

# Installation eine Linux Diskless Client (LDC) auf Basis von Unionfs

© Bernd Burre und Uwe Debacher 10/2005

Die Beschreibung geht davon aus, dass auf dem Serversystem eine funktionsfähige Installation von SuSE9.3 vorliegt. Die Beschreibung geht davon aus, dass der Server die IP 192.168.2.1 besitzt.

## 1. Zusätzliche Software:

Folgende Pakete müssen für diese Beschreibung vorhanden sein.

- dhcp
- bind
- kernel-quellen
- atftp
- syslinux
- nfs-utils
- ypserv
- Entwicklerwerkzeuge (gcc, make ,automake, autoconf, ...)

## 2. Vorbereitungen

Auf dem System sollte SuSEfirewall deaktiviert oder passend konfiguriert sein und die Dienste

- dhcpd
- named
- ypserv
- atftpd
- nfsserver (hier ist das Paket nfs-utils erforderlich)

müssen funktionsfähig konfiguriert werden. Für den atftpd muss das Verzeichnis /tftpboot als Arbeitsverzeichnis eingerichtet sein, das dann weiter vorbereitet wird.

```
cp /usr/share/syslinux/pxelinux.cfg /tftpboot
mkdir /tftpboot/pxelinux.cfg
mkdir /tftpboot/client
mkdir /tftpboot/client/ram
```

Dann das Archiv diskless.tgz entpacken

```
tar xvfz diskless.tgz
cp diskless/default /tftpboot/pxelinux.cfg/
```

und die Datei /tftpboot/pxelinux.cfg/default passend editieren

```
default linux

label linux
    kernel vmlinuz
    append initrd=initrd root=192.168.2.1:/tftpboot/client
```

Wichtig ist vor allem die IP-Adresse des Servers. Angegeben wird hier auch, in welchem Verzeichnis die Client-Installation zu finden ist. Hier ist das ein Verzeichnis unterhalb von /tftpboot. Das ist nicht zwingend, die Vorgabe bei LTSP z.B. ist /opt.

Nun muss das Verzeichnis noch per NFS exportiert werden (/etc/exports).

```
# See the exports(5) manpage for a description of the syntax of this file.
# This file contains a list of all directories that are to be exported to
# other computers via NFS (Network File System).
# This file used by rpc.nfsd and rpc.mountd. See their manpages for details
# on how make changes in this file effective.
/tftpboot/client 192.168.2.0/255.255.255.0(ro,no_root_squash,sync)
```

danach sollte der NFS-Server (neu) gestartet werden.

Für einen eventuellen ersten Test nun Kernel und initrd nach /tftpboot kopieren.

```
cp /boot/vmlinuz /tftpboot
```

```
cp /boot/initrd /tftpboot
```

ein Client-System sollte nun in der Lage sein den Kernel zu booten und müsste dann mit einer Kernel-Panic sterben.

### 3. Installation des Client-Systems

Auf dem Server-System kann nun mit YaST die Client-Installation erfolgen. Dazu dient der Menüpunkt „Installieren in Verzeichnis“. Hier muss unter Optionen noch der Zielpfad richtig eingestellt werden.

In YaST erfolgt eine normale Auswahl der zu installierenden Pakete und die Installation läuft wie gewohnt ab.

Am Ende der Installation muss noch eine Datei in das Client-System kopiert werden, die YaST nicht anlegt:

```
cp /var/adm/YaST/ProdDB/prod_00000001 /tftpboot/client/var/adm/YaST/ProdDB/
```

Nun machen wir zunächst einmal auf dem Serversystem ein Online-Update.

Im nächsten Schritt wird Unionfs im Serversystem compiliert und installiert, es stört dort auf alle Fälle nicht (die Beschreibung geht von der momentan aktuellen Versionsnummer 1.0.14 aus).

```
cd /usr/src/linux
make cloneconfig
make prepare_all
cd /tmp
tar xfvz /root/diskless/unionfs-1.0.14.tar.gz
cd unionfs-1.0.14
```

Hier jetzt eine Datei `fistdev.mk` anlegen mit folgendem Inhalt

```
EXTRACFLAGS=-DNODEBUG
```

damit wird das Modul ohne Debug-Informationen compiliert und damit kleiner.

