4 Vorgänge automatisch starten

Systemverwalter, die Linux-Server einrichten und verwalten wollen, sollten sich mit

- Betriebsarten,
- Zeitsteuerung von Prozessen und
- dem hintergründigen Wirken von Dämonen gründlich vertraut machen.

Administratoren, die bisher nur mit proprietären Servern von Novell und Microsoft gearbeitet haben, sollten sich spätestens hier mit diesen grundlegenden Linux-Konzepten vertraut machen; Unix- und Linux-Profis können getrost weiterblättern.

- Abschnitt 4.1 (Run-Level) beschreibt Betriebsarten zum Starten und Stoppen des Systems, für Verwaltungsarbeiten und für Mehrbenutzerbetrieb mit und ohne Netz oder Dienste.
- Zeitgesteuerte Einzelaufträge mit dem `at`-Befehl finden Sie im Abschnitt 4.2.
- Regelmäßige Vorgänge mit `Cron` (Abschnitt 4.3) nehmen Systemverwaltern viele Routinearbeiten ab.
- Der Superdämon `Inetd` bzw. *Xinetd* (Abschnitt 4.4) kann im Hintergrund viele Kommunikationsdienste an straffen Zügeln lenken.

4.1 Die Run-Level von SuSE-Linux

Das Mehrbenutzer-Betriebssystem Linux kennt verschiedene Betriebszustände (*Run-Level*) für normales Arbeiten, Wartung und Neustart.

Ein normaler Boot-Vorgang bringt das Linux-System in den Run-Level 5, bei dem die grafische Oberfläche mit voller Netzwerkunterstützung aktiviert ist und mehrere Benutzer gleichzeitig mit dem System arbeiten können.

Hinweis: In den aktuellen Versionen hat SuSE die Benennung der Run-Level stark verändert. Falls Sie Erfahrungen mit älteren Linux-Versionen haben, sollten Sie sich in `/sbin/init.d.README` mit den Änderungen vertraut machen.

Das Wechseln der Run-Level stoppt und startet Programme. So enthalten u. a. viele Konfigurationsbeschreibungen die Anweisung, die Netzwerkprogramme mit

```
init 1
init 5
```

neu zu starten. Dies wechselt zweimal den System-Zustand (Run-Level).

SuSE-Linux 9.2 kennt die folgenden Run-Level:

Run-Level	*Bedeutung*
0	Halt
S	Single User Mode
1	Single User Mode ohne Netzwerk
2	Multi User ohne Netzwerk
3	Multi User mit Netzwerk
4	Unbenutzt
5	Multi User mit Netzwerk und grafischer Anmeldung
6	Reboot

Tabelle 4.1: Die Run-Level von SuSE-Linux

Bei anderen Distributionen können die Nummern leicht abweichen.

Mit dem Befehl

- `init 0` hält man das System an, ebenso wie mit dem Befehl `halt`.
- `init S` wechseln Sie in den Single User Mode, bei dem Ihnen nur eine einzige Konsole zur Verfügung steht. Sie müssen sich nach dem Wechsel neu anmelden.
- `init 1` wechselt man in den Modus *Single User ohne Netzwerk*. Dies stoppt u. a. alle Programme, die mit dem Netzwerk zusammenhängen. Sie müssen sich nach dem Wechsel erneut anmelden.
- `init 2` wechselt das System wieder in den Modus *Multi User ohne Netzwerk* und stoppt alle Netzwerkprogramme.
- `init 3` aktiviert man den *Multi User Modus mit Netzwerk* und startet dabei alle Netzwerkprogramme neu.
- `init 5` aktiviert man den Modus Multiuser mit Netzwerk und grafischer Anmeldung und startet sowohl das Netzwerk als auch die grafische Oberfläche neu.
- `init 6` startet man das System neu, bewirkt also ein `reboot`.

Programme, die auf einen Wechsel des Run-Levels reagieren sollen, müssen im Ordner `/etc/init.d` ein Programm-Skript besitzen, das auf die Kommandoparameter `start` bzw. `stop` reagieren kann.

Für den im zweiten Kapitel nachinstallierten DHCPD heißt das Programm-Skript `dhcpd`.

Mit

```
/etc/init.d/dhcpd start
```

ruft man den DHCPD auf und mit

```
/etc/init.d/dhcpd stop
```

beendet man ihn wieder.

