Engineer

systemd

le gestionnaire du démarrage et des services sous Linux

Ce que c’est

Sur presque toutes les principales distributions Linux, le processus portant l’identifiant 1 est systemd : le gestionnaire du système et des services qui initialise la machine au démarrage, lance et surveille ses services pendant son fonctionnement, puis arrête tout dans l’ordre à l’extinction. Le projet écrit son nom en minuscules, jamais SystemD ni System D, et le d final est l’ancienne marque Unix d’un démon. Il appartient à la famille des systèmes d’initialisation, avec les scripts d’initialisation SysV et Upstart qu’il a remplacés sur la plupart des distributions, OpenRC et runit que choisissent encore certaines petites distributions, et launchd qui assure le même rôle sur macOS.

Tout ce que systemd gère est une unité, décrite dans un court fichier texte composé de sections et de paires clé-valeur. Son manuel compte 11 types : service, socket, target, device, mount, automount, timer, swap, path, slice et scope. Un service exécute un programme. Un temporisateur lance une autre unité aux échéances prévues. Une socket conserve un port d’écoute et lance son service à la première connexion. Une cible regroupe des unités dans un état que la machine peut atteindre, comme un système multi-utilisateur avec réseau. Les exigences et l’ordre sont des réglages séparés : « ceci a besoin de cela » et « ceci démarre après cela » sont donc deux lignes différentes. systemd les transforme en une transaction de tâches de démarrage et d’arrêt qu’il peut vérifier avant de l’exécuter.

Deux choix de conception le distinguent des systèmes d’initialisation à scripts shell qui l’ont précédé. Chaque service s’exécute dans son propre groupe de contrôle Linux : le gestionnaire suit ainsi les processus de ce groupe, peut les arrêter ensemble, limiter leur mémoire et comptabiliser leur consommation de processeur. Le projet est aussi une suite, et pas seulement un gestionnaire : son site répertorie un démon de journalisation, le journal, ainsi que des composants

pour les ouvertures de session, la configuration réseau, la synchronisation de l'heure et la résolution de noms. Les distributions choisissent quels compléments activer ; le gestionnaire au PID 1 est la partie presque universelle.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce que systemd ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. L'entretien quotidien des serveurs, de la messagerie, des comptes et des mises à jour, pour que les outils indispensables restent sans histoire.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. 10 serveurs PostgreSQL et MS SQL sur Ubuntu — 10 serveurs de bases de données déployés et maintenus

10 serveurs PostgreSQL et MS SQL sur Ubuntu Contactez-nous →

Pourquoi c'est important

Sur un serveur Linux, la question est rarement de savoir s'il faut exécuter systemd, puisque la distribution l'a déjà choisi. La vraie décision est de l'utiliser comme superviseur et ordonnanceur, ou d'empiler par-dessus un autre gestionnaire de processus, la politique de redémarrage d'un démon de conteneurs ou cron.

L'utiliser directement réunit l'ordre de démarrage, la politique de redémarrage, les journaux, les limites de ressources et l'isolation dans un fichier par service, lu par le gestionnaire déjà en fonctionnement, sans second processus superviseur. Une unité peut plafonner la mémoire avec `MemoryMax=`, rendre les répertoires du système d'exploitation en lecture seule pour son service avec `ProtectSystem=`, et s'exécuter sous un utilisateur éphémère avec `DynamicUser=`. `systemd-analyze security` évalue l'exposition de chaque service sur une échelle de 0,0 à 10,0. Une tâche lancée par un temporisateur, contrairement à une ligne de crontab, écrit dans le journal, bénéficie de la même isolation que tout service et affiche sa dernière et sa prochaine exécution dans `systemctl list-timers`. Cette compétence est aussi transférable : un ingénieur capable de lire un fichier d'unité peut exploiter la plupart des serveurs Linux, quelle que soit leur distribution.

Les coûts sont réels eux aussi. Les unités sont propres à Linux : une équipe qui livre aussi sur macOS écrit donc des listes de propriétés launchd, et une équipe qui livre sur les BSD écrit des scripts rc. Dans la plupart des conteneurs, systemd est entièrement absent : l'application elle-même est le PID 1, et la politique de redémarrage relève de ce qui exécute le conteneur. Quand conteneurs et

services ordinaires partagent un hôte, Quadlet de Podman réduit cet écart en transformant la description d'un conteneur en unité de service ordinaire : les deux types de charge sont ainsi placés sous un seul superviseur.

À quoi cela ressemble en pratique

La plateforme propre à l'entreprise s'appuie sur systemd à trois endroits.

