Engineer

systemd

das Programm, das unter Linux alles andere startet

Was es ist

Auf fast jeder verbreiteten Linux-Distribution hat systemd die Prozess-ID 1. Dieser System- und Dienstmanager fährt den Rechner beim Booten hoch, startet und überwacht seine Dienste während des Betriebs und beendet beim Herunterfahren alles in der richtigen Reihenfolge. Das Projekt schreibt seinen Namen klein, niemals SystemD oder System D. Das abschließende d ist die alte Unix-Kennzeichnung eines Daemons. systemd gehört zur Familie der Init-Systeme, neben den SysV-Init-Skripten und Upstart, die es auf den meisten Distributionen abgelöst hat, OpenRC und runit, die einige kleinere Distributionen weiterhin wählen, sowie launchd, das unter macOS dieselbe Aufgabe erfüllt.

Alles, was systemd verwaltet, ist eine Unit. Eine kurze Textdatei mit Abschnitten und Schlüssel-Wert-Paaren beschreibt sie. Das Handbuch zählt 11 Typen: service, socket, target, device, mount, automount, timer, swap, path, slice und scope. Eine Service-Unit führt ein Programm aus. Eine Timer-Unit startet eine andere Unit nach Zeitplan. Eine Socket-Unit hält einen lauschenden Port bereit und startet ihren Dienst bei der ersten eingehenden Verbindung. Ein Target fasst Units zu einem erreichbaren Systemzustand zusammen, etwa einem Mehrbenutzersystem mit Netzwerk. Anforderungen und Reihenfolge sind separate Einstellungen: „Dies benötigt jenes“ und „Dies startet nach jenem“ stehen deshalb in zwei verschiedenen Zeilen. systemd formt daraus eine gemeinsame Transaktion aus Start- und Stoppaufträgen, die es vor der Ausführung prüfen kann.

Zwei Entwurfsentscheidungen unterscheiden es von den früheren Init-Systemen aus Shell-Skripten. Jeder Dienst läuft in seiner eigenen Linux-Control-Group. Der Manager verfolgt deshalb die Prozesse in dieser Gruppe, kann sie gemeinsam stoppen, ihren Speicher begrenzen und ihren CPU-Verbrauch erfassen. Außerdem ist das Projekt sowohl Manager als auch Programmsammlung. Seine Website nennt einen Protokollierungsdaemon, das Journal, und Komponenten

für Anmeldungen, Netzwerkkonfiguration, Zeitsynchronisierung und Namensauflösung. Die Distributionen wählen, welche dieser zusätzlichen Bestandteile sie aktivieren. Nahezu überall vorhanden ist der Manager mit PID 1.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist systemd?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Server, E-Mail, Konten und Updates werden im Alltag betreut, damit die Dinge, auf die alle angewiesen sind, unauffällig funktionieren.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. 10 PostgreSQL-/MS-SQL-Server auf Ubuntu — 10 bereitgestellte und funktionsfähig gehaltene Datenbankserver

10 PostgreSQL-/MS-SQL-Server auf Ubuntu Kontakt aufnehmen →

Warum es wichtig ist

Auf einem Linux-Server lautet die Frage selten, ob systemd laufen soll; das hat die Distribution bereits entschieden. Die eigentliche Entscheidung ist, ob es auch als Supervisor und Zeitplaner eingesetzt wird oder ob ein weiterer Prozessmanager, die Neustartrichtlinie eines Containerdaemons oder cron darübergelegt wird.