In früheren Versionen von SuSE-Linux lagen diese Programm-Skripte im Verzeichnis `/sbin/init.d/`. Wer bisher mit älteren Versionen gearbeitet hat, muss sich hier umstellen. Hilfreich hierbei können die symbolischen Links sein, die SuSE jeweils im Verzeichnis `/usr/sbin` ablegt und bei denen jeweils die Buchstaben `rc` dem Namen vorangestellt sind. Für den DHCPD finden Sie also in `/usr/sbin` den Link `rcdhcpd`. Da das Verzeichnis `/usr/sbin` im Suchpfad aller Benutzer liegt, können Sie den DHCPD auch ohne Pfadangabe mit

```
rcdhcpd start
```

aufrufen und mit

```
rcdhcpd stop
```

wieder beenden.

Statt das Programm mit den Aufrufen von der Konsole aus zu starten und zu stoppen, können Sie die grafische Oberfläche benutzen.

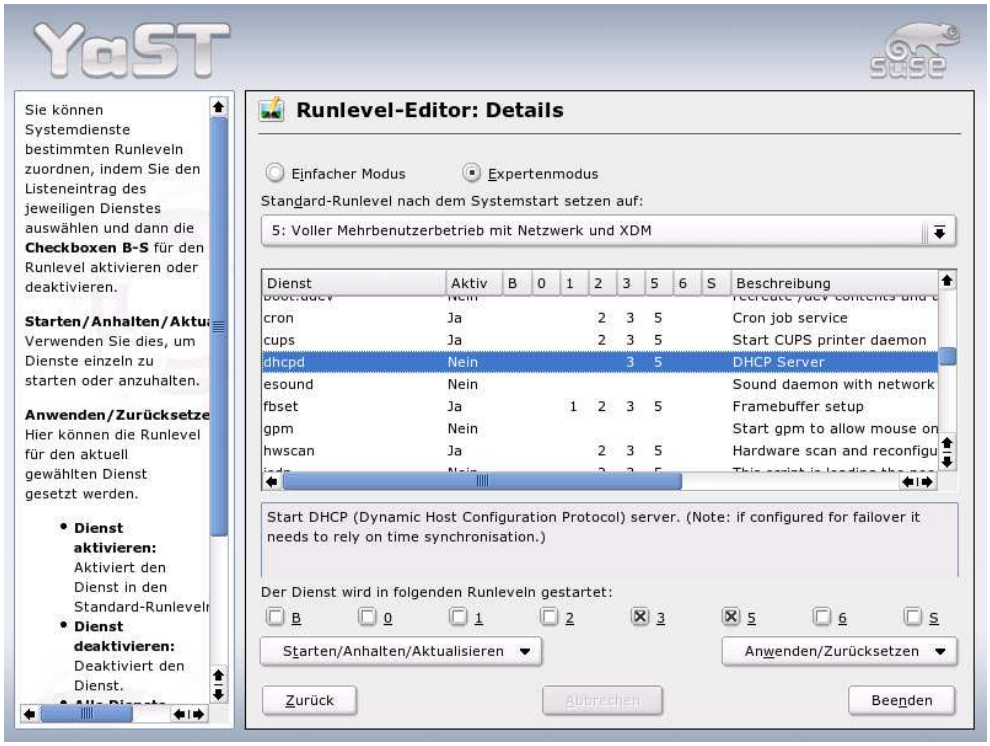

Abbildung 4.1: Run-Level-Editor

Sie finden im YaST-Kontrollzentrum unter *System* • *Runlevel-Editor* • *Experten-Modus* eine Auswahl aller Dienste, für die ein Start-Skript im Verzeichnis `/etc/init.d` vorliegt.

Wenn Sie dort den Leuchtbalken auf die Zeile für den dhcpd führen und auf *Starten/Anhalten/Aktualisieren* • *Starten* klicken, führt YaST im Hintergrund den Befehl *dhcpd start* aus. Wenn Sie hier mit YaST arbeiten, sollten Sie für den dhcpd die Run-Level 3 und 5 auswählen.

Das dreiteilige Start-Skript `dhcpd` ist einigermaßen lesbar. Der erste Teil beschreibt, in welchen Run-Leveln der Dämon laufen soll und welche anderen Dienste vorher gestartet sein müssen. YaST und andere Konfigurationstools werten diesen Teil aus, auch wenn er durch #-Zeichen auskommentiert ist.

Der nächste Abschnitt nennt die Pfade zu Programm und Konfigurationsdateien und das Skript, welches das Vorhandensein einiger Dateien prüft.

Erst im dritten Teil folgen die Programmzeilen für die einzelnen Parameter, mit denen Sie das Skript aufrufen können.

