Linux Commands Cheat Sheet 💻 🐧

We created this Linux Commands Cheat Sheet initially for students of our DevOps & Linux Bootcamp. But we're now sharing it with any and all DevOps Engineers, SysAdmins, and Developers that want to learn and remember some of the key Linux Commands and have a quick reference guide to the basics of Linux.

If you've stumbled across this cheatsheet and are just starting to learn Linux, you've made a great choice!

Linux powers the internet. It's everywhere. From the smallest to the biggest companies like Amazon, Microsoft, SpaceX. They're all using Linux on their backend, so it's great to learn if you're interested in becoming a DevOps Engineer or SysAdmin.

However, if you're stuck in an endless cycle of YouTube tutorials and want to start building real world projects and actually get hired, then come join the Zero To Mastery Academy.

You'll learn Linux + devops from actual industry professionals alongside thousands of students in our private Discord community.

You'll not only learn to become a top 10% DevOps Engineer by learning advanced topics most courses don't cover, but you'll also build awesome projects that you can add to your portfolio and wow employers with!

Just want the cheatsheet? No problem! Please enjoy and if you'd like to submit any suggestions, feel free to email us at support@zerotomastery.io

Contents

The Linux Terminal

Getting Help in Linux

MAN Pages

- Use `man command` to view the manual for a command.
  - **Example**: `man ls`

The man pages are navigated using the `less` command with shortcuts:

- `h` : Get help within `less`.
- `q` : Quit `less`.
- `enter` : Show next line.
- `space` : Show next screen.
- `/string` : Search forward for a string.
- `?string` : Search backward for a string.
- `n` / `N` : Next/previous appearance of the search term.

Checking Command Type

- `type rm`: Check if `rm` is a shell built-in or an executable file.
    - **Example**: `rm is /usr/bin/rm`
- `type cd`: Check if `cd` is a shell built-in.
    - **Example**: `cd is a shell builtin`

Getting Help for Shell Built-in Commands

- `help command`: Get help for shell built-in commands.
    - **Example**: `help cd`
- `command --help`: Get help for executable commands.
    - **Example**: `rm --help`

Searching Man Pages

- `man -k uname`: Search for `uname` in all man pages.
- `man -k "copy files"`: Search for "copy files" in man pages.
- `apropos passwd`: Search for `passwd` related man pages.

Keyboard Shortcuts

```
TAB TAB: Display all commands or filenames starting with wr
CTRL + L: Clear the current line.
CTRL + D: Close the shell.
CTRL + U: Cut the current line.
CTRL + A: Move cursor to start of the line.
Ctrl + E: Move cursor to the end of the line.
CTRL + C: Stop the current command.
CTRL + Z: Sleep the running program.
CTRL + ALT + T: Open a terminal.
```

Bash History

- `history`: Display the history.
- `history -d 100`: Remove a specific line from history.
- `history -c`: Clear the entire history.
- `echo $HISTFILESIZE`: Print the number of commands saved in the history file.
- `echo $HISTSIZE`: Print the number of history commands saved in memory.
- `!!`: Rerun the last command.

- `!20` : Run a specific command from history.
- `!-10` : Run the last nth command.
- `!abc` : Run the last command starting with `abc`.
- `!abc:p` : Print the last command starting with `abc`.
- `CTRL + R` : Reverse search through history.
- `HISTTIMEFORMAT="%d/%m/%y %T "` : Record date/time of each command (add to `~/.bashrc` for persistence).

Getting Root Access (sudo, su)

- `sudo command` : Run a command as root (for users in `sudo` or `wheel` group).
- `sudo su` : Become root temporarily.
- `sudo passwd root` : Set the root password.
- `passwd username` : Change a user's password.
- `su` : Become root (if root has a password).

