SmartClient Betriebshandbuch

Jörg Steffens

17. Juli 2009

Inhaltsverzeichnis

1 Server-Installation und -Konfiguration

Das SmartClient-Konzept sieht einige zentrale Serverdienste vor, die wegen notwendiger Redundanz und Lastverteilung auf mehrere Server aufgeteilt werden sollten.

Die Vorgehensweise zur Einrichtung der Server-Installationen ist dabei wie folgt:

1. Konzeption und Aufteilung der Server-Dienste auf die Server
2. Partitionierung der Server-Festplatten
3. Grundinstallation der Server mit allen notwendigen Diensten in der Paketauswahl
4. Installation der SmartClient-spezifischen Zusatzpakete
5. Konfiguration des LDAP für die Kommunikation mit den Clients
6. Test der Installation durch Anschluss eines neuen Clients

Dieses wird in den folgenden Abschnitten detailliert dargestellt.

1.1 Benötigte Server-Dienste

Folgende Dienste werden zum Betreiben einer SmartClient Infrastruktur benötigt:

- LDAP
- DNS
- DHCP
- TFTP
- Rsync
- Syslog
- NFS (optional)

Prinzipiell können alle Dienste auf einem einzigen Server laufen. Bei größeren Installationen bzw. Produktivbetrieb empfehlen wir die Aufteilung auf mehrere Server, wobei soweit möglich die Redundanz- bzw. Replikationsmechanismen der Dienste eingesetzt werden sollten.

1.2 Aufteilung der Server-Dienste

Ein typischer SmartClient Server Aufbau besteht aus vier zentralen Servern mit unterschiedlichen Aufgabenbereichen. Die zentralen Dienste des SmartClient-Konzepts sind wie folgt auf die Server aufgeteilt:

Server	Aufgaben	Beispiel IP
`server1`	LDAP-Master, DNS-Master, DHCP-Master, TFTP	10.1.110.1
`server2`	LDAP-Slave, DNS-Slave, DHCP-Slave, TFTP	10.1.110.2
`server3`	Rsync, NFS	10.1.110.3
`server4`	Syslog, Admin, Backup	10.1.110.4

Neben den zentralen Servern wird jeweils ein Server pro Außenstelle als *Caching-Server* verwendet. Diese Aufgabe können *dezentrale Server* aber auch normale SmartClient Workstations erledigen.

Wegen der unterschiedliche Aufgaben müssen die zentralen Server auch unterschiedlich konfiguriert sein. Die Grundinstallation wird aber möglichst einheitlich gestaltet, so dass auch Pakete installiert werden, die nicht auf allen Servern im Einsatz sind.

1.3 Partitionierung

Die Partitionsgrößen sind abhängig von der Bestimmung des Rechners. Um in der Partitionierung flexibel zu bleiben, bietet sich die Verwendung eines Logical Volume Manager (LVM) an.

Grundsätzlich wird folgende Partitionierung vorgeschlagen:

Mountpunkt	Größe	Aufgabe
swap	1 G	Swap
/	1 G	System
LVM:		
/tmp	1G	Temporäre Dateien
/usr	4G	System
/opt	2G	System
/home	1G (10G)	Benutzer-Verzeichnisse, ggf. zentral für alle Benutzer
/srv	1G (10G)	u.a. Distributions- und Update-Quellen
/var	4G	System. Variable Anteile
/var/log	2G	Log-Dateien, ggf. zentraler Log-Server
/var/lib/smartclient	1G (40G)	SmartClient Images

Wenn ein Server mehr Plattenplatz in einem bestimmte Bereich benötigt, werden die in Klammern gesetzten Werte verwendet (`/home` für den NFS-Server der Home-Verzeichnisse und den Backup-Server, `/export` für den Distributionsserver, `/var/lib/smartclient` für den Image (Rsync) Server).

1.4 Grundinstallation der Server

Auf allen Servern wird der SuSE Linux Enterprise Server 8 (SLES8) mit minimaler Paketauswahl (Minimal-System) installiert.

Zusätzlich Pakete, die auf allen Servern installiert werden:

`bind`	DNS Server.
`bind-utils`	von bind9 benötigt.*SLES hat nur bind9!!*
`dhcp-base`, `dhcp-server`	DHCP Server (automatische IP Adressvergabe)
`xntp`	Network Time Protocol. Zeitabgleich übers Netzwerk.
`atftp`	Trivial File Transfer Protocol. Zum booten von Rechnern übers Netzwerk benötigt.
`rsync`	effizientes Tool zur Datensynchronisierung übers Netzwerk.
`openldap2`	LDAP Server.
`nss_ldap`	wird u.a. zur Benutzerverwaltung über LDAP benötigt.
`pam_ldap`	wird zur Authentifizierung verwendet.
`sudo`	Zur Rechtevergabe von `ldap2dns` und `ldap2dhcp`.

Spezielle SmartClient-Pakete, die auf allen Servern installiert sein sollten:

`sc_base`	Verzeichnisstrukturen, Administrationsbenutzer und grundlegende Konfigurationsdateien.
`perl-SmartClient`	Perl Module, die von den anderen SmartClient Programmen verwendet werden.
`sc_tools`	Nützliche (kleine) Tools.

1.4.1 Server Aktualisierungen

Es wird empfohlen, nach der Installation und in regelmäßigen Abständen bzw. bei sicherheitskritischen Fehlern, die über das SuSE Maintenance Web veröffentlicht werden, Updates durchzuführen. Wie diese Updates durchzuführen sind, kann der Systemdokumentation entnommen werden.

1.5 Grundlegende Konfiguration der Server

1.5.1 Namensauflösung:

Als DNS Server sind die beiden eigenen DNS Server und in der Suchliste die zu ergänzenden Domains angegeben.

Die entsprechende Datei `/etc/resolv.conf` sieht folgendermaßen aus:

```
nameserver 10.1.110.1
nameserver 10.1.110.2
search server.meine-domain.de meine-domain.de
```

Achtung: Workstations haben typischerweise einen Namen wie `pc00034.gs.region.smartclient.de`.

Versucht man den Rechner über die Kurzform `pc00034.gs.region` aufzulösen, kann dieses fehlschlagen, da `gs` die Länderdomaine für South Georgia und die South Sandwich Inseln (Inseln im Süd Atlantischen Ozean) ist.

1.5.2 SSH - Secure Shell

Es gibt zwei Protokoll-Versionen von SSH. Bei der aktuellen Installation verwenden wir ausschließlich die Version 2, da diese sicherer ist und mittlerweile den Standard darstellt.

`ssh_config`

Die Datei `/etc/ssh/ssh_config` steuert den von diesem Rechner aus initiierten ssh-Verbindungsaufbau.
Zur Optimierung des Netzwerkverkehrs wird zwischen Workstations und Servern unterschieden.
Einige Sicherheitseinstellungen, wie z.B. `StrictHostkeyChecking`, müssen für `leases2ldap` und den Backup-Rechner aufgeweicht werden, damit ein Betrieb im Batch Modus möglich wird:

```
#   $OpenBSD: ssh_config,v 1.10 2001/04/03 21:19:38 todd Exp $

# This is ssh client systemwide configuration file. See ssh(1)
# for more information. This file provides defaults for users,
# and the values can be changed in per-user configuration
# files or on the command line.

# Configuration data is parsed as follows:
#  1. command line options
#  2. user-specific file
#  3. system-wide file
# Any configuration value is only changed the first time
# it is set. Thus, host-specific definitions should be at
# the beginning of the configuration file, and defaults
# at the end.

#
# workstation configuration
# faster encription, compression, less checking
#
Host ws*
    # ssh workstation key changes after update, so
    # don't check them.
    # don't check known_hosts, because we need a
    # working batch mode
   StrictHostKeyChecking no

    # use compression
   Compression yes

#
# server configuration
#
```

```
Host *.server.smartclient.de
    # nothing at the moment, take defaults

#
# Site-wide defaults for various options
#
Host *
#   ForwardAgent no
#   ForwardX11 no
#   RhostsAuthentication no
#   RhostsRSAAuthentication no
#   RSAAuthentication yes
#   PasswordAuthentication yes
#   BatchMode no
#   CheckHostIP yes
#   StrictHostKeyChecking ask
   StrictHostKeyChecking no
#   IdentityFile ~/.ssh/identity
#   IdentityFile ~/.ssh/id_rsa
#   IdentityFile ~/.ssh/id_dsa
#   Port 22
   Protocol 2,1
#   Cipher 3des
#   Ciphers aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,\
#     arcfour,aes192-cbc,aes256-cbc
#   EscapeChar ~
```

ssh_keys

SSH bietet mehrere Möglichkeiten der Authentisierung an. Ohne weitere Konfiguration findet die Authentisierung über die normalen Unix-Passwörter statt. Diese unterliegen allerdings den Unix-typischen Einschränkungen (z.B. maximale Länge von 8 Zeichen). Mehr Sicherheit und Möglichkeiten bieten Public-Key SSH-Schlüssel. Ein Schlüsselpaar mit DSA Verschlüsselung kann vom Benutzer mit dem Kommando

```
ssh-keygen -t dsa
```

erzeugt werden. Dabei wird man nach einer Passphrase gefragt, wobei im Gegensatz zu Benutzerpasswörtern beliebige und beliebig lange Zeichenketten verwendet werden können. Die Sicherheit ist abhängig von der Komplexität des Passworts (keine Wörter/Sätze, die in Büchern stehen) und (entscheidend) von der Länge.

Nach dieser Eingaben wird das Schlüsselpaar erzeugt und in den Dateien

`~/.ssh/id_dsa`
(privater Schlüssel; darf nicht von anderen Benutzern gelesen werden)

und

`~/.ssh/id_dsa.pub`
(öffentlicher Schlüssel; darf von jedem gelesen werden)

abgelegt.

Der öffentliche Schlüssel kann dann auf die Zielsysteme an die Datei

`~/.ssh/authorized_keys`

des entsprechenden Zielbenutzers angehängt werden:

```
echo id_dsa.pub >> ~/.ssh/authorized_keys
```

Beim nächsten Einloggen auf dem Zielsystem wird dann nach der Passphrase gefragt (Fallback ist Passwort).

ssh–agent

Um nicht bei jedem Einloggen die Passphrase eingeben zu müssen (was gerade bei automatisch ablaufenden Aktionen auch gar nicht möglich ist) kann man den SSH-Agenten verwenden.

Beim Arbeiten von einem X-Arbeitsplatz aus kann man die Verwendung des SSH-Agent in der Datei

`~/.xsession`

durch Setzen der dort enthaltenen Variablen

```
use_ssh="yes"
```

und

```
sshagent="yes"
```

einstellen.

Nach dem Neustart von X muss dann lediglich beim Aufruf des Kommandos

```
ssh-add ~/.ssh/id_dsa
```

die Passphrase einmal eingegeben werden.

Danach ist das Einloggen auf allen Rechnern, auf denen der *Public Key* des Benutzers auf dem Arbeitsplatzrechner entsprechend über die Datei `authorized_keys` bekannt ist, ohne Eingabe von Passphrase und Passwort möglich.

Dieses Verfahren eignet sich allerdings nicht zur Verwendung mit Serverdiensten, da diese normalerweise nicht unter X laufen. Es kann aber entsprechend angepasst werden indem man den SSH-Agenten aus einer Shell heraus startet. Subprozesse

dieser Shell können dann den Agenten benutzen und sich auf die entsprechenden Rechner ohne die sonst manuell notwendige Passworteingabe über das ssh-Protokoll einloggen.

Dieses wird durch das Ausführen des Skriptes `sc_ssh_agent.sh` (aus dem Paket `sc_tools`) erreicht:

```
source /usr/bin/sc_ssh_agent.sh
```

Einige SmartClient-Programme, z.B. `leases2ldap` greifen auf diesen Mechanismus zurück.

1.5.3 Network Time Protocol (NTP)

Um auf allen Rechner eine einheitliche Zeit zu garantieren, sollte der Zugriff auf einen Zeitserver per NTP konfiguriert werden. Falls kein Zeitserver zur Verfügung steht und ein Zugriff auf das Internet hierfür nicht möglich oder erwünscht ist, sollte einer der zentralen SmartClient Server zum Zeitserver auserkoren werden. Damit wird gewährleistet, dass alle Rechner die selbe Zeit haben. Es ist dann aber nicht gewährleistet, dass dies die richtige Uhrzeit ist.

Dazu wurde im LDAP ein `Service`-Eintrag `ntp` erzeugt. Durch diesen wird der DNS Name `ntp.server.smartclient.de` einer IP Adresse zugeordnet.

Auf allen Servern wird durch

```
chkconfig xntpd on
```

dafür gesorgt, dass der Zeitserver beim Booten des Rechners automatisch gestartet wird.

Auf den NTP-Clients muss zusätzlich der Eintrag

```
server ntp
```

in die Datei `/etc/ntp.conf` eingefügt werden.

1.5.4 LDAP Client

Die Benutzerdaten werden im zentralen LDAP-Verzeichnis gepflegt. Damit auf allen Servern die Authentifizierung über diese Daten möglich ist, müssen folgende Einstellungen vorgenommen werden:

Die Konfiguration kann komfortabel über YaST2 durchgeführt werden. Dazu muss zunächst das Paket `yast2-ldap-client` installiert werden. Anschließend können über `yast2` → `Netzwerkdienste` →`LDAP-Client` folgende Einstellungen gemacht werden:
„LDAP base dn“, die „LDAP-Wurzel“, wird dabei auf

`ou=user,o=smartclient,c=de`

gesetzt, weil nur unterhalb davon Benutzer definiert sind. Möchte man bei allgemeinen LDAP Suchanfragen den ganzen Baum miteinbeziehen, kann man zur Vereinfachung die Wurzel

`o=smartclient,c=de`

verwenden.

Die LDAP Server werden mit ihrer IP Adresse angegeben (im Beispiel 10.1.110.1 und 10.1.110.2), da es sonst beim Ausfall des DNS Servers zu einem Timeout von 30 Sekunden kommt.

Soll der Zugriff auf den Server verschlüsselt (TLS/SSL) erfolgen, muss die entsprechende Option ausgewählt werden (der Server muss auch dementsprechend konfiguriert sein).

Die Datei `/etc/openldap/ldap.conf` sollte anschließend folgende Einträge enthalten:

```
BASE ou=user,o=smartclient,c=de
HOST  10.1.110.1 10.1.110.2
```

Damit die Benutzerauflösung wirklich über LDAP erfolgt, muss die Datei `/etc/nsswitch.conf` folgende Zeilen beinhalten:

`passwd: files ldap`

`shadow: files ldap`

`group:  files ldap`

Die ursprünglichen Einträge für `passwd, shadow und group` müssen auskommentiert werden.

Auch diese Anpassung wird bei der Verwendung von Yast2 zur Konfiguration automatisch durchgeführt.

Zum Testen dieser Funktionalität sollte man den *Name Service Caching Daemon* (nscd) stoppen (`rcnscd stop`) und dann mit dem Kommando `id` ***Benutzername*** überprüfen, ob die Benutzerauflösung funktioniert.

1.5.5 Konfiguration der allgemeinen SmartClient-Pakete

sc_base

In der Konfigurationsdatei `/etc/smartclient/base` müssen die Einträge

```
ALL_LDAPHOST="ldap"
ALL_LDAPBASE="o=smartclient,c=de"
```

gesetzt werden. Der Rest kann unverändert bleiben.

Für die LDAP- und DNS-Server empfiehlt es sich, statt des DNS-Namens „ldap“ die IP-Adresse einzutragen. Alternativ kann der Name „ldap“ auch in der Datei `/etc/hosts` definiert werden.

Die Skripte `ldap2dns.pl` und `ldap2dhcp.pl` benutzen mittlerweile den Konfigurationsparameter `ALL_LDAPSERVERS`:

```
ALL_LDAPSERVERS="ldap01,rw,1;ldap02,ro,1"
```

Dies dient der Verteilung der Zugriffe auf mehrere LDAP-Server. Voraussetzung ist, dass eine korrekte Replikation der LDAP-Server aufgesetzt wurde. Es können beliebig viele LDAP-Server eingetragen werden. Bei einer Master/Slave Replikation darf nur der Master als *rw* („readwrite“, d.h. der Server unterstützt LDAP-Schreiboperationen) angegeben werden, handelt es sich um einen Slave, so muss *ro* (read only) eingetragen sein, da bei der Master-Slave-Replikation der Slave-Server nur LDAP-Leseoperationen unterstützt. Falls die LDAP-Server eine Master/Master Replikation unterstützen (z.B. Novell NDS), können auch mehrere Server mit *rw* konfiguriert werden.

Durch die Angabe der Priorität (im Beispiel bei beiden Servern eine „1“) kann die Verteilung der Anfragen zusätzlich beeinflusst werden.

Logrotate

Logfiles werden im Verzeichnis `/var/log/` gespeichert. Die für das SmartClient Framework spezifischen Logfiles werden in dem Unterverzeichnis `/var/log/smartclient/` abgelegt. In diesem Verzeichnis legen z.B. das Script `leases2ldap` sowie der Passwort-Generator ihre Log-Files ab.

Da diese Nachrichten recht umfangreich werden können, empfiehlt es sich, den `logrotate`–Mechanismus einzusetzen:

In `/etc/logrotate.conf` bzw. den Dateien in `/etc/logrotate.d/` wird definiert, was mit Logfiles geschehen soll, die eine gewisse Größe überschreiten. `logrotate`

wird automatisch einmal täglich durch `cron` aufgerufen, so dass die entsprechenden Dateien stetig überwacht werden.

Im vorliegenden Fall sollte eine Konfigurations-Datei `smartclient` im Verzeichnis `/etc/logrotate.d/` hinzugefügt werden:

```
/var/log/smartclient/* {
    compress
    dateext
    maxage 365
    rotate 99
    size=+1024k
    create 644 root root
    notifempty
    missingok
    copytruncate
}
```

Diese Datei legt fest, dass alle Dateien im `smartclient`-Unterverzeichnis komprimiert und mit Tagesdatum versehen gespeichert werden sollen, sobald sie die Größe von 1 MByte (1024 KByte) überschreiten. Danach wird eine neue Datei angelegt, mit den Rechten `o=rw,go=r (644)` und dem Besitzer *root* und der Gruppe *root*. Es werden maximal 99 Kopien einer Datei aufbewahrt, und alle Dateien, die älter als 365 Tage sind, werden gelöscht. Keiner der Dienste wird neu gestartet.

1.6 Installation der zentralen Server

1.6.1 Server 1 (LDAP Master, bind, dhcp)

LDAP Server (Master (read/write))

Das Paket `sc_ldap` muss installiert werden. Dieses enthält sowohl die Konfigurationsdateien für LDAP-Master sowie LDAP-Slave.

Die durch das Paket installierte Datei `/etc/openldap/slapd.conf.master` muss an die Stelle `/etc/openldap/slapd.conf` kopiert und angepasst werden. Hier muss auch das Replikationsverhalten eingestellt werden.

Durch

```
chkconfig ldap on
```

wird dafür gesorgt, dass der LDAP-Server beim Booten des Systems automatisch gestartet werden.

Mit dem Kommando `rcldap start` kann der LDAP-Server direkt gestartet werden.

Zum Zeitpunkt der Installation ist nur ein Grundgerüst (Skeleton) der Daten bekannt. Das Skeleton liegt als Beispiel für die Wurzel `o=smartclient,c=de` in der Datei

`/usr/share/doc/packages/sc_ldap/sc_skeleton.ldif`

vor.

Nach dem Kopieren dieses Grundgerüsts nach `/etc/openldap/smartclient.ldif` müssen durch Suchen/Ersetzen in einem Texteditor kundenspezifische Anpassungen vorgenommen werden. Die Anpassungen betreffen hauptsächlich den Austausch der *rootdn* von o=smartclient,c=denach `o=ihre-organisation,c=de`.

Weiterhin müssen die Dateien `sc_slapd_master.conf` und `sc_ldap.conf` nach `slapd.conf` bzw. `ldap.conf` kopiert werden. In der Datei `slapd.conf` muss noch die Zeile

```
include /etc/openldap/schema/smartclient-objectclasses.schema
```

in die Include-Section mit aufgenommen werden.

Danach kann das initiale Einspielen des Grundgerüsts an LDAP Daten mit dem Befehl

```
ldapadd -h ldap -D cn=admin,o=smartclient,c=de -W -x -f /etc/openldap/
smartclient.ldif
```

erfolgen (einspielen der Datei `smartclient.ldif` (`-f`) auf den Rechner „ldap“ (`-h`) als Benutzer *admin* (`-D`) und dabei nach dem Passwort fragen (`-W`) mit einfacher Authentisierung (`-x`)).

ldap2dns

erzeugt aus den LDAP Daten die DNS Konfiguration. Ggf. wird der DNS durch das Kommando `ndc` zum neu laden von Zonen-Dateien veranlasst.

Der Kopf der Zonendateien wird durch die Datei `/var/named/ldap_generated/dns-zonefile.header` definiert. Diese muss an das entsprechende Netzwerk angepasst werden.

Die Datei `named.conf` muss angepasst werden. Dazu kopiert man zunächst das Beispiel aus dem Paket `sc_ldap2dns` nach `/etc/named.conf` und ändert dann ggf. die Einträge für die `forwarder` etc.:

```
cp /usr/share/doc/packages/sc_ldap2dns/named.conf.master /etc/named.
conf
```

Die dort in der letzten Zeile inkludierte Datei wird erst von `ldap2dns` erzeugt, weshalb hier zunächst die ordnungsgemäße Konfiguration des LDAP überprüft werden muss.

Durch

```
chkconfig named on
```

wird dafür gesorgt, dass der Name-Server beim Booten des Systems automatisch gestartet wird.

`ldap2dns` selbst wird durch `leases2ldap` (siehe Kapitel 1.6.1 auf der nächsten Seite) gestartet.

