



OPENSUSE TUMBLEWEED · ROLLING · ZYPPER · YAST · KDE

openSUSE Handbook

For advanced users

Run the system from the inside. zypper in depth, Snapper snapshots and rollback, firewall, systemd, networking, users, SSH and containers.

CONTENTS

1. zypper in depth
2. Snapshots and rollback with Snapper
3. Firewall (firewalld)
4. systemd services
5. Logs: journalctl
6. Networking: nmcli, ip, ss
7. Users and permissions
8. SSH
9. Podman, Docker, Distrobox
10. Text processing and pipelines
11. Archives, backups, syncing
12. Customizing the shell
13. Performance and power

Yellow italic text (like `<package>`) marks the parts you fill in. In the terminal, paste is Ctrl+Shift+V and copy is Ctrl+Shift+C. Commands that start with `sudo` change the system; read what they do before running them.



sudogrimoire.com/opensuse/

Free, searchable handbooks for Linux, Windows and macOS

zypper in depth

More zypper for daily maintenance.

```
$ sudo zypper dup --no-allow-vendor-change
```

Distribution upgrade that keeps packages from their current repository.

```
$ sudo zypper addlock <package>
```

Locks a package so updates don't change it; `removeLock` undoes it.

```
$ zypper search --installed-only <word>
```

Searches only installed packages.

```
$ sudo zypper verify
```

Checks dependencies and fixes problems.

```
$ sudo zypper clean --all
```

Cleans the package cache.

```
$ sudo zypper modifyrepo --priority 90 <repo>
```

Sets repository priority (lower number wins).

```
$ less /var/log/zypp/history
```

The full history of installed and removed packages.

Snapshots and rollback with Snapper

openSUSE's default Btrfs layout and Snapper make undoing a bad update easy.

```
$ sudo snapper list
```

Lists snapshots with their numbers and descriptions.

```
$ sudo snapper create --description "<note>"
```

Takes a snapshot now.

```
$ sudo snapper status <old>..<new>
```

Files that changed between two snapshots.

```
$ sudo snapper undochange <old>..<new>
```

Reverts the changes between two snapshots.

```
$ sudo snapper rollback
```

After booting a read-only snapshot, makes it the default system.

TIP Your home folder is a separate subvolume and is not rolled back.

Firewall (firewalld)

openSUSE uses firewalld, and it is on by default.

```
$ sudo firewall-cmd --state
```

Is the firewall running.

```
$ sudo firewall-cmd --list-all
```

The active zone, open services and ports.

```
$ sudo firewall-cmd --permanent --add-  
service=ssh
```

Allows a service permanently.

```
$ sudo firewall-cmd --permanent --add-  
port=8080/tcp
```

Opens a port permanently.

```
$ sudo firewall-cmd --reload
```

Applies the permanent rules.

systemd services

systemd manages services (background daemons), timers and the boot process.

```
$ systemctl status <service>
```

Shows status, the latest log lines and the PID.

```
$ sudo systemctl start|stop|restart <service>
```

Starts / stops / restarts a service.

```
$ sudo systemctl enable --now <service>
```

Starts it at boot and starts it now.

```
$ sudo systemctl disable --now <service>
```

Stops it from starting at boot and stops it now.

```
$ sudo systemctl mask <service>
```

Prevents the service from being started at all.
`unmask` undoes it.

```
$ systemctl --failed
```

Lists failed units.

```
$ systemctl list-units --type=service --  
state=running
```

Running services.

```
$ systemctl list-unit-files --state=enabled
```

Units that start at boot.

```
$ systemctl list-timers
```

Scheduled jobs (systemd's answer to cron).

```
$ systemctl cat <service>
```

Shows the unit file and any drop-ins.

```
$ sudo systemctl edit <service>
```

Creates an override (drop-in) without touching the packaged file.

```
$ sudo systemctl daemon-reload
```

Makes systemd reread unit files after you change them.

```
$ systemctl --user enable --now <service>
```

User services; definitions live under
`~/.config/systemd/user/`.

```
$ loginctl enable-linger $USER
```

Lets user services run even when you are not logged in.

```
$ systemd-analyze blame
```

Lists how long each service took at boot.

```
$ systemd-analyze critical-chain
```

Shows the critical path of the boot.