Wer sich für die Firmengeschichte der Firma SuSE interessiert, dürfte sich auch für die Copyright-Zeilen am Anfang der Datei interessieren. In späteren Versionen wird hier vielleicht auch ein Hinweis auf Novell auftauchen.

`/etc/init.d/dhcp` (Dateianfang):

```
#! /bin/sh
# Copyright (c) 1996, 1997, 1998 S.u.S.E. GmbH
# Copyright (c) 1998, 1999, 2000, 2001 SuSE GmbH
# Copyright (c) 2002, 2003 SuSE Linux AG
#
# Author: Rolf Haberrecker <rolf@suse.de>, 1997, 1998, 1999
#         Peter Poeml <poeml@suse.de>, 2000, 2001, 2002, 2003
#
# /etc/init.d/dhcpd
#   and its symbolic link
# /usr/sbin/rcdhcpd
#
### BEGIN INIT INFO
# Provides:                     dhcpd
# Required-Start:               $local_fs $remote_fs $network
# X-UnitedLinux-Should-Start:   $named $syslog $time
# Required-Stop:                $local_fs $remote_fs $network
# X-UnitedLinux-Should-Stop:    $named $syslog
# Default-Start:                3 5
# Default-Stop:                 0 1 2 6
# Short-Description:            DHCP Server
# Description:                  Start DHCP (Dynamic Host
#                               Configuration Protocol)
```

```
#                                   server. (Note: if configured
#                                   for failover it
#                                   needs to rely on time
#                                   synchronisation.)
### END INIT INFO


if [ -s /etc/sysconfig/dhcpd ]; then

        . /etc/sysconfig/dhcpd

else
        # pre 8.0

        # Source SuSE config
        . /etc/rc.config

        test -s /etc/rc.config.d/dhcpd.rc.config && \
              . /etc/rc.config.d/dhcpd.rc.config

        # Determine the base and follow a runlevel link name.
        base=${0##*/}
        link=${base#*[SK][0-9][0-9]}

        # Force execution if not called by a runlevel
        # directory.
        test $link = $base && START_DHCPD=yes
        test "$START_DHCPD" = yes || exit 0

fi
```

Das Skript kennt etliche unterschiedliche Parameter. Neben `start` und `stop` kennt es u. a. auch `reload`, `restart` und `status`. Das Kommando `restart` stoppt den dhcpd und startet ihn nach 3 Sekunden wieder, *reload* liest die Konfiguration erneut und mit `status` testet das Programm, ob der *dhcpd* läuft.

Im ersten Teil wertet das Programm aus, ob es direkt von der Konsole aus gestartet wurde oder über einen Wechsel der Run-Level. Bei einem Start über Run-Level beachtet es die Variable `START_DHCPD` aus der Konfigurationsdatei `/etc/sysconfig/dhcpd`. Nur wenn diese den Wert *yes* hat, startet das Programm.

Jetzt fehlt noch die Kopplung an den Wechsel der Run-Levels. Dazu gibt es unterhalb von `/etc/init.d` für jeden Run-Level ein Verzeichnis, also

- `/etc/init.d/rc0.d`,
- `/etc/init.d/rc1.d`,
- `/etc/init.d/rc2.d`,

- /etc/init.d/rc3.d,
- /etc/init.d/rc4.d,
- /etc/init.d/rc5.d,
- /etc/init.d/rc6.d,
- /etc/init.d/rcS.d.

In diesen Ordnern befinden sich ggf. Verweise (*Softlinks*) auf die Start-/Stopp-Dateien im Ordner /etc/init.d, Für den DHCPD sind dies die Links

- S14dhcpd und
- K08dhcpd.

Der Buchstabe S steht hier für *Start,* der Buchstabe K für *Kill* (Beenden). Beim Wechsel in den Run-Level 5 ruft das Linux-System alle Links im Verzeichnis rc5.d, die mit einem *S* beginnen, mit dem Parameter start auf. Die Zahl gibt eine Reihenfolge an; je höher die Zahl, desto später startet das zugehörige Programm.

Beim Verlassen eines Run-Levels wertet Linux die Links aus, die mit einem *K* beginnen. Das zugehörige Programm-Skript startet dann mit dem Parameter stop.

Die Distribution der SuSE-CD/DVD installiert die Start-Skripten und Links der Programme automatisch. Bei Programmen, die vor ihrem Start noch konfiguriert werden müssen, stehen in der Konfigurationsdatei in /etc/sysconfig/ die entsprechenden Startschalter (z. B. DHCP_START) noch auf no.

