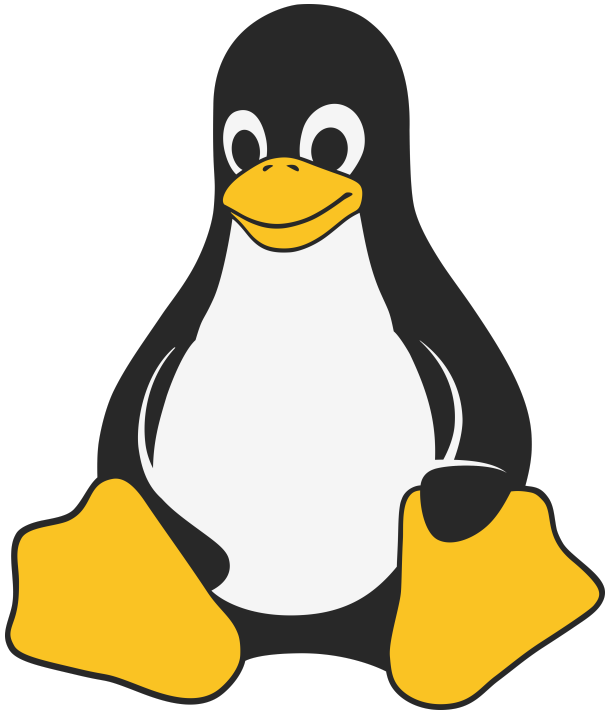
Systemd - Vorläufer, Konkurrenten, Ideengeber



Inhaltsverzeichnis

- SMF - Service Management Facility (Solaris)
- Launchd (Apple)
- Systemd
- Links

Überblick

Heute ist *Systemd* das das dominierende Init-System für Linux. Im Laufe der Unix- und Linux-Geschichte gab eine Vielzahl von Init-Systemen.

Diese Init-Systeme definieren in verschiedener Weise, wie ein Unix-/Linux-System nach dem Laden des Kernels initialisiert wird. Sie sind für die Initialisierung der Hardware, das Einhängen und Aushängen von Dateisystemen, das Starten und Beenden von Diensten beim Hoch- und Herunterfahren des Systems verantwortlich.

Zum Inhaltsverzeichnis ...

Durch die Mängel und Unzulänglichkeiten des traditionellen *SysVinit* (System V Init) entstanden viele Alternativen und Konkurrenten in verschiedenen Unix-Derivaten und Linux-Distributionen.

Dieses Dokument hat nicht den Anspruch, die verschiedenen Init-Systeme umfassend zu beschreiben. Es soll nur einen Überblick über die wichtigsten Vorläufer, Konkurrenten und Ideengeber von *Systemd* geben.

Hier die wichtigsten:

- *SysVinit* (System V Init)
- *Upstart*
- *OpenRC*
- *runit*
- *Launchd* (Apple)
- *SMF* - Service Management Facility (Solaris)
- *Systemd*

Zum Inhaltsverzeichnis ...

SysVinit (System V Init)

SysVinit ist das traditionelle Init-System für Unix-Systeme. Es wurde ursprünglich für AT&T Unix System V entwickelt.

Merkmale

- Definition sog. Runlevels
- Starten von Diensten durch Shell-Skripte, die in einer festen Reihenfolge sequentiell abgearbeitet werden
- Shell-Skripte in den Verzeichnissen `/etc/init.d` und `/etc/rc.d`

Zum Inhaltsverzeichnis ...

Runlevels der LSB (Linux Standard Base) 4.1.0

Runlevel	Name	Beschreibung
0	Off	System herunterfahren
1	Single-User mode	Administrationsmodus, Nur ein Benutzer (root) kann sich anmelden, nur root-Dateisystem gemountet
2	Multi-User mode	Keine Netzwerkschnittstellen aktiviert, keine Netzwerkdienste, nur lokale Dateisysteme gemountet

Zum Inhaltsverzeichnis ...

Runlevel	Name	Beschreibung
3	Multi-User mode with network	Netzwerkschnittstellen aktiviert, Netzwerkdienste aktiviert, Netzwerkdateisysteme gemountet, keine grafische Oberfläche
4	Not used/User-definable	Benutzerdefinierter Runlevel
5	Full mode	wie Runlevel 3, aber mit grafischer Oberfläche
6	Reboot	System neu starten

Kritikpunkte:

- Sequentielle Abarbeitung der Dienste führt zu langen Bootzeiten (keine Parallelisierung möglich)
- Shell-Skripte sind schwer wartbar und fehleranfällig
- Keine Möglichkeit, Dienste automatisch neu zu starten, wenn sie abstürzen
- Shell-Skripte starten viele Prozesse (langsam).
- Viele verschiedene Runlevel-Definitionen bei verschiedenen Unix-Derivaten und Linux-Distributionen

Zum Inhaltsverzeichnis ...