Linux Paths

Paths

- **Absolute**: Starts with `/`
- **Relative**: Relative to the current location
- `.` : Current working directory
- `..` : Parent directory
- `~` : User's home directory

Changing Directories

- `cd` : To user's home directory
- `cd ~` : To user's home directory
- `cd -` : To the last directory
- `cd /path_to_dir` : To `path_to_dir`
- `pwd` : Print the current working directory

Installing Tools

- `sudo apt install tree`: Installs the `tree` command

Using Tree

- `tree directory/`: Example: `tree .`
- `tree -d .`: Print only directories
- `tree -f .`: Print absolute paths

The ls Command

Usage: `ls [OPTIONS] [FILES]`

Listing Directories

- `ls`, `ls .`: Current directory
- `ls ~ /var /`: Multiple directories

Options

- `-l`: Long listing
- `-a`: All files (including hidden)
- `-1`: Single column
- `-d`: Directory information
- `-h`: Human-readable sizes
- `-S`: Sort by size
- `-X`: Sort by extension
- `--hide`: Hide specific files
- `-R`: Recursive listing
- `-i`: Inode number

Disk Usage

- `du -sh ~`: Size of home directory

File Timestamps and Date

- `ls -lu`: Access time (`atime`)
- `ls -l`, `ls -lt`: Modification time (`mtime`)

- `ls -lc`: Change time (`ctime`)
- `stat file.txt`: All timestamps
- `ls -l --full-time /etc/`: Full timestamps

Modifying Timestamps with Touch

- `touch file.txt`: Create or update timestamps
- `touch -a file`, `touch -m file`: Modify `atime` or `mtime`
- `touch -m -t 201812301530.45 a.txt`: Specific date/time
- `touch -d "2010-10-31 15:45:30" a.txt`: Both `atime` and `mtime`
- `touch a.txt -r b.txt`: Copy timestamps

Date and Calendar

- `date`: Current date/time
- `cal`, `cal 2021`, `cal 7 2021`: Calendars
- `cal -3`: Previous, current, next month
- `date --set="2 OCT 2020 18:00:00"`: Set date/time

Sorting with ls

- `ls -l`: Sorted by name
- `ls -lt`: Sorted by `mtime`, newest first
- `ls -ltu`: Sorted by `atime`
- `ls -ltu --reverse`: Reverse order

Viewing Files (cat, less, more, head, tail, watch)

Displaying File Contents

- `cat filename`: Display content
- `cat -n filename`: Line numbers
- `cat filename1 filename2 > filename3`: Concatenate

Less Shortcuts

- `h` : Help
- `q` : Quit
- `enter` : Next line
- `space` : Next screen
- `/string` : Search forward
- `?string` : Search backward
- `n` / `N` : Next/previous search result

Tail and Head

- `tail filename` : Last 10 lines
- `tail -n 15 filename` : Last 15 lines
- `tail -n +5 filename` : Starting with line 5
- `tail -f filename` : Real-time updates
- `head filename` : First 10 lines
- `head -n 15 filename` : First 15 lines

Monitoring Commands

- `watch -n 3 ls -l` : Refresh every 3 seconds

Working with Files and Directories

Creating and Updating Files

- `touch filename` : Create a new file or update timestamps.

Creating Directories

- `mkdir dir1` : Create a new directory.
- `mkdir -p mydir1/mydir2/mydir3` : Create nested directories.

The cp Command

Copy files and directories:

- `cp file1 file2`: Copy `file1` to `file2`.
- `cp file1 dir1/file2`: Copy to another directory with a different name.
- `cp -i file1 file2`: Prompt before overwrite.
- `cp -p file1 file2`: Preserve permissions.
- `cp -v file1 file2`: Verbose output.
- `cp -r dir1 dir2/`: Recursively copy directories.
- `cp -r file1 file2 dir1 dir2 dest_dir/`: Copy multiple items to a destination.

The mv Command

Move or rename files and directories:

- `mv file1 file2`: Rename a file.
- `mv file1 dir1/`: Move to a directory.
- `mv -i file1 dir1/`: Prompt before overwrite.
- `mv -n file1 dir1/`: Prevent overwriting.
- `mv -u file1 dir1/`: Update based on modification time.
- `mv file1 dir1/file2`: Move and rename.
- `mv file1 file2 dir1/ dir2/ dest_dir/`: Move multiple items.