Im Kapitel 12, *Firewalling und Masquerading*, finden Sie ein eigenes Programm-Skript /etc/init.d/maske. Wenn dieses im Run-Level 3 aktiv sein soll, müssen Sie in /etc/init.d/rc3.d folgende Links anlegen:

```
ln -s  /etc/init.d/maske  /etc/init.d/rc3.d/S40maske
ln -s  /etc/init.d/maske  /etc/init.d/rc3.d/K40maske
```

Damit startet das Programm beim Wechsel in den Run-Level 3 und stoppt beim Verlassen des Run-Levels 3.

Ein Muster für eigene Startprogramme finden Sie in der Datei /etc/init.d/skeleton.

/etc/init.d/skeleton (Auszug, Dateianfang):

```
#! /bin/sh
# Copyright (c) 1995-2004 SUSE Linux AG, Nuernberg, Germany.
# All rights reserved.
#
# Author: Kurt Garloff
# Please send feedback to http://www.suse.de/feedback/
#
# /etc/init.d/FOO
#   and its symbolic link
```

```
# /(usr/)sbin/rcFOO
#
#    This program is free software; you can redistribute it
#    and/or modify it under the terms of the GNU General
#    Public License as published by the Free Software
#    Foundation; either version 2 of the License, or
#    (at your option) any later version.
#
#    This program is distributed in the hope that it will be
#    useful, but WITHOUT ANY WARRANTY; without even the
#    implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
#    See the  GNU General Public License for more details.
#
#    You should have received a copy of the GNU General Public
#    License along with this program; if not, write to the
#    Free Software Foundation,
#    Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
#
# Template system startup script for some example
# service/daemon FOO
#
# LSB compatible service control script; see
# http://www.linuxbase.org/spec/
#
# Note: This template uses functions rc_XXX defined in
# /etc/rc.status on
# UnitedLinux (UL) based Linux distributions. If you want to
# base your script on this template and ensure that it works
# on non UL based LSB compliant Linux distributions, you
# either have to provide the rc.status
# functions from UL or change the script to work without them.
#
### BEGIN INIT INFO
# Provides:          FOO
# Required-Start:    $syslog $remote_fs
# Should-Start: $time ypbind sendmail
# Required-Stop:     $syslog $remote_fs
# Should-Stop: $time ypbind sendmail
# Default-Start:     3 5
# Default-Stop:      0 1 2 6
# Short-Description: FOO XYZ daemon providing ZYX
# Description:       Start FOO to allow XY and provide YZ
#       continued on second line by '#<TAB>'
#       should contain enough info for the runlevel editor
#       to give admin some idea what this service does and
#       what it's needed for ...
#       (The Short-Description should already be a good hint.)
### END INIT INFO
#
```

```
# Any extensions to the keywords given above should be
# preceeded by X-VendorTag- (X-UnitedLinux- X-SuSE- for us)
# according to LSB.
#
# Notes on Required-Start/Should-Start:
# * There are two different issues that are solved by
#    Required-Start
#    and Should-Start
# (a) Hard dependencies: This is used by the runlevel editor
#     to determine which services absolutely need to be
#     started to make the start of
#     this service make sense. Example: nfsserver should have
#     Required-Start: $portmap
#     Also, required services are started before the dependent
#     ones. The runlevel editor will warn about such missing
#     hard dependencies
#     and suggest enabling. During system startup, you may
#     expect an error,
#     if the dependency is not fulfilled.
# (b) Specifying the init script ordering, not real (hard)
#     dependencies. This is needed by insserv to determine
#     which service should be started first (and at a later
#     stage what services can be started
#     in parallel). The tag Should-Start: is used for this.
#     It tells, that if a service is available, it should be
#     started before. If not, never mind.
# * When specifying hard dependencies or ordering
#   requirements, you can
#   use names of services (contents of their Provides:
#   section) or pseudo names starting with a $.
#   The following ones are available
#   according to LSB (1.1):
#       $local_fs         all local file systems are mounted
#                         (most services should need this!)
#       $remote_fs        all remote file systems are mounted
#                         (note that /usr may be remote, so
#                         many services should Require this!)
#       $syslog           system logging facility up
#       $network          low level networking (eth card, ...)
#       $named            hostname resolution available
#       $netdaemons       all network daemons are running
#   The $netdaemons pseudo service has been removed in LSB 1.2.
#   For now, we still offer it for backward compatibility.
#   These are new (LSB 1.2):
#       $time           the system time has been set correctly
#       $portmap        SunRPC portmapping service available
#   UnitedLinux extensions:
#       $ALL            indicates that a script should be insert
ed
```

```
#                       at the end
# * The services specified in the stop tags
#   (Required-Stop/Should-Stop)
#   specify which services need to be still running when this
#   service is shut down.
#   Often the entries there are just copies or a subset
#   from the respective start tag.
# * Should-Start/Stop are now part of LSB as of 2.0,
#   formerly SUSE/Unitedlinux used X-UnitedLinux-Should-
#   Start/-Stop.
#   insserv does support both variants.
# * X-UnitedLinux-Default-Enabled: yes/no is used at
#   installation time
#   (%fillup_and_insserv macro in %post of many RPMs) to
#   specify whether
#   a startup script should default to be enabled after
#   installation. It's not used by insserv.
#
# Note on runlevels:
# 0 - halt/poweroff
# 6 - reboot
# 1 - single user
# 2 - multiuser without network exported
# 3 - multiuser w/ network (text mode)
# 5 - multiuser w/ network and X11 (xdm)
```

