FreeBSD Anleitung für Linux®-Benutzer

John Ferrell

Version: 679e5dc7fa

2016-09-30 17:21:23 +0000 von Bjoern Heidotting.

Zusammenfassung

Dieses Dokument soll Linux®-Benutzer mit den Grundlagen von FreeBSD vertraut machen.

Inhaltsverzeichnis

1. Übersicht

Dieses Dokument beschreibt einige der technischen Unterschiede zwischen FreeBSD und Linux®, damit sich Linux®-Anwender schnell mit den Grundlagen von FreeBSD vertraut machen können.

Dieses Dokument geht davon aus, dass FreeBSD bereits installiert ist. Lesen Sie das Kapitel Installation von FreeBSD des FreeBSD-Handbuch für die Hilfe bei der Installation.

2. Standard-Shell

Linux®-Benutzer sind oft überrascht, dass Bash nicht die Standard-Shell in FreeBSD ist. Genau genommen ist Bash nicht einmal in der Standardinstallation enthalten. FreeBSD benutzt stattdessen tcsh(1) als Standard-Shell für root, sowie die Bourne shell-kompatible sh(1) als Standardshell für Benutzer. sh(1) ist der Bash sehr ähnlich, besitzt jedoch einen kleineren Funktionsumfang. In der Regel werden Skripte für die sh(1) auch mit der Bash laufen. Der umgekehrte Fall trifft jedoch meistens nicht zu.

Bash und weitere Shells können unter FreeBSD mit Paketen und der Ports-Sammlung installiert werden.

Nachdem Sie eine andere Shell installiert haben, benutzen Sie chsh(1) um die Standard-Shell für einen Benutzer zu ändern. Es wird empfohlen, die Standard-Shell des Benutzers `root` unverändert bleibt, da Shells, welche nicht im Basissystem enthalten sind, in `/usr/local/bin` installiert werden. Im Falle eines Problems ist vielleicht das Dateisystem, auf dem sich `/usr/local/bin` befindet, nicht eingehängt ist. In einem solchen Fall hätte der Benutzer `root` keinen Zugriff auf die Standard-Shell, was ihn daran hindern würde, sich am System anzumelden und das Problem zu beheben.

3. Pakete und Ports: Installation von Software in FreeBSD

FreeBSD bietet zwei Methoden zur Installation von Anwendungen: Binärpakete und kompilierte Ports. Jede Methode hat ihre eigenen Vorteile:

- Schnellere Installation, insbesondere bei größeren Anwendungen.
- Es wird kein Verständnis darüber benötigt, wie Software kompiliert wird.
- Es muss kein Compiler installiert werden.
- Bieten die Möglichkeit, Installationsoptionen anzupassen.
- Eigene Korrekturen können angewendet werden.

Wenn für die Installation der Anwendung keine Änderungen nötig sind, kann auch das Paket installiert werden. Kompilieren Sie den Port, wenn die Anwendung eine Änderung an den voreingestellten Optionen erfordert. In diesem Fall kann ein angepasstes Paket mit `make Paket` erstellt werden.

Eine vollständige Liste aller Ports und Pakete finden Sie hier.

3.1. Pakete

Pakete sind vorkompilierte Anwendungen, sozusagen FreeBSD-Äquivalente von `.deb`-Dateien unter Debian/Ubuntu basierten Systemen und `.rpm`-Dateien von Red Hat/Fedora basierten Systemen. Pakete werden mit `pkg` installiert. Das folgende Kommando installiert beispielsweise Apache 2.4:

```
# pkg install apache24
```

Weitere Informationen zu Paketen finden Sie im Abschnitt 4.4 des FreeBSD Handbuchs: Benutzen von pkg zur Verwaltung von Binärpaketen.

3.2. Ports

Die FreeBSD Ports-Sammlung ist ein Gerüst aus `Makefiles` und Korrekturen, um Anwendungen aus dem Quellcode unter FreeBSD zu installieren. Wenn Sie einen Port installieren, wird das System den Quellcode herunterladen, die benötigten Korrekturen anwenden, den Quellcode kompilieren und die Anwendung und die erforderlichen Abhängigkeiten installieren.

Die Ports-Sammlung, oder einfach Ports genannt, kann mit portsnap(8) nach `/usr/ports` installiert werden.

Um einen Port zu kompilieren, wechseln Sie in das Verzeichnis des Ports und starten Sie den Bau-Prozess. Das folgende Beispiel installiert Apache 2.4 aus der Ports-Sammlung:

```
# cd /usr/ports/www/apache24
# make install clean
```

Ein Vorteil von Ports bei der Installation von Software ist die Möglichkeit, die Installationsoptionen anzupassen. In diesem Beispiel wird spezifiziert, dass zusätzlich das Modul mod_ldap installiert werden soll:

```
# cd /usr/ports/www/apache24
# make WITH_LDAP="YES" install clean
```

Lesen Sie Benutzen der Ports-Sammlung für weitere Informationen.