Die direkte Nutzung legt Startreihenfolge, Neustartrichtlinie, Protokolle, Ressourcenlimits und Sandboxing in einer Datei pro Dienst zusammen. Diese liest der ohnehin laufende Manager, ohne einen zweiten Supervisorprozess. Eine Unit kann Speicher mit `MemoryMax=` begrenzen, mit `ProtectSystem=` die eigenen Verzeichnisse des Betriebssystems für ihren Dienst schreibgeschützt machen und mit `DynamicUser=` unter einem vorübergehend angelegten Benutzer laufen. `systemd-analyze security` bewertet auf einer Skala von 0,0 bis 10,0, wie stark jeder Dienst exponiert ist. Anders als eine crontab-Zeile protokolliert ein per Timer gestarteter Auftrag ins Journal, erhält dasselbe Sandboxing wie jeder Dienst und zeigt seinen letzten und nächsten Lauf in `systemctl list-timers`. Die Kenntnisse lassen sich auch anderswo einsetzen: Wer als Engineer eine Unit-Datei lesen kann, kann die meisten Linux-Server betreiben, unabhängig von deren Distribution.

Auch die Nachteile sind real. Units sind auf Linux beschränkt. Ein Team, das zusätzlich für macOS ausliefert, schreibt deshalb launchd-Property-Lists; für die BSD-Systeme werden rc-Skripte benötigt. In den meisten Containern gibt es überhaupt kein systemd: Die Anwendung selbst ist PID 1, und die

Neustartrichtlinie gehört zu dem System, das den Container betreibt. Teilen sich Container und gewöhnliche Dienste einen Host, verkleinert Podmans Quadlet diese Lücke. Es macht aus einer Containerbeschreibung eine normale Service-Unit, sodass beide Arten von Arbeitslast unter einem Supervisor laufen.

Wie es in der Praxis aussieht

Die eigene Plattform des Unternehmens nutzt systemd an drei Stellen.

Im Implementierungsbericht vom September 2026 war der im Beitrag über alle öffentlichen Dienste des Unternehmens auf einem Host mit 512 MB beschriebene Produktionshost ein Einkernrechner mit ungefähr 464 MB nutzbarem Speicher. Darauf liefen ein Webserver mit TLS für sieben Domains, die Git-Forge, der Onion-Dienst, je ein Server für Gemini und Gopher sowie die Sperrung von Einbruchsversuchen. Backup-Automatisierung ist separat implementiert; daraus folgen keine nachgewiesenen aktiven verschlüsselten Sicherungen. Bei so wenig Speicher musste jeder ständig laufende Prozess seinen Speicherverbrauch rechtfertigen. systemd war deshalb der Supervisor, ohne einen zweiten darübergelegten Supervisorprozess. Die Container liefen als Quadlet-Units unter demselben Manager statt unter einem eigenen Containerdaemon. Als zu klären war, ob der Host zusätzlich einen Onion-Dienst tragen konnte, zeigten dokumentierte Stichproben über einen Tag, dass der verfügbare Speicher nie unter ungefähr 310 MB von 464 sank.

Die Implementierung von Sicherungen und Wiederherstellungstests verwendet systemd-Units für einen Datenbankdump, sofern er konfiguriert ist, einen verschlüsselten, deduplizierten restic-Snapshot über SFTP auf Speicher bei einem anderen Anbieter, die Bereinigung nach Aufbewahrungsregeln und eine Integritätsprüfung. Ein separater monatlicher Ablauf stellt eine bekannte Datei wieder her und vergleicht ihre Bytes. Sicherung und Wiederherstellung haben getrennte Erfolgssignale. Die Timer hängen von einem konfigurierten Repository ab. Laut Implementierungsbericht vom 2026-10-27 waren sie ohne ein solches deaktiviert; eine tatsächlich abgeschlossene Wiederherstellung ist darin nicht dokumentiert. Die Arbeit stieß außerdem auf eine der unten beschriebenen tückischeren Eigenheiten von systemd: Die Passphrase liegt in einer geschützten Datei, die die Units lesen. Bei einem über die Umgebung der Unit übergebenen Wert würden nämlich Escape-Sequenzen verarbeitet; ein Backslash könnte ihn dadurch unbemerkt verändern.

Der Technologieüberblick nennt die allgemeine Leitlinie: gewöhnliche systemd-Units als Standard, Docker und Kubernetes nur für Arbeitslasten, die ihren zusätzlichen Aufwand rechtfertigen.