`ldap2dns` läuft unter der Identität des Benutzers *dnsadmin*. Dieser Dienst darf nicht als *root* gestartet werden, da sonst das Überschreiben von Dateien nicht mehr funktioniert.

ldap2dhcp

erzeugt aus den LDAP-Daten die DHCP-Konfiguration. Ggf. wird der DHCP-Daemon neu gestartet und das Kommando zum Neustart des DHCP-Daemons auf anderen Servern zurückgegeben.

Die `dhcpd.conf` wird im Verzeichnis `/etc/smartclient/dhcpd/` erstellt. Deshalb muss die Standardposition dieser Konfigurationsdatei auf diese Position verlinkt werden. Dies erledigt folgendes Kommando:

```
ln -sf /etc/smartclient/dhcpd/dhcpd.conf /etc/dhcpd.conf
```

Sollen bestimmte Teile statisch gehandhabt werden, müssen diese in der Datei `etc/smartclient/dhcpd/dhcpd.conf.header` eingetragen werden. Dies betrifft z.B. die Netzwerke, für die der DHCP-Server aktiv sein soll, die aber nicht im LDAP gepflegt werden.

Der Master-Server stellt optional auch dynamische IP-Bereiche zur Verfügung. Nur dadurch wird die Erkennung neuer Rechner durch `leases2ldap` möglich (siehe Kapitel 1.6.1 auf der nächsten Seite).

Das Netzwerkinterface, auf dem der DHCP-Server aktiv sein soll, muss in die Datei `/etc/sysconfig/dhcpd` (Variable `DHCPD_INTERFACE`) eingetragen werden.

Durch

```
chkconfig dhcpd on
```

wird dafür gesorgt, dass der DHCP-Server beim Booten des Systems automatisch gestartet wird.

`ldap2dhcp` selbst wird durch `leases2ldap` (siehe Kapitel 1.6.1) gestartet.

`ldap2dhcp` läuft unter der Benutzer-Identität des Benutzers *dhcpadmin*.

leases2ldap

überprüft regelmäßig die Datei `/var/lib/dhcp/dhcpd.leases` nach neuen Rechnereinträgen. In diese Datei werden vom DHCP-Server alle Clients eingetragen, die eine Adresse aus dem dynamischen Adress-Bereich zugewiesen bekommen haben.

`leases2ldap` ruft die beiden nachfolgend beschriebenen Skripte `ldap2dns` und `ldap2dhcp` in vorgegebenen Zeitabständen auf, welche über den Parameter `LEASES2LDAP_TIMEOUT` in der Datei `/etc/smartclient/leases2ldap` festgelegt werden. Zusätzlich werden beide Skripte ausgeführt, falls die Datei `/var/lib/dhcp/dhcpd.leases` verändert wurde.

`leases2ldap` läuft unter der Benutzer-Identität von *root*.

Zum Anlegen neuer Rechner im LDAP-Verzeichnis benötigt `leases2ldap` Schreibzugriff auf den LDAP-Server.

Dazu ist im LDAP ein entsprechender Admin-Benutzer mit Passwort eingetragen (`cn=leases2ldap,ou=admins,ou=global,o=`smartclient`,c=`de).

Dieses muss mit den Eintragungen in der Konfigurationsdatei `/etc/smartclient/leases2ldap` übereinstimmen.

Da das Script `leases2ldap` auch Aktionen auf anderen Rechnern (DNS-Slave, DHCP-Slave) steuert, benötigt es passwortlosen Zugriff auf diese Systeme.

Dazu muss über die Kommandozeile des LDAP-Master-Servers ein Schlüsselpaar erzeugt werden, wobei der **public**-Key jeweils zu *dnsadmin* und *dhcpadmin* des DNS/DHCP-Slave-Servers kopiert wird:

```
ssh-keygen -t dsa

scp /root/.ssh/id_dsa.pub dhcpadmin@DHCPSLAVE:.ssh/ldapkey.id_dsa.pub
scp /root/.ssh/id_dsa.pub dnsadmin@DNSSLAVE:.ssh/ldapkey.id_dsa.pub

ssh dhcpadmin@DHCPSLAVE
cat .ssh/ldapkey.id_dsa.pub >> .ssh/authorized_keys
exit

ssh dnsadmin@DNSSLAVE
cat .ssh/ldapkey.id_dsa.pub >> .ssh/authorized_keys
```

```
exit
```

sc_password_generator

Workstations müssen die Möglichkeit besitzen, schreibend auf ihren eigenen Eintrag im LDAP-Verzeichnis zuzugreifen (z.B. für die Inventarisierung bzw. um ihre aktuelle Image-Versionsnummer einzutragen). Eine Workstation kann über ihre IP eindeutig einem LDAP-Eintrag zugeordnet werden. Leider kann die Authentifizierung beim Zugriff auf den LDAP-Server nicht direkt über die IP-Adresse erfolgen, sondern nur anhand eines Passwortes. Dieses Passwort kann einer Workstation aber im Falle eines Updates noch gar nicht bekannt sein.

Der Mechanismus zum Überwinden dieser Einschränkung sieht folgendermaßen aus:

- Die Workstation baut eine Verbindung zum Server `password` auf.
- Der Server `password` überprüft, von welcher Quell-IP die Anfrage kommt und sucht den passenden Workstation-Eintrag im LDAP-Verzeichnis. Für diesen Eintrag erzeugt der Server ein zufälliges Passwort, trägt es im LDAP-Objekt der Workstation ein und sendet es an die aufrufende Workstation zurück.
- Die Workstation erhält ihr Passwort und kann damit schreibend auf den LDAP-Server zugreifen.

Zum Erzeugen von neuen Workstation-Passwörtern im LDAP-Verzeichnis benötigt `password_generator` Schreibzugriff auf den LDAP-Server.

Im LDAP-Verzeichnis existiert ein entsprechender Admin-User mit Passwort (`cn=passgenerator,ou=admins,ou=global,o=`smartclient`,c=de`). Dieses Passwort muss mit den Eintragungen in der Konfigurationsdatei `/etc/smartclient/password_generator` übereinstimmen.

Die Portnummer für den Zugriff auf den Passwortserver ist in `/etc/smartclient/base` durch die Variable `ALL_PW_PORT` definiert. Die Standardeinstellung ist Port 6200. Dieser Eintrag muss beim Passwort-Server, den Clients und dem Rechner zum Erzeugen von Netboot-Dateien identisch sein.

Durch `chkconfig /etc/init.d/sc_password_generator on` wird dafür gesorgt, dass der Password Generator auch beim nächsten Booten gestartet wird.

tftpboot

TFTP (Trivial File Transfer Protocol) ist eine abgespeckte FTP-Version, die keine Benutzerauthentisierung vorsieht und über UDP, nicht TCP, läuft. Dieser Dienst

wird direkt von bootfähigen Netzwerkkarten unterstützt, um die initiale Software vom Server zu laden.

In diesem Kapitel wird beschrieben, welche Voraussetzungen geschaffen werden müssen, damit diese Software später von den Workstations per TFTP abgerufen werden kann.

Der Benutzer *tftpadmin* ist durch das Paket `sc_base` im System vorhanden. Damit man neue Netboot-Software für diesen Benutzer einspielen kann, muss er ein Passwort zugewiesen bekommen (`passwd tftpadmin`), oder es muss ein ssh-Key (vergl. Kapitel 1.5.2 auf Seite 9) in seinem Homeverzeichnis abgelegt sein.

Das entsprechende Datenverzeichnis wird mit folgenden Befehlen erzeugt:

```
mkdir -p /var/lib/smartclient/tftpboot/
chown -R tftpadmin.tftp /var/lib/smartclient/tftpboot/
```

Damit der TFTP-Daemon beim nächsten Booten gestartet wird, muss er mit

```
chkconfig atftp on
```

aktiviert werden.

Die Daten im TFTP Verzeichnis bestehen aus

- dem Netzwerk-Bootloader und der entsprechenden Konfiguration (`pxelinux.0`) aus dem `syslinux` Paket,
- dem Bootkernel,
- der initialen RAM Disk (`initrd.gz`).

Die Initiale Ram-Disk ist eine SmartClient-Komponente und für den Kunden angepasst.

1.6.2 Server 2 (LDAP Slave, bind Slave, dhcp)

LDAP Server (Slave)

Das Paket `sc_ldap` muss installiert werden. Dieses enthält sowohl die Konfigurationsdateien für den LDAP-Master- sowie den LDAP-Slave-Server.

Die Datei `/etc/openldap/slapd.conf.slave` muss an die Stelle `/etc/openldap/slapd.conf` kopiert und angepasst werden.

Durch

```
chkconfig ldap on
```

wird dafür gesorgt, dass der LDAP-Server beim Booten des Systems automatisch gestartet wird.

DNS-Slave

Für den Datentransfer zwischen DNS-Master und DNS-Slave wird auf Standard-Funktionen von DNS zurückgegriffen (mit etwas Hilfe von `ssh`, um neue Zonen auf dem Slave bekannt zu machen). Deshalb wird das Skript `ldap2dns` auf dem Slave *nicht* benötigt. Das Paket `sc_ldap2dns` ist aber trotzdem hilfreich und sollte daher installiert werden, da es auch die vorbereitete Konfigurationsdatei von `bind` enthält.

Diese wird mit dem Kommando

```
cp /usr/share/doc/packages/sc_ldap2dns/named.conf.master /etc/named.conf
```

an die richtige Stelle kopiert. An dieser Datei müssen noch die Anpassungen für einen Slave-Server vorgenommen werden.

Wesentlich ist dabei die Zeile

```
include "/var/named/ldap_generated/slave.zones";
```

am Ende der Datei `/etc/named.conf`.

Der DNS Slave erhält seine DNS-Daten durch automatische Zonentransfers vom Master zum Slave. Wenn neue Zonen hinzu kommen, überträgt der Master die Datei

```
/var/named/ldap_generated/slave.zones
```

vom Master zum Slave (dafür ist es erforderlich, dass der unter Kapitel 1.6.1 auf Seite 19 beschriebene ssh-Zugriff eingerichtet ist).

Durch

```
chkconfig named on
```

wird dafür gesorgt, dass der Name-Server beim nächsten Bootvorgang automatisch gestartet wird.

Das Kommando `sudo` muss derart konfiguriert werden, dass der Benutzer `dnsadmin` das Kommando `ndc` ausführen kann. Dieses geschieht durch den Aufruf von `visudo`, wobei folgende Zeile in die Konfigurationsdatei eingefügt wird:

```
dnsadmin    server1=NOPASSWD:/etc/init.d/named reload
```

ldap2dhcp

sollte genau wie auf dem Master (siehe Kapitel 1.6.1 auf Seite 18) konfiguriert werden.

`ldap2dhcp` wird von `leases2ldap` des *Master*-Rechner aufgerufen. Für den Slave-Modus wird es mit folgenden Parametern aufgerufen:

`-secondarymode` sorgt dafür, dass `ldap2dhcp` nicht selbst wieder sekundäre Server aufrufen will,

`-nodynrange` legt fest, dass selbst keine dynamische IP-Range für unbekannte Rechner zur Verfügung gestellt wird,

`-notauthoritative` regelt, dass bei bekannten Rechnern die entsprechende IP-Adresse übermittelt wird; unbekannten Rechner werden vom sekundären DHCP-Server einfach ignoriert (standardmäßig würde der DHCP-Server bei einer Anfrage nach einer IP-Adresse, die er nicht bedienen kann, zurückliefern, dass er nicht zuständig ist (*authoritative*); diese Rückmeldung kommt hier nicht; nur der Master-DHCP ist *authoritative*)

Zusätzlich muss das Kommando `sudo` so konfiguriert werden, dass es dem Benutzer *dhcpadmin* ermöglicht wird, das Startskript `/etc/init.d/dhcpd` auszuführen. Dieses geschieht durch den Aufruf von `visudo`, wobei folgende Zeile eingefügt wird:

```
dhcpadmin    server1=NOPASSWD:/etc/init.d/dhcpd restart
```

sc_password_generator

sollte zur Ausfallsicherheit auch auf diesem Server eingerichtet werden. Die Einrichtung erfolgt genau wie im Kapitel 1.6.1 auf Seite 20 beschrieben.

tftpboot

sollte zur Ausfallsicherheit auch auf diesem Server eingerichtet sein. Die Einrichtung erfolgt genau wie unter Kapitel 1.6.1 auf Seite 20 beschrieben.

1.6.3 Server 3 (Rsync, NFS)

sc_rsync

setzt auf dem Paket `rsync` auf und stellt eine Umgebung her, um per `rsync` Images verteilen zu können.

Im `sc_rsync`-Paket ist die Konfigurationsdatei `/etc/smartclient/rsyncd.conf` enthalten. Damit sie vom Rsync Daemon verwendet wird, muss mit dem Befehl

```
ln -sf /etc/rsyncd.conf /etc/smartclient/rsyncd.conf
```

ein entsprechender Link angelegt werden.

Das Kommando `chkconfig rsync on` sorgt dafür, das der Rsync-Daemon beim nächsten Booten automatisch gestartet wird.

rsync.client2server stellt einen Wrapper zur Verfügung, um eine Client-Installation zum Server zu kopieren. Diese Kopie kann dann später als Image eingesetzt werden.

Da der Referenz-Rechner und die verwendete Exclude-Liste normalerweise feststehen, können zur Vereinfachung weitere spezielle Wrapper Skripte für bestimmte Fälle angelegt werden.

Das Paket `sc_rsync` enthält das Beispiel Skript `/usr/share/doc/packages/sc_rsync/rsync.client2server.pc00257`. In diesem ist der Rechner mit `excludelist.full` vorkonfiguriert. Dieses Skript kopiert und passt man an die eigene Umgebung an. Das Skript wird am Besten nach `/usr/local/sbin/` kopiert.

Ist dies geschehen, kann ein Image eines Clients erzeugt und auf den Server gespielt werden. Dort kann es dann als Referenz-Image dienen. Weitere Details sind im Kapitel 3.10 auf Seite 72 beschrieben.

NFS

Installations-Server: Der NFS-Server dient auch als Datenquelle für manuell aufzusetzende Rechner.

Dazu werden die SuSE DVDs bzw. CD-ROMs mit den entsprechenden Updates und evtl. kundenspezifische Software auf den Server kopiert.

Diese liegen unterhalb von `/srv/dist/` und werden in der Datei `/etc/exports` mit folgender Zeile freigegeben:

```
/export/software 10.0.0.0/255.0.0.0(ro,insecure)
```

geeigneter als http server

Home: Dieser Server dient auch als NFS-Server für die Benutzer-Verzeichnisse. Dies ist aber eigentlich Aufgabe der dezentralen Server und deshalb dort genauer beschrieben (siehe 1.7.1 auf Seite 26).

1.6.4 Server 4 (Syslog, Admin, Backup)

syslog

Damit dieser Rechner syslog-Meldungen von anderen Rechnern in Empfang nehmen kann, muss in der Datei `/etc/sysconfig/syslog` die Variable `SYSLOGD_PARAMS="-r"` gesetzt werden. Im LDAP muss dieser Rechner als Service `loghost.server.smartclient.de` eingetragen sein.

Backup der Homeverzeichnisse:

Das Skript `/etc/cron.daily/sc_backuphomes.sh` sichert die Home-Verzeichnisse aller Fileserver nach `/home`. Wenn weitere Fileserver hinzukommen, müssen diese im Skript ergänzt werden. Durch die Position des Skriptes in `/etc/cron.daily/` wird das Skript einmal pro Tag (um 0:14 Uhr) ausgeführt. Das Backup wird über `rsync` realisiert.

Achtung: Wenn der Benutzer Dateien aus seinem Verzeichnis löscht, werden diese auch aus dem Backup gelöscht. Man benötigt also zusätzlich noch eine permanente Sicherung (z.B. Bandlaufwerk).

Da das Backup Skript per Cron Job läuft, ist eine Passwort- oder Passphrase-Eingabe nicht möglich. Es wird daher mit einen ssh-Key mit leerer Passphrase gearbeitet. Dadurch erhält der Nutzer *root* dieses Rechners Root-Zugriff auf die Fileserver ohne Passworteingabe und auch ohne ssh-Agenten.

Durch `ssh-keygen -t dsa` wird ein ssh-Schlüssel mit leerer Passphrase erzeugt. Der Inhalt der Datei `~/.ssh/id_dsa.pub` muss dann an die Datei `/root/.ssh/authorized_keys` der Dateiserver, von denen ein Backup durchgeführt werden soll, angehängt werden. Siehe auch Kapitel 1.5.2 auf Seite 11.

1.7 Installation der dezentralen Arbeitsgruppen-Server

Für die verschiedenen Lokationen sind Arbeitsgruppen-Server vor Ort vorgesehen. Diese dienen als Datei– (NFS) und Druckserver (CUPS). In der Hauptstelle wird diese Funktionalität von den SmartClient Servern (speziell Server 3) mit übernommen. Um den Text nicht unnötig zu verkomplizieren, wird im folgenden trotzdem von dezentralen Servern gesprochen.

1.7.1 Verschiedene Server

Router

Für den Zugriff aufs Netzwerk muss der Rechner als Router eingerichtet werden. Nach dem Einbinden der Netzwerk-Interfaces und dem Setzen der Standardroute (über YaST) muss die Variable `IP_FORWARD="yes"` in der Datei `/etc/sysconfig/sysctl` gesetzt werden.

DHCP-Relay

Damit die Endgeräte der Lokation eine IP Adresse zugeordnet bekommen, muss ein DHCP-Relay eingerichtet werden, da DHCP-Pakete nicht geroutet werden. Das DHCP-Relay nimmt Broadcast DHCP Anfragen entgegen und sendet sie als gerichtete IP-Pakete weiter an die zentralen DHCP-Server.

Zunächst muss das Paket `dhcp-relay` installiert werden. Die Konfiguration wird über die Datei `/etc/sysconfig/dhcrelay` vorgenommen. Hier müssen die DHCP-Server und die zu verwendenden Interfaces eingetragen werden. Damit Anfragen an beide DHCP-Server weitergeleitet werden, sollten auch beide Server eingetragen werden, z.B.

```
DHCRELAY_SERVERS="10.1.110.1 10.1.110.2"
```

So werden Anfragen an beide DHCP Server geschickt. Der DHCP Server, der als erstes antwortet, wird verwendet.

Damit der DHCP Relay Daemon beim nächsten Booten auch automatisch startet, werden mit `chkconfig dhcrelay on` die entsprechenden Links gesetzt.

NFS

Die Benutzerverzeichnisse werden auf den dezentralen Dateiservern gespeichert. Daher müssen die entsprechenden Home-Verzeichnisse für die Benutzer auf dem Dateiserver ihres Standortes angelegt werden.

Die Datei `/etc/export` muss entsprechend angepasst werden (siehe `man 5 exports`).

So sind die Benutzerverzeichnisse nur durch den Plattenplatz begrenzt. Wenn der Platzbedarf pro Benutzer feststeht, sollte eine entsprechende Quota-Umgebung geschaffen werden, wodurch jedem User eine maximale Größe an Festplattenplatz zugeordnet wird.

CUPS (Druckserver)

Der Cups Server muss installiert und konfiguriert werden.

Die Einrichtung von Druckern wird im Kapitel 3.5 auf Seite 59 beschrieben.

tftpboot

Zur Minimierung der benötigten WAN Bandbreite bietet es sich an, auch dezentral TFTP Server einzusetzen.

Die Installation erfolgt dabei wie im Kapitel 1.6.1 auf Seite 20 beschrieben.

Rsync

Die Einrichtung des Rsync-Dienstes erfolgt wie in Kapitel 1.6.3 auf Seite 23 beschrieben.

Referenz Images werden hier aber nicht von einer Workstation erzeugt, sondern vom zentralen Rsync Server zu allen dezentralen Servern kopiert.

Dafür müssen auf dem zentralen Rsync Server die dezentralen Server in der Datei

`/etc/smartclient/rsync`

in die Variable

`RSYNC_SLAVE_SERVERS`

aufgenommen werden.

Danach kann durch das Kommando `distributeimage.sh` (aufgerufen auf dem zentalen Rsync-Server) alle (oder das angebene) Image zu den dezentralen Rsync Servern kopiert werden (siehe auch Kapitel 3.10.3 auf Seite 77).

2 Client-Installation und -Konfiguration

Mit dem Installationsverfahren des SmartClient-Konzepts können nahezu beliebige Installationen von einem Referenz-PC auf viele Rechner verteilt und anschließend zentral administriert werden.

Die Vorgehensweise zur Einrichtung einer neuen Client-Installation ist dabei wie folgt:

1. Grundinstallation des Referenz-PCs mit allen gewünschten Client-Anwendungen und notwendigen Netzwerkdiensten in der Paketauswahl
2. Installation der SmartClient-spezifischen Zusatzpakete
3. Konfiguration des Clients
4. Erzeugung des Client-Images auf dem Installations-Server
5. Test der Installation durch Anschluss eines neuen Clients

Dieses wird in den folgenden Abschnitten detailliert dargestellt.

2.1 Grundinstallation eines Referenz-PCs

2.1.1 Partitionierung

Es wird davon ausgegangen, dass ein Client im Falle eines Dateisystemfehlers einfach neu installiert wird, d.h. lokal werden keine relevanten Daten abgelegt.

Für die lokale System-Partition des Referenz-PCs kann ein `ext2`- oder ein `ext3`-Dateisystem verwendet werden. Hinzu kommt eine lokale Swap-Partition zur Erweiterung des verfügbaren Hauptspeichers.

Die für die vorgenommene Installation empfohlene Partitionierung (siehe `/etc/fstab`) sieht wie folgt aus:

swap		1 G
/	ext3	4 G

Die Partitionsgrößen für die zu klonenden Rechner können im LDAP festgelegt werden (siehe Kapitel 3.10.2 auf Seite 74). Eine nachträgliche Veränderung im LDAP führt bei einem Update zu einer Neuinstallation des Rechners.

2.1.2 Paketinstallation

Die Anzahl der installierten Pakete sollte so gering wie möglich gehalten werden. Dies minimiert die Anzahl der Supportfälle und sorgt für eine schnelle Installation der Rechner.

Deshalb sollte die Basis Paketauswahl möglichst klein, aber inklusive X-Windows sein.

