Install-Server für mehrere Betriebssysteme

Brückenschlag

Aus verschiedensten Gründen ist in großen, heterogenen Umgebungen eine ganze Reihe von Install-Servern zu finden – pro Betriebssystem-Variante mindestens einer. Der folgende Artikel zeigt, wie ein Linux-Server neben Red Hat und Suse auch Solaris und Irix bedient. Udo Seidel

Fast jede Betriebssystem-Installation läuft nach dem gleichen Schema ab: Der Rechner bootet ein Mini-Betriebssystem, das ein Installationsprogramm enthält und startet. Das Mini-Betriebssystem besteht aus dem OS-Kernel und dem Dateisystem-Image mit den notwendigen Programmen und Konfigurationen. Die Dateien sind auf lokalen Medien wie Festplatte, CD-ROM, DVD oder USB-Stick gespeichert oder über das Netzwerk ladbar.

Der Installer erwartet eine Reihe von Angaben, die der Benutzer entweder interaktiv eingibt oder in einer so genannten Antwortdatei bereitstellt. Die zu installierende Software befindet sich entweder auf einem lokalen Datenträger oder ist über ein Netzwerk-Share erreichbar. Der letzte Schritt ist die Anpassung des Betriebssystems an die Bedürfnisse des Benutzers.

Netzboot mit PXE

Für das Vorhaben eines gemeinsamen Installations-Servers stellt sich die Frage, ob sich die oberflächlichen Parallelen der OS-Installation auch im technischen Detail fortsetzen. Da wäre zunächst der Bootprozess, also das Laden des Mini-Betriebssystems inklusive Installer-Start. Lokale Bootmedien sind ab einer gewissen Größe der Rechnerlandschaft umständlich, der PXE-Boot (Pre-Execution Environment) ist die elegante, Netzwerk-basierte Alternative [1]. Das PXE-Protokoll ist eine Art Zusammenfassung des Trivial File Transfer Protocol (TFTP) und des Dynamic Host Configuration Protocol DHCP.

Die Netzwerkkarte initiiert den PXE-Boot. Dazu sendet sie eine DHCP-Anfrage ins Netz und erhält in der Antwort neben den TCP/IP-relevanten Informationen auch die IP-Adresse des so genannten Bootservers. Mit Hilfe von TFTP lädt die Karte dann einen Bootloader herunter und führt ihn im Speicher auf. All diese Protokolle sind in RFCs der IETF spezifiziert und auf deren Website abrufbar [2].

Die Aufgabe des Bootloaders ist es, einen OS-Kernel mit den gewünschen Optionen zu laden. Dazu holt er sich die notwendigen Dateien mittels TFTP und führt sie im Arbeitsspeicher aus. Treten keine Fehler auf, ist der Rechner mit einem

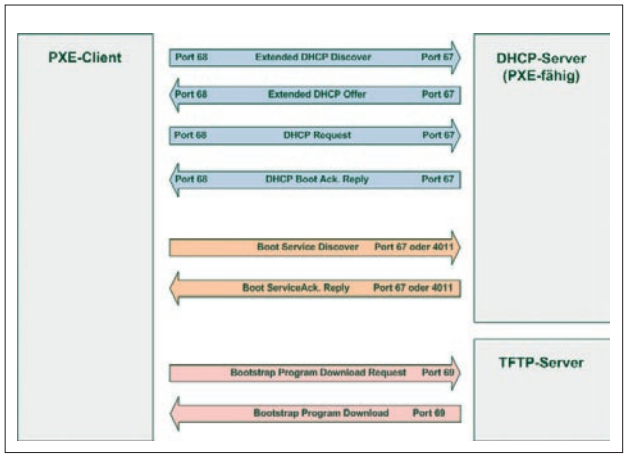

Abbildung 1: Der PXE-Bootprozess im Einzelnen.

Mini-Betriebssystem gebootet und hat den Installer gestartet. Die verwendeten Technologien sind in verschiedenen RFCs (Requests for Comments) spezifiert und damit prinzipiell OS-unabhängig.