Logs: journalctl

System logs are kept in the systemd journal. openSUSE keeps the systemd journal persistent; there is no separate syslog by default.

```
$ journalctl -b
```

Everything since this boot.

```
$ journalctl -b -1 -p err
```

Errors from the previous boot. The first place to look after a crash.

```
$ journalctl -u <service> -f
```

Follows a service's log live.

```
$ journalctl -p warning --since "1 hour ago"
```

Warnings and worse from the last hour.

```
$ journalctl --since today --grep "<word>"
```

Searches today's log for text.

```
$ journalctl -k
```

Kernel messages only (like dmesg).

```
$ journalctl --list-boots
```

Lists recorded boots.

```
$ journalctl --disk-usage
```

Space used by the logs.

```
$ sudo journalctl --vacuum-time=2weeks
```

Deletes logs older than two weeks. `--vacuum-size=500M` also works.

```
$ coredumpctl list
```

Records of crashed programs (if systemd-coredump is installed). `coredumpctl info <PID>` gives details.

```
$ sudo dmesg -w
```

Follows the kernel ring buffer live (to see what happens when you plug in USB).

Networking: nmcli, ip, ss

On desktop installs NetworkManager runs the network; its command-line tool is `nmcli` and its text UI is `nmtui`.

```
$ nmcli device status
```

Network devices and their state.

```
$ nmcli device wifi list
```

Visible Wi-Fi networks with signal strength.

```
$ nmcli device wifi connect "<SSID>" password "  
<password>"
```

Connects to a Wi-Fi network.

```
$ nmcli connection show
```

Saved connections.

```
$ nmcli connection up|down "<name>"
```

Brings a connection up / down.

```
$ sudo nmcli connection modify "<name>"  
ip4.method manual ip4.addresses  
192.168.1.50/24 ip4.gateway 192.168.1.1  
ip4.dns "1.1.1.1 9.9.9.9"
```

Sets a static IP. Then `nmcli connection up "<name>"`.

```
$ nmtui
```

Menu-driven text UI; useful without a graphical desktop.

```
$ sudo hostnamectl set-hostname <new-name>
```

Changes the computer's name.

```
$ ip -br a
```

Interfaces and IP addresses, briefly.

```
$ ip route
```

Routing table and default gateway.

```
$ resolvectl status
```

DNS servers (when systemd-resolved is used). Otherwise `cat /etc/resolv.conf`.

```
$ ss -tulpn
```

Listening TCP/UDP ports and which program owns them (sudo to see the program).

```
$ ping -c 4 <address>
```

Reachability test.

```
$ tracepath <address>
```

The route packets take.

```
$ curl -I https://<site>
```

Fetches a site's HTTP headers.

```
$ dig <domain>
```

DNS lookup (bind-utils package).

Users and permissions

On this system sudo rights come from membership in the `wheel` group.

```
$ id
```

Your user ID and groups.

```
$ sudo useradd -m -G wheel -s /bin/bash <user>
```

New user with a home folder and sudo rights.
Then `sudo passwd <user>`.

```
$ sudo usermod -aG <group> <user>
```

Adds a user to a group (without `-a` it removes them from other groups). Takes effect after logging in again.

```
$ sudo userdel -r <user>
```

Deletes a user along with their home folder.

```
$ passwd
```

Changes your own password.

```
$ chmod +x <file>
```

Makes a file executable.

```
$ chmod 644 <file>
```

Owner reads/writes, others read. 755 is typical for folders and scripts.

```
$ chmod -R u+rwX,go-rwx <folder>
```

Makes a folder accessible only to its owner.

```
$ sudo chown -R <user>:<group> <path>
```

Changes the owner.

```
$ setfacl -m u:<user>:rw <file>
```

Gives a specific user extra permissions (ACL).
`getfacl` shows them.

```
$ sudo -i
```

Opens a root shell. Type exit when done.

```
$ sudo visudo -f /etc/sudoers.d/<name>
```

Edits sudo rules with syntax checking.

