bee/sec

Systemd – Past, Present, and Future

Open-Source-Treffen, München
28. Oktober 2016

Nils Magnus
Lead Consultant Security Architecture
nils.magnus@beesec.de

Überblick

bee/sec

Über kaum eine Systemkomponente wurde in den letzten Jahren so leidenschaftlich gestritten wie Systemd. Das liegt wohl nicht zuletzt mit daran, dass Systemd eine äußerst zentrale Funktion im Linux-Betriebssystem implementiert – und zwar auf eine Art und Weise, die viele Anwender und Systemverwalter zunächst überraschte.

Der Vortrag stellt zunächst die Diskussion um die Thematik in einen technikhistorischen Kontext sowie betrachtet die Situation und Probleme, die ursächlich zur Entwicklung von Systemd geführt haben. Anschließend stellt er Kritik und Argumente gegenüber und protokolliert den (noch andauernden) Entscheidungsfindungsprozess. Abschließend blickt er in die Zukunft und versucht einzuordnen, welche Auswirkungen Systemd auf die Weise hat, wie wir zukünftig Systeme entwerfen und betreiben.

Nils Magnus befasst sich als Berater, Systemarchitekt und Fachutor seit über 20 Jahren mit Linux- und Unix-Systemen sowie mit Open Source. Als Vorstand für den LinuxTag und die German Unix Users Group hat viele Konferenzen geplant. Er arbeitet als Lead Consultant Security Architecture arbeitet er für bee/sec in Köln, München und Berlin.

Über bee/sec

Als auf Informationssicherheit spezialisiertes Beratungsunternehmen unterstützen wir unsere Kunden in folgenden Bereichen:

Information Security Management Systeme

Security Assessments

Security Architecture

Penetration Testing

Dabei berücksichtigen wir gleichermaßen **Technologie** und **Organisation**.

Vergangenheit (1969 - 200x)

Kontext

bee/sec

- Kernel bootet unter PID 0
- Letzte Amtshandlung des Bootens:
  Neuen Prozess mit PID 1 erzeugen
- „Vater aller Userland-Prozesse“
- Terminert ein Prozess, so erhält sein Parent-Prozess ein SIGCHILD
- Sonderfall: Ist der Parent-Prozess auch schon terminiert, so erhält stattdessen PID 1 das Signal

Vater aller Prozesse

bee/sec

- Traditionell startet der Kernel /bin/init
- Dieser Prozess startet weiteren Prozesse:
  - Login-Prozesse auf TTYs
  - Eine X-Session mit Login-Manager
  - Diverse Systemkonfigurationen
    (z. B. Netzkonfiguration, Kerneleinstellungen)
  - Diverse Server-Hintergrundprozesse
    (z. B. Mailserver, Webserver, sshd, etc.)

Drei Dekaden Userland-Start

bee/sec

Präkambrium: /etc/rc.local

- Ein Skript, in der Admin von Hand und mit einem Editor alle Konfigurationen einträgt
- Netzwerkkonfiguration, Server, alles
- Maximal automatisierungsfeindlich
- Alles sequenziell: Hängt ein Skript, hängt der Bootprozess

Mittelalter: SystemV, rc + /etc/init.d/*

- Neues Konzept adressiert Automatisierung: Runlevel
- Einzelne Skripte sind aber immer noch nicht parallelisierbar

Kritik am Init-System

bee/sec

- Nicht robust: Ein Fehler in einem Init-Skript hält den kompletten Bootvorgang auf (z. B. DNS-Server nicht erreichbar, NFS-Server reagiert nicht)
- Der Bootvorgang dauert sehr lange
- Abhängigkeiten zwischen Abläufen müssen explizit verwaltet werden (durch Dateinamen!)