Es empfiehlt sich, die Paketauswahl genau zu prüfen und nicht benötigte Pakete (wie z.B. `wvdial`, `ypbind`, isdn-Pakete) abzuwählen.

Folgende Pakete werden benötigt und sollten installiert werden:

`autofs`	: automatisches Einbinden von Netzwerklaufwerken
`cups-client`	: Zugriff auf CUPS Drucker Server
`xntp`	: Zeiteinstellung über den Zeitserver
`rfb`	: Bildschirm Spiegelung übers Netzwerk (zur Benutzerbetreuung)
`rsync`	: wird von SmartClient zur Generierung des Referenz Images benötigt
`bind-utils`	: wird vom ISC dhcp-client benötigt
`nss_ldap`	: Benutzerauthentisierung über LDAP
`pam_ldap`	: Benutzerauthentisierung über LDAP
`perl-ldap`	: wird von SmartClient benötigt
`perl-Net-IP`	: wird von SmartClient benötigt
`perl-Unix-Syslog`	: wird von SmartClient benötigt

SmartClient spezifische Pakete:

`sc_base`	: Verzeichnisstruktur und Konfigurationen
`perl-SmartClient`	: spezielle Perl-Module
`sc_mkwsconfig`	: Herstellung der Konfigurationsdaten für den LDAP Zugriff
`sc_hardware`	: Hardwareerkennung
`sc_ldap2authorizedkeys`	: Liest Public-Keys aus dem LDAP und speichert sie in `~/.ssh/authorized_keys`
`sc_rsyncDeltaImages`	: nachinstallieren von Delta Images
`sc_inventar2ldap`	: Ablegen von Client-Informationen im LDAP
`sc_print`	: Druckereinrichtung

Für die Funktion als *Caching-Server* werden zusätzlich noch folgende Pakete benötigt:
`cups-server`: Eigenständiger Zugriff auf die Drucker
`atftp` : TFTP-Server um andere Workstations booten zu lassen

2.2 Konfiguration

2.2.1 Verschiedenes

Anlegen von Benutzern

Da die Benutzerverwaltung zentral erfolgt, muss kein lokaler Benutzer existieren. Um Zugriffsmöglichkeit zu bieten, falls bei einem Netzwerkausfall die zentralen Server nicht erreichbar sind, kann zu diesem Zweck ein lokaler Benutzer angelegt werden.

Bei lokalen Benutzern im Zusammenspiel mit zentral verwalteten Benutzern muss die Lage der Heimatverzeichnisse beachtet werden. Diese werden bei zentral verwalteten Benutzern vom zentralen Server gemountet. Bei einem lokalen Benutzer sollte das Heimatverzeichnis lokal liegen und nicht in Konflikt mit den zu mountenden Heimatverzeichnissen der anderen Benutzer geraten.

Daher bietet es sich an, vor Anlegen eines lokalen Benutzers die Datei `/etc/default/useradd` so anzupassen, dass die Heimatverzeichnisse von neu anzulegenden Benutzern nicht in `/home` erzeugt werden, sondern z.B. in `/usr/local/home`:

```
HOME=/usr/local/home
```

Beim Anlegen eines Benutzers auf dem Client mittels `useradd` oder `yast` wird dem neuen Benutzer nun ein Heimatverzeichnis unterhalb von `/usr/local/home` zugewiesen.

dhcp-Client

Die Netzwerkkarte wird über YaST für die automatische Adress-Zuweisung per DHCP konfiguriert. Auch der Rechner-Name und Nameserver sollten per DHCP akzeptiert werden.

sc_base

In der Datei `/etc/smartclient/base` müssen die Einträge

```
ALL_LDAPHOST="ldap"
ALL_LDAPBASE=o=smartclient,c=de
ALL_NETWORK_INTERFACES="eth0"
```

eingetragen werden, wobei das Netzwerkinterface (im Beispiel `eth0`) entsprechend zu wählen ist. Alle übrigen Einträge können unverändert bleiben.

Startskripte deaktivieren

Das Starten von Diensten nimmt beim Booten der Clients Zeit in Anspruch. Um diese Zeit möglichst gering zu halten, sollte überprüft werden, welche der gestarteten Dienste auf den Clients wirklich notwendig sind.

Mit `chkconfig` erfolgt die Ausgabe aller installierter Startskripte und ihres Aktivierungs-Zustands („`off`“ bzw. „`on`“). So kann leicht überblickt werden, welche dieser Dienste aktiviert sind. Mit `chkconfig startskript off` kann das Starten eines Dienstes deaktiviert werden. So ist auf den Clients normalerweise kein LVM (Logical Volume Manager), kein RAID und kein Crypto-Dateisystem im Einsatz, so dass die Skripte `boot.lvm`, `boot.md` und `boot.crypto` deaktiviert werden können. Wenn kein Joystick und keine Soundkarten vorhanden sind, können auch `alsasound` und `joystick` ausgeschaltet werden.

Die Konfiguration kann auch über das YaST2-Modul (`yast` → `System` → `Runlevel-Editor` erfolgen.

mkwsconfig

`mkwsconfig` muss über das Kommando

```
rcsc_mkwsconfig start
```

gestartet werden.

Damit es bei jeden Booten startet, werden mit

```
chkconfig sc_mkwsconfig on
```

die entsprechenden Links in den Runlevel-Verzeichnissen gesetzt.

LDAP

Die Benutzerauthentisierung erfolgt über LDAP. Dafür muss der Zugriff auf den primären und auch sekundären LDAP Server eingerichtet werden.

Die Konfiguration erfolgt wie in Kapitel 1.5.4 auf Seite 13 beschrieben.

Darüber hinaus ist es vorteilhaft, wenn die LDAP-Server über ihre IP-Adressen anstatt mit ihren Namen angegeben werden, da sonst bei einem Ausfall der DNS-Server selbst ein lokales Einloggen als *root* nur mit erheblicher Zeitverzögerung möglich ist (30 Sekunden Timeout für die Nameserver-Abfrage).

Die Einstellungen können auch über `YaST2` vorgenommen werden.

Hardware-Erkennung

Bei der Einrichtung einer neuen Client-Konfiguration muss beachtet werden, dass das erzeugte Image nachher auch auf allen eingesetzten, unterschiedlichen Rechnern ohne manuelle Nachkonfiguration funktionsfähig ist.

Wenn unterschiedliche Hardware zum Einsatz kommt, ist die automatischen Hardware–Erkennung zur korrekten Konfiguration der Peripheriegeräte notwendig.

Da es sich um Desktop Rechner handelt, wird von folgender Ausstattung ausgegangen:

1. eine IDE Festplatte
2. eine Netzwerkkarte (Ethernet oder Token-Ring)
3. eine Grafikkarte

Die automatische Erkennung dieser Client-Komponenten wird durch das Paket `sc_hardware` abgebildet.

Die automatische Hardware-Erkennung muss rechtzeitig zu Anfang des Bootvorgangs aufgerufen werden und ohne Benutzerinteraktion vonstatten gehen.

Standardmäßig wird bei SuSE Linux Rechnern die automatische Hardware-Erkennung durch das `hwscan`-Startskript beim Booten durchgeführt. Dieses Skript wartet allerdings ggf. auf Benutzereingaben und muss daher mit dem Kommando

```
chkconfig hwscan off
```

deaktiviert werden.

Nach der Installation des `sc_hardware` Pakets müssen ein paar Änderungen am System vorgenommen werden. Damit keine (evtl. ungewollten) Änderungen automatisch am System durchgeführt werden, wurde darauf verzichtet, diese automatisch per RPM-Postinstall vorzunehmen.

sc_hardware_net In der Datei `/etc/init.d/network` muss die Variable `Required-Start` oder besser `Should-Start` (ab Suse Linux 9.1) um `sc_hardware_net` ergänzt werden.

Danach müssen die Kommandos

```
chkconfidg sc_hardware_net on
chkconfig network on
```

ausgeführt werden.

Damit wird das Netzwerk erst dann gestartet, wenn `sc_hardware_net` die entsprechende Hardware erkannt hat.

sc_hardware_X Die Grafikkartenerkennung basiert auf dem Suse–Tool `sax2`.

Zur Einrichtung muss in der Datei `/etc/init.d/xdm` die Variable `Required-Start` oder besser `Should-Start` (ab Suse Linux 9.1) um `sc_hardware_X` ergänzt werden.

Danach müssen die Kommandos

```
chkconfig sc_hardware_X on
chkconfig xdm on
```

ausgeführt werden.

Die Grafikkartenerkennung nimmt ca. 20 Sekunden in Anspruch. Außerdem wird die Bildschirmausgabe durch die Grafikkartenerkennung beeinflusst. Dies könnte den Anwender gegebenenfalls irritieren, weshalb bei einer Neuinstallation von Clients darauf hingewiesen werden sollte.

Aus diesem Grund wird die Grafikkartenerkennung normalerweise nicht bei jedem Neustart durchgeführt, sondern lediglich nach einem Update.

Geregelt wird dies über die Existenz der Datei `/tmp/sax_has_runned`.

Ist diese vorhanden, wird keine Erkennung durchgeführt. Das Löschen dieser Datei führt also zu einer Erkennung beim nächsten Start. Alternativ kann man die Erkennung auch manuell mit

```
rcsc_hardware_X force-start
```

erreichen.

Zusätzlich existiert in `/etc/smartclient/sc_hardware` die Variable `SC_HARDWARE_START`. Wird diese auf `ONBOOT` gesetzt, findet die Hardware-Erkennung für das X Window System bei jedem Boot-Vorgang statt.

Kapitel 3.4 auf Seite 59 beschreibt, wie mit kritischen Monitoren umgegangen werden kann.

sc_hardware_audio Um beim Vorhandensein von Soundkarten zu gewährleisten, dass diese korrekt angesprochen werden, gibt es das Skript `sc_hardware_sound`, welches Soundkarten beim Systemstart erkennt und automatisch konfiguriert.

Damit die Soundkarten-Aktivierung erst nach der Konfiguration durchgeführt wird, muss im Startskript `/etc/init.d/alsasound` die Variable `Required-Start` oder besser `Should-Start` (ab Suse Linux 9.1) um den Eintrag `sc_hardware_audio` ergänzt werden.

Mit

```
chkconfig sc_hardware_audio on
chkconfig alsasound on
```

werden die entsprechenden Startlinks gesetzt.

Nach Durchführung der Soundkarten-Erkennung legt das Skript `sc_hardware_sound` eine Datei `/var/tmp/sc_hardware_audio` an, deren Vorhandensein beim nächsten Bootvorgang verhindert, dass die Soundkarten- Erkennung durchgeführt wird. So wird diese Erkennung nur nach Updates ausgeführt. Soll eine Neu-Konfiguration der Soundkarte erzwungen werden, kann dies durch Aufruf (`rcsc_hardware_audio restart`) erreicht werden.

Anpassung des Boot-Loaders

Also Bootloader wird standardmäßig GRUB verwendet. Dieser startet ein graphisches Auswahl-Menü. Da beim Darstellen Bootloaders des die automatische Hardware-Erkennung der Clients noch nicht durchgelaufen ist, sollte das grafische Boote-Menü deaktiviert werden.

Dazu wird in der Datei `/boot/grub/menu.lst` die Zeile `gfxmenu ...` entfernt.

Weiterhin sollte der Timeout beim Booten, bevor der Default-Eintrag des Boot-Menüs aktiviert wird, heruntergesetzt werden. Der voreingestellte Wert beträgt 8 Sekunden. Über den Parameter `timeout` kann hierfür auch ein andere Wert gesetzt werden.

Netzwerklaufwerke

Für das Einbinden der Netzwerklaufwerke wie `/home` oder `/Gemeinsam` wird `autofs` verwendet, da dies resistenter gegen einen temporären Ausfall des NFS-Servers ist als das statische Mounten über einen Eintrag in der Datei `/etc/fstab`.

Hierfür ist `autofs` ab der Version 4 einzusetzen. Ab Suse Linux 10.0 ist dies Standard. Bei älteren Distributionen musste expliziet das Paket `autofs4` ausgewählt und entsprechenden Einstellungen vorgenommen werden.

Durch `chkconfig autofs on` wird dafür gesorgt, dass der Automounter beim Booten automatisch gestartet wird.

Die Konfigurationsdatei `/etc/auto.master` muss um die Einträge für die einzubindenen Laufwerke ergänzt werden. Im Beispiel müssen die lokalen Verzeichnisse `/home` und `/gemeinsam` existieren. Die Datei sieht dann wie folgt aus:

`/etc/auto.master`:

```
/net        /etc/auto.net
/home       /etc/auto.home
/gemeinsam  /etc/auto.gemeinsam
```

Die Konfigurationsdateien `/etc/auto.home` und `/etc/auto.gemeinsam` müssen angelegt werden.

Durch folgende Datei wird für jeden Benutzer automatisch sein Heimatverzeichnis gemountet (Voraussetzung: das Heimatverzeichnis auf dem File-Server liegt dort unter `/home/`*`Benutzername`*):

`/etc/auto.home`:

```
*       -fstype=nfs,hard,intr,nodev,nosuid,nolock,nonstrict,\
        rsize=8192,wsize=8192                    server:/home/&
```

Beim Zugriff auf z.B. `/home/tux` würde automatisch das Verzeichnis „`tux`“ (als Unterverzeichnis von `home`) als Mountpunkt erzeugt und das Verzeichnis vom Server dorthin gemountet. Erfolgt für einen bestimmten Zeitraum kein Zugriff mehr auf das Verzeichnis, z.B. weil der Benutzer sich abgemeldet hat, so wird ein umount durchgeführt.

Drucker

Das verwendete Drucksystem ist CUPS (siehe Kapitel 1.7.1 auf Seite 26).

Zentrale Druckdienste Eine feste Konfiguration der Drucker sollte nicht vorgenommen werden, da diese Konfiguration später auf allen Rechnern funktionieren soll. CUPS kann so konfiguriert werden, dass es selber im Netzwerk nach verfügbaren CUPS Servern sucht (Browse). Alternativ kann auch ein fester Druckserver eingetragen werden. Über diesen Namen sollte jeder Rechner seinen nächstgelegenden Druckserver finden. Standard Name ist `cups`.

Einen Mischung zwischen eigenen Browsen und festen Druckerver ist über die Konfigurationsoption

```
BrowsePoll cups:631
```

in der Datei `/etc/cups/cupsd.conf`, ggf. bei gleichzeitige Deaktiverung des allgemeinen Browsen, aktivert werden.

Dadurch versucht der Client sich mit dem Server `cups` zu verbinden und seine Drucker zu übernehmen.

sc_ldap2lpadmin erläutern

Abschnitt über Konfiguration von lokalen Druckern

Standarddrucker Für die automatische Konfiguration eines Druckers pro Client muss die Variable `DEFAULT_PRINTER` aus der Datei `/etc/sysconfig/printer` entfernt werden. Ansonsten würde nur dieser verwendet und nicht der Standarddrucker des Netzwerkes.

Zuletzt muss noch das Skript `sc_ldap2defaultprinter` nach `/etc/init.d/` kopiert und das Kommando

```
chkconfig sc_ldap2defaultprinter on
```

ausgeführt werden.

Die Konfiguration von Standarddruckern ist im Kapitel 3.5.4 auf Seite 62 beschrieben.

Soll ein ähnliches Verhalten wie bei Windows erreicht werden, wo jeder Benutzer die gewünschten Drucker auf seinem System selbst konfiguriert, kann die Einrichtung von lokalen Druckerwarteschlangen für Benutzer freigegeben werden. Dazu muss in `/etc/cups/cupsd.conf` der Eintrag `SystemGroup` auf die gewünschte Gruppe gesetzt werden.

Syslog

Die Log-Meldungen der Clients sollen auf dem zentralen Log-Host gespeichert werden. Dabei sollen nur wichtige Meldungen dort abgelegt werden. Der folgende, zusätzliche Eintrag in `/etc/syslog.conf` sorgt dafür, dass alle Meldungen der Priorität „error“ oder schlimmer an den Log-Host weitergeleitet werden:

```
*.err @loghost
```

So sind alle wichtigen Meldungen der Clients zentral auf dem Log-Server verfügbar. (Der SmartClient-Server bzw. ein dazu bestimmter Server sollte unter dem Namen „loghost“ erreichbar sein.)

Zugriff auf einen Zeitserver

Um die Zeit-Einstellung auf allen Clients zu koordinieren, sollten die Clients auf einen Zeitserver zugreifen. Der SmartClient-Server sollte auch als Zeitserver eingerichtet sein (siehe 1.5.3 auf Seite 13) und unter dem Namen „ntp“ erreichbar sein.

Über `chkconfig xntpd on` wird dafür gesorgt, dass beim Booten automatisch Zeit-Informationen vom Server geholt werden.

Der anzusprechende Server wird in die Datei `/etc/ntp.conf` eingetragen:

```
server ntp
```

Passwort

Alle Benutzer-Accounts inklusive Passwörter und Ablaufdaten für Passwörter sind im LDAP-Verzeichnis hinterlegt.

LDAP würde bei Passwortänderungen aber alle vorgeschlagenen Passwörter ohne weitere Sicherheitsprüfung annehmen.

Überprüfungen, ob ein Passwort gewissen Kriterien genügt, werden deshalb auf der lokalen Workstation vorgenommen.

Wesentlich dabei ist die Konfigurationsdatei `/etc/login.defs`. Dort kann u.a. festgelegt werden:

`PASS_MIN_LEN`	5	Minimale Passwortlänge
`PASS_CHANGE_TRIES`	2	Anzahl der Versuche, bis ein eigentlich zu schwaches Passwort trotzdem angenommen wird.
`LOGIN_RETRIES`	3	Anzahl der negativen Versuche bis ein Login-Vorgang abgebrochen wird.

dies beschreibt noch nicht alles. Trotz > 5 *kann ein Password too easy sein. cracklib?*

2.2.2 KDE

Die Desktop Umgebung KDE ist ein sehr mächtiges Gebilde mit unzähligen Einstellmöglichkeiten. Einige Dinge werden in jeder zentral administrieten Umgebung benötigt, wie z.B. das zentral konfigurierte Menü. In den folgenden Abschnitten wird ein Überblick über die eingesetzten KDE Konfigurationsanpassungen gegeben. Für weitergehende Informationen bietet die Seite http://www.kde.org/areas/sysadmin/ einen guten Startpunkt.

Datei Hierarchie

Aus [Kdeb]:

KDE and KDE applications look up files by scanning the directory trees. The directory trees are in order of precedence. When a file is present in multiple directory trees, the file from the first listed tree takes precendence. Normally, the tree located in the user's home directory has the highest precedence. This is also the directory tree where changes are written to.

For configuration files the story is slightly different. If there are multiple configuration files found in the directory trees with the same name their content is combined. The precedence order of the directory trees plays a role here. When

two files define the same configuration key, the file with the highest precedence determines which value is used for the key.

KDE Umgebungsvariablen bestimmen, welche Verzeichnisse als Konfigurationsverzeichnisse ausgewertet werden:

KDEHOME This environment variable determines the location of the user level directory tree and is used by KDE applications for creating and saving files. This directory tree has the highest precedence; files or settings found in this directory tree will take precendent over any files or settings found in other directory trees.

This directory tree is, as the name already suggests, normally located in the home directory of the user.

If the environment variable has a value that starts with a tilde ( ), the tilde is replaced with the users home directory at runtime. In order to use this care must be taken to add proper quoting, otherwise the shell might do the expansion already resulting in undesired behaviour in combination with su.

KDEDIRS Since KDE 3.x it is possible to specify more than a single system level directory tree. This makes it possible to let groups of users use a directory tree dedicated for their group. Such an additional directory tree can contain additional applications, specialized application resources or a specific set of default configurations suitable for the group. Specifying default configurations this way instead of using a /etc/skel construction has as advantage that changes in the default configuration can be made after the account of the user has been created.

The directories in KDEDIRS should be seperated with a colon (:). The directories are listed in precedence order, the first directory has highest precedence, the last one lowest precedence. Since a group level directory tree should normally override any settings present at the system level, one would list the group level directory tree before the system level directory tree.

KDEDIR aus Kompabilitätsgründen mit KDE 2.x. Wird auch verwendet, wenn KDEDIRS nicht gesetzt ist. This environment variable defines where the system level KDE directory tree is located. All data files such as icons, soundfiles and menu description files that accompany applications are usually installed in this directory tree.

Beispiel:

```
KDEHOME='~/.kde3'
KDEDIRS='/var/lib/kde-profiles/smartclient1:/var/lib/kde-profiles/kiosk:/etc/opt
```

Achtung: bei Suse Distributionen werden im Standard KDE Startskript `/opt/kde3/bin/startkde` diese Variablen entfernt. Um dies zu verhindern, sind die Zeilen

```
unset KDEDIR
unset KDEDIRS
```

durch

```
#unset KDEDIR
#unset KDEDIRS
```

zu ersetzen. Diese Modifikation muss nach jedem KDE Update vorgenommen werden. Alternativ kann man seit KDE 3.3.2 diese Variablen in einer Datei `/opt/kde3/env/*.sh` setzen.

Anstatt die `KDEDIRS` Variable manuell zu setzen, kann sie bei SmartClient auch bei den Einstellungen des Benutzers im LDAP hinterlegt und durch entsprechende Skripte ausgewertet werden.

Über das Kommando `kde-config --path config` lassen sich diese Einstellungen nach dem Anmelden überprüfen.