4. Systemstart

Viele Linux®-Distributionen verwenden das SysV init System, während FreeBSD das traditionelle BSD-init(8) benutzt. Unter BSD-init(8) gibt es keine Runlevel und `/etc/inittab` existiert auch nicht. Stattdessen wird der Systemstart von rc(8) Skripten gesteuert. Beim Systemstart liest `/etc/rc` `/etc/rc.conf` und `/etc/rc.conf.local` um herauszufinden welche Dienste gestartet werden müssen. Die jeweiligen Dienste werden dann gestartet, indem die entsprechenden Skripten in `/etc/rc.d/` und `/usr/local/etc/rc.d/` ausgeführt werden. Diese Skripte sind ähnlich wie die Skripte in `/etc/init.d/` unter Linux®-Systemen.

Die Skripte in `/etc/rc.d/` sind für Anwendungen aus dem „Basissystem“, wie beispielsweise cron(8), sshd(8), und syslog(3). Die Skripte in `/usr/local/etc/rc.d/` gehören zu den vom Benutzer installierten Anwendungen, wie zum Beispiel Apache und Squid.

Da FreeBSD als komplettes Betriebssystem entwickelt wird, werden die vom Benutzer installierten Anwendungen nicht als Teil des „Basissystems“ angesehen. Diese Anwendungen werden in der Regel als Pakete oder Ports installiert. Um die Anwendungen vom Basissystem zu separieren, werden diese unterhalb von `/usr/local/` installiert. Die Binärdateien der installierten Anwendungen werden in `/usr/local/bin/` gespeichert, die Konfigurationsdateien in `/usr/local/etc/` , und so weiter.

Dienste werden über einen Eintrag in `/etc/rc.conf` aktiviert. Die Standardeinstellungen des Systems stehen in `/etc/defaults/rc.conf` und werden von den Einstellungen in `/etc/rc.conf` überschrieben. Lesen Sie rc.conf(5) für weitere Informationen über die verfügbaren Einträge. Wenn Sie zusätzliche Anwendungen installieren, lesen Sie die Nachrichten um zu erfahren, wie Sie alle dazugehörigen Dienste aktivieren.

Die folgenden Einträge in `/etc/rc.conf` aktivieren sshd(8) sowie Apache 2.4, wobei Apache mit SSL-Unterstützung gestartet werden soll.

```
# enable SSHD
sshd_enable="YES"
# enable Apache with SSL
apache24_enable="YES"
apache24_flags="-DSSL"
```

Sobald ein Dienst in `/etc/rc.conf` aktiviert ist, kann er ohne einen Neustart des Systems gestartet werden:

```
# service sshd start
# service apache24 start
```

Wenn ein Dienst nicht aktiviert wurde, kann er auf der Kommandozeile mit `onestart` gestartet werden:

```
# service sshd onestart
```

5. Netzwerkkonfiguration

Anstelle einer allgemeinen *ethX*-Kennzeichnung, die von Linux® benutzt wird, um Netzwerkschnittstellen zu identifizieren, verwendet FreeBSD den Treibernamen gefolgt von einer Nummer. Die folgende Ausgabe von ifconfig(8) zeigt zwei Intel®Pro 1000 Netzwerkschnittstellen (`em0` und `em1`):

```
% ifconfig
```

```
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 10.10.10.100 netmask 0xffffff00 broadcast 10.10.10.255
        ether 00:50:56:a7:70:b2
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 192.168.10.222 netmask 0xffffff00 broadcast 192.168.10.255
        ether 00:50:56:a7:03:2b
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
```

Mit ifconfig(8) kann einer Schnittstelle eine IP-Adresse zugeordnet werden. Damit diese nach einem Neustart erhalten bleibt, muss die IP-Konfiguration in `/etc/rc.conf` eingetragen werden. Die folgenden Einträge in `/etc/rc.conf` konfigurieren den Rechnernamen, die IP-Adresse und das Standard-Gateway:

```
hostname="server1.example.com"
ifconfig_em0="inet 10.10.10.100 netmask 255.255.255.0"
defaultrouter="10.10.10.1"
```

Benutzen Sie die folgenden Einträge um die Schnittstelle über DHCP zu konfigurieren:

```
hostname="server1.example.com"
ifconfig_em0="DHCP"
```

6. Firewall

FreeBSD verwendet nicht Linux® IPTABLES als Firewall. Stattdessen hat der Benutzer unter FreeBSD die Wahl zwischen drei Firewalls, die auf Kernel-Ebene arbeiten:

- PF
- IPFILTER
- IPFW

PF wurde vom OpenBSD Projekt entwickelt und nach FreeBSD portiert. PF wurde als Ersatz für IPFILTER entwickelt und die Syntax ist der von IPFILTER sehr ähnlich. PF kann zusammen mit altq(4) werden um QoS-Funktionen bereitzustellen.

