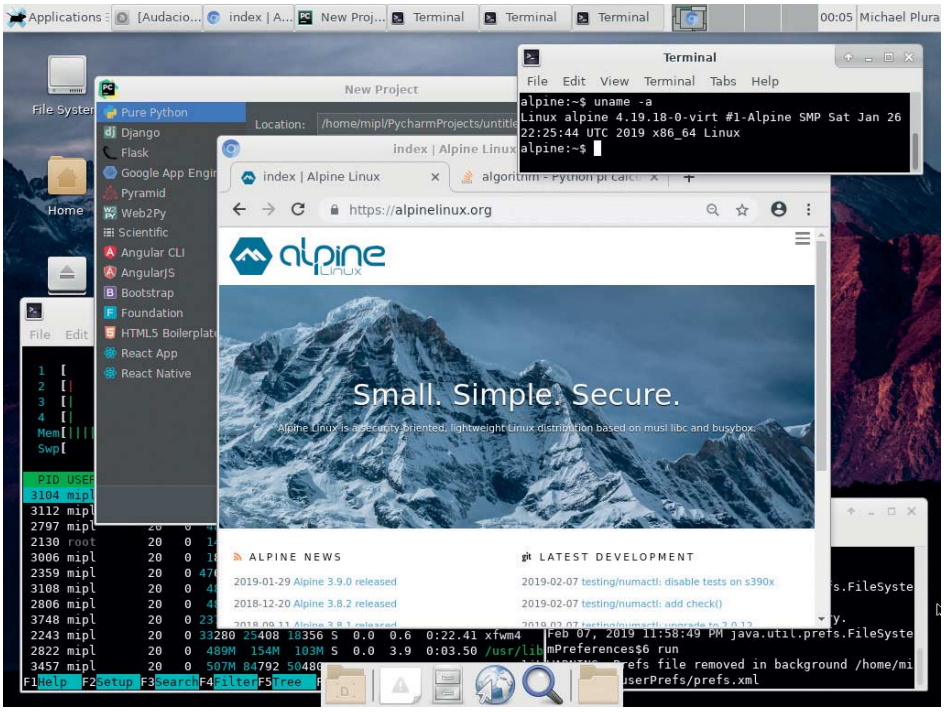

Alpine Linux 3.9.0: Zurück zu OpenSSL

Federleicht

Michael Plura

Docker-Images mit Alpine Linux werden fast doppelt so häufig heruntergeladen wie die von Ubuntu und Debian zusammen. Die auf Sicherheit optimierte, einfache und winzige Distribution kann jedoch mehr.

Das Alpine-Linux-Team rund um Natanael Copa veröffentlichte Version 3.9.0 seiner auf Sicherheit, Einfachheit und Minimalismus optimierten Linux-Distribution. Die Basis ist ein Kernel 4.19, BusyBox 1.29, musl libc 1.1.20 mit GCC 8.2.0 als C-Compiler. Neu ist die Unterstützung für AMRv7. Das SquashFS-Dateisystem zum Betrieb aus dem Speicher ist nun signiert, und es gibt Verbesserungen bei GRUB.

Zu viele Änderungen in LibreSSL veranlassten die Entwickler dazu, zurück zu OpenSSL zu wechseln. Bereits vor einem Monat entfernten sie das seit April 2017 nicht mehr freie grsecurity/PaX-Framework für einen gehärteten Linux-Kernel aus Alpine Linux 3.8.1. Trotzdem schützen Position Independent Executables (PIE) mit „stack smashing protection“ Alpine Linux.

Seinen Ursprung nahm Alpine Linux im LEAF-Projekt (Linux Embedded Appliance Framework Project), das wiederum fußt auf dem legendären „Router on a Floppy“ (Linux Router Project) aus den späten 90er-Jahren. Alpine Linux ist also eine minimalistische Linux-Distribution: Es startet von einem Read-only-Medium und läuft komplett im Arbeitsspeicher. Dabei ist es aber so klein wie möglich, um als Basis für einfache Server oder Firewall-/Router-, Proxy- oder VPN-Appliances zu fungieren.

Alpine Linux ist nicht mit den großen Distributionen vergleichbar, eignet sich mittlerweile aber nicht nur für Appliances. So nutzt Docker es als Basis für Container, wie stark, das belegen zum Zeitpunkt, als diese Zeilen entstanden, folgende Download-Zahlen auf Docker Hub:

Alpine	1 986 214 848
Ubuntu	915 265 673
Debian	177 416 107

Alpine Linux lässt sich aber auch für Server und mit Einschränkungen als Desktop-System einsetzen. Die Basisinstallation belegt 13 + 126 MByte (*/boot* und */*). Zum Vergleich: Ein nacktes Debian oder Devuan GNU/Linux benötigt knapp 400 MByte und lässt sich mit Tricks auf unter 200 MByte drücken, ein RHEL liegt hingegen bei 700 bis 900 MByte. Wie schaffen es die Alpine-Linux-Entwickler, ein so kleines, aber funktionsfähiges System einzurichten?