Unterverzeichnisse

Directory	Description
share/applnk/	Contains .desktop files describing the KDE-menu.
share/apps/	Contains applications specific data files. Each application has a sub-directory here for storing its files.
share/autostart/	beinhaltet .desktop Dateien, die beim Starten von KDE mitgestartet werden sollen.
share/config/	Contains configuration files. Configuration files are normally named after the application they belong to followed by rc. There are also files that are specific to a component and as such referenced by all applications that use that component. A special case is `kdeglobals`, this file is read by all KDE applications.
share/config/session/	This directory is used by session management and is normally only available under $KDEHOME. At the end of a session KDE applications store their state here. The file names start with the name of the application followed by a number. The session manager `ksmserver` stores references to these numbers when saving a session in `ksmserverrc`.
share/doc/HTML/	Documentation of KDE applications is stored here. Documentation is categorized by language and the application it belongs to.
share/icons/	Under this directory icons are stored. Icons are categorized per theme, dimension and usage category.
share/mimelnk/	In this directory .desktop files that describe mimetypes are stored.
share/services/	This directory contains .desktop files that describe services. Services and Applications are very similar, the major difference is that a Service is usually used by other Services or Applications while an Application is in general started by the user. Services do not appear in the KDE menu.
share/servicetypes/	This directory contains .desktop files that describe servicetypes. A servicetype usually represents a certain programming interface. Applications and Services include in their .desktop files the servicetypes that they provide.
share/sounds/	This directory contains sound files.
share/templates/	This dir contains templates for creating files of various types. A template consists of a .desktop file that describes the file and that includes a reference to a file in the .source sub-directory. The templates in this directory appearem in the *Create New* menu available on the desktop and in the file browser. When a user selects a template from the menu its source file is copied.
share/wallpapers/	This directory contains images that can be used as background picture.

Applikationen hinzufügen

KDE Verzeichnisse

kde-config –path config

share

Menü: share/applnk/2xa.desktop

/usr/share/applications/ ???

/usr/share/mime/packages/neu.xml anlegen und update-mime-database /usr/share/mime/ aufrufen. erzeugt xml Datei in entsprechenden Unterverzeichnissen unter /usr/share/mime/ Wer wertet es aus?

```
<?xml version="1.0" encoding="UTF-8"?>
<mime-info xmlns="http://www.freedesktop.org/standards/shared-mime-info">
  <mime-type type="application/x-2x">
        <comment xml:lang="en">2x en</comment>
        <comment xml:lang="de">2x de</comment>
        <glob pattern="*.2xa"/>
  </mime-type>
</mime-info>
```

erst nach /opt/kde3/share/mimelnk/application//x-2x.desktop funktionstüchtig

```
   [Desktop Entry]
Comment=test Email Message
Hidden=false
Icon=message
MimeType=application/x-2x
Patterns=*.2xa;*.2XA
Type=MimeType
Comment[de]=test Outlook E-Mail
```

Applikationsberechtigungen

Häufig ist es erwünscht, dass bestimmte Applikationen nur von Mitgliedern bestimmter Gruppen ausgeführt werden können. Auch die Menü-Einträge oder Desktop-Icons für die entsprechenden Applikationen sollen nur sichtbar sein, falls der Benutzer die Applikation ausführen darf.

Dazu wird eine Gruppe für eine bestimmte Applikation unter `users` → `groups` → `applications` angelegt und die entsprechenden Benutzer als Mitglieder aufgenommen.

Auf dem Client wird eine Berechtigungs-Datei im Verzeichnis `/etc/permissions.d` erzeugt. Wichtig ist, dass diese Datei den Namen eines installierten rpm-Paketes haben muss, um ausgewertet zu werden. Hier bietet sich z.B. der Name `sc_base` an.

In diese Datei werden die Dateien eingetragen, deren Berechtigungen geändert werden sollen. Beim Aufruf von `SuSEconfig` bzw. `SuSEconfig --module permissions` werden die Dateien im Verzeichnis `/etc/permissions.d/` ausgewertet und die Berechtigungen der angegebenen Dateien bzw. Verzeichnisse dementsprechend verändert.

Um Menü-Einträge oder Desktop-Icons unter KDE zu beeinflussen, muss sowohl die ausführbare Datei als auch die Datei, die den Eintrag im Menü bzw. das Desktop-Icon beschreibt, so in die Permissions-Datei eingetragen werden, dass alle Rechte für „others“ entfernt werden und das Ausführ-Recht bzw. Lese-Recht für die Gruppe gegeben ist. Dabei wird die Gruppe der Dateien auf die dafür angelegte Applikations-Gruppe gesetzt.
Beispiele für Einträge in der Datei `/etc/permissions.d/sc_base`:

```
/opt/kde3/bin/kmail                                          root.kmail      750
/etc/opt/kde3/share/applnk/SuSE/Applications/KMail.desktop   root.kmail      640
```

Durch diese Einträge ist sowohl das KMail-Binary nur noch von Mitgliedern der Gruppe *kmail* ausführbar, als auch der Menü-Eintrag nur noch von Mitgliedern dieser Gruppe lesbar, so dass der Eintrag nur für diese Gruppenmitglieder sichtbar ist.

Bei Veränderungen an der Berechtigungs-Datei muss darauf geachtet werden, dass `SuSEconfig` aufgerufen wird.

Ändern sich bei Benutzern Gruppen-Mitgliedschaften, so ist eine Neu-Anmeldung unter KDE notwendig. Weiterhin muss ein Neu-Aufbau der Menü-Strukturen unter KDE erzwungen werden.

Weiterhin muss der KDE System COnfiguration CAche (ksycoca) (normalerweise unter `$HOME/$USER/cache-$HOST/ksycoca` abgelegt) neu aktualisiert werden. Hierzu dient das Kommando `kbuildsycoca --noincremental`. In einer SmartClient Umgebung steht es in dem Skript `sc_kde_menu_refresh` zur Verfügung. Damit es bei jedem Start einer KDE Sitzung ausgeführt wird, kann es unter `/opt/kde3/share/UnitedLinux/addon-scripts/` hinterlegt werden.

So ist bei einer Neu-Anmeldung des Benutzers unter KDE sichergestellt, dass die Menü-Strukturen aktuell sind.

Anstatt die `.desktop`-Datei für den lesenden Zugriff durch den Benutzer zu sperren kann man alternativ bei neueren KDE-Versionen (ab 3.2) auch eine `TryExec` Direktive hinzufügen. Beispielsweise steht in `KMail.desktop` folgender Eintrag:

```
Exec=kmail -caption "%c" %i %m
```

Hier kann eine Zeile

```
TryExec=kmail
```

hinzugefügt werden. Dadurch wird der Menü-Eintrag nur für die Benutzer sichtbar, die ein „kmail“-Binary im Pfad haben und die Berechtigung, das Binary auszuführen.

Erläuterung Hidden Attribut und dessen Nutzung

Eine Liste der Standard Optionen in .desktop Dateien ist unter http://freedesktop.org/Standards/desktop-entry-spec/ zu finden.

Die Konfiguration von Berechtigungen über `sudo` (siehe `man sudo`) ist auch denkbar.

XDG Mit XDG wird versucht, einen Konfigurationsdatei Standard für verschiedene Windowmanager (primär KDE und Gnome) zu etablieren. KDE unterstützt dies optional ab Version 3.2. Die systemweie Konfiguration hierzu liegt unter `/etc/xdg/menus/`, die Benutzer spezifische unter `.config/menus`. Weitere Informationen sind unter http://freedesktop.org/Standards/ und speziell unter http://freedesktop.org/Standards/basedir-spec/ zu finden.

KDE Kontrollzentrum

Einstellmöglichkeiten des KDE Kontrollzentrum mit `kcontroledit` anpassen. Dieses Tool basiert auf kmenuedit und speichert seine Einstellungen auch unter `~/.config/menus/applications-kmenuedit.menu`. Zur Systemweitenverwendung muss diese Datei ins Verzeichnis `/etc/xdg/menus/applications-kmenuedit.menu` kopiert werden. Ein Ablegenunter einem anderen Namen ist leider nicht möglich, da sie dann nicht zum richtigen Zeitpunkt ausgewertet wird.

Besser geeignet ist die Konfiguration mittels eines `[KDE Control Module Restrictions]` Abschnittes in der Datei `share/config/kdeglobals` eines KDE Profils.

Die Liste aller Kontrol-Module kann mit dem Befehl `kcmshell --list` ermittelt werden. Zu dem Modulnamen muss in dem `[KDE Control Module Restrictions]` Abschnitt noch ein `kde-` voran– und ein `.desktop=false` nachgestellt werden.

Beispiel:

```
[KDE Control Module Restrictions]
kde-obex.desktop=false
kde-keyboard=false
```

Applikationseinstellungen

Primär werden Applikationen über ihre GUIs bzw. das KDE Kontrollzentrum (`kcontrol`) konfiguriert.

Dies aber nicht in jedem Fall ausreichend, da erstens nicht alle Parameter zugänglich sind und zweitens weitere Einstellungen zum setzen von unveränderbaren Voreinstellungen notwendig sind (siehe Kapitel 2.2.2 auf der nächsten Seite).

In diesem Fall können Einstellungen für Applikationen direkt in den Konfigurationsdateien der dem Profil entsprechenden KDE Verzeichnissen vorgenommen werden.

Ein Vorteil von KDE Konfigurationsdateien ist, dass in den Benutzereinstellung nur die Differenzen zu den Einstellungen gespeichert werden. Dadurch wirken sich Änderungen in den Standardeinstellungen direkt auf den Benutzer aus, sofern dieser auch vorher die Standardeinstellungen verwendet haben. Nachteil dieser Methode ist aber, dass meine keine vollständigen Konfigurationsdateien abgelegt werden, sondern nur die vom Standard unterscheidenen Parameter. Somit stellt sich bei der Konfiguration per Texteditor die Herausforderung, erstmal die entsprechenden Variablen zu kennen.

Eine Erleichterung hierzu stellt das GUI des Tool KConfigEditor (`http://extragear.kde.org/apps/kconfigeditor.php`) dar. Dieses stellt alle möglichen Konfigurationsdateien in einem Baum dar und dazu jeweils alle (?) passenden Variablen, ggf. auch zusammen mit einer Erläuterung.

Autostart

Damit Anwendungen automatisch mit KDE zusammen gestartet werden, müssen die zugehörigen `.desktop` Dateien in das `share/autostart/` Verzeichnis eines verwendeten Profiles abgelegt werden.

Zusätzlich sind noch Autostart spezifische Einstellungen möglich. Der wichtigste Parameter ist `X-KDE-autostart-condition`.

Am Beispiel der Applikation krandrtray lautet er

```
X-KDE-autostart-condition=krandrtray:General:Autostart:true
```

Dies bedeutet: überprüfe den Parameter Autostart der Sektion `[General]` aus der Konfigurationsdatei krandrtray. Der Default Wert ist „true“.

Details hierzu sind unter [TBS02] im Kapitel „Automatically Running an Application on KDE Start“ zu finden.

Systemweite Autostart Anwendungen sind im Verzeichnis `/opt/kde3/share/autostart/` definiert und könnten somit dort oder mit der dort unter Parameter `X-KDE-autostart-condition` angegebenen Bedingung deaktiviert werden.

Programme die mit Autostart gestartet werden, werden normalerweise automatisch von Session Management ausgeschlossen, so dass sie beim nächsten einloggen nicht mehrfach gestartet werden (von Autostart und Sessionmanager, der alte Session wieder herstellt).

Spezielle KDE Anwendungen

SuSE Hardware Watcher Ab Suse Linux Prof 8.2 bzw. Suse Linux Desktop 1.0 kommt der SuSEWatcher zum Einsatz. Diese Applikation dockt im KDE-Panel an und öffnet ein Popup-Fenster, falls neue Hardware detektiert wurde.

Da die Hardware beim SmartClient automatisch konfiguriert wird, ist diese Funktionalität unerwünscht. Allerdings kann der SuSEWatcher nicht gänzlich deaktiviert werden, da er auch für das Erzeugen von Desktop-Icons für Wechselmedien wie CD-ROM- und Floppy-Laufwerk zuständig ist.

Durch Hinzufügen des folgenden Eintrags in der Konfigurationsdatei `/opt/kde3/share/config/hwsusekickerrc` wird das Popup-Fenster nach der Erkennung neuer Hardware deaktiviert:

```
[General]
DoNotShowPopup=true
```

Anpassung des Login-Screen (KDM) Falls gewünscht, kann der KDM (KDisplay Manager), der das grafische Login ermöglicht, angepasst werden.

Die gut kommentierte Konfigurationsdatei `/etc/opt/kde3/share/config/kdm/kdmrc` sorgt z.B. durch Anpassung des Eintrags `ShowUsers=None` nach `ShowUsers=NotHidden` dafür, dass beim Login alle verfügbaren Benutzer angezeigt werden.

KDE Kiosk

Ursprünglich für Internet Terminals gedacht, ist der primäre Fokus des Kiosk Projektes das Rechte Management eines KDE Systems. Dies wird hauptsächlich dadurch erreicht, dass dem Benutzer die Möglichkeit genommen werden kann, bestimmte Einstellungen zu verändern. Dies kann Parameterweise, Abschnittsweise oder sogar Dateiweise erfolgen. Ein entsprechend vor Veränderung geschützter Parameter wird durch den Suffix `[$i]` (für immutable) gekennzeichnet. Desweiteren ist es möglich, Parameter nicht statisch zu setzen, sondern ihren Wert durch Umgebungsvariablen bzw. sogar Shellskripten bestimmen zu lassen. Weitere Detail hierzu sind unter [Kdec] nachzulesen.

Ein generelles, sehr gravierendes Problem ist (Stand: KDE 3.3), dass es für den Anwender nicht ersichtlich ist, ob es sich um eine geschützte Einstellung handelt

oder nicht. D.h. der Anwender kann Einstellungen vornehmen, diese werden nur nicht gespeichert. Es ist anzunehmen, das dies die Anwender verwirren wird.

Des weiteren bleibt dieselbe Problematik wie schon vorher beim editieren von Umgebungsvariablen bestehen: man muss die Parameter erst kennen um sie sperren zu können.

Zur Erleichterung gibt es das Kiosk Tool: (`http://extragear.kde.org/apps/kiosktool.php`), in dem sinnvolle Voreinstellungen vorgenommen werden können. Das Kiosk Tool hat auch schon ein Profil Management vorgesehen, dass aber nicht unbedingt verwendet werden muss. Angenehm daran ist, dass die Einstellungen nicht bei den Benutzereinstellungen (`~/.kde/`), sondern in einem extra Profil Verzeichnis gespeichert werden.

Unter [Mal04] wird ein darauf aufbauendes Verfahren teilweise recht detailiert beschrieben. Leider wird auf viele der bestehenden Einschränkungen und Schwierigkeiten nicht eingegangen.

Verfahrensweise

Sollen KDE Profile, ggf. kaskadierte KDE Profile erstellt werden, so ist es sinnvoll, grundlegende Einstellungen mit den Kiosk–Tool vorzunehmen. Dieses Profil sollte dann möglichst auch nicht mehr per Hand modifiert werden.

Das KDE Applikationsmenü sollte hiervon getrennt betrachtet werden. Während das hinzufügen von Applikationen keine Schwierigkeit darstellt, ist das Entfernen von Applikationen nicht vorgesehen. Deshalb bietet es sich an, statt die bestehenden Menüs zu modifizieren, ein komplett eingenes zu definieren.

Kiosk Profile

- kiosktool

allgemeines KDE Profil

- Test Benutzer
- `~/.kde/` löschen
- per GUI Einstellungen vornehmen
- kcontroledit
- (kconfigeditor)
- Einstellungen betrachten und sinnvolle mit entsprechendes Profil Verzeichnis übernehmen. Nicht alle, da einiges nicht sinnvoll (Session (gestartete Applikationen, Fensterpositionen)), bzw. überflüssig so dass man schnell die Übersicht verliert.

- Testen mit weiteren Benutzer (jedesmal `~/.kde/` löschen)

weitere KDE Profile

- runterliegende Profile einbinden, ansonsten wie allg. Profil.
- darauf achten, nur die für dieses Profil gewollten Einstellungen ins Profil Verzeichnis übernehmen.

Profile ändern

- backup
- kiosktool: profil laden, verändern
- allg. Profil: `~/.kde/` löschen, Profile nach `~/.kde/` kopieren. Änderungen vornehmen. diff mit Backup bilden und sinnvolle Parameter übernehmen. testen.

2.2.3 Nicht-KDE Applikationen

Mozilla/Firebird

Der Webbrowser Firefox gehört zur Mozilla Familie und wird wie dieser konfiguriert.

Hilfreich ist in diesem Zusammenhang das `Custumer Configuration Kit` Plugin (siehe [CCK06]) Die erstellte Konfiguration kann mit dem Kommando

```
firefox -install-global-extension EINSTELLUNGEN.xpi
```

systemweit installiert werden.

Die aktuelle Konfiguration im Browser mit `about:config` abgefragt werden. Informationen zu den eingebundenen Plugins sind mit `about:plugins` ersichtlich.

`/usr/bin/firefox` ist ein Link auf das Startskript `/opt/MozillaFirefox/bin/firefox.sh`. Dieses wertet auch die Umgebungsvariable `MOZ_DEBUG=1` aus.

Plugins werden automatisch durch das Skript `/opt/MozillaFirefox/bin/add-plugins.sh` eingebunden, dass auch als RPM Postinstall Skript aufgerufen wird.

Eine einfache Zuordnung von Mimetypen zu Applikationen kann über die Datei `/etc/mailcap` vorgenommen werden, z.B. `application/pdf;/usr/bin/acroread "%s"`.

2.2.4 User Skeleton Verzeichnis

Das Skeleton-Verzeichnis (`/etc/skel`), das beim Anlegen von neuen Benutzern verwendet wird, um die Home-Verzeichnisse mit Vorgaben zu versehen, liegt auf dem Dateiserver der jeweiligen Lokation.

Da die initiale Konfiguration aber mit den installierten Anwendungen harmonisiert, wird der Inhalt des Skeleton hier beschrieben, deren weitere Verwaltung aber in Kapitel 3.6 auf Seite 63.

Das Default Skeleton Verzeichnis einer SuSE Linux Installation enthält um die 25 Dateien bzw. Verzeichnisse. Diese steuern im Einzelfall u.a. das Verhalten des X-Servers, was nicht erwünscht ist. Um ein einfach zu administrierendes System zu erhalten, ist es notwendig, Benutzer möglichst einheitlich (am besten an einer zentralen Stelle) handhaben zu können.

Deswegen sollte von einem leeren Skeleton Verzeichnis ausgegangen und nur die Dateien angelegt werden, die unbedingt notwendig sind.

Gerade Einstellungen in den Dateien wie `.xinitrc`, `.xsession`, `.bashrc`, `.profile` sollten wenn möglich in den entsprechenden globalen Konfigurationsdateien vorgenommen werden.

was fehlt?

2.3 Test der Installation durch Anschluss eines neuen Clients

Nach der Synchronisation des Referenz-PCs sollte sichergestellt sein, dass folgende Scripte gestartet sind:

- `sc_leases2ldap`
- `sc_password_generator`

Damit können nun neue Clients via PXE-Boot vom DHCP-Server ein entsprechendes Boot-Image bekommen und darüber dann das Image des Referenz-PCs erhalten. Clients, die bereits im LDAP-Server eingetragen sind, können durch ein Zurücksetzen der `dbkImageVersion` auf den Wert 0 via PXE-Boot die Differenz des neuen Referenz-Images zu ihrer aktuellen Installation erhalten.

3 Betriebshandbuch

Die Basis des SmartClient Systems bildet der zentrale Verzeichnisdienst LDAP. Dieser kann auf unterschiedliche Weisen modifiziert werden. Die präferierte Methode ist die Nutzung der SmartClient Webmin Module. Diese ermöglichen es, die wichtigsten SmartClient Aktionen durchzuführen.

Daneben kann das LDAP Verzeichnis auch direkt mit beliebigen LDAP Browsern modifiziert werden. Dabei ist aber zu beachten, dass viele Syntax- und Plausibilitätsüberprüfungen dann nicht durchgeführt werden können. Falsche Daten im LDAP können im schlimmsten Fall das gesammte System unbrauchbar machen.

Nachfolgend werden die möglichen administrativen Tätigkeiten aufgezeigt, die während des Betriebs durchgeführt werden können bzw. müssen.

3.1 Administrations-Tools

3.1.1 Webmin

3.1.2 LDAP Tools

phpLDAPadmin

phpLDAPadmin ist ein webbasiertes, generisches LDAP Tool.

gq

Um mit gq auf einen LDAP Server zuzugreifen, muss dieser zunächst in den Voreinstellungen von gq eingetragen werden. Um den LDAP Server vollständig mit gq administrieren zu können, muss mindestens die Version 0.6.0 verwendet werden. Zur Konfiguration wird der Reiter *Servers* unter dem Menüpunkt *File/Preferences* aktiviert. Mit dem Button *New* wird dann der Dialog zum Eintragen eines LDAP Servers aufgerufen. Ein Konfigurationsbeispiel ist in den Screenshots 3.1 auf der nächsten Seite gezeigt.

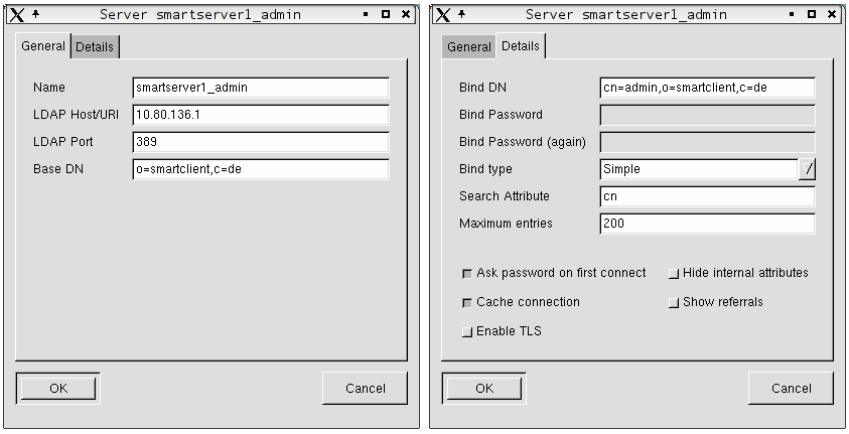

Abbildung 3.1: Konfiguration von `gq`

Nach dem Eintragen eines neuen Servers muss `gq` beendet und neu gestartet werden, erst dann taucht der neue Server zur Auswahl beim Reiter *Browse* auf. In der Browse-Ansicht kann durch den LDAP-Baum navigiert werden.