Die Probleme und Unzulänglichkeiten von SysVinit führten zur Entwicklung vieler Alternativen wie *Upstart*, *OpenRC*, *runit*, *Launchd*, *SMF* und schließlich *Systemd*.

Upstart

Upstart wurde von 2006 bis 2014 von Canonical für Ubuntu entwickelt. 2015 stellte Canonical die Entwicklung von Upstart ein und wechselte zu *Systemd*.

Merkmale

- Event-basiertes Init-System
- Parallelisierung der Dienste
- Automatisches Neustarten von Diensten möglich

OpenRC

OpenRC ist ein Init-System, das ab 2007 von Roy Marples für NetBSD und Gentoo Linux entwickelt wurde. Es wird auch heute noch in einigen Distributionen verwendet: Gentoo, Alpine Linux, Devuan etc.

Zum Inhaltsverzeichnis ...

Merkmale

- Shell-Skripte in den Verzeichnissen `/etc/init.d` und `/etc/rc.d`
- Ähnlich wie *SysVinit*, aber mit einigen Verbesserungen
- Besser strukturierte Skripte mit Funktionen *start*, *stop*, *restart*, *status*
- Dependency-Management, Dependency-Graph
- Parallelisierung der Dienste möglich

Zum Inhaltsverzeichnis ...

runit

runit wurde seit 2004 von Gerrit Pape entwickelt für NetBSD, FreeBSD, OpenBSD, Linux, macOS und Solaris.

Merkmale

- Sehr einfach gehaltenes Init-System: schlank, modular, portabel
- Keine Runlevels, keine Shell-Skripte
- Parallelisierung der Dienste
- Automatischer Neustart von Diensten möglich

Zum Inhaltsverzeichnis ...

SMF - Service Management Facility (Solaris)

SMF wurde ab 2005 von Sun Microsystems für Solaris entwickelt. Es ist ein sehr mächtiges und flexibles Init-System, das auch die Init-Skripte von SysVinit unterstützt. Zentrales *SMF*-Konzept ist der *Service*.

Merkmale

- Zentrales Konzept des *Service*
- Genau definierte Zustände eines Service: `online`, `offline`, `disabled`, `maintenance`, `degraded`, `legacy_run`, `uninitialized`
- Service-Definitionen in XML-Dateien
- konfigurierbare Dependencies (Abhängigkeiten) zwischen Services
- Automatischer und unterbrechungsfreier Neustart von Diensten möglich (z.B. bei Absturz, Upgrade oder Konfigurationsänderung)
- Konfigurierbare Ausführlichkeit der Boot-Meldungen

Zum Inhaltsverzeichnis ...

- Delegation von Aufgaben an Benutzer, die nicht root sind
- Paralleler Start von Services (abhängig von den konfigurierten Dependencies)
- Automatischer Neustart von Services nach Fehler
- Abwärtskompatibilität zu SysVinit-Init-Skripten: Diese können von SMF gestartet und überwacht werden. Automatischer Neustart nicht möglich.

Launchd (Apple)

Launchd wurde ab 2005 von Apple für macOS, später für iOS und watchOS entwickelt. Es ist ein sehr mächtiges und flexibles Init-System, das auch auf FreeBSD portiert wurde.

`launchd` ist ein Daemon, der weitere Daemons startet und überwacht. `launchctl` ist ein Kommandozeilen-Tool zur Steuerung und Verwaltung von `launchd`.

Merkmale

- `launchd` ist ein Daemon, der weitere Dienste nach Bedarf (wenn ein Request für den Dienst ansteht) startet und überwacht. So müssen weniger Dienste permanent laufen.
- Socket-Aktivierung: Dienste werden erst gestartet, wenn ein Client eine Verbindung aufbaut.
- Parallelisierung der Dienste (auch abhängiger Dienste)
- Automatischer und unterbrechungsfreier Neustart von Diensten möglich (z.B. bei Absturz, Upgrade oder Konfigurationsänderung)

Zum Inhaltsverzeichnis ...