Diese Datei können Sie an Ihre Bedürfnisse anpassen. Relativ neu in dieser Datei ist der Block `INIT INFO`. Hier geben Sie an, in welchen Run-Leveln Ihr Programm aktiv sein soll und welche anderen Dienste zuvor bereits gestartet bzw. gestoppt sein müssen.

Mit diesen Informationen kann das Programm `insserv` im Verzeichnis `/etc/init.d/` automatisch die passenden symbolischen Links für die Run-Level anlegen.

Über den Wechsel der Run-Level startet das System Programme, die ständig aktiv sein sollen. Daneben gibt es auch Anwendungsfälle, bei denen ein Programm zu einem ganz bestimmten Zeitpunkt aktiv sein soll. Hierzu dient die Zeitsteuerung mit `at` und `cron`:

- Mit `at` startet man ein Programm einmalig zu einem bestimmten Zeitpunkt. Eine typische Anwendung sind Wartungsarbeiten, die dann ausgeführt werden sollen, wenn keine Benutzer angemeldet sind.
- Will man ein Programm regelmäßig zu einem bestimmten Zeitpunkt aufrufen, so bietet sich `cron` an. Alle regelmäßigen Wartungsarbeiten und statistischen Auswertungen gehören zu den möglichen Anwendungsfällen.

Lesen Sie in den nächsten beiden Abschnitten mehr über Zeitsteuerung.

4.2 Zeitgesteuerte Einzelaufträge

Mit dem Befehl `at` können dafür Berechtigte zu einem bestimmten Zeitpunkt Programme ausführen, z. B. ein zeitaufwändiges Programm nachts starten.

Tipp: Mit `at` kann man nur Programme starten, für deren Ausführung man auch die notwendigen Rechte besitzt. Für das Beispiel sollte man als `root` angemeldet sein, da normale Benutzer mit `find` nicht in allen Verzeichnissen suchen dürfen.

Um z. B. alle Dateien zu finden, die keinem Benutzer gehören, kann man den `find`-Befehl in der folgenden Form einsetzen:

```
find  /  -nouser
```

Der Suchvorgang ist recht zeitaufwändig, da `find` dazu alle Dateien untersuchen muss. Dateien ohne Benutzer entstehen, wenn man Benutzer löscht, die Dateien außerhalb ihrer Home-Verzeichnisse abgelegt haben. Da die Suche in größeren Systemen recht lange dauern kann, sollte man diese Suche auf einen ruhigen Zeitpunkt, z. B. 22:00 Uhr, verschieben. Dazu gibt man ein:

```
at  22:00
```

Am veränderten Eingabezeichen gibt man den eigentlichen Befehl

```
at> find  /  -nouser
```

ein und schließt die Eingabe dann mit Strg D ab.

```
boss:~ # at 22:00
warning: commands will be executed using /bin/sh
at> find / -nouser
at> <EOT>
job 1 at 2004-10-16 22:00
boss:~ #
```

Die Zeitpunkte für die Ausführung kann man auf verschiedene Arten angeben, wie hier im Beispiel über `HH:MM`, aber auch mit `now +2 hours`. Damit würde das Programm in zwei Stunden starten. Statt `hours` sind auch die Angaben `minutes`, `days` und `weeks` und absolute Zeitangaben wie `teatime` (16:00 Uhr) und `midnight` möglich.