Das Installationsprogramm benötigt grundsätzlich zwei verschiedene Informationspakete. Das sind erstens die Konfigurationsdetails des Rechners wie Rechnername, Software-Auswahl, Zeitzone, Root-Passwort und Festplatten-Einteilung. Das zweite Informationspaket beschreibt, wo und wie der Installer auf die zu installierenden Softwarepakete zugreifen kann. Typisch sind hier die entsprechenden Daten eines Fileservers: IP-Adresse oder Hostname und Name des entsprechenden Share.

Auslieferung

Überwiegend kommt das Network File System (NFS) zur Anwendung. Moderne Linux-Distributionen können aber auch auf CIFS, HTTP oder FTP zurückgreifen. Beide Informationspakete kann der Admin dem Installer per Antwortdatei bereitstellen, sodass die eigentliche OS-Installation ohne Benutzerinteraktion erfolgt. Name und Syntax dieser Antwortdatei und die Art und Weise der Bereitstellung ist aber stark herstellerabhängig. Tabelle 1 listet für einige Unix-Derivate und Linux-Distributionen den Namen der automatisierten Installation auf. Zum Glück – und auch nicht unerwartet – spielt beim Zugriff auf die Antwortdatei NFS ein große Rolle.

Selbst auf technischer Ebene verwenden die meisten OS-Installer dieselben Technologien, die zudem durch RFCs standardisiert sind. Damit ist die Idee eines gemeinsamen Install-Servers durchaus realistisch. Der mehrstufige Aufbau einer automatischen OS-Installation erlaubt es, gegebenenfalls bestimmte Aufgaben auf andere dedizierte Server auszulagern. Prinzipiell gibt es nun zwei Wege, um sich dem Aufbau eines Install-Servers zu nähern. Der einfachere Zugang ist die Bereitstellung der Softwarepakete für die eigentliche Installation. Der technisch interessantere Weg ist es allerdings, mit dem Bootserver anzufangen. Diesen Weg wählt dieser Artikel.

Bis zu vier verschiedene Techniken kommen zur Anwendung, bis der zu installierende Rechner das Installationsprogramm gestartet hat (Tabelle 2). Typischerweise erfüllt ein einziger Rechner alle Funktionen. Es ist aber auch möglich, für jede Aufgabe einen dedizierten Server aufzusetzen. Im Labor des Autors etwa fungiert ein Linux-basiertes NAS-Gerät von Netgear als PXE-, TFTP- und NFS-Server, während ein geclustertes Open BSD die DHCP-Dienste bereitstellt.

Für den Install-Server unter Linux benötigt der Admin eine Reihe von Softwarepaketen, um die entsprechenden Daemon-Prozesse konfigurieren und starten zu können. Für den PXE-Teil verwenden die meisten Distributionen das Syslinux-Paket [3]. Die DHCP-Implementation stammt vom Internet Systems Consortium [4]. Einen TFTP-Server erhält man entweder dort, wo es auch die neuesten Linux-Kernels gibt [5], oder bei Freshmeat [6].

Pixie me!

Wie in Abbildung 1 dargestellt beginnt der PXE-Bootprozess mit einer DHCP-Kommunikation. Neben den für den TCP/IP-Stack relevanten Angaben erhält der Rechner hier auch die Information, von welchem Server er welche Datei als Bootloader herunterladen soll. Die entsprechenden Anweisungen lauten »next-server« und »filename«. Hinter der IP-Adresse von »next-server« verbirgt sich der TFTP-Server. Die Angabe »filename« bezieht sich auf das Wurzelverzeichnis des TFTP-Daemon. In den meisten Fällen verwendet der Admin »/tftpboot« als eine Art Chroot-Verzeichnis, sodass die Angaben in der DHCPD-Konfiguration relativ zu diesem Verzeichnis zu sehen sind – doch dazu später mehr (Listing 1).