The rm Command

Remove files and directories:

- `rm file1`: Remove a file.
- `rm -v file1`: Verbose removal.
- `rm -r dir1/`: Remove a directory.
- `rm -rf dir1/`: Force removal without prompt.
- `rm -ri file1 dir1/`: Prompt for each removal.

Secure File Deletion

- `shred -vu -n 100 file1`: Securely overwrite and remove a file.

Piping and Command Redirection

Piping Examples

- `ls -lSh /etc/ | head`: View the top 10 largest files.
- `ps -ef | grep sshd`: Check if `sshd` is running.
- `ps aux --sort=-%mem | head -n 3`: Top 3 processes by memory.

Command Redirection

Redirect output and errors:

- `ps aux > processes.txt`: Output to a file.
- `id >> users.txt`: Append output.
- `tail -n 10 /var/log/*.log > output.txt 2> errors.txt`: Separate output and errors.
- `tail -n 2 /etc/passwd /etc/shadow > all.txt 2>&1`: Redirect all to one file.
- `cat /var/log/auth.log | grep "fail" | wc -l`: Count occurrences.

Finding Files with locate and find

locate

- `sudo apt install plocate`: Install `plocate`.
- `sudo updatedb`: Update the database.
- `locate filename`: Find a file by name.
- `locate -i filename`: Case insensitive search.
- `locate -b '\filename'`: Exact name search.
- `locate -r 'regex'`: Regular expression search.
- `locate -e filename`: Check file existence.
- `which command`: Show command path.

find

Search with various options:

- `find ~ -type f -size +1M`: Files over 1MB.
- Options include `-type`, `-name`, `-iname`, `-size`, `-perm`, `-links`, `-atime`, `-mtime`, `-ctime`, `-user`, and `-group`.

Searching for Text Patterns with grep

Usage: `grep [OPTIONS] PATTERN FILE`

Options

- `-n`: Print line number.
- `-i`: Case insensitive.
- `-v`: Invert match.
- `-w`: Match whole words.
- `-a`: Include binary files.
- `-R`: Recursive search.
- `-c`: Count matches.
- `-C n`: Context display (n lines around the match).

Extracting ASCII Characters from Binary Files

- `strings binary_file`: Example `strings /bin/ls`.

VIM - Text Editor

Modes

- Command Mode: Default on entry.
- Insert Mode: Editing text.
- Last Line Mode: Save/exit commands.

Config File

- VIM settings: `~/.vimrc`.

Commands

- `i`, `I`, `a`, `A`, `o`: Enter Insert Mode.
- `:w!`, `:q!`, `:wq!`, `:e!`: Save/quit commands in Last Line Mode.
- `x`, `dd`, `ZZ`, `u`, `G`, `$`, `0`, `^`: Editing commands in Command Mode.
- `/string`, `?string`, `n`, `N`: Search commands in Command Mode.
- `vim -o file1 file2`: Open files in stacked windows.
- `vim -d file1 file2`: Highlight differences.

Navigation

- `Ctrl+w`: Switch between files.

Account Management

```
## Account Management
/etc/passwd # users and info:
/etc/shadow# users' passwords
/etc/group# groups

## User Commands
useradd [OPTIONS] username # Create user.
usermod [OPTIONS] username # Modify user.
userdel -r username        # Delete user.

## Group Commands
groupadd group_name # Create group.
groupdel group_name # Delete group.

## Examples
useradd -m -d /home/john -c "C++ Developer" -s /bin/bash -G
usermod -aG developers,managers john # Example of modifying
```

Monitoring Users

```
## Commands
who -H    # User info.
id        # User info.
whoami    # User info.
w         # System usage.
uptime    # System usage.
last      # Login history.
last -u username # Login history for a specific user.
```

File Permissions

Understanding Permissions

- **Legend**: `u` (user), `g` (group), `o` (others), `a` (all), `r` (read), `w` (write), `x` (execute), `-` (no access).

Displaying Permissions

- `ls -l /etc/passwd`: View file permissions.
- `stat /etc/shadow`: Detailed permission stats.

