Référence rapide systemd

Gestion des services, unités, minuteries et journalctl

Gestion des services

Commandes de base

```
systemctl start nginx
systemctl stop nginx
systemctl restart nginx
systemctl reload nginx    # reload config
systemctl status nginx
```

Activer / Désactiver

```
systemctl enable nginx    # start at boot
systemctl disable nginx   # remove from boot
systemctl enable --now nginx  # enable + start
systemctl is-enabled nginx
```

États d'un service

active (running)	Le service est en cours d'exécution normalement
active (exited)	A été exécuté une fois et s'est terminé avec succès
inactive (dead)	Le service est arrêté
failed	Le service a planté ou s'est terminé avec une erreur
activating	Le service est en cours de démarrage

Fichiers d'unité

Emplacement des fichiers d'unité

/etc/systemd/system/	Unités créées par l'administrateur (priorité maximale)
/run/systemd/system/	Unités générées à l'exécution
/usr/lib/systemd/system/	Unités installées par les paquets
~/.config/systemd/user/	Unités au niveau utilisateur

Unité de service de base

```
[Unit]
Description=My Application
After=network.target
[Service]
ExecStart=/usr/bin/myapp --config /etc/myapp.conf
Restart=on-failure
User=appuser
[Install]
WantedBy=multi-user.target
```

Appliquer les modifications

```
systemctl daemon-reload   # reload unit files
systemctl restart myapp   # apply changes
```

Minuteries

Unité minuterie

```
[Unit]
Description=Run backup daily
[Timer]
OnCalendar=*-*-* 02:00:00
Persistent=true
[Install]
WantedBy=timers.target
```

Syntaxe OnCalendar

***-*-* 02:00:00**	Tous les jours à 2 h 00
Mon *-*-* 09:00:00	Chaque lundi à 9 h 00
***-*-01 00:00:00**	Premier jour de chaque mois
hourly / daily / weekly	Planifications abrégées

Gestion des minuteries

```
systemctl list-timers --all
systemctl start backup.timer
systemctl enable backup.timer
systemd-analyze calendar "daily"
```

Cibles

Cibles courantes

multi-user.target	Démarrage normal, multi-utilisateur, sans interface graphique
graphical.target	Bureau graphique complet
rescue.target	Mode de secours mono-utilisateur
emergency.target	Shell minimal, root uniquement
network-online.target	Réseau entièrement configuré
timers.target	Toutes les unités minuterie prêtes

Commandes de cibles

```
systemctl get-default
systemctl set-default multi-user.target
systemctl isolate rescue.target
systemctl list-dependencies graphical.target
```

Journalctl

Consulter les journaux

```
journalctl -u nginx        # logs for unit
journalctl -u nginx -f     # follow (tail)
journalctl -u nginx --no-pager
journalctl -b              # current boot only
```

Filtrer les journaux

```
journalctl --since "2026-03-01"
journalctl --since "1 hour ago"
journalctl -p err          # errors and above
journalctl _PID=1234
```

Niveaux de priorité

emerg (0)	Système inutilisable
alert (1)	Action immédiate requise
crit (2)	Condition critique
err (3)	Condition d'erreur
warning (4)	Condition d'avertissement
info (6)	Informatif
debug (7)	Messages de débogage

Maintenance des journaux

```
journalctl --disk-usage
journalctl --vacuum-size=500M
journalctl --vacuum-time=30d
```

Réseau

networkctl

```
networkctl list
networkctl status eth0
networkctl up eth0
networkctl down eth0
```

systemd-resolve

```
resolvectl status
resolvectl query example.com
resolvectl flush-caches
resolvectl statistics
```

Attente réseau

```
# In unit file [Unit] section:
After=network-online.target
Wants=network-online.target
```

Montages

Unité de montage

```
[Unit]
Description=Mount data volume
[Mount]
What=/dev/sdb1
Where=/mnt/data
Type=ext4
Options=defaults,noatime
[Install]
WantedBy=multi-user.target
```

Unité automount

```
[Unit]
Description=Automount data on access
[Automount]
Where=/mnt/data
TimeoutIdleSec=300
[Install]
WantedBy=multi-user.target
```

Convention de nommage

/mnt/data	Fichier d'unité : **mnt-data.mount**
/var/lib/app	Fichier d'unité : **var-lib-app.mount**

Le chemin de montage avec `/` remplacé par `-`, tiret initial supprimé

Environnement

Définir des variables d'environnement

```
[Service]
Environment=APP_ENV=production
Environment=PORT=8080
EnvironmentFile=/etc/myapp/env
```

Format du fichier d'environnement

```
# /etc/myapp/env
APP_ENV=production
DATABASE_URL=postgres://localhost/db
SECRET_KEY=changeme
```

Durcissement du service

ProtectSystem=strict	Système de fichiers en lecture seule sauf chemins autorisés
ProtectHome=true	Masquer /home, /root, /run/user
NoNewPrivileges=true	Empêcher l'escalade de privilèges
PrivateTmp=true	/tmp isolé pour le service
ReadWritePaths=/var/lib/myapp	Autoriser les écritures dans des chemins spécifiques

Dépendances

Directives d'ordre et de dépendance

After=b.service	Démarrer après b (ordre uniquement)
Before=b.service	Démarrer avant b (ordre uniquement)
Requires=b.service	Dépendance forte ; échec si b échoue
Wants=b.service	Dépendance souple ; ne pas échouer si b échoue
BindsTo=b.service	S'arrêter quand b s'arrête
Conflicts=b.service	Ne peut pas s'exécuter en même temps que b

Référence rapide systemd

Inspecter les dépendances

```
systemctl list-dependencies nginx
systemctl list-dependencies --reverse nginx
systemd-analyze dot nginx.service | dot -Tsvg > deps.svg
```

Modèles courants

Politiques de redémarrage

`Restart=no`	Ne jamais redémarrer (par défaut)
`Restart=on-failure`	Redémarrer en cas de code de sortie non nul
`Restart=always`	Toujours redémarrer (pour les démons)
`RestartSec=5`	Attendre 5 secondes avant de redémarrer
`StartLimitBurst=3`	Redémarrages max dans l'intervalle
`StartLimitIntervalSec=60`	Intervalle pour le comptage de rafales

Surcharger sans modifier

```
systemctl edit nginx  # creates drop-in
# /etc/systemd/system/nginx.service.d/override.conf
systemctl cat nginx   # show effective config
systemctl revert nginx  # remove overrides
```

Analyse système

```
systemd-analyze                 # boot time
systemd-analyze blame           # per-unit time
systemd-analyze critical-chain
systemctl list-units --failed
```