Dieser beispielhafte PF-Eintrag erlaubt eingehende SSH-Verbindungen:

```
pass in on $ext_if inet proto tcp from any to ($ext_if) port 22
```

IPFILTER ist eine von Darren Reed entwickelte Firewall. Diese Firewall ist nicht FreeBSD-spezifisch und wurde bereits auf andere Betriebssysteme portiert, einschließlich NetBSD, OpenBSD, SunOS, HP/UX, und Solaris.

Die Syntax für IPFILTER zum Erlauben von eingehenden SSH-Verbindungen lautet:

```
pass in on $ext_if proto tcp from any to any port = 22
```

Die Firewall IPFW wird von FreeBSD entwickelt und betreut. Sie kann zusammen mit dummynet(4) eingesetzt werden, um Traffic-Shaping zu realisieren und verschiedene Arten von Netzwerkverbindungen zu simulieren.

Die Syntax für IPFW zum Erlauben von eingehenden SSH-Verbindungen ist:

```
ipfw add allow tcp from any to me 22 in via $ext_if
```

7. FreeBSD aktualisieren

Es gibt zwei Methoden um ein FreeBSD-System zu aktualisieren: aus den Quellen oder über binäre Updates.

Die Aktualisierung aus den Quellen ist etwas komplexer, bietet aber das höchste Maß an Flexibilität. Dieser Prozess beinhaltet die Synchronisation der Quellen über einen Subversion-Server von FreeBSD. Sobald die lokale Kopie aktualisiert wurde, kann eine neue Version des Kernels und des Userlands kompiliert werden.

Binäre Updates funktionieren in etwa so, als wenn Sie ein Linux®-System mit `yum` oder `apt-get` aktualisieren. In FreeBSD können Sie freebsd-update(8) benutzen, um binäre Updates herunterzuladen und zu installieren. Diese Updates können mit cron(8) zu festgelegten Zeitpunkten durchgeführt werden.

Anmerkung

Wenn Sie cron(8) verwenden um Updates zu planen, benutzen Sie `freebsd-update cron` in der crontab(1), um die Möglichkeit zu verringern, dass alle Maschinen die Updates zur gleichen Zeit laden:

```
0 3 * * * root /usr/sbin/freebsd-update cron
```

Weitere Informationen über die Aktualisierung aus den Quellen und Binär-Updates finden Sie im Kapitel FreeBSD aktualisieren des FreeBSD Handbuchs.

8. procfs: Verschwunden, aber nicht vergessen

In einigen Linux®-Distributionen kann man in `/proc/sys/net/ipv4/ip_forward` feststellen, ob IP-Weiterleitung aktiviert ist. In FreeBSD wird stattdessen sysctl(8) verwendet, um diese und andere Systemeinstellungen anzuzeigen.

Auf einem FreeBSD-System kann der folgende Befehl benutzt werden, um festzustellen ob IP-Weiterleitung aktiviert ist:

```
% sysctl net.inet.ip.forwarding
net.inet.ip.forwarding: 0
```

Benutzen Sie `-a` um alle Einstellungen des Systems anzuzeigen:

```
% sysctl -a | more
```

Wenn eine Anwendung procfs benötigt, fügen Sie den folgenden Eintrag in `/etc/fstab` ein:

```
proc                /proc           procfs  rw,noauto       0       0
```

Die Option `noauto` verhindert, dass `/proc` beim Booten automatisch eingehängt wird.

Das Dateisystem kann ohne Neustart eingehängt werden:

```
# mount /proc
```

9. Häufig verwendete Kommandos

Einige Kommando-Äquivalente sind wie folgt:

Linux®-Kommando (Red Hat/Debian)	FreeBSD Äquivalent	Aufgabe
`yum install` *Paket* / `apt-get install` *Paket*	`pkg install` *Paket*	Paket aus einem entfernten Repository installieren.
`rpm -ivh` *Paket* / `dpkg -i` *Paket*	`pkg add` *Paket*	Ein lokales Paket installieren

Linux®-Kommando (Red Hat/Debian)	FreeBSD Äquivalent	Aufgabe
`rpm -qa` / `dpkg -l`	`pkg info`	Installierte Paket anzeigen
`lspci`	`pciconf`	PCI-Geräte anzeigen
`lsmod`	`kldstat`	Geladene Kernelmodule anzeigen
`modprobe`	`kldload` / `kldunload`	Kernelmodule laden/entladen
`strace`	`truss`	Systemaufrufe verfolgen

10. Fazit

Dieses Dokument hat einen Überblick über FreeBSD geboten. Lesen Sie das FreeBSD Handbuch für eine tiefergehender Abdeckung dieses und weiterer Themen, welche nicht in diesem Dokument behandelt sind.