Ist der Bootloader »pxelinux.0« geladen, sucht er seine Konfigurationsdatei auf dem TFTP-Server. Diese Datei macht Angaben über den zu ladenden OS-Kernel und eventuell notwendige Parameter. Dazu erwartet »pxelinux.0« ein Verzeichnis namens »pxelinux.cfg«, das sich am gleichen Ort wie »pxelinux.0« befindet. Das Installieren von mehr als einem Rechner vom gleichem Bootserver ist ein gewünschtes Feature.

Die Konsequenz ist die Möglichkeit, mehrere Bootloader-Konfigurationen in einem Verzeichnis zu verwalten. Daher sucht »pxelinux.0« zunächst nach einer Konfi-

Listing 1: DHCP-Konfiguration

```
01 # cat /etc/dhcpd.conf
02 allow bootp;
03 authoritative;
04
05 subnet 192.168.5.0 netmask 255.255.255.0 {
06         option routers                  192.168.5.254;
07         option subnet-mask              255.255.255.0;
08
09         option domain-name              "beispiel.
   de";
10         option domain-name-servers      192.168.5.34,
   192.168.5.5;
11         range 192.168.5.151 192.168.5.155;
12         default-lease-time 21600;
13         max-lease-time 43200;
14         get-lease-hostnames true;
15 }
16 ...
17 host rhel {
18         hardware ethernet 00:0F:1F:14:F7:1C;
19         fixed-address 192.168.5.205;
20         next-server 192.168.5.10;
21         filename "rhel54/pxelinux.0";
22 }
23 ...
24 #
```

Tabelle 1: Betriebssysteme und ihre Autoinstaller

OS-Name	Autoinstaller
HP-UX	Ignite
Irix	Roboinst
AIX	NIM
Solaris	Jumpstart
Red Hat/Fedora	Kickstart
(Open) Suse	Autoyast

Tabelle 2: Phasen der OS-Installation

Phase	Protokoll
Bootloader	PXE (TFTP, DHCP)
Mini-OS	TFTP
Installer	NFS

gurationsdatei, deren Name sich aus der MAC-Adresse und dem ARP-Codetyp ergibt (RFC 826). Der ARP-Typecode für Ethernet ist 1 und ist der MAC-Adresse vorangestellt. Die Hexzahlen sind mit Kleinbuchstaben anzugeben, Bindestriche dienen als Separator. Aus der Ethernet-MAC-Adresse »00:0F:1F:14:F7:1C« wird damit »01-00-0f-1f-14-f7-1c«.

Gibt es diese nicht, verwendet der Bootloader die IP-Adresse – allerdings in hexadezimaler Form. Hier muss der Admin aber Großbuchstaben verwenden und auf Trennzeichen verzichten. Im Falle von 192.168.5.205 ergibt sich damit »C0A805CD«. Durch sukzessives Entfernen einer Hexzahl erweitert »pxelinux.0« den Such-Radius (siehe Listing 2). Sind auch diese Bemühungen vergeblich, sucht PXE-Linux nach einer Datei namens »default«.