Changing Permissions

- `chmod u+r filename`: Add read to user.
- `chmod u+r,g-wx,o-rwx filename`: Adjust multiple permissions.
- `chmod ug+rwx,o-wx filename`: Set multiple permissions.
- `chmod ugo+x filename`: Add execute to all.
- `chmod a+r,a-wx filename`: Modify all permissions.

Absolute Mode

- `chmod 777 filename`: Set all permissions for all.
- `chmod 755 filename`: Read & execute for group and others.
- `chmod 644 filename`: Read-only for group and others.

Special Permissions

- **SUID**: `chmod u+s executable_file`.
- **SGID**: `chmod g+s projects/`.
- **Sticky Bit**: `chmod o+t temp/`.

UMASK

- Display: `umask`.
- Set new value: `umask new_value`.

Ownership

- Owner: `chown new_owner file`.

- Group: `chgrp new_group file`.
- Both: `chown new_owner:new_group file`.
- Recursive: `chown -R new_owner file`.

File Attributes

- Display: `lsattr filename`.
- Change: `chattr +-attribute filename`.

Processes

Process Viewing

- `type rm`: Check if `rm` is built-in or executable.
- `ps`: Processes in current terminal.
- `ps -ef`, `ps aux`, `ps aux | less`: System processes.
- `ps aux --sort=%mem | less`: Sort by memory usage.
- `ps -ef --forest`: ASCII process tree.
- `ps -f -u username`: Processes by user.
- `pgrep -l sshd`, `pgrep -f sshd`, `ps -ef | grep sshd`: Check for `sshd`.
- `pstree`, `pstree -c`: Hierarchical process tree.

Dynamic Real-Time View

- `top`: Start system monitor.
- `top` shortcuts: `h` for help, `space` for refresh, `d` for delay, etc.
- `top -d 1 -n 3 -b > top_processes.txt`: Top in batch mode.
- Install `htop` for an interactive view.

Killing Processes

- `kill -l`: List signals.
- `kill pid`, `kill -SIGNAL pid1 pid2 ...`: Send signals.

- `kill $(pidof process_name)`: Kill using `pidof`.

Background and Foreground Management

- `command &`: Run in background.
- `jobs`: List jobs.
- `Ctrl + Z`: Stop process.
- `fg %job_id`: Resume in foreground.
- `bg %job_id`: Resume in background.
- `nohup command &`: Immune to hangups.

Networking

Getting Network Interface Information

- `ifconfig`: Enabled interfaces.
- `ifconfig -a`, `ip address show`: All interfaces.
- `ifconfig enp0s3`, `ip addr show dev enp0s3`: Specific interface.
- `ip -4 address`: Only IPv4 info.
- `ip -6 address`: Only IPv6 info.
- `ip link show`, `ip link show dev enp0s3`: L2 info, including MAC.
- `route`, `route -n`, `ip route show`: Default gateway.
- `systemd-resolve --status`: DNS servers.

Setting Network Interfaces

- `ifconfig enp0s3 down`, `ip link set enp0s3 down`: Disable interface.
- `ifconfig enp0s3 up`, `ip link set enp0s3 up`: Enable interface.
- `ifconfig -a`, `ip link show dev enp0s3`: Check status.
- `ifconfig enp0s3 192.168.0.222/24 up`, `ip address add 192.168.0.112/24 dev enp0s3`: Set IP.
- `ifconfig enp0s3:1 10.0.0.1/24`: Secondary IP.
- `route del default gw`, `ip route del default`, `ip route add default via`: Default gateway.

- `ifconfig enp0s3 hw ether`, `ip link set dev enp0s3 address`: Change MAC.

Netplan for Static Network Configuration on Ubuntu

1. Stop/Disable NetworkManager.
2. Create/Modify YAML in `/etc/netplan`.
3. Apply config: `sudo netplan apply`.
4. Verify: `ifconfig`, `route -n`.