Zum Suchen mit `gq` kann der Reiter *Search* verwendet werden. Es gibt die Suchmodi *search* und *filter*. Im search-Modus werden allerdings nur bestimmte Attribute durchsucht, wie *cn*, *ou*, *uid*, d. h. es kann nach Workstation-Namen, Usern oder Organisationseinheiten gesucht werden, nicht jedoch z. B. nach einer MAC-Adresse. *warum soll das nicht gehen? Ich habe leider keine version 0.6 mehr da.*

Im *filter*-Modus können LDAP Filterausdrücke gemäß RFC 2554 benutzt werden. Eine Dokumentation solcher Filter befindet sich in der man-Page *ldap_search*, die im Paket `openldap2-devel` enthalten ist. Ein Beispiel für die Suche nach einer MAC-Adresse (z. B. `(macAddress=02:fe:a5:bc:dc:4e)` wäre (der folgende Ausdruck enthält eine UND-Verknüpfung):

```
(&(macAddress=02:fe:a5:bc:dc:4e)(objectClass=dbkWorkstation))
```

Häufig benötigt man Informationen über Rechner. Die Syntax für Suchanfragen im `gq` ist am LDAP-Standard orientiert.

Sucht man z. B. alle Workstations, die nicht die aktuelle Image Version – etwa die Version `2002071701` – besitzen, sähe eine Suchabfrage folgendermaßen aus:

```
(&(objectclass=dbkWorkstation)(!(dbkImageVersion=2002071701)))
```

Dieser Suchfilter liefert alle Objekte zurück, die die Objektklasse dbkWorkstation haben (dies entspricht dem LDAP-Filter „`(objectclass=dbkWorkstation)`“ und „`&`“ nicht „`!`“ die Image-Version `2002071701` haben „`!(dbkImageVersion=2002071701)`“).

Kommandozeilentools

Fals `gq` nicht eingesetzt werden kann, bieten sich noch die sehr leistungsstarken Tools `ldapsearch`, `ldapsearch`, `ldapadd` und `ldapmodify`*doku ldap tools?* an. Diese Tools eignen sich auch für die Entwicklung einfacher Shell-Skripte.

Mit `ldapsearch` können LDAP Suchanfragen durchgeführt werden. Dabei können auch die oben erwähnten Filterausdrücke nach RFC 2554 verwendet werden. Außerdem können optional die Attribute des Suchergebnisses spezifiziert werden. Die Ausgabe erfolgt im LDIF Format, so daß sie mit `ldapmodify` oder `ldapadd` verwendet werden kann.

Ein Beispiel für eine Suchabfrage mit `ldapsearch`:

```
ldapsearch -x -LLL -h localhost -b "o=smartclient,c=de" \
"(macAddress=00:04:76:22:2f:e6)" macAddress dbkImageVersion
```

3.1.3 LDAP Server

OpenLDAP

Ändern des LDAP Admin Passwortes Das Admin-Passwort des LDAP ist in der Datei `/etc/openldap/slapd.conf` in der Zeile `rootpw` festgelegt. Dieses oberste Passwort kann nicht im LDAP direkt eingestellt werden, da es auch schon zum Aufsetzen des LDAPs benötigt wird.

3.1.4 SmartClient LDAP Suchschema (Treesearch)

LDAP selbst stellt nur eine Hierarchie von Daten zur Verfügung. Die Art, wie diese Daten interpretiert werden bleibt der Applikation überlassen. Eine Besonderheit von SmartClient ist die Verwendung von Standard-Einstellungen. Dies ist folgendermassen implementiert: Wenn ein LDAP Attribut gesucht wird, so wird erst der eigene Eintrag betrachtet. Sollte es dort nicht definiert sein, so wird der *standards*-Eintrag für den *deviceType* gesucht. Wird auch in diesem kein passenes Attribut gefunden, so wird eine Ebene höher im LDAP nach dem gleichen Schema gesucht. Wird er auch dort nicht gefunden, wird wieder höher gesucht, bis die *standards*-Einträge der Wurzel des LDAPs erreicht sind. Normalerweise ist davon auszugehen, dass spätestens dort der Eintrag definiert ist.

Im SmartClient Kontext wird dieses Verhalten „treesearch“ genannt.

Beispiel

Als Beispiel soll ausgehend von dem Rechner cn=pc00257,ou=test,o=smartclient, c=de ein „treesearch“ durchgeführt werden.

Der Rechner ist vom *deviceTypeDn* cn=linuxPc,ou=global,o=smartclient,c=de. Alle *standards*-Einträge für diesen *deviceTypeDn* sind als cn=standardsLinuxPc eingepflegt.

Als Suchreihenfolge ergibt sich dann:

cn=pc00257,ou=test,o=smartclient,c=de Der Eintrag für den ein Attribut gesucht wird, wird zuerst befragt. Ausserdem wird dieser Eintrag zu Hilfe genommen, um den *deviceTypeDn* zu bestimmen, falls er nicht schon von vornherein bei der Suchanfrage bekannt war und der Suchanfrage übergeben worden ist.

cn=standardsLinuxPc,cn=pc00257,ou=test,o=smartclient,c=de Ein so gearteter Eintrag würde nicht gesucht werden, da der zu Grunde liegende Eintrag cn=pc00257,ou=test,o=smartclient,c=de kein LDAP Container ist und daher keine Standards-Objekte enthalten kann. Eine Suche könnte keine Elemente finden und würde daher ein leeres Ergebnis bringen.

Daher steht der Eintrag hier nur, damit die Suchreihenfolge besser verstanden werden kann.

ou=test,o=smartclient,c=de Eine Ebene höher wird als nächstes untersucht.

cn=standardsLinuxPc,ou=test,o=smartclient,c=de Dieser Eintrag wurde gefunden, da er der *standards*-Eintrag für PC's vom *deviceTypeDn* des zu suchenden Eintrags.

o=smartclient,c=de Wieder eine Ebene höher als die letzte Ebene – der zwischengeschobene *standards*-Eintrag wird hier nicht berücksichtigt.

cn=standardsLinuxPc,o=smartclient,c=de Wieder ein *standards*-Eintrag. Dies ist der letzte Eintrag, wenn o=smartclient,c=de als LDAP Basis konfiguriert ist.

3.2 Server

Clients und Server werden im LDAP getrennt behandelt. Die Einträge der Clients sind dafür optimiert, automatisch erzeugt werden zu können. Dafür hat ein Client auch immer nur einen, durchnummerierten Namen.

Server hingegen werden von vielen Workstations aus angesprochen. Sie erhalten einen Namen nach einer bestimmten Syntax (z. B. `server01`). Zusätzlich erhalten

Server im SmartClient Kontext auch noch Namen, die den zur Verfügung gestellten Dienst beschreibt. Dadurch wird es für die Clients möglich, einfach den Dienst, den sie erreichen wollen durch den entsprechenden Namen zu finden (z. B. `nfs`, `loghost`, `ldap`). Durch die DNS Hierachie, die der Hierachie des LDAP entspricht, kann außerdem gewährleistet werden, dass eine Workstation immer den ihr am nächsten stehenden Server verwendet.

Um eine klare Trennung zwischen Server und angebotenem Dienst (Service) zu gewährleisten, wird dies im LDAP auch durch zwei verschiedene Objektklassen repräsentiert:

dbkServer und dbkService.

3.2.1 Die Objektklasse dbkServer

Die Objektklasse dbkServer repräsentieren die physikalischen Server, wie in dem DNS Eintrag `server01.server.smartclient.de`. Die Objektklasse dbkService hingegen hält Informationen über die die angebotenen Dienste, wie `ldap`, `nfs`, `loghost` vor.

Zum Anlegen neuer Einträge verwendet man am Besten das Tool `gq`, nimmt einen vorhanden Eintrag und modifiziert ihn.

Als erstes sollte im LDAP der Server angelegt werden, dann die dazugehörigen Services.

Die zu befüllenden Attribute für dbkServer (siehe auch Tabelle 3.1 auf der nächsten Seite) sind im einzelnen:

macAddress Die MAC Adresse des Servers. Zur Zeit kann nur eine MAC Adresse eingetragen werden. Hat der Rechner mehrere Netzwerkkarten und soll dies berücksichtigt werden, so kann für die weiteren Karten jeweils ein Service eingerichtet werden. Die MAC Adressen sollten aus sechs zweier Kombinationen der Ziffern „1–6“ und der Buchstaben „a–f“ bestehen. Die Kombinationen sind mit „:“ zu trennen. Ein Beispiel für eine MAC Adresse könnte dann `00:12:aa:34:bb:cd` sein.

ipHostNumber Dieses Attribut enthält die IP Adresse des Rechners. Es ist ein multi-value Feld, d. h. es können pro Eintrag mehrere IP Adressen angegeben werden. In den meisten Fällen wird es aber sinnvoller sein, die zusätzlichen IP Adressen in Service Objekten zu verpacken. Die zu verwendenden IP Adressen bestehen aus vier Zahlengruppen aus dem Bereich „0–255“, die jeweils mit einem Punkt getrennt werden (z. B. `10.0.1.2`).

dbkDnsName Soll zusätzlich zum *cn* des Server-Objektes noch weitere DNS Namen für den Server konfiguriert werden, so können diese in diesem Attribut

eingetragen werden. Diese Namen werden als `cname` in die DNS Konfiguration mit aufgenommen.

ipServiceProtocol Dieses Feld hat rein dokumentativen Charakter. Es führt die Namen der Services auf, die dieser Server bereitstellt. Für einen FTP Server könnte dort also der String `ftp` eingetragen werden. Wird der Server auch noch als Webserver genutzt, so könnte zusätzlich noch `http` hinzugenommen werden.

dbkPubKey Wird der Server zur Administration per ssh von SmartClient Workstation genutzt, so sollte der öffentliche Schlüssel (*public key*) des `sshd` des Servers in dieses Feld eingetragen werden (oder auch mehrere, so vorhanden und falls sinnvoll). Auf den SmartClient Workstation wird dieser öffentliche Schlüssel zur Konfiguration administrativer Zugänge genutzt.

dbkAssociatedServiceDN Zur Gruppierung von Services zu den bereitstellenden Servern kann dieses rein dokumentativ genutzte Attribut genutzt werden. Es enthält die *dn* der Services, die auf diesem Server laufen. In dem oben genannten Beispiel des `ftp` und `http` Servers wären dann z. B. die Verweise auf cn=ftp,o=smartclient,c=de und cn=www,o=smartclient,c=de eingetragen (zu beachten ist natürlich, dass es die Objekte, auf die Verwiesen wird, auch existieren müssen und von der Objektklasse dbkService sein sollten).

LDAP Attribut		Beispiel + Beschreibung
macAddress		`00:60:94:2F:C8:F8`, kann mit dem Komanndo `ifconfig` herausgefunden werden.
ipHostNumber		z. B. 10.1.10.1, die IP des Servers
dbkDnsName		`smartserver01`, `smartserver02`, etc. – jeder physikalisch vorhandene Server. Diese werden im Name-Server eingetragen.
ipServiceProtocol		`dns`, `dhcp`, `tftp`, `ldap`, `password`, `rsync`, `ntp` (alle Protokolle, die vom jeweiligen Server angeboten werden)
dbkPubKey	may	Hier kann der Eintrag aus `/root/.ssh/id_dsa.pub` eingetragen werden. Wird von den Clients gelesen und in deren Datei `/root/.ssh/authorized_keys` geschrieben. Damit wird ein Zugang ohne Eingabe des Passworts ermöglicht.
dbkAssociatedServiceDN	may	DN von Services, die auf diesem Server laufen

Tabelle 3.1: Attribute der Objektklasse dbkServer

3.2.2 Die Objektklasse dbkService

Die Objektklasse dbkService stellt einen IP basierten Dienst dar. Die einzelnen Attribute (siehe auch Tabelle 3.2 auf der nächsten Seite) sind dabei wie folgt:

ipHostNumber Wie bei Server-Objekten kann hier eine (oder auch mehrere) IP Adresse eingetragen werden, unter der der Service erreichbar ist.

dbkDnsName Analog zum Server können auch hier weitere DNS Namen eingetragen werden. Diese werden wie gehabt als `cname` im DNS konfiguriert.

dbkServiceName *habe ich glaube ich noch gar nicht benutzt.fs*

dbkServiceStatus Kann zur Schaltung des Dienstes genutzt werden. Ist hier ein von `TRUE` unterschiedlicher Wert eingetragen, so wird der gesamte Eintrag bei der Erzeugung der DNS Dateien übersprungen.

dbkServiceEmail Hier sollen Mail-Adressen eingetragen werden können, an die Fehlermeldungen bezüglich des Dienstes geschickt werden. Dieses Feld wird zur Zeit aber nicht ausgewertet.

dbkAssociatedServerDN In diesem Feld werden alle Server Objekte referenziert, die diesen Dienst zur Verfügung stellen. Er wird bisher nur zu dokumentarischen Zwecken benutzt. Im Falle eines Lastbalancierten Webservers könnten hier als alle *dn* der Server Objekte eingetragen werden, aus denen der virtuelle Webserver besteht.

3.2.3 Strukturierung

Zentrale Server/Services werden in dem Container ou=server,o=smartclient,c=de abgelegt. Server können aber auch für die Lokationen definiert werden. Für die Plus Geschäftsstelle wurde der dezentrale Server/Services in dem entsprechenden Bereich im LDAP eingetragen.

Die zentralen Server haben eine Doppelrolle:

Einmal dienen sie als zentrale Server/Services für alle Geschäftsstellen. Deshalb sind sie im Server Container abgelegt.

Einige Dienste (Datei- und Druckserver) stellen sie aber nur den Workstations am zentralen Standort bzw. Hauptverwaltung zur Verfügung. Deshalb wurde `nfs` und `cups` als Services in den Container der *hv* (Hauptverwaltung) eingetragen. Diese verweisen dann auf die Server im Service Container.

LDAP Attribut		Beispiel + Beschreibung
ipHostNumber		Hier wird die IP des Servers festgelegt, auf dem der jeweilige Service implementiert ist.
dbkDnsName		Ein Eintrag pro Service: ntp01, rsync01, tftp01, password01, dhcp01, syslog, localhost, popup01, etc.
dbkServiceName		Beschreibt die jeweiligen Services. In der Regel ist hiervon der *dbkDnsName* abgeleitet. Dient zur Information.
dbkServiceStatus	may	TRUE, Wenn hier nicht TRUE eingetragen ist, wird dbkDnsName auch nicht im DNS eingetragen. Hiermit können Dienste temporär „abgeschaltet“ werden, ohne sie aus dem LDAP löschen zu müssen.
dbkServiceEmail	may	Mail-Adresse für Fehlermeldungen bzgl. dieses Dienstes. Wird nicht automatisch ausgewertet.

Tabelle 3.2: Attribute der Objektklasse dbkService

Lastbalancierung per round-robin

gehört das hierher? fs Mit den Objektklassen dbkService und dbkService können sowohl vorwärts als auch rückwärts auflösende DNS round-robin Einträge erzeugt werden. Dies ist möglich, da ein Server mehrere Dienste durch einzelne Service-Objekte bereitstellen kann und jeder Server/Service mehrere IP Adressen erhalten kann.

3.3 Clients

3.3.1 Hinzufügen eines Client-Rechners

Normalerweise werden neue Client-Rechner automatisch durch das Perl-Modul `SmartClient::Leases2ldap` (siehe Kapitel 1.6.1 auf Seite 19) innerhalb des Programmes `sc_ldap2dnsdhcpd` dem LDAP hinzugefügt.

Dieses geschieht, sobald dieser Client eingeschaltet wird und die Netzwerkkarte eine DHCP Anfrage verschickt, um eine IP zu bekommen. Der DHCP-Server erkennt dieses und ermittelt für das Netz, in dem der Rechner steht, die dort nächste freie IP Adresse. Damit wird auch der Name des Rechner festgelegt und der Rechner in dem entsprechenden Netz im LDAP eingetragen.

Sollte die automatische Eintragung neuer Rechner nicht erwünscht sein, kann dies durch Zuweisen des Wertes `FALSE` an das LDAP Attribut *dbkDynamicIp* in dem Standards-Objekt des jeweilen Netzes deaktiviert werden.

Neue Rechner können auch manuell über das Tool `gq` eingetragen werden. Die nötigen Attribute sind in Tabelle 3.3 auf der nächsten Seite zusammengefasst. Im einzelnen sind die Attribute die folgenden:

macAddress Wie bei den Objektklassen dbkServer und dbkService.

dbkImageVersion Jede Workstation hat von dem eingespielten Image eine Image-Versionsnummer. Diese wird hier automatisch bei der Installation eingetragen. Zusätzlich kann mit diesem Attribut die Art der Installation gesteuert werden.

dbkCachingImageServer Gibt an, ob dieser Arbeitsplatz als Image-Server für andere Rechner genommen werden darf. Kann die Werte `TRUE` oder `FALSE` enthalten.

dbkStatus Wird von diversen Skripten als Statusfeld über diesen Rechner genutzt. So überschreibt beispielsweise jedes Update der Maschine diesen Eintrag mit der letzten erfolgreichen oder fehlgeschlagenen Aktion.

dbkSerialNumber Kann von Client-Skripten genutzt werden um eine Identifizierung der Workstation-Hardware zu ermöglichen. Das Attribut wird derzeit nicht von den Server-Skripten genutzt.

dbkErrorFlags *Wird derzeit in der Debeka als Schalter zur Steuerung der OVO-Probes genutzt.*

dbkDeltaImageVersion Wenn dem Client *Delta-Images* zugewiesen sind, so wird in diesem Attribut mitprotokolliert, welche Versions der jeweiligen Delta-Images installiert sind. Das Format des Attributes ist dabei *dn:version*. Wenn das Delta-Image den dn cn=delta,ou=global,o=smartclient,c=de und die Versionsnummer `123` hätte, so würde das Attribut bei installiertem Delta-Image den Wert `cn=delta,ou=global,o=smartclient,c=de:123` haben.

dbkStandardPrinterDn Zeigt auf den Drucker, der als Standard auf der Workstation eingestellt wurde. Siehe Kapitel 3.5.4 auf Seite 62.

dbkPrinterGroupDn Kann verwendet werden um eine Auswahl von Druckern einzuschränken. In den Druckergruppe können Drucker zusammengefasst werden, die für den Benutzer der Arbeitstation besonders intereressant sind.

userPassword Nimmt die Password-Credentials auf und ermöglicht es den Client-Skripten mithilfe des Passwort-Servers die eigenen Einstellungen anzupassen.

LDAP Attribut		Beispiel + Beschreibung
macAddress		Die MAC Adresse der Netzwerkkarte (`ip addr`)
dbkImageVersion		Die Versionummer des installierten Images.
dbkCachingImageServer		TRUE, FALSE – Wird hier nicht verwendet und muss deshalb auf FALSE stehen.
dbkStatus		Wird automatisch gesetzt.
dbkSerialNumber	may	
dbkErrorFlags	may	Wird automatisch gesetzt.
dbkDeltaImageVersion	may	Die Versionsnummern der installierten Delta Images. Wird automatisch gesetzt.
dbkStandardPrinterDn	may	Siehe Kapitel 3.5.4 auf Seite 62.
dbkPrinterGroupDn	may	Wird nicht immer verwendet.*wovon?*
userPassword	may	Wird vom Password-Server automatisch gesetzt.
dbkDeviceTypeDn	may	Zur Unterscheidung unterschiedlicher Image Typen
ipHostNumber	may	Falls die IP-Adresse nicht aus dem Namen der Workstation errechnet werden kann oder soll

Tabelle 3.3: Attribute der Objektklasse dbkWorkstation

dbkDeviceTypeDn Gibt Auskunft über den *devicetype* des Rechners. Dieser wird dazu genutzt, die Standards für den Arbeitsplatz aus der Lokation zu lesen und bei der Anlage den Namen des Rechners zu errechnen.

ipHostNumber In diesem Feld kann die IP-Adresse des Rechners eingetragen werden. Dies geschieht normalerweise durch die Server-Skripte bei der Neuanlage oder dem Umzug der Rechner und sollte nicht geändert werden.

Sollten grössere Mengen von Rechner aufgenommen werden, kann eine LDIF Datei mit den entsprechenden Informationen erstellt werden und diese mit einem `ldapadd` Kommando zum LDAP hinzugefügt werden.

3.3.2 Entfernen eines Client-Rechners

Ein Client-Rechner kann mit Hilfe des Tools `gq` auch aus dem LDAP-Verzeichnis gelöscht werden. Hierzu wählt man die Funkton `Delete` mit der rechten Maustaste auf dem entsprechenden LDAP-Eintrag (in der Rubrik GS) aus.

Wichtig: Damit der Rechner nicht automatisch wieder im LDAP eingetragen

wird, darf er nicht in der Datei `/var/lib/dhcp/dhcpd.leases` stehen. Dies ist bei festen-IP Adressen normalerweise auch nicht der Fall.

Wichtig: Wird ein Rechner aus LDAP gelöscht, so muss der Administrator dafür sorgen, dass Serverdienst `sc_ldap2dnsdhcpd` davon in Kenntnis gesetzt wird. Dafür stehen zwei *xmlrpc*-Schnittstellen zur Verfügung. Siehe hierzu auch Kapitel 4.4 auf Seite 91.

3.4 Monitore

Bei der automatische Einrichtung der Grafikunterstützung mittels `sc_hardware_X` (siehe Kapitel 2.2.1 auf Seite 33) wird der Monitor mittels DDC2 u. a. danach gefragt, welche Auflösungen er zur Verfügung stellt. Die höchste dieser Auflösung wird dann verwendet. Leider liefern manche Monitore nicht die optimalen Werte zurück. Manchmal liefern sie sogar Werte zurück, die sie nicht darstellen können. So ist beim Einsatz von 17 Zoll Monitoren eine Auflösung von 1024×768 zu empfehlen, und nicht 1280×1024.

Um hiermit umzugehen, können im LDAP Einstellungen für Monitore vorgenommen werden. *Beschreibung LDAP Einstellungen* Für Systeme mit mehreren angeschlossenen Monitoren ist dies sogar zwingend.