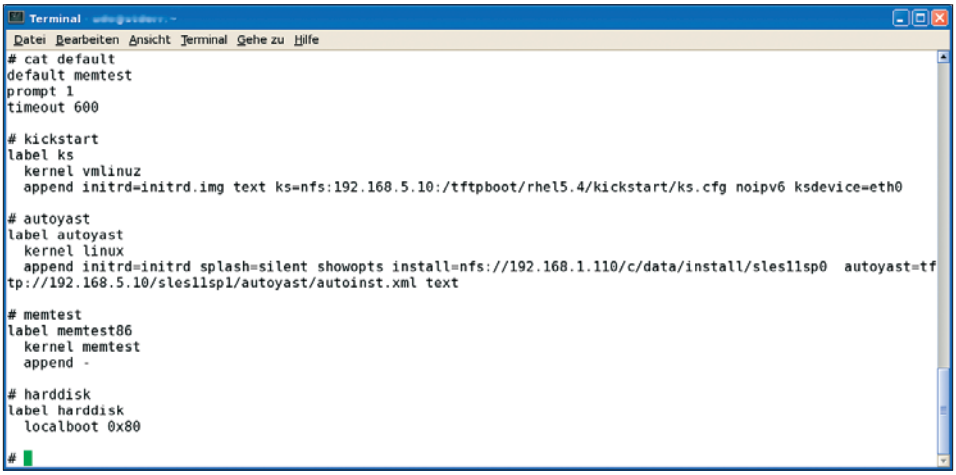

Abbildung 2: PXE-Linux-Konfiguration inklusive Anweisungen für Kickstart (RHEL) und Autoyast (SLES).

Neuere Versionen von PXE-Linux verstehen zusätzlich Nicht-Standard-DHCP-Optionen. So kann der Admin beispielsweise den Standard-Suchpfad für die Konfigurationsdatei überschreiben oder einen automatischen Reboot bei auftretenden TFTP-Fehlern veranlassen.

Listing 2: Suchmuster für PXE-Linux

```
01 /pxelinux.cfg/01-00-0f-1f-14-f7-1c
02 /pxelinux.cfg/C0A805CD
03 /pxelinux.cfg/C0A805C
04 /pxelinux.cfg/C0A805
05 /pxelinux.cfg/C0A80
06 /pxelinux.cfg/C0A8
07 /pxelinux.cfg/C0A
08 /pxelinux.cfg/C0
09 /pxelinux.cfg/C
10 /pxelinux.cfg/default
```

Listing 3: PXE-Grub-Konfiguration für Solaris

```
01 # cat menu.lst.010024BE3A0DEF
02 default=0
03 timeout=5
04 title Solaris Jumpstart
05         kernel /solaris10/multiboot kernel/unix -B
   install_media=192.168.5.10:/data/install/solaris10
   - install nfs://192.168.5.10/tftpboot/solaris10/
   jumpstart/config.tar text
06         module /solaris10/x86.miniroot
07 title Solaris
08         kernel /boot/multiboot kernel/unix -B
   install_media=cdrom
09         module /boot/x86.miniroot
10 title Solaris Serial Console ttya
11         kernel /boot/multiboot kernel/unix -B
   install_media=cdrom,console=ttya
12         module /boot/x86.miniroot
13 title Solaris Serial Console ttyb (for lx50, v60x and
   v65x)
14         kernel /boot/multiboot kernel/unix -B
   install_media=cdrom,console=ttyb
15         module /boot/x86.miniroot
```

Mehrstufig

Der Aufbau der Konfigurationsdatei von PXE-Linux ist recht einfach, sie verwendet folgende Schlüsselwörter: »DEFAULT«, »LABEL«, »KERNEL« und »APPEND«. Die Anweisung »KERNEL« beschreibt, welche Datei als OS-Kernel zu laden ist. Dabei muss das nicht unbedingt ein „echter“ OS-Kernel sein, es kann sich auch um einen weiteren (staging) Bootloader handeln. Mit »APPEND« weist der Admin dem zu ladenden Kernel weitere Argumente zu. Typisch sind hier der Verweis auf eine Ramdisk, die das Root-Dateisystem enthält. Die Anweisungen »KERNEL« und »APPEND« fasst ein »LABEL« zusammen. Der Label-Name ist Referenzpunkt für den Anwender, um einen bestimmten Kernel für den Bootprozess auszuwählen.

Über weitere Anweisungen kann der PXE-Admin einen Kernel als Standard bestimmen, den der Bootloader bei fehlender Benutzereingabe auswählt. Der konkrete Inhalt der Konfigurationsdatei von PXE-Linux hängt vom zu installierenden Betriebssystem ab. Ein Beispiel für zwei verschiedene Linux-Distributionen ist in Abbildung 2 angegeben.