Tipp: Sollten Sie nicht die beschriebene Rückmeldung erhalten, sondern eine Fehlermeldung, dann läuft der `at`-Dämon nicht, und Sie müssen ihn erst starten (z. B. über den Run-Level-Editor).

Unerledigte Aufträge zeigt `atq` an:

```
boss:~ # atq
1        2004-10-16 22:00   a   root
```

Anhand der Jobnummer eines Auftrages (im Bild *job 1*) kann man diesen wieder löschen:

```
boss:~ # atrm 1
```

Der `at`-Dämon gibt Daten statt auf den Bildschirm in eine Mail an den Auftraggeber aus.

Um regelmäßig nach herrenlosen Dateien zu suchen, benutzt man besser `cron`.

4.3 Regelmäßige Vorgänge mit cron

Für regelmäßige Vorgänge ist `cron` ein besseres Werkzeug als `at`. Für solche Vorgänge führt man in einer so genannten `crontab`-Tabelle die Vorgänge und die Zeiten auf, an denen man diese ausführen will.

In der Grundeinstellung dürfen alle Benutzer, die nicht in der Datei `/var/spool/cron/deny` verzeichnet sind, eine `crontab` anlegen. In der Grundinstallation sind hier nur `gast` und `guest` eingetragen.

Eine derartige `crontab`-Tabelle könnte folgendermaßen aussehen:

```
# roots crontab
#
# min hour day month dayofweek (1=Mo,7=Su) command
15 22 * * * /usr/bin/find / -nouser
```

So startet der Suchbefehl jeden Tag um 22:15 Uhr. Ein Stern steht als Jokerzeichen für alle Zeiten. Der Suchbefehl startet also an jedem Tag, in jedem Monat und an jedem Wochentag um 22:15 Uhr.

Eingeben kann man diese Tabelle als Benutzer `root` durch:

```
crontab  -e
```

Die Möglichkeiten der Zeitangabe sind recht vielfältig. Mit z. B.

```
# roots crontab
#
# min hour day month dayofweek (1=Mo,7=Su) command
15 22 * * 1-5 /usr/bin/find / -nouser
```

würde die Suche nur an Werktagen ablaufen.

Zu den Anwendungen, die Sie regelmäßig per `cron` ausführen sollten, gehört die Datensicherung – das Backup. Hinweise hierzu finden Sie oben im Kapitel 2.

Neben diesen zeitgesteuerten Aufträgen, die an einen Benutzer gebunden sind, kennt Cron auch Aufträge, die an einen allgemeinen Zeitpunkt gebunden sind:

- stündlich (hourly),
- täglich (daily),
- wöchentlich (weekly),
- monatlich (monthly).

Für derartige Aufträge besitzt Cron Verzeichnisse unterhalb von `/etc`, z. B. `/etc/cron.daily` für die täglichen Aufträge. Alle Programme, die sich in einem dieser Verzeichnisse befinden, führt `cron` immer dann durch, wenn der entsprechende Zeitpunkt gekommen ist. Außer bei `cron.hourly` ist das immer ein Zeitpunkt um Mitternacht.

4.4 Der Super-Dämon inetd für Internet-Dienste

Für viele Internet-Dienste wie POP, SMTP, FTP und Telnet findet man weder ein Start-Skript in `/etc/init.d/` noch einen zeitgesteuerten Aufruf.

Das spart Ressourcen, da die zugehörigen Programme erst bei Bedarf starten. Für das Starten ist der Super-Dämon `inetd` zuständig. SuSE installiert dazu den `xinetd`, eine neuere Version des inetd; das *x* im Namen steht für *extended* (erweitert).

Der xinetd ist flexibler konfigurierbar und auch stärker auf Sicherheit ausgerichtet als sein Vorgänger.

Konfigurieren können Sie `xinetd` in der recht kurzen und übersichtlichen Datei `/etc/xinetd.conf`.