Kritik an Init-Diensten

bee/sec

- Etwas schlecht automatisierbar
- Kein wirklicher Standard: Jeder Hersteller hat das selbst implementiert
- Unterschiedliche Semantik und Funktionalität (nur start/stop? Auch restart? Reload?)
- Viele Anbieter bringen gar keine eigenen Init-Skripte mit
- Viele Skripte sind nicht in der Lage, abgestürzte Dienste zu erkennen und neu zu starten

Gegenwart (200x-heute)

Alternative Ansätze

bee/sec

- Solaris Service Management Facility (SMF)
  XML – muss man noch mehr sagen?
- OpenRC: Gentoo
- Upstart: zeitweise der Favorit von Ubuntu
- Launchd, Epoch, finit: Your Milage May Vary

New Kid on the Block: Systemd

bee/sec

- Ab 2010 von Kai Sievers und Lennart Poettering entworfen
- Implementiert und betreut von einer seither wachsenden Community
- Knapp 20 Github Comitter
- rund 100 Teilnehmer bei der systemd.conf von 2015 und 2016

Adoption Systemd

bee/sec

- Effektiv heute bei allen „großen“ Linux-Distributionen umgesetzt:
  - Fedora, CentOS, RHEL, Atomic
  - OpenSUSE, LEAP, SLES
  - Debian und Ubuntu
- Hat Ende 2014 fast zum Schisma bei Debian geführt: Lange Diskussion der Community
- Devuan ist Systemd-freies Debian
- Unterschiedliches Bild bei den „selbstbauenden“ Distributionen: Arch Linux und Gentoo

Kritik an Systemd

bee/sec

- Integriert mehr Funktionalität als das vergleichbare SysV-Init
- Besonders beim Debugging des Boot-Prozesses oft trickreich
- Performance und Binärformate im Journald
- Auf Social-Level war die Projekteinführung etwas holperig

Ökosystem Systemd

bee/sec

Eine ganze Reihe von Werkzeugen ist im Umfeld von Systemd entstanden:

- Networkd: Konfiguration von Netzen
- Firewalld: Klare Schnittstelle statt Shellskripten, die Netfilter-Kommandos absetzen
- Journald: Zentrale Sammelstelle für Logfiles
- Nspawnd: Leichtgewichtige „Alternative“ für Docker, rkt, LXC oder LXD

Zukunft (2016+)

Kernbaustein in Containern

bee/sec

- Die Rolle von Services in Containern hat sich noch nicht vollständig gefunden
- Soll ein Container-Service (z. B. Docker Engine) Container starten, überwachen und ggf. neustarten?
- Oder soll das ein Daemon wie Systemd übernehmen, der unter PID 1 läuft?

Ausblick

bee/sec

- Der Austausch einer sehr zentralen Komponente ist immer von Schmerzen und Unzufriedenheit begleitet
- Gerade 3rd-Party-Anbieter müssen noch mehr und noch bessere Systemd-Units liefern
- Glücklicherweise hat sich die zeitweise emotional aufgeladene Diskussion etwas beruhigt
- Das Potenzial von Systemd ist noch lange nicht ausgereizt

Vielen Dank für‘s Interesse!

Fragen?

bee security GmbH
Siegburger Straße 215
50679 Köln / Germany

Nils Magnus
/ Lead Consultant

T +49 (0) 221 – 888 798 0
M +49 (0) 176 – 2312 4557
E nils.magnus@beesec.de

Backup & Feedback

Sammlung

bee/sec

- Spezielle Systemcalls, die Systemd braucht.
- Systemd kann auch alte Service-Skripte verarbeiten (das bringt aber nicht so viele Vorteile)
- 30 Jahre Reifung, Änderungen, Schlecht ohne Führerschein
- Services serialisieren.
- Wie sind Udev und Systemd verbunden?
- Debian benutzt meist noch die klassischen init-Skripte und wandelt die automatisch

Unit-Files

bee/sec

Aufbau der Unit-Files

- In /etc/systemd/system/* sind Links auf /usr/lib/systemd/system/*
- /run/systemd/system/*(automatisch generiert)
- /etc/systemd/xxxx.conf
-

http://0pointer.de/blog/projects/security.html

Wie geht es weiter?

bee/sec

Diskussion
rund um Systemd ab
21:30 Uhr