OpenSSH Configuration and Management

Installation

Ubuntu

```
sudo apt update && sudo apt install openssh-server openssh-client
```

CentOS

```
sudo dnf install openssh-server openssh-clients
```

Server Connection

```
ssh -p 22 username@server_ip  # Connect using default SSH p
ssh -p 22 -l username server_ip  # Connect with a specific
ssh -v -p 22 username@server_ip  # Connect in verbose mode

# Ubuntu
sudo systemctl status ssh  # Check SSH status
sudo systemctl stop ssh    # Stop SSH service
sudo systemctl restart ssh # Restart SSH service
sudo systemctl enable ssh  # Enable SSH to start on boot
sudo systemctl is-enabled ssh  # Check if SSH is enabled or

# CentOS
sudo systemctl status sshd  # Check SSH status
sudo systemctl stop sshd    # Stop SSH service
sudo systemctl restart sshd # Restart SSH service
sudo systemctl enable sshd  # Enable SSH to start on boot
sudo systemctl is-enabled sshd  # Check if SSH is enabled o
```

Security Configuration

Edit /etc/ssh/sshd_config and then apply changes by restarting SSH:

- Change port: Port 2278
- Disable root login: PermitRootLogin no
- Restrict user access: AllowUsers user1 user2
- Configure firewall to filter SSH access
- Enable Public Key Authentication, disable password-based login
- Use SSH Protocol 2 only
- Set client session intervals and max attempts for security

Remember to consult the man page (`man sshd_config`) for detailed configuration options.

File Transfer Techniques with SCP and RSYNC

SCP Usage

```
# Copy local file to remote host
scp a.txt john@80.0.0.1:~
scp -P 2288 a.txt john@80.0.0.1:~ # Custom port

# Copy from remote to local
scp -P 2290 john@80.0.0.1:~/a.txt .

# Copy entire directory to remote
scp -P 2290 -r projects/ john@80.0.0.1:~
```

RSYNC Commands

```
# Sync local directory to local backup
sudo rsync -av /etc/ ~/etc-backup/

# Mirror directory, deleting extraneous files from dest
sudo rsync -av --delete /etc/ ~/etc-backup/

# Exclude files during sync
rsync -av --exclude-from='~/exclude.txt' /source/ /dest/
```

```
# Sync over SSH with custom port
sudo rsync -av -e 'ssh -p 2267' /etc/ student@192.168.0.108
```

Exclude Patterns Example

```
# exclude.txt could include patterns like:
*.avi
music/
abc.mkv

# Exclude specific file types during transfer
rsync -av --exclude='*.mkv' /source/ /dest/
```

WGET for File Download

```
# Install wget
sudo apt install wget # Ubuntu
sudo dnf install wget # CentOS

# Basic file download
wget https://example.com/file.iso

# Resume incomplete download
wget -c https://example.com/file.iso

# Download with bandwidth limit
wget --limit-rate=100k https://example.com/file.iso

# Download multiple files
wget -i urls.txt # urls.txt contains list of URLs

# Recursive download for offline viewing of a website
wget -mkEpnp http://example.org
```

Use these commands to efficiently copy files and directories across systems and for downloading content from the internet, ensuring data synchronization and maintaining web accessibility.

NETSTAT and SS Usage

```
# Display all ports and connections
sudo netstat -tupan
sudo ss -tupan
```

```
# Check if port 80 is open
netstat -tupan | grep :80
```

LSOF Commands

```
# List open files
lsof

# Files opened by a specific user
lsof -u username

# Files opened by a specific command/process
lsof -c sshd

Open files for TCP ports in LISTEN state
lsof -iTCP -sTCP:LISTEN
lsof -iTCP -sTCP:LISTEN -nP
```

Use these commands to monitor network connections, check for open ports, and view files opened by users or processes, especially for security and troubleshooting.

Nmap Scanning Guide

```
# SYN Scan (root required)
nmap -sS 192.168.0.1

# TCP Connect Scan
nmap -sT 192.168.0.1

# Scan All Ports
nmap -p- 192.168.0.1

# Scan Specific Ports
nmap -p 20,22-100,443,1000-2000 192.168.0.1

# Service Version Detection
nmap -p 22,80 -sV 192.168.0.1

# Ping Scan Network
nmap -sP 192.168.0.0/24

# Skip Host Discovery
nmap -Pn 192.168.0.0/24

# Exclude Specific IP from Scan
```

```
nmap -sS 192.168.0.0/24 --exclude 192.168.0.10

# Output Scan to File
nmap -oN output.txt 192.168.0.1

# OS Detection
nmap -O 192.168.0.1

# Aggressive Scan
nmap -A 192.168.0.1

# Read Targets from File & Output to File without DNS Resol
nmap -n -iL hosts.txt -p 80 -oN output.txt
```

Only scan your own networks and systems, or those you have explicit permission to test. Unauthorized scanning can be illegal.