Das x86-Solaris 10 verwendet wie praktisch alle Linux-Distributionen seit einiger Zeit Grub als Bootloader und erwartet ihn auch bei Installationen übers Netzwerk. Daher lädt PXE-Linux nicht direkt den Solaris-Kernel, sondern eine PXE-fähige Version von Grub: »pxegrub«. Die entsprechende DHCPD-Konfiguration zeigt Abbildung 3. Das Laden des Solaris-Kernels mit den entsprechenden Optionen übernimmt Grub, der seine Konfigurationsdatei ebenso wie PXE-Linux auf dem Bootserver erwartet. Eine Konfiguration von PXE-Linux entfällt.

Ebenso wie PXE-Linux kann PXE-Grub rechnerspezifische Konfigurationen ver-

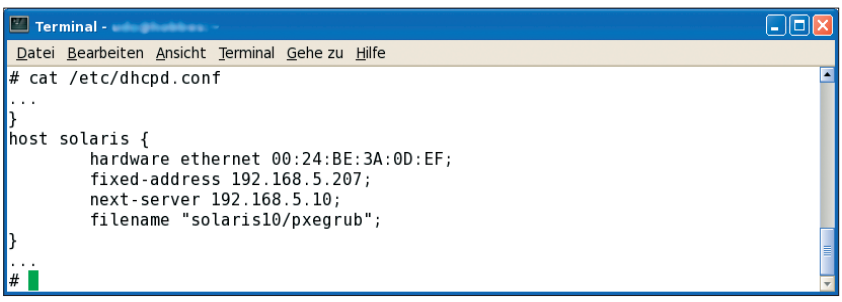

Abbildung 3: DHCPD-Konfiguration für x86-Solaris mit Grub.

walten. Dazu sucht »pxegrub« auf dem Bootserver nach einer Datei namens »menu.lst.*Hardware-ID*«. Die Hardware-ID ergibt sich aus dem ARP-Typecode und der MAC-Adresse in Hex-Darstellung – alles ohne Trennzeichen. Ist diese Datei nicht vorhanden, sucht Grub nach »/boot/grub/menu.lst« im Wurzelverzeichnis des TFTP-Servers. Wie PXE-Linux versteht auch PXE-Grub Nicht-Standard-DHCP-Optionen. Dann kann der Admin über die DHCP-Option 150 einen vollständigen Pfad zur PXE-Grub-Konfiguration festlegen.

Die Konfiguration von PXE-Grub sollte für den gestandenen Linux-Admin kein Problem darstellen, da sie dem normalen Grub sehr ähnelt. Ein augenscheinlicher Unterschied ist das Laden eines Programms »multiboot« zusätzlich zum OS-Kernel. Dieses Programm entspricht der Multiboot-Spezifikation [7], es übernimmt die Kontrolle über den Bootprozess von PXE-Grub. PXE-Grub hat den

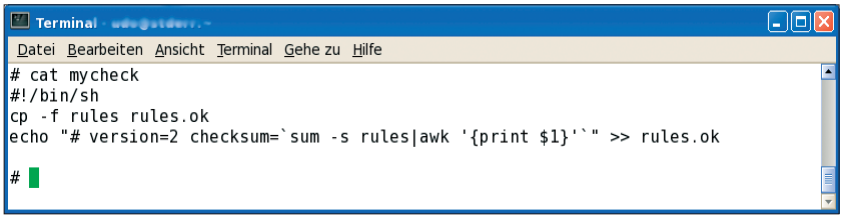

Abbildung 4: Nachbildung des Solaris-Skripts »check« zur Erzeugung der »rules.ok«-Datei.

Betriebssystemkern zwar in den Speicher geladen, die Aktivierung geschieht aber erst durch »multiboot«. Die notwendigen Binaries »pxegrub« und »multiboot« sind nicht Bestandteil von PXE-Linux. Sie befinden sich im Lieferumfang von x86-Solaris (Listing 3).