Dans le compte rendu de réalisation de septembre 2026, l'hôte de production décrit dans les services publics de l'entreprise réunis sur un hôte de 512 Mo était une machine à un cœur disposant d'environ 464 Mo de mémoire utilisable. Y résidaient un serveur web assurant TLS pour sept domaines, la forge Git, le service onion, un serveur Gemini et un serveur Gopher, ainsi qu'un dispositif de blocage des intrusions. L'automatisation des sauvegardes est réalisée séparément ; cela n'établit pas que des sauvegardes chiffrées étaient actives. À cette taille, chaque processus résident devait justifier sa mémoire : systemd était donc le superviseur, sans second processus superviseur superposé, et les conteneurs fonctionnaient comme unités Quadlet sous le même gestionnaire plutôt que sous leur propre démon de conteneurs. Lorsqu'il a fallu décider si l'hôte pouvait aussi accueillir un service onion, les échantillons de mesure consignés sur une journée ont montré que la mémoire disponible ne descendait jamais sous environ 310 Mo sur 464.

La réalisation des sauvegardes et des tests de restauration utilise des unités systemd pour un export de base de données là où il est configuré, un instantané restic chiffré et dédupliqué vers un stockage chez un autre fournisseur par SFTP, la purge selon la politique de rétention et un contrôle d'intégrité. Un processus mensuel distinct restaure un fichier connu et compare ses octets ; sauvegarde et restauration ont des signaux de réussite séparés. Les temporisateurs dépendent d'un dépôt configuré : le compte rendu du 2026-10-27 indique qu'ils étaient désactivés faute de dépôt et ne consigne aucune restauration réelle achevée. Le travail a aussi rencontré l'un des pièges de systemd décrits plus bas : la phrase secrète se trouve dans un fichier protégé que lisent les unités, car une valeur passée par l'environnement de l'unité subit l'interprétation des séquences d'échappement et une barre oblique inverse pouvait la modifier silencieusement.

La pile technique fixe la règle générale : des unités systemd ordinaires par défaut, Docker et Kubernetes étant réservés aux charges qui les justifient.

Où les problèmes surviennent

Une unité modifiée continue de fonctionner avec ses anciens réglages. systemd lit les fichiers d'unité en mémoire : une modification sur disque ne fait donc rien tant que `systemctl daemon-reload` n'a pas reconstruit la vue du gestionnaire, et `systemctl cat` peut afficher un fichier différent de celui que le gestionnaire en cours d'exécution utilise. Effectuer les modifications avec `systemctl edit`, qui recharge pour vous, ou recharger puis redémarrer le service.

Un secret arrive au service avec un caractère en moins. Les valeurs d'`Environment=` sont traitées selon les règles de systemd pour les guillemets : les séquences d'échappement de style C sont interprétées, et une barre oblique inverse littérale doit être doublée. La documentation de systemd avertit aussi que les variables d'environnement sont visibles par des clients non privilégiés via D-Bus. Placer les secrets dans un fichier et le transmettre avec `LoadCredential=`, qui remet au service une copie en lecture seule accessible uniquement à son utilisateur et à root.

Une mise à jour de paquet annule discrètement une correction locale. Les modifications apportées au fichier du fournisseur sous `/usr/lib/systemd/system/` sont écrasées à la prochaine mise à jour du paquet. Placer les changements dans un complément de configuration sous `/etc/systemd/system/<unit>.d/`, fusionné par-dessus et conservé lors des mises à jour. Les listes de commandes s'additionnent au lieu de se remplacer : un complément qui change la commande de démarrage l'efface donc d'abord par une affectation vide :

```
# /etc/systemd/system/app.service.d/override.conf
[Service]
ExecStart=
ExecStart=/usr/local/bin/app --config /etc/app/app.toml
Restart=on-failure
```

Une tâche nocturne manque son exécution quand le serveur est éteint. Un temporisateur calendaire dont l'échéance est passée pendant l'arrêt de la machine attend simplement le créneau suivant, car `Persistent=` vaut false par défaut. Définir `Persistent=true` pour les sauvegardes et les autres tâches qui doivent s'exécuter : l'exécution manquée se déclenche lorsque le temporisateur redevient actif, sous réserve d'un éventuel délai aléatoire configuré.

Un service qui plante reste arrêté. `Restart=` vaut `no` par défaut : un service qui meurt reste donc arrêté jusqu'à ce que quelqu'un le remarque. Définir `Restart=on-failure` pour les services de longue durée, en sachant qu'une fois la limite de démarrages de l'unité atteinte, le gestionnaire abandonne et qu'un démarrage manuel doit attendre la fin de l'intervalle ou un `systemctl reset-failed` .

Pour aller plus loin

- systemd(1) — le gestionnaire lui-même : les types d'unité, la séquence de démarrage et les signaux auxquels il répond.
- systemd.unit(5) — les fichiers d'unité, les dépendances, les limites de démarrage et les répertoires de compléments qui les surchargent.
- systemd.service(5) — les types de service, la politique de redémarrage et la remise à zéro des listes de commandes.
- systemd.exec(5) — l'environnement d'exécution, les options d'isolation et les informations d'identification.
- systemd.timer(5) — les temporisateurs calendaires et monotones, notamment `Persistent=` .
- systemctl(1) — la ligne de commande pour démarrer, modifier, recharger et inspecter les unités.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le parcours de déploiement, les risques opérationnels et les personnes d'astreinte. Nous pouvons définir un changement avec des vérifications, une procédure de reprise et une responsabilité claire.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce que systemd ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/systemd/