3.5 Drucker

3.5.1 Hinzufügen eines Druckertyps (Referenz-Drucker)

Druckerdefinitionen werden im `global` Container abgelegt. Dort sind die Eigenschaften des Druckers anhand der Attribute der Objektklasse dbkRefPrinter wie in Tabelle 3.4 auf der nächsten Seite beschrieben, zu befüllen. Die Attribute sind im einzelnen

dbkPrinterLanguage Art der Druckersteuerungssprache. Der Inhalt dieses Feldes wird derzeit von der Skripten nicht genutzt. Es enthält momentan die Werte `pcl` oder `ps`.

dbkDrivername Beschreibender Name des Druckertypes, der mit diesem Druckertreiber genutzt werden kann. Dient dem Administrator zur Übersicht und wird im Client als Druckertyp angezeigt.

dbkPpd Name der *Printer Page Description*-Datei (PPD), mit der der Drucker konfiguriert wird. Hier ist nicht der komplette Pfad zur Datei angegeben werden. Es reicht der einfache Dateiname, beispielsweise `HP2100_6.ppd`.

dbkPrinterDeviceUri Gibt an, wo die zu druckenden Daten vom Client hingeschickt werden sollen. In dem *Uniform Resource Interface* (URI) kann der Platzhalter `%dbkDnsName%` genutzt werden. Dieser Platzhalter wird bei Systemstart des Clients durch den DNS Namen des realen Druckers ersetzt. Für Netzwerkdrucker, die wie eine HP-JetDirect-Box angesprochen werden, kann der URI-Eintrag `socket://%dbkDnsName%:9100` genutzt werden.

LDAP Attribut		Beispiel + Beschreibung
dbkPrinterLanguage	must	`ps`, `bjc` – Druckersprache
dbkDrivername	must	`HP LaserJet 2100 Series` – Name des Druckertreibers für Windows
dbkPpd	must	`HP2100_6.ppd` – Printer Description Datei
dbkPrinterDeviceUri	must	`socket://%dbkDnsName%:9100` – das *Uniform Resource Interface*
dbkPrinterDeviceUriOptional	may	Optionales *Uniform Resource Interface*. Wird derzeit nicht ausgewertet

Tabelle 3.4: Attribute der Objektklasse dbkRefPrinter

Sollten feste Druckerserver eingesetzt und diese manuell konfiguriert werden, werden diese Attributen nicht ausgewertet.

3.5.2 Hinzufügen eines Druckers

Drucker sind Geschäftsstellen zugeordnet und werden auch dort angelegt.

Achtung:

Drucker müssen, bevor sie das erste Mal am Netzwerk angeschaltet werden, manuell in den LDAP eingetragen werden, da per DHCP Drucker nicht zuverlässig von Workstations unterschieden werden können.

Ein Drucker hat zwei Namen.

Der erste ist der durchnummerierte Name, der auch seine IP wiederspiegelt.

Der zweite Name ist beliebig und dient dazu den Drucker für den Anwender genauer zu beschreiben. Häufig wird der Typ zusammen mit der Position (Etage, Zimmer) verwendet.

Beispiele für durchnummerierte Namen können in Tabelle 3.5 auf der nächsten Seite nachgeschlagen werden.

d. h. IP-Adresse − Netzwerk-Adresse in der Darstellung als eine 5-stellige Dezimalzahl.

IP-Adresse	Netzmaske	relevante IP	Hexadezimal	Dezimal	Name
00.00.00.01	255.255.0.0	00.00.00.01	00000001	1	pr000001
00.00.01.00	255.255.0.0	00.00.01.00	00000100	256	pr000256
10.01.00.01	255.255.0.0	00.00.00.01	00000001	1	pr000001
10.01.01.00	255.255.0.0	00.00.01.00	00000100	256	pr000256
10.01.0E.05	255.255.0.0	00.00.0E.05	00000E05	3589	pr003589

Tabelle 3.5: Namensgebung durch IP-Adresse relativ zur Netzwerkadresse

Die Attribute des dbkPrinter Objektes sind wie in Tabelle 3.6 gezeigt zu setzen.

LDAP Attribut		Beispiel + Beschreibung
dbkRefPrinterDn	must	Verweist auf einen Referenz-Drucker, z. B. `cn=hp2100tn,ou=global,o=smartclient,c=de`
dbkDnsName	must	Zusätzlicher DNS-Name des Druckers.
macAddress	may	MAC Adresse des Druckers.
ipServicePort	may	Port-Nummer des Druck-Dienstes.

Tabelle 3.6: Attribute der Objektklasse dbkPrinter

Beim Einsatz von Druckservern dienen diese Einträge nur der Vollständigkeit, werden aber ansonsten nicht ausgewertet. Allerdings könnten diese Informationen genutzt werden, um einen dezentralen Linux (CUPS) Druckserver automatisch zu konfiguieren. Z.Zt. müssen diese Druckserver manuell (mit dem CUPS Webfrontend) konfiguiert werden.

3.5.3 Entfernen eines Druckers

Auch ein Drucker kann per Mausklick mit `gq` aus dem LDAP entfernt werden. Danach muss er auch aus dem Druckserver entfernt werden (CUPS Webfrontend). Es sollte darauf geachtet werden, dass dieser Drucker nicht als Standarddrucker verwendet wird.

Dies kann mit dem Tool `gq` unter `Search`→`Filter` mit folgendem LDAP-Filter ermittelt werden:

```
(dbkStandardPrinter=DRUCKERNAME)
```

3.5.4 Setzen von Standarddruckern

Zur Erfüllung von folgenden Anforderungen bzgl. der Bereitstellung von Druckern muss wie folgt vorgegangen werden:

- Alle Workstations einer Lokation sollen alle Drucker dieser Lokationen ansprechen können.
- Eine Lokation/Netzwerk hat einen Standarddrucker.
- Pro Workstation kann ggf. ein Standarddrucker definiert werden.
- Der Anwender hat die Möglichkeit, sich ggf. einen anderen Standarddrucker einzustellen.

Workstation

Pro Workstation kann ein Standarddrucker in dem Attribut *dbkStandardPrinterDn* definiert werden. Dieser Druckername muss bei einem der Drucker dieses Netzwerkes als *dbkDnsName* Attribut definiert sein.

Beim nächsten Booten der Workstation oder durch Aufruf von

```
/etc/init.d/sc_ldap2defaultprinter start
```

wird die Umgebungsvariable `PRINTER` entsprechend im globalen Profil gesetzt.

Wenn bei einer Workstation der LDAP-Eintrag des Standardprinters gelöscht wird, bleibt dieser bis zum nächsten Update der Workstation bestehen. Der direkte Austausch des Standarddruckers bei einer Workstation wird jedoch unmittelbar vollzogen (ohne Update).

Die Attribute *dbkStandardPrinterDn* und *dbkPrinterGroupDn* dienen zum Aufsetzen von Druckerservern bzw. Druckergruppen.

Netzwerk

Wenn eine Workstation keinen eigenen Standarddrucker definiert hat, wird der Standarddrucker des Netzwerks (Geschäftsstelle) verwendet. Der Standarddrucker kann für den Druckserver mit dem Kommando

```
lpadmin -h SERVERNAME -d STANDARDDRUCKER??
```

festgelegt werden.

Alternativ kann auch hierfür das CUPS Webfronend verwendet werden.

3.6 Benutzer

Benutzerdaten werden im LDAP gepflegt. Die Attribute der Objektklassen posixAccount und shadowAccount sind in den Tabellen 3.7 und 3.8 aufgelistet.

LDAP Attribut		Beispiel + Beschreibung
cn	must	Benutzerkennung
uid	must	Benutzerkennung/Login (identisch mit cn)
uidNumber	must	Benutzernummer
gidNumber	must	primäre Gruppennummer
homeDirectory	must	Benutzer-Heimatverzeichnis
userPassword	may	Benutzerpasswort (verschlüsselt)
loginShell	may	Login-Shell (z. B. `/bin/bash`)
gecos	may	Beschreibung des Benutzers
description	may	zusätzliche Beschreibung

Tabelle 3.7: Attribute der Objektklasse posixAccount für POSIX-Accounts von Benutzern

LDAP Attribut		Beispiel + Beschreibung
shadowLastChange	must	Datum (als „Tage seit dem 1.1.1970“), an dem das Passwort das letzte Mal geändert wurde
shadowMin	must	Tage bevor das Passwort geändert werden darf
shadowMax	must	Tage nach denen das Passwort geändert werden muss
shadowWarning	must	Tage vor Ablaufen des Passworts, wann gewarnt wird
shadowInactive	may	Tage nach Ablaufen des Passworts, bis der Account gesperrt wird
shadowExpire	must	Datum (als „Tage seit dem 1.1.1970“), an dem der Account gesperrt wird
shadowFlag	must	reserviertes Feld

Tabelle 3.8: Zusätzliche Attribute für shadow-Passwörter in der Objektklasse shadowAccount

Zur Erfüllung der folgenden Vorgaben ist die folgende Konfiguration notwendig:

Nach dem Neuanlegen eines Benutzers soll dieser sein Passwort direkt ändern müssen (*shadowLastChange* muss daher beim Anlegen eines Users auf 32 Tage in die Vergangenheit gesetzt werden).

Danach soll die Gültigkeit 30 Tage betragen, 3 Tage vor Verfallen soll eine Warnung erzeugt werden, die endgültige Sperrung nach Ablauf des Passworts soll nach 120 Tagen erfolgen.

Dies führt zu folgenden Einstellungen:

shadowLastChange z. B. 11829 (für 21.06.2002 - 30 Tage)

shadowMin 0

shadowMax 30

shadowWarning 3

shadowInactive 120

shadowExpire -1

Die eigentlichen Home-Verzeichnisse liegen auf dem entsprechenden Dateiserver.

3.6.1 Benutzer hinzufügen

Mit dem Skript `adduser.sh` können Benutzer angelegt werden. Die Parameter werden per Dialog abgefragt. Der Dialog kann an die Bedürfnisse des Kunden angepasst werden. Da auch der Dateiserver abgefragt wird, kann das Skript an zentraler Stelle ausgeführt werden, so das das Skeleton-Verzeichnis nicht auf mehrere Server synchronisiert werden muss. Der Ablauf von `adduser.sh` stellt sich folgendermaßen dar:

- Dialog mit Abfrage der Daten über den neuen Benutzer
- Eintragung des neuen Benutzers in LDAP
- Anlegen des Home-Verzeichnisses und kopieren des Skeleton auf den Dateiserver
- evtl. Aufruf von Skripten zur Anpassung von Einstellungen auf dem Dateiserver

Beispieldialog

Ein Beispieldialog von `adduser.sh` (Benutzereingaben sind kursiv dargestellt):

```
server04:~ # adduser.sh
Login Name: test01
Userid number [1003]:
Nachname: Testmeier
Vorname: Egon
Planstelle: 123/45
NFS-Server: nfs.gs0001.gs.smartclient.de
```

Beschreibung der einzelnen Eingaben:

Login Name: Benutzerkurzname, der beim Einloggen benutzt wird. Darf keine Sonderzeichen und Umlaute enthalten, nur a-z und 0-9.

Userid number: Numerische Benutzer ID. Das Skript ermittelt durch eine LDAP Anfrage die nächste freie Benutzer ID, die als Vorgabewert benutzt wird, wenn an dieser Stelle nichts eingegeben wird (nur Enter).

Nachname, Vorname, Planstelle: Diese Daten werden im LDAP in das sog. *Gecos* Attribut eingetragen

NFS-Server: NFS-Server auf dem das Home-Directory des Benutzers liegen soll. Dieser Servername wird auch in den LDAP Eintrag des Benutzers (Attributname *Host*) aufgenommen, hat jedoch zunächst rein informativen Charakter.

Anschließend werden die Daten des neuen Benutzer-Accounts im LDIF Format zur Kontrolle dargestellt. Der Eintrag wird erst nach der Bestätigung in den LDAP-Server geschrieben.

3.6.2 Skeleton Verzeichnis

Beim Anlegen eines Benutzers mit `adduser.sh` wird das Skeleton Verzeichns `/etc/skel` per `rsync` über `ssh`[1] auf den im Dialog spezifizierten NFS-Server in das Home-Verzeichnis des Benutzers kopiert.

Dieses Skeleton Verzeichnis sollte nur die absolut notwendigen Benutzerkonfigurationen enthalten. Wann immer möglich, sollte auf Systemvorgaben gesetzt werden, die auf dem Referenzrechner angelegt sind.

3.6.3 Nachträgliches Veränderungen in den Benutzer-Verzeichnissen

Weit schwieriger als das Anlegen eines Benutzers mit Standardwerten (Skeleton) sind nachträgliche Veränderungen. Es ist natürlich möglich, in jedem Homeverzeichnis manuelle Anpassungen vorzunehmen. Für einige Fälle kann jedoch das Skript `distribute_skel.sh` nützlich sein. Damit kann man entweder für alle Benutzer das gesamte Homeverzeichnis durch das Skeleton ersetzen (Achtung: alle bestehende Dateien werden gelöscht!), dasselbe für einzelne Benutzer, oder einzelne Dateien oder Verzeichnisse für alle oder einzelne User hinzufügen oder löschen.

```
usage: distribute_skel.sh -a|-u <username> [-f <file>]|[-d <dir>] [-t]
    -a            : all users
    -u <username> : copy skel only for the specified user
    -f <file>     : copy only the specified file
```

[1] hier wird `rsync` über `ssh` verwendet, u. a. da `scp` keine Links kopieren kann

```
-d <dir>      : copy only the specified directory
-t            : dry run (only shows what would be done)
-r            : remove the file or directory specified with -f or -d

Files or Directories specified with -f or -d must always be relative
path names.
```

Das Skript funktioniert z.Zt. nur pro Dateiserver, d. h. es muss zunächst das aktuelle Skeleton `/etc/skel` vom zentralen Server auf den jeweiligen NFS-Server kopiert werden.

3.6.4 Benutzerdaten ändern

Benutzerdaten können im LDAP geändert werden, z. B. mit `gq`.

Passwort zurücksetzen

Zum Ändern des Benutzerpasswortes wählt man den entsprechenden User an und schreibt in das Feld *userPassword* im Klartext das neue Passwort. Dabei ist darauf zu achten, dass die Passwortverschlüsselung auf `Crypt` steht. Die Änderung wird per `Apply` zum LDAP geschickt. Beim nächsten Betrachten (oder durch ein `Refresh`) sieht man dann den per Crypt verschlüsselten Passwort-Eintrag.

Passwort-Änderung für Benutzer erzwingen

Der Benutzer wird in regelmäßigen Zeitabständen dazu aufgefordert, sein Passwort zu ändern. Dies kann auch dazu verwendet werden, eine Passwort-Änderung des Benutzers zu erzwingen.

Zuständig dafür ist die Objektklasse shadowAccount. Dort ist der Wert *shadowMax* festgelegt, der angibt, wie viele Tage ein Passwort maximal gültig ist. Standard sind im Beispiel 30 Tage.

Das Datum der letzten Passwortänderung ist in der Variablen *shadowLastChange* im Format „Tage seit dem 1.1.1970“ gespeichert. Diesen Wert erhält man durch das Skript `days.sh`, welches ohne Parameter den Wert für heute liefert. Möchte man den Wert für den „19.07.2002“ haben, ruft man `days.sh 20020719` auf. Durch das Setzen des Wertes auf einen Tag, der mehr als 32 Tage zurück liegt (es können Ungenauigkeiten wegen Tageszeit und Zeitzonen auftreten), kann eine Passwortänderung des Benutzers erzwungen werden. Man sollte den Wert aber nicht zu weit zurücksetzen, da nach einer bestimmten Zeit nach Ablauf des Accounts der Zugang vollständig gesperrt wird (im Beispiel 120 Tage). Dann müsste der Administrator den *shadowLastChange*- Wert erst einmal wieder auf einen gültigen Wert stellen. *abschnitt nach shadow. Verweis auf shadow*

3.6.5 Benutzer entfernen

Benutzer können gelöscht werden, indem sie einfach aus dem LDAP entfernt werden. Danach kann auch ihr Home-Verzeichnis auf dem entsprechenden Dateiserver gelöscht werden.

Alternativ kann man auch den Zugang ab einem bestimmten Datum über das LDAP Attribut *shadowExpire* sperren.

3.7 Geschäftsstellen

3.7.1 LDAP

Für das Anlagen einer neuen Geschäftsstelle muss im LDAP ein entsprechender Container unterhalb von `ou=GS,o=smartclient,c=de` hinzugefügt werden.

Wesentlich ist das Anlegen des Objektes `standards` unterhalb der neuen Geschäftsstelle. Dieses besteht hauptsächlich aus der Objektklasse dbkLocation.

Mit der Objektklasse dbkUpdate kann innerhalb der `standards` für eine Geschäftsstelle (oder einer Region) auch unterschiedliches Update-Verhalten definiert werden.

Folgende Attribute können gesetzt werden:

wird dbkPrinterGroupDn verwendet? MAY? Dummywert?

3.7.2 Dezentraler Server

Das Aufsetzen des lokalen Servers erfolgt wie im Kapitel 1.7 auf Seite 25 beschrieben.

3.8 Standards für verschiedende Geräte

3.8.1 Definition dbkDeviceType

Für die Unterscheidung verschiedener Gerätetypen ist die Objektklasse dbkDeviceType eingeführt worden. Die neuen Attribute für diese Objektklasse sind in Tabelle 3.11 auf der nächsten Seite zu sehen. Die genaue Funktion der Attribute *dbkObjectIdFormatString* und *dbkObjectIdGenerator* werden in Kapitel 3.8.1 auf Seite 70 eingehender erläutert.

LDAP Attribut		Beispiel + Beschreibung
ipNetworkNumber	must	Netzwerk IP 10.1.0.0
ipNetmaskNumber	must	Netzwerk Maske 255.255.0.0
dbkDhcpRange	must	dynamische IP Adressen, die temporär für neue Workstation verwendet werden. 10.1.1.101, 10.1.1.199
dbkDhcpFixedRange	must	IP Adressen die fest zu Workstations zugeordnet werden 10.1.1.1, 10.1.1.99
dbkDefaultGw	must	Default Gateway 10.1.110.3
dbkDynamicIp	must	aktivieren des dynamischen IP Bereiches. Ohne diese können neue Workstation nicht automatisch eingerichtet werden. TRUE (FALSE)
dbkPrinterGroupDn	must	Standard Druckergruppen. Da es ein MUST Attribut ist, muss es mit einen Dummy Wert gefüllt werden, falls es nicht verwendet werden soll.

Tabelle 3.9: Attribute der Objektklasse dbkLocation

In dem globalen Konfigurations-Zweig können jetzt Objekte mit der Objektklasse dbkDeviceType angelegt werden. Diese Objekte können die folgenden Attribute enthalten:

description Enthält eine kurze Zusammenfassung, welche Geräte mit diesem „DeviceType“ markiert werden sollen.

dbkAuthorName Sollte mit dem Kürzel des Erstellers des Objektes befüllt sein.

dbkObjectIdFormatString Dieser String enthält die Formatierungsanweisungen für die Benennung von neuen Objekten in LDAP. Der String kann Platzhalter der Form `${ID_DES_GENERATORS}` enthalten. Hierbei muss die gewählte Id (`ID_DES_GENERATORS`) in einem Wert des Feldes *dbkObjectIdGenerator* vorkommen. Ein Beispiel für die jetzigen Standardbenennung von Arbeitsstationen wäre `pc${pcid}`.

dbkObjectIdGenerator Für jede Id in dem Feld *dbkObjectIdFormatString* muss eine Generator-Konfiguration in diesem Feld eingetragen werden. Dabei ist das generelle Format `type=`*typ*`,id=`*id*`[,format=`*format*`]`. Die verschiede-

LDAP Attribut		Beispiel + Beschreibung
dbkDoTftpBoot	must	von welchem Rechner wird übers Netz gebootet (LOCALBOOT: immer von Festplatte booten, WAN: zentraler Server (`tftp.smartclient.de`), Rechnername)
dbkDoRsync	must	von welchem Rechner werden Updates (per rsync) geholt WAN (NONE, Rechnername)
dbkUpdateTime	must	Zeit für einen automatischen Update. 0
dbkTimeout	may	Timeout bei einem automatischen Update

Tabelle 3.10: Attribute der Objektklasse dbkUpdate

nen Generator-Typen sind in Kapitel 3.8.1 auf der nächsten Seite aufgeführt. Implementierte Generatoren sind derzeit die Typen `IP` und `UNIQUE`. Die Id kann frei gewählt werden, sollte aber dem regulären Ausdruck

```
^[a-zA-Z]+[a-zA-Z_0-9]*$
```

genügen. Das Format ist das *sprintf* Format und hat als Voreinstellung den Wert `%05d`.

Attribut	*Typ*	*Spezielles*	*Beschreibung*
dbkDeviceTypeDn	DN	single-value	Referenz auf ein Objekt zur Typisierung der Arbeitsstation
dbkObjectIdFormatString	String	single-value	Spezieller Format String zur Erstellung des *cn* der Arbeitsstation.
dbkObjectIdGenerator	string	multi-value	Aufzählung und Konfiguration der benutzten Generatoren.

Tabelle 3.11: Neue Attribute für die Typisierung und Benennung von Arbeitstationen

Namensvergabe durch den Namensgenerator

Wird ein Name für ein neues Objekt – z. B. eine Workstation – gesucht, so sucht der Namensgenerator zuerst nach dem konfigurierten dbkDeviceType. In diesem Objekt sind die Regeln für den Namensgenerator hinterlegt. Die Werte in dem

multi-value Attribut *dbkObjectIdGenerator* spezifizieren die zu verwendenden Generatortypen und deren zu nutzende Id's. Das Attribut *dbkObjectIdFormatString* gibt die Position der Werte aus den einzelnen Generatoren an. Als Platzhalter für die gewählten Generatortypen dient ein String der Form `${ID}`. Die Zeichenkette ist dabei frei wählbar, und muss in einem der Generatortypen ebenso als Id angegeben sein. Generatoren des Types `UNIQUE` werden als letztes ausgewertet. Pro Generator darf nur ein `UNIQUE`-Generatortyp verwendet werden. Die Reihenfolge der anderen Generatoren sollte keine Rolle spielen und kann nicht beeinflusst werden.