Irix

Bei Irix, der Unix-Variante von Silicon Graphics, ist der Administrator in Sachen PXE-Boot eingeschränkt. Anstelle von DHCP verwendet der Irix-Installer nämlich das ältere BOOTP (Listing 4). Obwohl der DHCP-Daemon BOOTP-Anfragen auch bedienen kann, ließ sich SGIs Unix erst mit einem waschechten BOOTP-Daemon zur Mitarbeit bewegen. Die Angaben zum Bootserver, die in den anderen Fällen der DHCP-Server bereitstellt, muss der installierende Benutzer im Command PROM eingeben.

Das sieht für Roboinst ungefähr so aus: »boot -f bootp()192.168.5.10:/dist/sa(sash64) mrmode=custom mrconfig=192.168.5.10:/custom/octane«. Die für den Bootprozess benötigten Programme stellt dann wieder der TFTP-Server bereit. Das gilt für die »sash«

(Standalone Shell), den Disk-Initialisierer »fx« und natürlich den Irix-Kernel. Die verwendeten Pfade sind relativ zum Root-Verzeichnis des TFTP-Daemon. Die Binärdateien befinden sich auf den Irix-Installations-CDs.

Mit dem Booten des OS-Kernels über das Netzwerk ist der schwierigste Teil der Arbeit erledigt. Nun muss sich der Admin darum kümmern, dass der Installer alle Fragen vorab beantwortet bekommt und dass die Softwarepakete im Netz auch verfügbar sind.

Wie von selbst

Damit der Installer ohne Benutzer-Rückfragen durchläuft, benötigt er eine Datei, die alle Antworten bereithält. Typischerweise liegt diese Antwortdatei auf einem NFS-Share oder ist per TFTP erreichbar. Die Information, wie der Installer an diese Datei herankommt, muss der Admin schon beim Booten des OS-Kernels mitgeben. Demzufolge gilt es, die Konfiguration von PXE-Linux anzupassen. Im Falle von Red Hat und Suse ist das recht einfach und bereits in Abbildung 2 dargestellt. Beide Linux-Distributionen verwenden jeweils eine Datei.

Sowohl Solaris als auch Irix gehen hier einen komplexeren Weg. Der Solaris-Installer Jumpstart erwartet zwei Dateien: »sysidcfg« und »rules«. In der ersten gibt der Admin Hardware-unabhängige Daten wie Zeitzone, Root-Passwort, Verzeichnisdienste und Hostname an. Die Festplatten-Einteilung und Informationen über die zu installierende Software erfährt der Installer über die Datei »rules«. Nähere Details sind dabei in weiteren Konfigurationsdateien untergebracht, auf die »rules« verweist.

Der signifikante Unterschied zu gängigen Linux-Installern ist nun, dass Jumpstart eine Art Prüfsumme der »rules«-Datei erwartet, die in »rules.ok« abgelegt ist. Das Solaris-Skript »check« funktioniert nicht ohne Weiteres unter Linux. Glücklicherweise lässt sich die Prüfsummenbildung aber mit einem Linux-Skript nachstellen (siehe Abbildung 4).

Da PXE-Linux aber nur mit einer Antwortdatei umgehen kann, gibt es bei Jumpstart zunächst ein Problem. Es ist aber nicht spezifisch für den Linux-Install-Server und tritt auch bei reinen Solaris-Umgebungen auf. Die Lösung ist einfach: alle Jumpstart-Konfigurationen in ein Archiv packen und per PXE-Linux auf das Archiv verweisen (Listing 5). Gültige Archivtypen sind Tar, Tar.gz, Zip und Tar.bz2. Das Beispiel in Listing 3 benutzt ein Tar-Archiv, die entsprechende Anweisung lautet »nfs://192.168.5.10/tftpboot/solaris10/jumpstart/config.tar«.