Minimal dank BusyBox und musl libc

BusyBox kommt in vielen eingebetteten Systemen wie Firewalls/Routern, NAS-Geräten, Settop-Boxen, Android und im initramfs von GNU/Linux zum Einsatz, wo die Ressourcen für eine herkömmliche Unix-Shell samt Toolset nicht ausreichen. Das Paket fasst die grundlegenden Funktionen (Single Unix Specification, SUS) zum Betrieb von Linux oder FreeBSD in einem einzigen Binärprogramm zusammen. Bei Alpine Linux sind das zurzeit über 80 Befehle in */bin* – fast alle sind ein Link auf */bin/busybox*.

Theoretisch lässt sich BusyBox auf über 200 Hilfsprogramme und gut 300 Befehle aufrüsten. Die aus einer *bash* unter GNU/Linux gewohnten Befehle sind größtenteils vorhanden, allerdings oft mit leicht eingeschränktem Funktionsumfang. Der englische Wikipedia-Eintrag zeigt unter „Commands“ eine Liste aller Befehle. BusyBox übernimmt ebenfalls die Aufgaben des *init*-Daemons und von *udev*, indem es selbst als Init-Prozess (PID 1) läuft und *mdev* verwendet. Das weitere Prozessmanagement übergibt es an das schlanke und moderne Init-System OpenRC – eine leichtgewichtige Alternative zum rasant wachsenden systemd. BusyBox verwendet die Almquist Shell *(ash)*, einen Clone der ursprünglichen Bourne-Shell von System V.4. Auch *ash* besitzt weniger Funktionen als Bash, beherrscht mittlerweile aber „Command Completion“ und andere Annehmlichkeiten.

Übrigens: BusyBox steht unter der GPLv2 und ist wegen des Zwangs zur Codeveröffentlichung bei Hardwareherstellern weniger beliebt, einer der zerstrittenen Entwickler bastelt am vergleichbaren ToyBox unter einer BSD-Lizenz.

Auch die mächtige GNU C Library *(glibc)* üblicher Distributionen musste anfangs der µClib weichen, wegen zu vieler Probleme ersetzte man diese aber 2014 durch die mit glibc weitgehend kompatible *musl libc*. Die steht unter der MIT-Lizenz und ist wesentlich leichtgewichtiger als die glibc und teilweise schneller. Leider läuft damit nicht jede Software.

Schlankes Init: OpenRC

Als einfaches und schlankes Init-System verwendet Alpine Linux das von Gentoo adaptierte OpenRC. Seit Version 0.25 enthält OpenRC *openrc-init* als optionalen PID1-Ersatz für das klassische */sbin/init*, bei Alpine Linux übernimmt diesen Teil jedoch BusyBox. OpenRC kommt unter anderem in Gentoo Linux, Artix und weiteren Systemen zum Einsatz; das systemd-freie Devuan GNU/Linux bietet es als Alternative zu SysV-Init an. Die Entwickler achten auf Portabilität zu anderen Systemen, und so ist es sogar unter FreeBSD, NetBSD und TrueOS verfügbar.

OpenRC startet Dienste sequenziell, kann diese aber auch parallel hochfahren und Abhängigkeiten berücksichtigen. Es trennt nicht nur Prozesse via *cgroups* sauber voneinander, sondern auch Code (init.d) und Konfigurationen (conf.d). Init-Skripte lassen sich für mehrere Komponenten bündeln (NFS: nfsd, portmap et cetera). OpenRC versucht nicht, wie systemd alles selbst zu machen und jedes Rad neu zu erfinden, sondern benutzt etablierte Werkzeuge für spezielle Aufgaben *(cron, syslog …)*. Entwicklern bietet es ausführliche Debugging-Funktionen. Es lässt sich optional – etwa zur Socket-Activation – um den Init-Supervisor S6 erweitern.

Konfiguration mit Anleihen von BSDs *rc*

Die globale Konfiguration von OpenRC erfolgt über */etc/rc.conf*, die entfernt an dieselbe Datei in den BSDs erinnert. Das Gentoo-Wiki zu OpenRC zeigt als hypothetisches Beispiel, wie man den SSH-Dienst so einrichtet, dass er nur mit Ethernet (eth0), nicht aber mit WLAN (wlan0) startet, solange das System im „Default"-Runlevel läuft. Wechselt der Runlevel zu „Office", verhält sich SSH genau umgekehrt. Runlevels können Nutzer frei definieren, sie lassen sich aus dem GRUB-Bootloader heraus starten oder im Betrieb mit *openrc <Runlevel>* wechseln.

Ein *rc-status* zeigt die aktuelle Konfiguration, *rc-update add|del <Dienst> <Runlevel>* schaltet Dienste ein und aus, *rc-update show* zeigt alle konfigurierten (Option *–v*: alle verfügbaren) Init-Skripte und deren Zuordnung zu Runlevels an. Das Alpine Wiki liefert zu OpenRC lediglich rudimentäre Informationen, Gentoo, Arch Linux sowie ein Blog-Beitrag zu Letzterem gehen weiter in die Tiefe (siehe ix.de/ix1903062).