Generatortypen

Die verschiedenen Generatortypen sind:

IP Errechnet aus der IP-Adresse des Objektes und der Netzwerkeinstellungen der Lokation eine relative Nummer. Diese wird in dem gewünschten Format zurückgegeben. Als Parameter kann dieser Generatortyp die Attribute `type`, `id` und `format` haben. Das Attribut `format` ist optional und hat – wenn nicht anders angegeben – den Wert `%05d`.

UNIQUE Dieser Generatortyp ersetzt den Platzhalter (z. B. `${hvpc}`) durch den LDAP-Platzhalter `*` und nutzt den so entstandenen String als Filter um im LDAP-Datenbestand nach allen ähnlichen Objekten zu suchen. Danach wird über die mögliche Wertemenge (angegeben durch die Parameter `min` und `max`) iteriert. Dir jeweils aktuelle Wert wird per `format` formatiert und an die Stelle des Platzhalters gesetzt. Ist der so erzeugte String eindeutig, so wird die so gewonnene Zeichenkette zurückgegeben.

Der LDAP Suchfilter kann mit dem Parameter `attribute` noch weiter angepasst werden. Dieser Parameter gibt an, welches LDAP Attribut durchsucht werden soll. Ist dieser Parameter nicht angegeben, so wird das Attribut *cn* angenommen.

LDAP Mit diesem Generatortypen kann auf Attribute aus dem LDAP-Baum bezug genommen werden. Dafür sind die Parameter `attribute` und `scope` anzugeben. Der Parameter `scope` ist optional und kann z. Zt. als alleinigen Wert `treesearch` beinhalten. Mit dem Parameter `attribute` kann ein LDAP-Attribut angegeben werden, dass mittels *treesearch* ausgehend vom – zu dem Zeitpunkt noch **nicht** existierenden Objektes – gesucht werden soll. Der Wert des gefundenen Attributes wird per `format` formatiert zurückgegeben. Wird kein Parameter `format` übergeben, so wird `%s` genutzt.

XMLRPC Dieser Generatortyp nutzt XMLRPC, um einen Namen zu generieren. Als Parameter sind die folgenden Werte zugelassen:

url bezeichnet die URL unter der der Namensgenerator aufgerufen werden kann.

method benennt die XMLRPC Methode, die der Server unter URL anbietet.

realm, user, pass Falls der XMLRPC Server zur Authentifizierung irgendwelche Credentials benötigt, so können diese mit diesen Parametern angegeben werden.

format kennzeichnet wie gehabt die gewünschte Formatierung für den Platzhalter und ist wie bei dem LDAP-Generatortypen mit `%s` voreingestellt.

KEY Eventuell benötigt der XMLRPC Server noch mehr Parameter. Diese können frei gewählt werden und als *KEY* in der Konfiguration zu diesem Generatortypen hinterlegt werden. *KEY* soll hier also als Platzhalter für einen tatsächlich benötigten Wert verstanden werden, so wäre es z. B. denkbar, dass der XMLRPC Server auf verschiedene LDAP Server zugreifen kann und den zu nutzenden LDAP Server in dem Attribut LDAPSERVER erwartet. Dann wäre ein *KEY* also `LDAPSERVER`.

Dem XMLRPC Server wird ein *XMLRPC-struct* übergeben. In dieser Datenstruktur sind alle angegeben Parameter als Schlüssel/Werte-Paar enthalten. Zusätzlich werden die folgenden Schlüssel mit Werten befüllt:

generatorId Ist mit der Id des Generatortyps befüllt.

generatorIdFormat Enthält das für diesen Generatortypen gewählte `format`.

generatorIdFormatstring Ist mit dem bis zu diesem Generatortypen ausgwerteten Formatstring befüllt.

ldapSearchBase ist gleich der Suchbasis für das gesamte SmartClient-Umfeld.

ipAddress Enthält die IP-Adresse, die das neu anzulegende Objekt erhalten wird.

netmask, network Bezeichnen die Netzwerkeinstellungen der Lokation, in die das neue Objekt einsortiert werden soll.

Beispiele für Belegungen der Formatstrings und deren Generatoren

In diesem Abschnitt sollen einige Beispiele für die Verwendung der Namensgeneratoren gegeben werden. Die Quellabschnitte sind im LDIF ähnlichen Format angegeben und sind in den Objekten der Klasse dbkDeviceType einzufügen.

Standard SmartClient Workstation Üblicherweise wird der Standardname für eine Workstation aus der IP Adresse und den Netzwerkeinstellungen der Lokation errechnet. Dies Verhalten kann mit dem Generatortyp `IP` nachgebildet werden.

```
dbkObjectIdFormatString: pc${gsip}
dbkObjectIdGenerator: type=IP,id=gsip,format=%05d
```

Diese Konfiguration würde Namen der Art *pc00256* erzeugen.

Regionalleitung Dies sind Arbeitsstationen, die in Kleinstgeschäftsstellen eingesetzt werden. Der Name der Arbeitsstationen wird aus der sogenannten Werbebereichsnummer gebildet. Zusätzlich sollen die Namen aber auch eindeutig sein. Das kann durch die Kombination von zwei Generatoren erreicht werden.

```
dbkObjectIdFormatString: rl${wb}${ziffer}
dbkObjectIdGenerator: type=LDAP,id=wb,format=%04d,attribute=ou
dbkObjectIdGenerator: type=UNIQUE,id=ziffer,format=%c,min=97,
 max=122
```

Diese Konfiguration würde zu Namen der Art *rl1234a* führen.

3.9 Boot-Einstellungen

3.9.1 Referenz Boot-Einstellungen

Die Referenz Boot-Einstellungen können mit der Objektklasse scRefBootOptions aus Tabelle 3.12 auf der nächsten Seite vorgenommen werden.

Diese Einstellungen werden primär von `ldap2dhcp` verarbeitet.

3.9.2 Zuweisen von Boot-Einstellungen

Über das Setzen von *scRefBootOptionsDn* können für alle Client-Rechner, Server und Drucker individuelle Boot-Einstellungen vorgenommen werden.

3.10 Referenz Images

Dieses Kapitel beschreibt den Umgang mit Referenz Images. Grundsätzlich gehören hierzu folgende Komponenten:

- Referenz Rechner (z.B. `refpc.smartclient.de`)
- Referenz Image LDAP Eintrag (z.B. `cn=refImage,ou=global,o=smartclient,c=de`)
- Rsync Server (z.B. `rsync.server.smartclient.de`)
- Referenz Images (-old, current, -new, -full) auf dem Rsync Server

LDAP Attribut		Beispiel + Beschreibung
dbkDhcpOptionsRemote	must	wenn das System nicht die aktuelle Software Version besitzt, wird diese DHCP Option für die entsprechende MAC-Adresse erzeugt filename “pxelinux.0“;
dbkDhcpOptionsLocal	must	wenn das System die aktuelle Software Version besitzt, wird diese DHCP Option für die entsprechende MAC-Adresse erzeugt filename “LOCALBOOT“;
scPxeCfgFileRemote	may	wenn das System die nicht aktuelle Software Version besitzt, wird im TFTP Verzeichnis wird ein Link (Namen = MAC-Adresse) zu dieser Datei erzeugt netboot
scPxeCfgFileLocal	may	wenn das System die aktuelle Software Version besitzt, wird im TFTP Verzeichnis wird ein Link (Namen = MAC-Adresse) zu dieser Datei erzeugt localboot

Tabelle 3.12: Attribute der Objektklasse scRefBootOptions

- Referenz Image Status Datei im Referenz Image (`/etc/smartclient/base:ALL_IMAGE_STATUS_FILE` bzw. `/etc/smartclient/base:ALL_IMAGES_STATUS_PATH` + cn für Delta Images)

3.10.1 Ablauf

- Erzeugung eines RefImage LDAP Eintrags (einmalig)
- Erzeugen einer neuen RefImage Version (kopieren eines Referenz Rechners zum Rsync Server)
  - Daten werden vom Referenz Rechner zum Rsync Server in das `-full` Image kopiert (ohne Auswertung der Exclude-Liste)
  - Daten werden von `-full` nach `-new` kopiert (unter Auswertung der Exclude-Liste)
  - ggf. manuelles setzen der neuen Version im LDAP Eintrag
- Aktivierung einer neuen Image Version
  - Aktuelles Image wird nach `-old` gesichert
  - Daten aus `-new` werden aktivert
  - Die Image Status Datei wird ausgewertet und der zugehörige LDAP-Eintrag wird aktualisiert
- Distribution einer Image Version auf dezentrale Server (optional)

3.10.2 Referenz Image Eintrag

Vorbereitet zur Nutzung eines Referenz Images muss ein entsprechender LDAP Eintrag erzeugt werden. Globale RefImage Definitionen werden im LDAP in dem Kontainer

`ou=global,o=smartclient,c=de`

abgelegt.

Für ein Referenz Image ist durch die `dbkRefImage` bestimmt.

Rsync Pfade

Dort enthaltene Pfadangaben beziehen sich auf die im Rsync Server konfiguierten Module. Diese sind unter `/etc/rsyncd.conf` definiert und müssen auch unter `/etc/smartclient/rsync:RSYNC_PATH` eingetragen sein.

Im Normalfall sind sie auf `/var/lib/smartclient/rsync/` gesetzt.

Versionsnummern

Die Versionsnummern sollten folgendermaßen gewählt werden: JJJJMMTTVV wobei

Format	Bedeutung
JJJJ	4-stellige Jahreszahl
MM	2-stelliger Monat
TT	2-stelliger Tag
VV	2-stellige Tagesversionsnummer

Die Tagesversionssnummer beginnt bei 01 und wird benötigt, wenn es an einem Tag mehr als eine Image-Version gibt.

Dieses Format wurde analog zur Konvention bei DNS Servern gewählt.

LDAP Eintrag

LDAP Attribut		*Beispiel + Beschreibung*
dbkPartitionsTable	must	Partitionstabelle. Zur Zeit werden nur die ersten vier primären Partitionen und ext2/3 Dateisysteme unterstützt. Dies kann bei Bedarf leicht erweitert werden. `1000 82 swap swap;4000 83 / ext3`
dbkInitRdScript	must	nach dem booten der Standard Initialen RAM Disk (initrd) übers Netzwerk, wird dieses Skript nachgeladen und ausgeführt. Dadurch könnten unterschiedliche Verhaltensweisen für die verschiedenen Images definiert werden. Das hier angegebene ist das Standard Skript: `config/initRdScripts/initRdScript.sh`
dbkExcludeList	must	Für den Update eines Images können bestimmte Dateien/Verzeichnisse ausgeschlossen werden. Mit der jetzt eingestellten werden einfach alle Dateien des Root Dateisystems übertragen. In Zukunft würde es Sinn machen z. B. die Dateien aus `/var/log/` auszuschliessen. Dies spart nicht nur Bandbreite, sondern würde das Zielsystem auch konsistenter halten. Excludelisten werden direkt von dem Program `rsync` ausgewertet. `config/excludelists/excludelist.full`
dbkImageLocation	must	Pfadangabe auf dem Rsync Server des eigentlichen Image Verzeichnisses. Der Zielpfad (als direktes Unterverzeichnis von `images`) wird automatisch angelegt. `images/refImage`
dbkImageVersion	must	Version des Images. 2002070503
dbkImageSize	must	Ungefähre Grösse des Images. Diese Angabe wird z .Zt. nicht ausgewertet. 1200
dbkExcludeListRunning	may	für zukünftige Updates im laufenden Betrieb

Image Status Datei

Die Image Status Datei beschreibt innerhalb eines Images deren Status. Sie dient auch zur Zuordnung zwischen RefImage LDAP Eintrag und physikalischen Images auf dem Rsync Server bzw. im laufenden System.

Die Image Status Datei für Komplett-Images ist durch die Variable `/etc/smartclient/base:ALL_IMAGE_STATUS_FILE` definert. Normalerweise hat sie den Wert `/etc/smartclient/images/refImage`.

Da durchaus mehrere Delta-Images auf einen System installiert werden können, wird der Name ihrer Status Datei gebildet durch den Pfad, der in der Variable `/etc/smartclient/base:ALL_IMAGES_STATUS_PATH` definert ist und dem *Common Name* (*cn*) Attribut des Delta-Images als Dateiname.

Eine Image Status Datei beinhaltet folgende Attribute:

	Image		
Attribut	*Referenz*	*Delta*	*Beschreibung*
SC_REFIMAGE_PC	C	–	refpc.smartclient.de
SC_REFIMAGE_CN	C	M	refImage
SC_REFIMAGE_DN	C	M	cn=refImage,ou=global,o=smartclient, c=de
SC_REFIMAGE_LOCAL_DN	A	A	cn=refImage,ou=config,ou=gs0010,o=smartclient,c=de
SC_REFIMAGE_VERSION	C (A)	M	2006102602
SC_REFIMAGE_LOCATION	C	M	images/refImage
SC_REFIMAGE_SIZE	C/A	A	5019

Dabei gibt die zweite bzw. für Delta–Images die dritte Spalte an, wie die Informationen in die Image Status Datei kommen:

M manuell einzutragen

C wird von `sc_rsync.client2server.pl` erzeugt, d.h. nur für komplett Images

A wird von `sc_activate_image.pl` ergänzt, d.h. erst bei der Aktivierung eines (Delta–)Images

3.10.3 Erzeugen einer neuen RefImage Version

Die Erzeugung einer neuen Version eines Referenz Images ist ein mehrstufiger Prozess. Im ersten Schritt werden die Daten vom Referenz Rechner zum Rsync Server übertragen. Um ein Image auszurollen, muss es überdies noch aktiviert werden.

Zur Erzeugung einer neuen Version dient das Skript `sc_rsync.client2server.pl`. Dieses wird auf dem Rsync Server ausgeführt und kopiert in Inhalt des Referenz Rechners an vorgesehende Stelle. Der Pfad zum Rsync Bereich wird über die Variable `/etc/smartclient/rsync:RSYNC_PATH` ermittelt, der genaue Image Pfad über den Referenz Image LDAP Eintrag.

Aufruf:

```
sc_rsync.client2server.pl --refpc refpc.smartclient.de --refimage "cn=refImage,o
c=de" --version 2006102602
```

Ablauf:

- Daten werden vom Referenz Rechner zum Rsync Server in das `-full` Image kopiert (ohne Auswertung der Exclude-Liste)
- Daten werden von `-full` nach `-new` kopiert (unter Auswertung der Exclude-Liste)
- Ermittelte Daten zum Image werden in innerhalb des `-new` Images in einer Status Datei hinterlegt (normalerweise `/etc/smartclient/images/refimage`). Siehe Kapitel 3.10.2 auf der vorherigen Seite.
- ggf. manuelles setzen der neuen Version im LDAP Eintrag

3.10.4 Aktivierung einer neuen Image Version

Bei der Aktivierung wird ein schon vorher erzeugtes Referenz Image „scharf“ geschaltet, d.h. ggf. führt dies auch zur Aktualisierung aller Workstations.

Zuständig hierfür ist das Skript `sc_activate_image.pl`.

Aufruf:

```
sc_activate_image.pl --admin "cn=admin,o=smartclient,c=de" --password ⟨PASSWORT⟩
"cn=refImage,ou=global,o=smartclient,c=de"
```

Ablauf:

- aktuelles Image wird nach `-old` gesichert
- Daten aus `-new` werden aktivert
- die Image Status Datei wird ausgewertet und der zugehörige LDAP-Eintrag wird aktualisiert

3.10.5 Distribution einer Image Version auf dezentrale Server

(optional)

3.11 Software-Aktualisierung eines Client-Rechners

Um eine Aktualisierung durchzuführen, muss eine Workstation übers Netzwerk booten.

Dieses Verhalten wird über LDAP Attribute gesteuert.

Diese werden von `ldap2dhcp` (siehe Kapitel 1.6.1 auf Seite 18) ausgewertet.

3.11.1 dbkUpdate in standards

Normalerweise besitzt eine Workstation nur die Objektklasse dbkWorkstation und nicht die Objektklasse dbkUpdate.

Das Attribut *dbkDoTftpBoot* der Objektklasse dbkUpdate (siehe Tabelle 3.10 auf Seite 69) im `standards` Objekt des Netzes stellt ein, ob Workstation überhaupt übers Netz booten dürfen. Standardmässig zeigt dieser Wert auf den lokalen TFTP-Server.

Allerdings wird die DHCP Konfiguration nur für die Rechner auf „Booten übers Netzwerk“ gestellt, die nicht die aktuelle Image Version besitzen.

Dies wird über den Eintrag *dbkImageVersion* in der jeweiligen Workstation festgelegt. Wenn die Versionsnummer mit dem aktuellen Image übereinstimmt, wird eingestellt, dass der Rechner über seine Festplatte bootet, ansonsten bootet er über den in *dbkDoTftpBoot* angegebenen Server.

Hat ein Rechner erstmal über das Netz gebootet, wird als nächstes über den Eintrag *dbkDoRsync* festgelegt, von welchem Server er sein Update bezieht.

Nach erfolgreichem Update schreibt die Workstation ihre neue Versionsnummer zurück in den LDAP und die Workstation booten beim nächsten Reboot von ihrer Festplatte.

3.11.2 Aktualisierung eines Client-Rechners erzwingen

Es kann vorkommen, das eine Workstation, obwohl sie eigentlich die aktuelle Image Version besitzen solten, sich nicht wie gewünscht verhält (z. B. überhaupt nicht mehr von Festplatte booten kann).

In diesem Fall kann man den Eintrag *dbkImageVersion* eines Workstation Objektes auf 0 setzen.

Damit bootet die Workstation beim nächsten Starten übers Netzwerk und führt einen Update durch.

Setzt man die Versionsnummer auf -1 so löscht die Workstation vor dem Update die gesammt Festplatte. Dies dauert wesentlich länger als ein normaler Update,

allerdings kann man danach absolut sicher sein, dass sich keine *nicht* im Referenz Image vorhandenen Informationen mehr auf der Workstation befinden.

dbkUpdate in der Workstation Objektklasse

Sollten in einem Netzwerk Updates deaktiviert sein, was mit `dbkDoTftpBoot=LOCALBOOT` in den `standards` der Lokation beschrieben ist, oder möchte man sein Update von einem anderen als dem Standardserver beziehen, kann man für einzelne Workstation den Update-Mechanismus einstellen.

Dazu fügt man die Objektklasse dbkUpdate zu einer Workstation hinzu. In diesem Fall wird die Image-Versionsnummer der Workstation für das Booten über das Netz auch nicht betrachtet, sondern allein der Wert von *dbkDoTftpBoot*.*stimmt das?* Bei einem Update wird zusätzlich zum Zurückschreiben der Versionsnummer dieser Wert auf `LOCALBOOT` gesetzt.

An einem geschäftsstellenweiten Update würde die Workstation mit eigener dbkUpdate-Objektklasse nicht teilnehmen. Dazu müsste diese Klasse erst wieder entfernt werden.

Zeitgesteuertes Update

Sollten lokale Server zur Verfügung stehen, so ist ein zeitgesteuertes Update nicht notwendig.

Aktualisierung aus Sicht einer Workstation

Sollte die Workstation zum erstenmal eingeschaltet werden, d. h. sie ist noch nicht im LDAP eingetragen, so muss das Attribut *dbkDynamicIp* der Geschäftstelle oder der globalen Standards auf `true` gestellt sein, damit der Client zunächst eine dynamische IP erhält.

1. Client wird angeschaltet
2. erhält IP (dynamische, wenn noch nicht im LDAP, sonst statische)
3. PXELINUX Bootloader wird per TFTP geladen
4. PXELINUX lädt Konfigurationsdatei entsprechend der eigenen MAC-Adresse, oder die Datei `default` falls keine individuelle Datei vorhanden ist.
5. Kernel + initrd werden per TFTP übertragen.
6. Die Datei `/linuxrc` aus der initrd wird ausgeführt.
7. Client lädt Netzwerkkarten-Treiber von der initrd

8. Client wartet auf statische IP
9. Referenz Image wird ermittelt (aus Workstation oder Standards ...)
   - *dbkInitRdScript*
   - *dbkPartitionsTable*
   - *dbkImageLocation*
   - *dbkExcludeList*
   - *dbkImageVersion*
10. Festplatte wird überprüft
    - Festplatte wird auf jeden Fall neu formatiert, falls *dbkImageVersion* $= -1$
    - Partitionstabelle wird geschrieben
    - Dateisystem wird überprüft und im Fehlerfall neu angelegt.
11. *dbkInitRdScript* wird geladen (`rsync`) und ausgeführt
    - rsync Server wird ermittelt (none, wan, lan, NAME)
    - rsync wird durchführt
    - Die aktuelle Versionssnummer wird im LDAP eingetragen
    - falls im LDAP *dbkDoTftpBoot* gesetzt war, wird es auf LOCALBOOT geändert
    - Client führt `lilo` bzw. `grub-install` aus und kann danach lokal booten.
    - Client stellt DHCP-Request und nutzt den Lease-Timeout um festzustellen, wann im LDAP eingetragen und zum DHCP-Server durchgedrungen ist, dass er über seine eigene Festplatte booten darf.
    - falls kexec konfiguriert ist, werden der neue Kernel und initrd geladen und ausgeführt. Ansonsten erfolgt ein Neustart.

3.12 Produktionsübernahmeverfahren

Im folgenden wird ein Vorschlag zu einem Produktionsübernahmeverfahren gemacht.