Neben der Verteilung der Jumpstart-Konfiguration sieht das Bereitstellen der eigentlichen Softwarepakete wie ein Kinderspiel aus. Ein ganz normaler NFS-Share reicht hier aus. Es gibt keine Besonderheiten zu beachten, wenn man von den normalen NFS-Interoperabilitäts-Herausforderungen absieht. Bei SGIs Roboinst ist die Lage noch ein bisschen komplizierter.

Listing 4: BOOTP-Konfiguration für SGIs Roboinst

```
01 iris:\
02         :ht=ether:\
03         :ha=0800690e3d81:\
04         :sm=255.255.255.0:\
05         :gw=192.168.5.254:\
06         :ip=192.168.5.12:
```

Listing 5: Jumpstart-Dateien in einem Tar-Archiv

```
01 # tar tf config.tar
02 add_guest_user
03 any_machine
04 finish
05 rules
06 rules.ok
07 set_root_login_console
08 sysidcfg
```

Prüfsumme unter Linux

Wie oben bereits gezeigt, sind Eingaben am Command PROM nötig, was eine Fernsteuerung der Installation nahezu unmöglich macht. Analog zu Jumpstart legt auch Roboinst eine Art Prüfsummendatei der Konfiguration an. Das zugehörige Skript »roboinst_config« funktioniert nach einer kleinen Anpassung auch unter Linux. Die Roboinst-Konfiguration lädt der zu installierende Rechner per TFTP vom Server.

Für die eigentliche Installation nutzt der Irix-Installer aber überraschenderweise kein NFS, sondern die Remote Shell (RSH). Auf dem Install-Server erwartet Roboinst einen Benutzer »guest«, der vom zu installierenden Rechner ohne Angabe eines Passwort erreichbar sein muss. Neben einem entsprechenden Eintrag in »rhosts« muss der Admin bei den meisten Linux-Distributionen auch passende Einträge in der »/etc/securetty« vornehmen. Es kommt noch dicker: Der Irix-Installer führt als Benutzer »guest« auf dem Install-Server »/bin/sh« aus, erwartet aber eine Korn-Shell. Die Installation mit Linux als Roboinst-Server gelingt erst, wenn »/bin/sh« ein Link auf »ksh« ist.

Was sonst noch geht

Red Hat, Suse, x86-Solaris und Irix sind nicht die einzigen Betriebssysteme, die sich über einen Linux-Install-Server installieren lassen. Die Reihe lässt sich fortsetzen mit Free BSD, Open BSD oder AIX. Wesentlich ist, dass bei der Netzwerk-basierten Installation die Standardprotokolle TFTP, DHCP und NFS zum Einsatz kommen. Die grundlegenden Herausforderungen für den Admin sind zum einen die Anpassung von PXE-Linux an die Anforderungen des zu installierenden OS. Die zweite Hürde ist die Konfiguration der automatischen OS-Installation. Beide Aufgaben sind mit etwas Geduld und Beharrlichkeit zu meistern. Dieser Artikel zeigt, dass der zentrale Install-Server unter Linux im Rechenzentrum keine Utopie ist. *(ofr)* ■

Infos

[1] PXE-Spezifikation: [http://www.pix.net/software/pxeboot/archive/pxespec.pdf]

[2] IETF-RFCs: [http://tools.ietf.org/html]

[3] Syslinux für PXE: [http://syslinux.zytor.com/wiki/index.php/PXELINUX]

[4] ISC DHCP: [http://isc.org/products/DHCP/]

[5] TFTP-Server: [http://www.kernel.org/pub/software/network/tftp/]

[6] ATFTP-Server: [http://freshmeat.net/projects/atftp]

[7] Grub Multiboot: [http://www.gnu.org/software/grub/manual/multiboot/multiboot.html]

Der Autor

Dr. Udo Seidel ist eigentlich Mathe-Physik-Lehrer und seit 1996 Linux-Fan. Nach seiner Promotion hat er als Linux/Unix-Trainer, Systemadministrator und Senior Solution Engineer gearbeitet. Heute ist er Leiter eines Linux/Unix-Teams bei der Amadeus Data Processing GmbH in Erding.