o systemie:

```
uname -a
```

Dodatkowe uwagi

- Niektóre polecenia wymagają uprawnień roota (sudo), zwłaszcza te, które wchodzą w interakcje z komponentami sprzętowymi.
- Urządzenia USB mogą nie zawsze być wyświetlane na liście, jeśli nie zostaną prawidłowo wykryte przez system.
- smartctl wymaga zainstalowania pakietu smartmontools.

Przydatne komendy

Wprowadzenie

Aby wyszukać plik według nazwy w bieżącym katalogu i podkatalogach:

```
find . -name "nazwa_pliku.txt"
```

Aby znaleźć pliki zmodyfikowane w ciągu ostatnich 7 dni:

```
find /path/to/search -type f -mtime -7
```

Aby zlokalizować puste pliki:

```
find /path/to/search -type f -empty
```

Korzystanie z funkcji ‘locate’

Aby szybko znaleźć plik:

```
locate filename.txt
```

Aby zaktualizować bazę danych lokalizacji:

```
sudo updatedb
```

Użycie funkcji ‘whereis’

Aby znaleźć plik binarny, źródło i instrukcję dla polecenia:

```
whereis ls
```

Użycie funkcji ‘which’

Aby znaleźć dokładną lokalizację polecenia w ścieżce systemowej:

```
which python3
```

Dodatkowe uwagi

- find umożliwia wyszukiwanie na podstawie różnych kryteriów, takich jak rozmiar, czas modyfikacji i typ.
- locate jest szybsze, ale opiera się na zaktualizowanej bazie danychb.
- whereis i które są przydatne do lokalizowania plików binarnych i zainstalowanego oprogramowania.