PCs:		Rsync Server:
refPC: Zugriff: LinuxAdmins Jede Konfigurationsänderung ist zu dokumentieren. U.U. bietet sich auch das Backup im Versionsmanagement CVS an. Die Liste der installierten Pakete sollte gesichert werden. Die Partitionierung des Rechners ist dem LDAP zu entnehmen.		
	rsync.dev →	Erstellen von Backups: image.dev.$VERSION Die dev Images können u.U. auch Dateien beinhalten, die nicht Teil der test- und prod-Images sind (z. B. RPM Datenbank, yast, ...)
	← rsync.dev	
devPCs: Zugriff: 1 pro LinuxAdmins, evtl. einer pro Entwickler zum Testen neuer Einstellungen werden die devPCs verwendet. Erfolgreiche Änderungen können manuell auf den refPC überspielt werden. Danach werden sie auf den devPCs wieder getestet. Sobald das dev Image stabil zu sein scheint		

	rsync.test →	Versionsnummer wird vom verwendeten devImage übernommen.
	← rsync.test	
testPCs: Zugriff: LinuxAdmins, Entwickler, Benutzerservice Austesten der gesammten Konfiguration (Checklisten). Sollten Fehler festgestellt werden, so sind diese auf den devPCs (und damit refPC) zu beheben. Am testPC dürfen keine Veränderungen vorgenommen werden. (Nach beheben des Fehlers wird eine neues Test-Image generiert.) Abnahme durch alle Verantwortlichen. Wenn im längeren Betrieb keine Fehler mehr festgestellt wurden		
	rsync.prod →	Versionsnummer wird vom testImage übernommen Erstellen von Backups: image.prod.$VERSION
	← rsync.prod	
prodPCs: Zugriff: LinuxAdmin, Benutzerservice, produktive Benutzer Verteilung erst an eine GS (zum Testen), danach an alle		

3.13 Fehlerfälle

3.13.1 Ein Client funktioniert nicht

Sofern ein Client Funktionsstörungen ausweisst, können folgende Schritte sukkzessive durchgeführt werden, um das Problem zu beheben:

1. Netz prüfen

2. Update des Clients
3. Löschen der Client-Installation und Neuinstallation

3.13.2 Die LDAP Server sind nicht mehr synchron

Normalerweise synchronisieren sich Master und Slave LDAP Server automatisch (siehe 1.6.1 auf Seite 16 und 1.6.2 auf Seite 21). Sollten diese aber einmal (wegen eines Konfigurationsfehlers, u.ä.) auseinander laufen, kann man diese auch einfach manuell synchronisieren. Die effektive Methode besteht darin, beide LDAP Server anzuhalten (`rcldap stop`) und dann die Dateien aus `/var/lib/ldap/` auf den anderen Rechner zu übertragen.

sync-repl erklären? reinit skript?

4 Schnittstellen

4.1 Einführung

Bei vielen der unten beschriebenen XML-RPC Aufrufen kann es passieren, dass der `sc_controld.pl` länger zur Kommunikation mit dem `sc_ldap2dhcpdnsd.pl` braucht, als der Benutzer zu warten bereit ist. In disem Fall gibt der Control-Daemon eine Antwort mit einer speziellen `id` zurück. Diese `id` kann dazu genutzt werden, die Antwort auf die gestellte Frage später abzuholen.

Allgemein kann der Daemon von Perl aus wie folgt angesprochen werden:

```
1 use strict;
2 use warnings;
3 use RPC::XML::Client;
4 my $proxy=RPC::XML::Client->new('http://server:2223');
5 # Aufruf ohne Parameter
6 my $result = $proxy->send_request('getRandom');
7 # Aufruf mit Parameter
8 my $result = $proxy->send_request('getResult', RPC::XML
    ::struct->new({'id' => 1}));
```

4.1.1 Allgemeine XML-RPC Funktionen

struct getResult(struct) Versucht die Antwort auf eine früher gestellte Anfrage zu holen.

Parameter Ein Struct mit

id (string) die Kennung, die eine vorherige Antwort bei einem Timeout zurückgegeben hat.

timeout (integer) in Sekunden, nachdem die Anfrage abgebrochen werden soll. (optional)

Rückgabe wie bei der ursprünglich gestellten Anfrage

struct dumpEntry(struct) Gibt den aktuellen Wert des LDAP-Eintrages aus dem `SmartClient::ScCache` Objektes aus und schreibt ihn ebenfalls in das Logger Objekt.

Parameter Ein Struct mit

dn (string) des Eintrages, der Ausgegeben werden soll

Rückgabe Ein Struct mit

id (string) ...
code (integer)
message (string)
hash (struct) in dem die gesamte Ausgabe von der internen Funktion `SmartClient::LdapCache::getInternalEntryHash` enthalten ist

string getHaMode() Liefert den HA-Modus des Servers, wie er in der globalen Konfigurations-Datei `/etc/smartclient/ldap2dns` definiert ist.

Parameter Keine

Rückgabe (string) `ha_mode_primary` oder `ha_mode_primary`

4.2 sc_leases2ldap

XML-RPC Funktionen

struct getDislocatedDevices() liefert die Liste der Systeme, die sich nicht in ihren definierten Netz befinden. Dabei werden nur die Systeme aufgeführt, die für den Administrator relevant sind, d.h.

action ACTION_ADD oder ACTION_MOVE,

permission_level PERM_LEVEL_ADMIN oder PERM_LEVEL_USER

permission PERM_LEVEL_UNDEF

Parameter Keine

Rückgabe Ein Struct mit

id (string) Schlüssel der Struktur/des Hashs. Derzeit identisch mit der MAC Adresse
mac_address (string)
lease_time_start (string)
lease_ip (string)
current_ldap_dn (string)
location_dn_new (string)
action (integer) durchzuführende Aktion
0: ACTION_UNDEF keine Aktion vorgeschlagen
1: ACTION_ADD MAC Adresse existiert noch nicht im LDAP. Kann hinzugefügt werden

2: ACTION_MOVE MAC Adresse existiert bereits an anderer Stelle im LDAP. LDAP Eintrag kann verschoben werden

3: ACTION_MANUAL MAC Adresse existiert mehrfach im LDAP. Dies ist ein Fehler. Manueller Eingriff erfoderlich. Erst danach kann sich dieser Zustand ändern

permission_level (integer) Umzugszulässigkeit

0: PERM_LEVEL_UNDEF keine Information über Umzugszulässigkeit vorhanden

1: PERM_LEVEL_AUTO Umzug okay (kein manueller Eingriff benötigt)

2: PERM_LEVEL_ADMIN Umzug nach manueller Bestätigung durch Administrator okay

4: PERM_LEVEL_USER Umzug nach Bestätigung des Benutzers (oder Administrators) zulässig

permission (integer)

-1: PERM_LEVEL_UNDEF keine Information vorhanden.

0: PERM_LEVEL_DENIED Erlaubnis zum Umziehen verweigert. Wird nicht mehr als Dislocated Device dargestellt

1: PERM_LEVEL_GRANTED Erlaubnis für den Umzug erteilt. LDAP Eintrag wird demnächst verschoben

struct getAllDislocatedDevices() wie `getDislocatedDevices`, jedoch werden alle Geräte zurückgegeben, bei denen `action` > `ACTION_UNDEF` ist.

struct denyPermission(struct) verbietet den Umzug eines Dislocated Devices. Danach wird es nicht mehr von der Funktion getDislocatedDevices aufgeführt.

Parameter

id (string)

Rückgabe

code (integer) Fehlercode

0 okay

1 Fehler: id existiert nicht

...

message (string) Statusmeldung

struct relocateDislocatedDevice(struct) Führt die unter id beschriebene Aktion durch, d.h. fügt entweder eine neue Workstation/Drucker hinzu oder verschiebt einen bestehen Eintrag in eine neue Lokation.

Es besteht die Möglichkeit, den Device Typen mit anzugeben, ansonsten bleibt der Bestehende erhalten bzw. wird der Lokations-Default verwendet.

Intern wird überprüft, ob dieser Umzug zulässig ist.

Parameter Ein Struct mit

id (string)
device_type_dn (string) Device-Type dn (optional)

Rückgabe Ein Struct mit

id (string) id
action (integer?) durchgeführte Aktion
code (integer) Fehlercode
- **0: XMLRPC_OK** okay, Aktion erfolgreich durchgeführt (auch bei ACTION_NOTHING)
- **1: XMLRPC_NO_SUCH_ID** Dislocated Device id ist nicht vorhanden
- **2: XMLRPC_NO_PERMISSION** keine Berechtigung diese Aktion durchzuführen (darf nicht vorkommen, da hiermit das Recht gesetzt wird)
- **4: XMLRPC_ERROR_MANUAL** Probleme mit Dislocated Device, z.B. bereits mehrfach im LDAP
- **8: XMLRPC_UNKNOWN_ACTION** angeforderte Aktion ist unbekannt
- **16: XMLRPC_LDAP_FAILURE** LDAP Fehler
- ...

message (string) Statusmeldung
subcode (integer) Unterfehlercode, z.B Umzug nicht erlaubt oder LDAP-Fehlercode
submessage (string) Unterfehlermeldung
old_ldap_dn (string)
new_ldap_dn (string)

(moveLdapEntry(String objectDn, String locationDn)) Verschiebt einen LDAP Entry (z.B. Workstation oder Drucker) im LDAP Baum in den angegeben Container.

Diese Funktion ist derzeit nicht implementiert, da hierfür auch eine Authentisierung stattfinden müsste

Parameter

1. String objectDn: derzeitiger DN
2. String locationDn: DN der Ziellokation

Rückgabe

1. Integer: Fehlercode

0 okay, Umzug erfolgreich durchgeführt
1 Fehler, Umzug gescheitert
2 Quelle Objekt existiert nicht
4 Ziel Lokation existiert nicht
8 Umzug nicht erlaubt
16 LDAP Fehler
...

2. String: Fehlermeldung
3. Integer: Unterfehlercode (wenn Umzug nicht erlaubt, oder LDAP Fehlercode)
4. String: Unterfehlermeldung
5. String: newDn

4.3 schedule_update Testdaemon

XML–RPC Funktionen

Funktionen:

sendWol(String IP) Simuliert ein Paket zum Aufwecken (Wake up On LAN) eines Rechners. Kein Rückgabewert.

Parameter

1. String: IP

Rückgabe

sendPing(String IP) Simuliert das anpingen einer IP Adresse. Dadurch soll erkennbar werden, ob ein simulierter Rechner ansprechbar ist.

Parameter

1. String: IP

Rückgabe

1. ping exit codes (system nicht erreichbar, pingend)

sendSshCommand(String IP, String command) Simuliert das einloggen per SSH auf einem System und dann das Ausführen des Kommandos `command`.

Parameter

1. String: IP
2. String: command Auswahl, aus reboot, halt

Rückgabe

1. ssh exit codes (system nicht erreichbar, exit code des kommandos)

setClientStatus(String IP, Hash) Setzt den Status einer Arbeitsstation. (Eventuell sollte hier die Möglichkeit gegeben sein, auch ganze Lokationen auf einmal mit einem Status zu versehen. Switch defekt, oder auch einfach nur die gleiche Netzanbindung und daher die gleiche rsync Zeit.)

Parameter

1. String: IP
2. Hash:
   wol ⇒ ON|OFF. Rechner wacht (nicht) per WOL auf. Ist WOL ausgeschaltet, so kann der Rechner nicht mehr angeschaltet werden, wenn er per ssh Kommando ausgeschaltet wird.
   power ⇒ ON|OFF. Rechner ist (aus|an).
   network ⇒ ON|OFF. Rechner ist per Netz (nicht) erreichbar. Dies impliziert sowohl, dass der Rechner aus ist, als auch, dass er nicht per WOL angeschaltet werden kann.
   localbootTime ⇒ Integer. Lokale Bootzeit in Sekunden
   netbootWanTime ⇒ Integer. Bootzeit übers WAN in Sekunden
   netbootLanTime ⇒ Integer. Bootzeit übers LAN in Sekunden
   rsyncWanTime ⇒ Integer. Zeit zum Syncen übers WAN in Sekunden
   rsyncLanTime ⇒ Integer. Zeit zum Syncen übers LAN in Sekunden
   weitere Fehler Fälle (Rechner kann in irgendeinem Vorgang (rsync, boot, shutdown) Fehler bringen.)

Rückgabe

1. Integer: Fehlercode
   0 okay. Setzen des Status erfolgreich
   1 Fehler.

getClientStatus(String IP) Erfragt den aktuellen Zustand einer Arbeitsstation.

Parameter

1. String: IP

Rückgabe

1. Integer: Fehlercode
   0 OK.
   1 Fehler. LDAP Fehler
2. Hash:
   wol ⇒ ON|OFF. Rechner wacht (nicht) per WOL auf

power ⇒ ON|OFF. Rechner ist (aus|an)
reachable ⇒ ON|OFF. Rechner ist per Netz (nicht) erreichbar
localbootTime ⇒ Integer. Lokale Bootzeit in Sekunden
netbootTime ⇒ Integer. Bootzeit übers Netz in Sekunden
netRsyncTime ⇒ Integer. Zeit zum Syncen übers Netz in Sekunden
localRsyncTime ⇒ Integer. Zeit zum Syncen übers Netz in Sekunden
weitere Fehler Fälle (Rechner kann Fehler bringen)

getStatus( ) Erfragt den aktuellen Zustand des Daemons.

4.4 sc_ldap2dnsdhcpd

check modifyTimestamp nach Änderungen in LDAP Containern. Diese werden dann neu erzeugt. Periodisch wird komplett neu erzeugt und eingelesen (konfigurierbar, default 1x Stunde) ?

XML-RPC Funktionen

struct notifyOfModification(array) Es wurden neue Devices in LDAP eingetragen oder verschoben. Dem Daemon wird hiermit ein Hinweis gegeben, dass dies passiert ist.

Parameter Array mit

1. (string) DNs der Einträge, die sich geändert haben

Rückgabe Struct mit

code (integer)
message (string)
id (string)
...

struct notifyOfDelete(array) Wie `notifyOfModification`. Nur ein wenig anders.

struct setUpdate(struct) Setzt einen Eintrag/Lokation auf ...

Parameter Ein Struct mit

location_dn (string) kann alternativ zu `workstations` genutzt werden.
device_type_dn (string) wählt einen speziellen Device-Type aus, dessen Update-Verhalten geändert werden soll

workstations (array) Liste mit DNs der Arbeitstationen, die angepasst werden sollen
tftpboot (string)
rsync (string)

Rückgabe Ein Struct mit

id (string)
code (integer)
message (string)
workstations (struct) mit Statusmeldungen pro Workstation
$location_dn (integer) Statusmeldung von `setUpdateEntry` für die Lokation (wenn `location_dn` übergeben wurde)

struct getScheduleUpdateWorkstation(struct) Holt eine Liste mit Workstations, die in einer Lokation auf Update stehen und gleichzeitig die Kriterien des angegebenen Device-Typs und RefImages erfüllen.

Parameter Ein Struct mit

location_dn (string) in der nach Workstations gesucht werden soll
refimage_dn (string) Name des RefImages, das für die Workstations ausgesucht sein soll
devicetype_dn (string) Device-Typ der Workstations

Rückgabe Ein Struct mit

code (integer)
id (string)
message (string)
workstations (struct) mit den DNs der Workstation als Schlüssel. Als zugehörige Werte sind die Rückgaben von internen Aufrufen von `ScCache::getDeviceStatus($dn)`

struct createConfig() Aus Hochverfügbarkeitsgründen wird die Konfiguration auf dem zweiten System immer versetzt und in Abstimmung mit dem ersten aktiviert (1. erstellt, aktiviert, notifySecondary, 2. erstellt, aktiviert)

Dieser Aufruf ist synchron, heißt wartet, bis Konfiguration erstellt wurde.

Parameter

Rückgabe

code (integer) Fehlercode (0: erfolgreich aktiviert)
state (string) ...

struct getDeviceStatus(struct) Liefert Informationen zu einem Device, speziell die derzeit gültigen Booteinstellungen (Netboot oder nicht). Die unten Aufgeführten Elemente sind nicht in jedem Fall in der Antwort, z.B. gibt es

für Printer und Server keine Image Versionen. Die Namen orientieren sich an den LDAP Attributnamen, aber z.B. für dbkRefImageVersion gilt dies nicht (Attribut heißt ebenfalls dbkImageVersion).

Im Fehlerfall (z.B. der übergene DN ist nicht im cache vorhanden) wird im Struct mit dem Key 'error' ein String mit einer Fehlermeldung zurückgegeben.

Parameter Ein Struct mit

device_dn (string) DN

Rückgabe Ein Struct mit

code (integer)
message (string)
dnsName (string) (kommt aus cn)
ipHostNumber (string)
macAddress (string)
dbkDeviceType (string) Device-Type: (noch zu klären)
dbkRefImageDn (string)
dbkRefImageVersion (integer)
dbkImageVersion (integer)
scRefBootOptionsDn (string)
netboot (boolean) (eingestellt im DHCP)
dbkDoRsync (string) (WAN, LAN, NONE, Rechnername)
dbkDoTftpBoot (string) (WAN, LOCALBOOT, Rechnername)
dbkUpdateTime (string) (< 0 kein Update)

struct waitForDeviceStatus(struct) Wartet, bis das per DN angegebene Device den Status aus dem Hash erreicht.

Achtung: Synchron

Parameter

device_dn (string) DN
netboot (boolean)
dbkImageVersion (string) IMAGEVERSION|undef
(timeout) (integer) Timeout in Sekunden. Optional (HTTP Timeout: 170 Sekunden)

Rückgabe

code (integer) Fehlercode
0: XMLRPC_OK okay, Aktion erfolgreich durchgeführt (auch bei ACTION_NOTHING)
1: XMLRPC_NO_SUCH_ID Objekt id ist nicht vorhanden

8: XMLRPC_UNKNOWN_ACTION angeforderte Aktion ist unbekannt
32: XMLRPC_TIMEOUT Timeout
message (string) Beschreibung code
device_dn (string) DN
netboot (boolean)
dbkImageVersion (string) IMAGEVERSION|undef

struct changeDeviceType(struct) Ändert den *dbkDeviceTypeDn* eines Eintrages und passt den Namen entsprechend an.

Parameter **dn** (string) DN der Workstation
device_type_dn (string) DN des neuen *dbkDeviceTypeDn*

Rückgabe Kompletter geänderter Eintrag der Workstation mit neuem Namen und neuem *dbkDeviceTypeDn*.

registerDeviceStatusChangeNotifier(String DN, Url callbackMethod, Integer repeat) Hinterlegt eine XMLRPC Schnittstelle, über die das aufrufende Programm über Änderungen an Device DN benachrichtigt werden will.

Diese Funktion ist derzeit nicht implementiert.

Parameter

1. (string) DN
2. (string) URL: XMLRPC Methode, über die der DN und der Status des Devices mitgeteilt werden kann.
3. (integer) repeat
   -1 Ab jetzt jede Änderung mitteilen.
   1..n n mal die Änderung mitteilen.

Rückgabe

1. (integer) Fehlercode
   0 OK.
   1 Fehler.
2. (string) Identifier mit dem der ChangeNotifier wieder entfernt werden kann

unregisterDeviceStatusChangeNotifier(String ID) Entfernt den ChangeNotifier aus den registrierten XMLRPC Methoden

Diese Funktion ist derzeit nicht implementiert.

Parameter

1. String: ID die bei registerDeviceStatusChangeNotifier-Aufruf zurückgeliefert wurde.

Rückgabe

1. (integer) Fehlercode
   0 OK.
   1 ID nicht registriert
   2 Kann ID nicht entfernen

A Literaturverzeichnis

[Bin03a] Karl-Eugen Binder. Erfahrungsbericht ber die Einfhrung einer LINUX-Desktop-Strategie. http://www-5.ibm.com/de/versicherungen/garmisch_2003_pdf/Thema_Neue_Technologien/Binder/Binder_27.pdf, 2003. IBM Anwenderkongress Versicherungswirtschaft.

[Bin03b] Karl-Eugen Binder. Erfahrungsbericht ber die Einfhrung einer LINUX-Desktop-Strategie. http://www.infologica.com/Infologica/artikel/download/company/Nr.11_GSE_2003_010_LINUX.pdf, 2003. GSE Herbsttagung.

[Bin04] Karl-Eugen Binder. Linux auf dem Desktop: Erfolgreiche Strategie der Stuttgarter Lebensversicherung. http://www.baden-wuerttemberg.de/sixcms/media.php/1082/P_Binder_bwconboss_01072004.pdf, Juni 2004. Baden-Wrttemberg: Connected (bwcon).

[CCK06] Client customization kit (cck). http://www.mozilla.org/projects/cck/, 2006.

[fre] freedesktop.org Standards. http://freedesktop.org/Standards/.

[Gol02] Patrick Goltzsch. Die Linux-Versicherung. *CIO*, 5, 2002. http://www.cio.de/index.cfm?PageID=259&cat=det&maid=1123#.

[Kdea] KDE for System Administrators. http://www.kde.org/areas/sysadmin/.

[Kdeb] KDE Filesystem Hierarchy. http://www.kde.org/areas/sysadmin/fsh.php.

[Kdec] Kiosk: The configuration framework in KDE3. http://webcvs.kde.org/cgi-bin/cvsweb.cgi/kdelibs/kdecore/README.kiosk.

[KMS02] M. Kulisch, A. Meyer, and J. Steffens. Ausgetauscht – Linux auf 3000 verteilten Arbeitsplätzen. *iX – Magazin fr professionelle Informationstechnik*, pages 40 – 44, März 2002. http://www.suse.com/de/business/products/frameworks/smartclient/misc/ix_0302_040_044_suse.pdf.

[Mal04] Adrian Malaguti. Polixy based Linux Desktop Environment. http://enterprise.kde.org/articles/, 2004.

[Mey02] Axel Meyer. Debeka: Open Source im kommerziellen Einsatz. http://www.bull.de/download/events/ados04/Debeka_AM.pdf, 2002.

[ope] *OpenLDAP Admin-Guide*. http://www.openldap.org/doc/admin/.

[Sch02] Ludger Schmitz. Debeka favorisiert Linux-Clients. *Computerwoche*, 13, 29. März 2002. http://www.suse.com/de/business/products/frameworks/smartclient/misc/SdrCW1302.pdf.

[TBS02] Cristian Tibirna, Waldo Bastian, and Malte Starostik. The KDE Administrators Guide. http://i18n.kde.org/doc/admin/, 2002. (incomplete).