`/etc/xinetd.conf`:

```
#
# xinetd.conf
#
# Copyright (c) 1998-2001 SuSE GmbH Nuernberg, Germany.
# Copyright (c) 2002 SuSE Linux AG, Nuernberg, Germany.
#

defaults
{
        log_type        = FILE /var/log/xinetd.log
        log_on_success  = HOST EXIT DURATION
```

```
        log_on_failure  = HOST ATTEMPT
#        only_from       = localhost
        instances       = 30
        cps             = 50 10

#
# The specification of an interface is interesting,
# if we are on a firewall.
# For example, if you only want to provide services from an
# internal network interface, you may specify your internal
# interfaces IP-Address.
#
#       interface       = 127.0.0.1

}

includedir /etc/xinetd.d
```

Mit dem #-Zeichen beginnen wie üblich die Kommentarzeilen. In dieser Konfigurationsdatei finden Sie nur allgemeinere Einstellungen, z. B. den Namen der Logdatei und welche Informationen darin aufgenommen werden sollen. Sie können hier auch regeln, dass Dienste überhaupt nur für ein bestimmtes Interface, also z. B. nur für *eth0*, starten können.

Am wichtigsten ist die letzte Zeile, die die Dateien im Verzeichnis `/etc/xinetd.d` in die Konfiguration aufnimmt. Wenn Sie wie im Kapitel 3 beschrieben das Paket *qpopper* installiert haben, finden Sie in diesem Verzeichnis jetzt eine Datei `qpopper`.

Mit dieser Datei legen Sie fest, was der xinetd bei Anfragen auf dem zugehörigen Port 110 anstellen soll.

```
/etc/xinetd.d/qpopper
#
# qpopper - pop3 mail daemon
service pop3
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = root
        server          = /usr/sbin/popper
        server_args     = -s
        flags           = IPv4
}
```

Sie finden hier die Angabe des Server-Programmes (*server*), eventuell mit zusätzlichen Startparametern (`server_args`), sowie Angaben zum Protokoll (*protocol*) und dem Benutzernamen (*user*) für diesen Dienst. Jeder Dienst arbeitet mit den

Rechten eines bestimmten Benutzers (hier `root`), so als ob dieser Benutzer das Programm selbst gestartet hätte.

Wenn Sie in der Konfigurationsdatei für einen Dienst die Zeile

```
disable         = yes
```

finden, ist dieser Dienst deaktiviert, und der xinetd wird ihn nicht starten.

Sie brauchen die Konfigurationsdateien nicht direkt zu bearbeiten. Einfacher geht das über das YaST-Kontollzentrum und die Funktion *Netzwerkdienste • Netzwerkdienste (inetd)*. Sie sehen eine Liste aller vorbereiteten Dienste und deren Status.

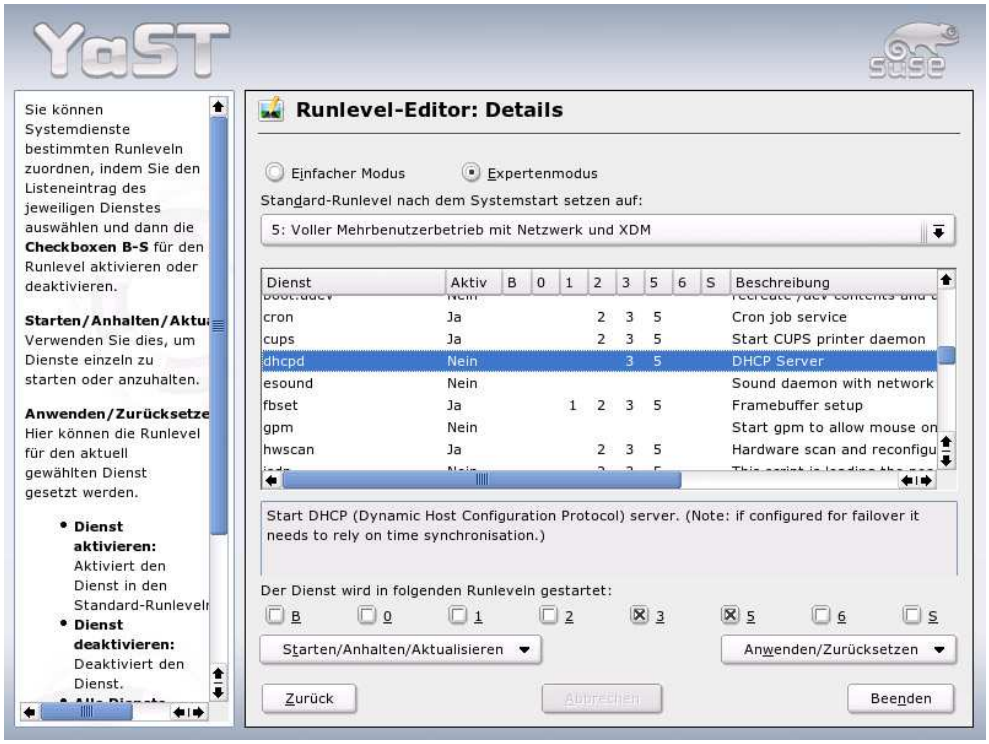

Abbildung 4.2: Konfiguration der Netzwerkdienste

Im Auslieferungszustand ist keiner der Dienste aktiviert, SuSE aktiviert den xinetd daher auch nicht. Sie müssen also im Zweifelsfall zuerst auf *Aktivieren* gehen, um den xinetd überhaupt zu starten.