TIP If you ticked “Use this password for system administrator”, root has the same password as your user.

SSH

```
$ sudo systemctl enable --now sshd
```

Starts the SSH server; open it in the firewall with `sudo firewall-cmd --permanent --add-service=ssh`.

```
$ ssh-keygen -t ed25519 -C "<Label>"
```

Creates a key pair (`~/.ssh/id_ed25519`).

```
$ ssh-copy-id <user>@<server>
```

Uploads your public key to the server; after that you log in without a password.

```
$ ssh <user>@<server> -p <port>
```

Connects.

```
$ ssh -L 8080:localhost:80 <user>@<server>
```

Tunnels local port 8080 to port 80 on the server.

```
$ scp <file> <user>@<server>:~/
```

Copies a file to the server. `-r` for folders.

TIP Define short names in `~/.ssh/config` :

```
Host myserver · HostName 192.168.1.10 · User user · IdentityFile ~/.ssh/id_ed25519
```

Then just `ssh myserver`.

TIP To turn off password logins, write `PasswordAuthentication no` into `/etc/ssh/sshd_config.d/10-local.conf` and restart the SSH service.

Podman, Docker, Distrobox

Distrobox lets you use other distributions' tools without cluttering the host; Podman or Docker run services. Podman is largely compatible with Docker commands and doesn't need root.

```
$ sudo zypper install podman distrobox
```

Installs Podman and Distrobox.

```
$ distrobox create -i  
registry.fedoraproject.org/fedora:latest -n  
<name>
```

Creates a container of another distribution (e.g. `-i archlinux`, `-i ubuntu:24.04`).

```
$ distrobox enter <name>
```

Enters it; your home folder is shared. Inside, `distrobox-export --app <app>` adds the app to the host menu.

```
$ podman run -it --rm debian bash
```

Starts a throwaway container.

```
$ podman ps -a
```

All containers.

```
$ podman images
```

Local images. `podman system prune -a` removes unused ones.

```
$ podman run -d -p 8080:80 -v  
~/site:/usr/share/nginx/html  
docker.io/library/nginx
```

Runs a service with a mounted folder.

```
$ sudo zypper install docker docker-compose &&  
sudo systemctl enable --now docker
```

Installs Docker and starts it.

```
$ sudo usermod -aG docker $USER
```

Use Docker without sudo (log in again). This group is equivalent to root access.

TIP To run a Podman container as a systemd service, use Quadlet: write a `.container` file under `~/.config/containers/systemd/`, run `systemctl --user daemon-reload` and start it.

Text processing and pipelines

| passes one command's output to the next. > writes to a file (overwriting), >> appends, 2>&1 sends errors to the same place. && runs if the previous command succeeded, || if it failed.

```
$ grep -rni "<text>" <folder>
```

Recursive, case-insensitive search with line numbers.

```
$ grep -v "^#" <file> | grep -v "^$"
```

Drops comments and blank lines; handy for reading config files.

```
$ sed -i 's/<old>/<new>/g' <file>
```

Find and replace in a file. Try without -i first and check the output.

```
$ awk '{print $1, $3}' <file>
```

Prints the 1st and 3rd column of each line.

```
$ sort <file> | uniq -c | sort -rn | head
```

Most frequently repeated lines.

```
$ wc -l <file>
```

Line count.

```
$ cut -d: -f1 /etc/passwd
```

First colon-separated field (user names).

```
$ find . -name "*.log" -mtime +7 -delete
```

Deletes .log files older than 7 days.

```
$ find . -type f -name "*.jpg" -print0 | xargs  
-0 -I{} cp {} ~/Pictures/
```

Copies found files one by one (safe with spaces in names).

```
$ <command> | tee <file>
```

Writes output to both the screen and a file.

```
$ echo "<line>" | sudo tee -a /etc/<file>
```

Appends to a root-owned file (sudo echo >> doesn't work).

```
$ diff -u <old> <new>
```

Differences between two files.

```
$ watch -n 2 <command>
```

Reruns a command every 2 seconds.

```
$ curl -s <URL> | jq '.'
```

Pretty-prints JSON (jq package).