Software Management with DPKG and APT

DPKG

- View .deb file info: `dpkg --info package.deb`
- Install from .deb: `sudo dpkg -i package.deb`
- List installed programs: `dpkg --get-selections` or `dpkg-query -l`
- Find by name: `dpkg-query -l | grep ssh`
- List package files: `dpkg -L openssh-server`
- Find owning package: `dpkg -S /bin/ls`
- Remove package: `sudo dpkg -r package`
- Purge package: `sudo dpkg -P package`

APT

- Update index: `sudo apt update`
- Install/update: `sudo apt install apache2`
- List upgradable: `sudo apt list --upgradable`
- Full upgrade: `sudo apt full-upgrade`
- Remove: `sudo apt remove package`
- Purge: `sudo apt purge package`

- Auto remove dependencies: `sudo apt autoremove`
- Clean cache: `sudo apt clean`
- List all packages: `sudo apt list`
- Search: `sudo apt list | grep nginx`
- Show package info: `sudo apt show nginx`
- List installed: `sudo apt list --installed`

Task Scheduling using Cron

```
crontab -e # Edit crontab
crontab -l # List tasks
crontab -r # Remove tasks

# Schedule Format:
* * * * * command # Every minute
15 * * * * command # Hourly
30 18 * * * command # Daily
3 22 * * 1 command # Weekly
10 6 1 * * command # Monthly
@yearly command # Yearly
@reboot command # At reboot
```

Getting System Hardware Information

General Hardware

```
lshw                  # Full hardware info
lshw -short           # Short format
lshw -json            # JSON format
lshw -html            # HTML format
```

CPU Information

```
lscpu                 # CPU details
lshw -C cpu           # Hardware-specific CPU details
lscpu -J              # JSON format
```

Memory Information

```
dmidecode -t memory           # RAM specs
dmidecode -t memory | grep -i size
dmidecode -t memory | grep -i max
free -m                       # Memory usage
```

PCI and USB Devices

```
lspci                         # PCI buses and connected dev
lspci | grep -i wireless
lspci | grep -i vga
lsusb                         # USB controllers and devices
lsusb -v                      # Verbose output
```

Storage Devices

```
lshw -short -C disk
fdisk -l                      # List disks
fdisk -l /dev/sda
lsblk                         # Block devices list
```

Network Devices

```
lshw -C network
iw list                       # Wi-Fi cards
iwconfig                      # Wi-Fi configuration
iwlist scan                   # Wi-Fi networks scan
```

System Information via /proc

```
cat /proc/cpuinfo             # CPU info
cat /proc/meminfo             # Memory info
cat /proc/version             # System version
uname -r                      # Kernel version
uname -a                      # All system info
```

Battery Power

```
acpi -bi                      # Battery info
acpi -V                       # All ACPI info
```

Working with Device Files (dd)

```
# Backup MBR
dd if=/dev/sda of=~/mbr.dat bs=512 count=1

# Restore MBR
dd if=~/mbr.dat of=/dev/sda bs=512 count=1

# Clone partition
dd if=/dev/sda1 of=/dev/sdb2 bs=4M status=progress
```

Use these commands to check hardware specifications and perform operations with device files safely.