Dazu steuern Sie den Leuchtbalken auf den gewünschten Dienst und aktivieren oder deaktivieren diesen Dienst dann mittels *Status wechseln*. Da YaST die Änderungen erst nach Klick auf *Weiter* in die Konfigurationsdateien übernimmt, zeigt es in der ersten Spalte der Zeile noch an, dass es sich um eine geänderte Einstellung handelt.

Mit einem Klick auf *Bearbeiten* können Sie auch die gesamte Konfigurationsdatei für einen Dienst bearbeiten.

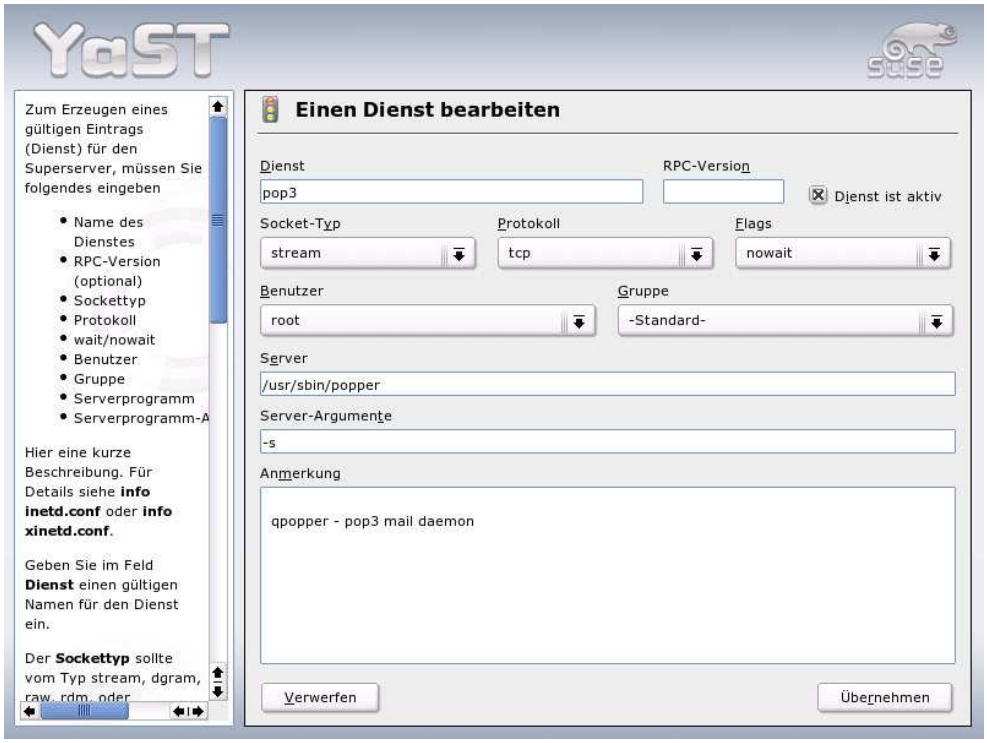

Abbildung 4.3: Einen Dienst bearbeiten

Sie sehen hier dieselben Informationen wie in der Konfigurationsdatei. Da das Formular die Gefahr von Fehleingaben verringert, hilft es beim Konfigurieren.

YaST unterscheidet drei Dienst-Zustände. Am häufigsten finden Sie deaktiviert (---), am wichtigsten ist wohl aktiviert (*An*). Weiter unten in der Liste stehen die nicht installierten Dienste (*NI*), für die SuSE die Konfiguration vorbereitet hat, das Paket mit dem Server-Dienst fehlt aber noch.

Wenn Sie dieses Buch bis zum Ende durchgearbeitet haben, werden sich in Ihrer Liste mehr aktivierte Dienste finden als jetzt. Da jeder aktivierte Dienst ein zusätzliches Risiko bedeutet, sollten Sie stets nur die Dienste aktivieren, die Sie auch benötigen, insbesondere, wenn Sie Ihren Rechner direkt mit dem Internet verbinden.