Wo es schiefgeht

Eine bearbeitete Unit läuft mit den alten Einstellungen weiter. systemd liest Unit-Dateien in den Speicher. Eine Änderung auf dem Datenträger bleibt deshalb wirkungslos, bis `systemctl daemon-reload` die Sicht des Managers neu aufbaut. `systemctl cat` kann eine Datei zeigen, die von der Konfiguration des laufenden Managers abweicht. Änderungen mit `systemctl edit` vornehmen, das automatisch neu lädt, oder neu laden und anschließend den Dienst neu starten.

Ein Geheimnis erreicht den Dienst mit einem fehlenden Zeichen. Werte in `Environment=` durchlaufen systemds eigene Regeln für Anführungszeichen. Dabei werden Escape-Sequenzen im C-Stil verarbeitet; ein wörtlicher Backslash muss verdoppelt werden. Die Dokumentation von systemd warnt außerdem, dass Umgebungsvariablen über D-Bus für unprivilegierte Clients sichtbar sind. Geheimnisse in einer Datei ablegen und mit `LoadCredential=` übergeben. Das stellt dem Dienst eine schreibgeschützte Kopie bereit, die nur sein Benutzer und root lesen können.

Ein Paketupgrade macht eine lokale Korrektur unbemerkt rückgängig. Änderungen an der Herstellerdatei unter `/usr/lib/systemd/system/` überschreibt das nächste Paketupdate. Änderungen in einem Drop-in unter `/etc/systemd/system/<unit>.d/` ablegen. Es wird darübergelegt und übersteht Upgrades. Befehlslisten werden ergänzt, nicht ersetzt. Ein Drop-in, das den Startbefehl ändert, leert ihn deshalb zunächst durch eine leere Zuweisung:

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

Ein nächtlicher Auftrag überspringt die Nacht, in der der Server ausgeschaltet war. Ein Kalender-Timer, der fällig war, während der Rechner ausgeschaltet war, wartet einfach auf den nächsten Termin, weil `Persistent=` standardmäßig false

ist. Für Sicherungen und andere zwingend auszuführende Aufträge `Persistent=true` setzen. Der versäumte Lauf startet dann, wenn der Timer wieder aktiv ist, gegebenenfalls mit einer konfigurierten Zufallsverzögerung.

Ein abgestürzter Dienst bleibt ausgefallen. Der Standardwert von `Restart=` ist `no` . Ein beendeter Dienst bleibt deshalb aus, bis es jemand bemerkt. Für dauerhaft laufende Dienste `Restart=on-failure` setzen. Dabei beachten: Sobald eine Unit ihr Startlimit erreicht, gibt der Manager auf. Ein manueller Start muss dann warten, bis das Intervall abgelaufen ist oder `systemctl reset-failed` ausgeführt wurde.

Weiterführende Lektüre

- systemd(1) — der Manager selbst: Unit-Typen, Bootreihenfolge und die Signale, auf die er reagiert.
- systemd.unit(5) — Unit-Dateien, Abhängigkeiten, Startlimits und die überschreibenden Drop-in-Verzeichnisse.
- systemd.service(5) — Diensttypen, Neustartrichtlinie und das Zurücksetzen von Befehlslisten.
- systemd.exec(5) — Ausführungsumgebung, Sandboxing-Optionen und Zugangsdaten.
- systemd.timer(5) — Kalender- und monotone Timer einschließlich `Persistent=` .
- systemctl(1) — die Befehlszeile zum Starten, Bearbeiten, Neuladen und Untersuchen von Units.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Bereitstellungsweg, die Betriebsrisiken und die Bereitschaftsverantwortlichen. Wir können eine Änderung mit Prüfungen, einem Wiederherstellungsweg und klaren Zuständigkeiten abgrenzen.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist systemd?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

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

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/systemd/