Service Management

```
# Analyze boot process
systemd-analyze
systemd-analyze blame

# List active units
systemctl list-units
systemctl list-units | grep ssh

# Service status
sudo systemctl status nginx.service

# Stop service
sudo systemctl stop nginx

# Start service
sudo systemctl start nginx

# Restart service
sudo systemctl restart nginx

# Reload service config
sudo systemctl reload nginx
sudo systemctl reload-or-restart nginx

# Enable service at boot
sudo systemctl enable nginx

# Disable service at boot
sudo systemctl disable nginx

# Check if service is enabled at boot
sudo systemctl is-enabled nginx

# Mask service
```

```
sudo systemctl mask nginx

# Unmask service
sudo systemctl unmask nginx
```

Ubuntu

```
sudo systemctl status ssh       # Check SSH service status
sudo systemctl stop ssh         # Stop SSH service
sudo systemctl restart ssh      # Restart SSH service
sudo systemctl enable ssh       # Enable SSH to start on bo
sudo systemctl is-enabled ssh   # Check if SSH is enabled o
```

CentOS

```
sudo systemctl status sshd       # Check SSHD service statu
sudo systemctl stop sshd         # Stop SSHD service
sudo systemctl restart sshd      # Restart SSHD service
sudo systemctl enable sshd       # Enable SSHD to start on
sudo systemctl is-enabled sshd   # Check if SSHD is enabled
```

Security Configuration

To configure security settings, edit `/etc/ssh/sshd_config`. Apply changes by restarting SSH. Key configurations include:

- Change SSH port:
  - `Port 2278`
- Disable root login:
  - `PermitRootLogin no`
- Restrict user access to specified users only:
  - `AllowUsers user1 user2`
- Configure firewall to filter SSH access
- Enable Public Key Authentication and disable password-based login
- Use SSH Protocol 2 only
- Set client session intervals and maximum attempts for increased security

Note: Consult the man page (`man sshd_config`) for detailed configuration options.

Bash Programming

Bash Aliases

```
alias                      # List all aliases
alias name='command'       # Create an alias
unalias name               # Remove an alias
```

Useful Aliases

```
alias c='clear'
alias cl='clear; ls; pwd'
alias root='sudo su'
alias ports='netstat -tupan'
alias sshconfig='sudo vim /etc/ssh/sshd_config'
alias update='sudo apt update && sudo apt dist-upgrade -y &
```

Interactive File Manipulation

```
alias cp='cp -i'
alias mv='mv -i'
alias rm='rm -i'
```

Bash Variables

```
variable="value"           # Define a variable
echo $variable             # Reference a variable
declare -r const=100       # Define a read-only variable
unset variable             # Unset a variable
env | grep PATH            # Find an environment variable
export PATH=$PATH:~/bin    # Modify the PATH variable
```

Special Variables

```
$0, $1, $2, ..., ${10}     # Script name & positional argume
$#                         # Number of positional arguments
```

```
"$*"                       # All positional arguments as a s
$?                         # Exit status of the last commanc
```

Program Flow Control

```
if [ condition ]; then command; fi
if [ condition ]; then command; else other_command; fi
if [ condition ]; then command; elif [ condition ]; then...
```

Test Conditions

```
# Numeric comparisons: -eq, -ne, -lt, -le, -gt, -ge
# File checks: -s, -f, -d, -x, -w, -r
# String comparisons: =, !=, -n (not zero), -z (is zero)
# Logical operators: && (and), || (or)
```

Loops and Functions

```
for i in {1..5}; do echo "Loop $i"; done                # Fc
while [ condition ]; do command; done                   # Wh
case "$variable" in pattern) command;; esac             # Ca
function name() { command; }                            # Fu
name() { command; }                                     # Al
name                                                    # Ca
```

Command Examples

```
crontab -e    # Edit crontab file
crontab -l    # List crontab entries
crontab -r    # Remove crontab entries
```

Combine these constructs to write effective bash scripts for task automation and system management.

If you want to learn Linux Commands and become a DevOps Engineer or Systems Administrator, check out our Linux Bootcamp Course and our DevOps Career Path!

Back To Top

Quick Links

Home
Pricing
Testimonials
Blog
Cheat Sheets
Industry
Newsletters
Community

The Academy

Courses
Career Paths
Career Path
Quiz
Web
Development
Machine
Learning & AI
Generative AI
Data Analytics
DevOps & Cloud
Design
Cyber Security
Lifetime
Challenge
Learning
Passport

Company

About ZTM
Ambassadors
Contact Us

Excellent
4.9 out of 5

Privacy Terms Cookies