- Konfiguration in *plist*-Dateien (Property List Files, XML-Dateien)
- Zentrale Konfigurationsdatei `/etc/launchd.conf`
- Zentrales Logging in `/var/log/system.log`

Kern-Merkmal und wichtigste Neuerung von *Launchd* ist die **Socket-Aktivierung**: Dienste werden erst gestartet, wenn ein Client eine Verbindung aufbaut. Nicht nur unabhängige Dienste, sondern auch Dienste, die voneinander abhängen, können so parallel gestartet werden. Dies führt zu schnelleren Bootzeiten. Die Abhängigkeiten zwischen den Diensten müssen nicht explizit konfiguriert werden.

Apple's *Launchd* wurde zu einem wichtigen technischen Ideengeber für *Systemd*. Die Technik der **Socket-Aktivierung** wurde von *Systemd* übernommen und wurde auch hier zu einem zentralen Konzept.

Zum Inhaltsverzeichnis ...

Systemd

Zitat aus Wikipedia: https://de.wikipedia.org/wiki/Systemd

Systemd ist eine Sammlung von Programmen, Hintergrundprogrammen (Daemons) und Bibliotheken für Linux-Betriebssysteme. Ihr zentraler Bestandteil ist der *Systemd* init-Prozess, der als erster Prozess (Prozess-ID 1) zum Starten, Überwachen und Beenden weiterer Prozesse dient. Es bietet aber auch andere Systemkomponenten an, deren Bandbreite vom Booten („*Systemd*-boot“) bis zum Logging („journald“) reichen.

Systemd wurde von Lennart Poettering, Kay Sievers (Red Hat Inc.) und anderen in C programmiert und wird als freie Software unter der GNU Lesser General Public License (LGPL) veröffentlicht.

Die Ideen und Konzepte zu *Systemd* entstanden aus der Betrachtung von bereits bestehenden modernisierten init-Systemen wie *launchd* von macOS und SMF (Service Management Facility) von Solaris. Es wurde am 10. April 2010 veröffentlicht.

Zum Inhaltsverzeichnis ...

Distributionen, die *Systemd* als vorgegebenen init-Dienst verwenden, sind Fedora ab Version 15, openSUSE ab Version 12.1, Mandriva 2011, Mageia ab Version 2, Arch Linux seit Oktober 2012, Red Hat Enterprise Linux ab Version 7, Tizen sowie siduction ab Version 2013.2, SUSE Linux Enterprise Server ab Version 12, Ubuntu ab Version 15.04 und Debian ab Version 8.

Merkmale

- Vereinheitlichung des Init-Systems für verschiedene Linux-Distributionen
- Deklarative Konfiguration in INI-Dateien (viel einfacher und wartbarer als Shell-Skripte)
- dennoch Abwärtskompatibilität zu SysVinit-Init-Skripten
- Läuft nur auf Linux-Systemen, setzt einen Linux-Kernel voraus, für andere Unix-Systeme nicht verfügbar
- Zuständig für
  - Initialisierung der Hardware

- ◦ Einhängen und Aushängen von Dateisystemen
  - ◦ Starten und Beenden von Diensten
  - ◦ Login-Management
  - ◦ Event-Logging
- Zentrales Konzept der **Units**: Service-Units, Device-Units, Mount-Units, Timer-Units, Socket-Units, Target-Units etc.
- Zentrale Technik der **Socket-Aktivierung**
- außerdem IPC (Inter-Process Communication) mit D-Bus (Desktop-Bus)

- Parallelisierung der Dienste: Nicht nur unabhängige Dienste, sondern auch Dienste, die voneinander abhängen, können parallel gestartet werden. Durch die Socket-Aktivierung können alle nahezu gleichzeitig gestartet werden.
- Sehr kurze Bootzeiten
- Automatischer und unterbrechungsfreier Neustart von Diensten möglich (z.B. bei Absturz, Upgrade oder Konfigurationsänderung)
- Zentrales Logging in `journald`

Links

- *SysVinit* bei Wikipedia (de)
- *SysVinit* bei Wikipedia (en)
- Runlevel bei Wikipedia (en)
- LSB (Linux Standard Base) bei Wikipedia (en)
- *Upstart* bei Wikipedia (en)
- *OpenRC* bei Wikipedia (en)
- *runit* bei Wikipedia (en)

Zum Inhaltsverzeichnis ...

- *Launchd* bei Wikipedia (en)
- *SMF* bei Wikipedia (en)
- *Systemd* bei Wikipedia (de)
- *Systemd* bei Wikipedia (en)
- *Systemd*-Homepage
- *Systemd*-Socket-Aktivierung (Lennart Poettering)