Services maison : le guide du local

Faire tourner ses propres services à la maison — messagerie, tableau de bord, sauvegarde, domotique — sans abonnement, sans nuage, sans envoyer ses données chez quelqu'un d'autre.

Version 1.0 — 2026-11-18

---

1. Le principe

Un service local a trois propriétés qu'un service cloud n'a pas :

- rien ne sort de votre réseau, sauf ce que vous décidez ;
- personne ne peut fermer le service en changeant ses conditions ;
- la panne est visible : c'est votre machine, vous pouvez regarder.

En contrepartie, vous devenez l'exploitant. Ce guide traite exactement de cela : comment installer, surveiller et sauvegarder un service pour qu'il tienne des années sans attention.

La règle d'or : un service que vous ne surveillez pas est un service qui tombera sans prévenir.

---

2. Choisir ce qu'on auto-héberge

Tout n'est pas worth la peine. Critères :

Auto-héberger	Laisser au cloud
données personnelles (santé, notes, photos)	service à fort débit (vidéo)
service que vous utilisez tous les jours	outil que vous oubliez
protocole ouvert (MQTT, ActivityPub, CalDAV)	application fermée sans API
besoin d'automatisation	service gratuit suffisant

Le piège classique : auto-héberger pour le principe, puis passer plus de temps à maintenir l'outil qu'à l'utiliser. Si un service gratuit vous convient et ne manipule pas vos données, gardez-le.

---

3. systemd : l'unité comme contrat

Une unité systemd décrit comment le service doit vivre : ce qu'il faut avant de le démarrer, ce qu'il faut faire s'il échoue, ce qu'il ne doit jamais pouvoir faire.

```
[Unit]
Description=Mon service
After=network.target mosquitto.service
Wants=mosquitto.service

[Service]
Type=simple
User=monuser
Group=gpio
ExecStart=/usr/bin/python3 /home/monuser/service/serveur.py
```

```
Restart=on-failure
RestartSec=10
NoNewPrivileges=true
ProtectSystem=full
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Quatre lignes qui changent tout :

- Restart=on-failure — un plantage ne laisse pas le service mort ;
- User= — le service tourne sans droits superutilisateur ;
- NoNewPrivileges, ProtectSystem=full — s'il est compromis, il ne peut pas écrire sur le système ;
- After= — il ne démarre pas avant son dépendance.

Ne jamais ajouter Restart=always sans réfléchir : un service qui plante en boucle redémarre en boucle. Préférer on-failure et comprendre la cause.

Timers plutôt que démons

Pour une tâche périodique, un *timer* est presque toujours préférable à un démon : il n'y a pas de processus permanent, donc rien à maintenir, et un échec ne laisse rien de coincé.

```
[Timer]
OnBootSec=2min
OnUnitActiveSec=60
AccuracySec=5
Unit=mon-service.service
```

---

4. Boucle locale : le défaut, pas l'exception

Un service qui n'a pas besoin d'être joignable depuis l'internet ne doit écouter que sur 127.0.0.1.

```
# Grafana
Environment=GF_SERVER_HTTP_ADDR=127.0.0.1
Environment=GF_USERS_ALLOW_SIGN_UP=false
Environment=GF_AUTH_ANONYMOUS_ENABLED=false
```

Deux raisons : une surface d'attaque nulle, et l'impossibilité d'exposer quelque chose par erreur. Pour accéder au service depuis l'extérieur, un tunnel (chiffrement de bout en bout, authentification) vaut mieux qu'une ouverture de port.

Vérifier : ss -ltnp | grep <port> doit montrer 127.0.0.1, jamais 0.0.0.0.

---

5. Surveiller : relever, pas supposer

Un système fiable ne se contente pas d'agir : il relit l'état qu'il croit avoir produit.

- « J'ai écrit le GPIO, donc le relais est fermé » → non. Relire la broche.
- « J'ai envoyé le message » → non. Lire le code de retour.
- « Le service est actif » → non. systemctl is-active et le journal.

Un bon tableau de surveillance pour une machine seule :

Grandeur	Où la lire	Seuil d'alerte
température du SoC	vcgencmd measure_temp	≥ 78 °C
throttling	vcgencmd get_throttled	≠ 0x0
tension du cœur	vcgencmd measure_volts	hors 0,8–1,4 V
unités en échec	systemctl --failed	≥ 1
disque	df /	≥ 90 %
synchronisation NTP	timedatectl show -p NTPSynchronized	non synchronisé
signal Wi-Fi	/proc/net/wireless	≤ −85 dBm

Deux écueils :

1. alerter sur tout → on ignore les alertes. Mieux vaut peu de signaux, mais des signaux qui comptent, dédupliqués (par exemple une nouvelle alerte d'un même type seulement après 6 h) ;
2. ne jamais tester l'alerte. Une alerte non testée est une alerte cassée.

---

6. Sauvegarder : la copie qui compte est celle hors de la machine

Une sauvegarde sur la même machine n'est pas une sauvegarde.

- Bundle git pour le code et les notes : un seul fichier, vérifiable (git bundle verify), transportable ;
- sommes de contrôle (sha256sum) pour détecter une corruption silencieuse ;
- au moins une copie hors de la machine — et idéalement chez deux prestataires différents.

Vérifier régulièrement, pas seulement écrire :

```
git fsck --full
sha256sum -c checksums.sha256
git bundle verify backups/depot-*.bundle
```

Une sauvegarde non testée n'existe pas.

---

7. Pièges observés (et évitables)

Piège	Symptôme	Remède
IPv6 configuré mais non routable	~24 s de blocage par requête	/etc/gai.conf → IPv4 d'abord, ou curl -4
bus 1-Wire flottant	périphériques fantômes 00-…	pull-up 4,7 kΩ
RPi.GPIO	instable sur Pi 5	lgpio / gpiod
base de données dans /home (700)	le service ne peut pas la lire	dossier partagé avec le bon groupe
source apt ajoutée sans vérification	apt-get update en erreur	vérifier le Release avant d'ajouter
RPC publics gratuits	403 / 429	toujours un repli, ne jamais faire planter la mesure
pkill -f <motif> dans un script	tue le script lui-même	filtrer par PID

---

8. Exemple complet

Une machine seule, quatre services :

```
collecteur  →  SQLite  →  Grafana (127.0.0.1:3000)
alarme      →  journaux + notification
```

C'est peu, et c'est suffisant pour des années de service. La complexité ne vient jamais du nombre de services, mais de l'absence de l'un de ces quatre piliers : surveillance, sauvegarde, redémarrage automatique, coupure de sécurité.

9. Limites

- Ce guide ne couvre pas la mise en ligne publique (c'est un sujet à part, avec ses responsabilités : journaux, mentions légales, notifications de violation).
- Les chemins et versions correspondent à Debian 13 / Raspberry Pi OS ; adapter à votre distribution.
- Aucun prix n'est garanti pour le matériel mentionné.