Von Platte oder aus dem RAM

Das Projekt stellt Alpine Linux in verschiedenen Konfigurationen bereit: Neben zwei unterschiedlich bestückten ISO-Images gibt es Netboot-Dateien und ein Mini-Root-System für x86/x86_64, ppc64le, s390x und armhf/aarch64. Eine auf den Betrieb in VMs optimierte Version für x86/x86_64 und ein Xen-Dom0-Image decken virtuelle Umgebungen ab. Für ARM-SoCs mit U-Boot-Bootloader und speziell den Raspberry Pi 1/2/3 gibt es armhf/aarch64-Archive, die auf eine mit VFAT(Win95) formatierte SD-Karte auszupacken sind. Neu in Version 3.9.0 ist die Unterstützung von ARMv7.

Die Images starten Alpine Linux als Livesystem samt root-Account ohne Passwort. Per *setup-alpine* nimmt man Installation und Anpassungen vor. Das Tool fragt die Basisinformationen wie Tastaturlayout, Rechnername, Netzwerk, Root-Passwort, Zeitzone und so weiter interaktiv ab. Die Eingabe eines Fragezeichens liefert in vielen Fällen eine kurze Hilfe. Bei einer Installation auf eine (virtuelle) Festplatte muss man „sys" als Nutzungsart angeben. Ein *reboot* schließt die Installation ab.

Auch die Paketverwaltung haben die Entwickler abgespeckt, sie ist extrem schnell. Die digital signierten Archive im *.apk*-Format (nicht zu verwechseln mit Android Application Package) bilden, liegen sie in einem Verzeichnis zusammen mit einer *APKINDEX.tar.gz*, ein Repository. Der Aufruf *apk index* generiert diesen Index. Die zu nutzenden Repositories definiert */etc/apk/repositories*. Für Appliances reicht der Umfang des „main"-Zweigs (5688 Pakete) aus, wer ein Desktop-System oder umfangreichere Dienste benötigt, sollte „community" (über 4000 Pakete zusätzlich) mit einbeziehen.

Ein *apk update* aktualisiert die Liste der verfügbaren Pakete, *apk upgrade* bringt das System auf den neuesten Stand. *apk add|del|fix|info|search* und weitere Optionen verwalten die Pakete. Erstaunlich ist die Geschwindigkeit: Ein *apk add mg* installiert einen MicroEmacs in einer Sekunde. Eine Installation im „sys"-Modus verhält sich wie jede andere Linux-Installation. Läuft Alpine Linux komplett im Speicher, muss man Änderungen im */etc*-Verzeichnis per *lbu commit -d* persistent auf den Datenträger schreiben, von dem aus es gestartet wurde.

Fazit

Alpine Linux ist ein einfaches und funktionelles Linux ohne den großen Ballast etablierter Distributionen. Es eignet sich nicht nur für Appliances, Container oder IoT-Geräte, sondern macht auch auf dem Server oder einer spartanischen Workstation regelrecht Spaß. (avr@ix.de)

Michael Plura

arbeitet in Schweden als freier Autor mit den Schwerpunkten IT-Sicherheit, Virtualisierung und freie Betriebssysteme.

Quellen

[1] Michael Plura; Betriebssysteme; Startbereit; Alternative Init-Systeme für Linux; *iX* 06/2017, S. 108

[2] Detaillierte Informationen zu OpenRC und BusyBox: ix.de/ix1903062

Alpine Linux per PXE

Alpine Linux startet auch per PXE von einem TFTP-Server. Es lässt sich so zur Installation oder als komplett im RAM laufendes Rettungs-/Livesystem nutzen. Dazu kopiert man aus *alpine-netboot-3.9.0-x86_64.tar.gz* die beiden Dateien *vmlinuz-vanilla* und *initramfs-vanilla* in ein Verzeichnis auf dem TFTP-Server, zum Beispiel nach */srv/tftp/alpine/*, und erweitert dessen *pxelinux.cfg/default* um den im Listing gezeigten Eintrag. Für virtuelle Maschinen kann man *„...-vanilla"* durch *„...-virt"* ersetzen, um Speicher für nicht benötigte HW-Treiber zu sparen. Schneller startet Alpine Linux, wenn es sich die Repositories der *APPEND*-Zeile nicht vom Alpine-CDN, sondern von einem Spiegel im LAN holen kann.

Listing: Ergänzung zu *pxelinux.cfg/default*

```
LABEL Alpine
  MENU LABEL Alpine Linux
  KERNEL alpine/vmlinuz-vanilla
  INITRD alpine/initramfs-vanilla
  APPEND ip=dhcp alpine_repo=http://dl-cdn.alpinelinux.org/alpine/v3.9/main/ modloop= ↵
         http://dl-cdn.alpinelinux.org/alpine/v3.9/releases/x86_64/netboot/modloop-vanilla
```