Archives, backups, syncing

```
$ tar -czvf <archive>.tar.gz <folder>
```

Archives a folder with gzip.

```
$ tar --zstd -cvf <archive>.tar.zst <folder>
```

Archives with zstd; faster than gzip and compresses better.

```
$ tar -xvf <archive>
```

Extracts any tar archive. Use `-C <folder>` for a target folder.

```
$ tar -tvf <archive>
```

Lists the contents without extracting.

```
$ zip -r <archive>.zip <folder>
```

Creates a zip. Extract with `unzip <archive>.zip`.

```
$ 7z x <archive>.7z
```

Extracts 7z / rar and similar formats (7zip package).

```
$ rsync -avh --progress <source>/ <target>/
```

Syncs a folder; copies only what changed. The trailing / means "the contents".

```
$ rsync -avh --delete --dry-run <source>/  
<target>/
```

Mirror, deleting from the target what's not in the source; see what would happen first with `--dry-run`.

```
$ rsync -avz -e ssh <source>/ <user>@<server>:  
<path>/
```

Remote backup over SSH.

```
$ rclone config
```

Sets up cloud targets such as Google Drive, OneDrive and S3; sync with `rclone sync`.

Customizing the shell

Bash settings live in `~/.bashrc` in your home folder. Add aliases and variables at the end of the file.

```
$ nano ~/.bashrc
```

Opens the shell configuration file.

```
alias update='sudo zypper dup'
```

System update in one word. Write it into the file and save.

```
alias ll='ls -lah --group-directories-first'
```

A handy detailed listing.

```
$ source ~/.bashrc
```

Loads the changes without opening a new terminal.

```
export PATH="$HOME/.local/bin:$PATH"
```

Adds your own scripts folder to PATH.

```
$ echo $SHELL
```

The shell you are using.

```
$ sudo zypper install zsh && chsh -s  
/usr/bin/zsh
```

Installs Zsh and makes it the default (log in again). Same method for Fish.

```
$ type <command>
```

Shows whether a name is an alias, a function or a file.

TIP When writing a script, put `#!/usr/bin/env bash` on the first line and `set -euo pipefail` on the second; it stops on errors and catches undefined variables. Then `chmod +x script.sh`.

Performance and power

KDE Plasma shows power profiles in the battery applet (power-profiles-daemon).

```
$ powerprofilesctl list
```

Power profiles and which one is active (power-profiles-daemon).

```
$ powerprofilesctl set performance
```

Switches profile: power-saver / balanced / performance.

```
$ sudo zypper install sensors
```

CPU and disk temperatures; run `sensors` after installing.

```
$ sudo zypper install btop nvidia-smi
```

Advanced system and GPU monitors.

```
$ sudo iotop -o
```

Processes writing the most to disk (iotop package).

```
$ systemd-analyze
```

Total boot time (firmware, loader, kernel, userspace).

```
$ sudo fstrim -av
```

Runs TRIM on SSDs. To do it weekly: `sudo systemctl enable --now fstrim.timer`.

```
$ upower -i $(upower -e | grep BAT)
```

Battery health, capacity and charge information.

```
$ free -h && swapon --show
```

RAM and swap areas.

Commands are written for openSUSE Tumbleweed with KDE Plasma (zypper, YaST, Btrfs). Most also work on Leap. Menu names follow the English interface. Commands that start with `sudo` change the system; read what they do before running them.