Nun das Modul übersetzen und installieren:

```
make
```

```
make install
```

Sollten beim Compilieren Fehler auftauchen, dann kontrollieren, ob die Pakete „ctags“ und „e2fsprogs-devel“ installiert sind. Wichtig ist vor allem, dass `unionfs.ko` erzeugt wird.

Für spätere Schritte wird noch ein Script benötigt, welches das Proc-Dateisystem auch nach einem `chroot` verfügbar macht.

```
cp diskless/clientprocmount.sh /root/bin/
```

### 4. Konfiguration des Client-Systems

Zuerst mittels

```
/root/bin/clientprocmount.sh start
```

das Proc-Dateisystem verfügbar halten. Überprüfen, ob YaST für das Client-System die Sprache richtig gesetzt hat oder einfach die Einstellung kopieren:

```
cp /etc/sysconfig/language /tftpboot/client/etc/sysconfig/language
```

Mittels

```
chroot /tftpboot/client
```

ins Client-System wechseln, dort SuSEconfig ausführen.

Danach YaST starten und ein Online-Update durchführen. Wichtig ist, dass die Kernel-Versionen von Server und Client übereinstimmen, sonst ist das Unionfs-Modul nicht funktionsfähig.

Als nächstes wird mittels

```
passwd root
```

das root- Password gesetzt.

Dann mit exit das Client-System wieder verlassen.

Nun wird das Unionfs-Modul in das Clientsystem kopiert.

```
cp /lib/modules/$(uname -r)/kernel/fs/unionfs.ko  
/tftpboot/client/lib/modules/$(uname -r)/kernel/fs/
```

Nun die Datei /tftpboot/client/etc/sysconfig/kernel editieren und dort folgende Module eintragen:

```
INITRD_MODULES="mii 8139too via-rhine"
```

die genaue Liste hängt natürlich von der benutzten Hardware ab. Hier sind die Treiber für die Netzwerkkarten RTL8139 und Via-Rhine vorgesehen.

Noch ein paar Dateien für den Boot-Vorgang überschreiben:

```
cp diskless/boot /tftpboot/client/etc/init.d/  
cp diskless/boot.localfs /tftpboot/client/etc/init.d/  
cp diskless/boot.soundsetup /tftpboot/client/etc/init.d/  
cp diskless/union.sh /tftpboot/client/etc/init.d/
```

Die Client-fstab editieren (/tftpboot/client/etc/fstab):

```
tmpfs /tmp tmpfs defaults 0 0  
devpts /dev/pts devpts mode=0620,gid=5 0 0  
proc /proc proc defaults 0 0  
usbfs /proc/bus/usb usbfs noauto 0 0  
sysfs /sys sysfs noauto 0 0
```

Die unionfs.sh editieren (/tftpboot/client/etc/init.d/union.sh):

Die Variablen „domain“ und „hostprefix“ müssen an das System angepasst werden.

Nun die initrd neu erzeugen und dafür sorgen, dass boot.soundsetup beim Systemstart des Clients gestartet wird (bei einem chroot immer an clientprocmount.sh denken, s.o.).

```
chroot /tftpboot/client  
depmod -a  
mkinitrd -D eth0  
insserv boot.soundsetup  
exit
```

Jetzt noch Kernel und initrd an die richtige Stelle kopieren

```
cp /tftpboot/client/boot/vmlinuz /tftpboot  
cp /tftpboot/client/boot/initrd /tftpboot
```

Nach diesen Schritten sollte sich das Client-System booten lassen. Idealerweise sollte das System im Runlevel 3 starten (der aktuelle Runlevel steht in der Datei /etc/inittab), dann kann man mittels sax2 relativ bequem die Grafik konfigurieren. Dazu kann es sinnvoll sein vom Serversystem aus alle Dateien unterhalb vom /tftpboot/client/var/cache/sax/files/ zu löschen.

Die entstandene Konfigurationsdatei

/etc/X11/xorg.conf kopiert man dann mittel scp auf das Serversystem.

```
scp /etc/X11/xorg.conf 192.168.2.1:/tftpboot/client/etc/X11/
```

## 5. Nachbesserungen

Mit VIA-Epia-Geräten taucht unter SuSE9.3 das Problem auf, dass bei einem Reboot das PXE nicht mehr funktioniert, man muss also jedesmal den Netzstecker ziehen. Das liegt daran, dass der via-rhine Treiber das PXE deaktiviert. Nach Anleitung aus dem Forum Viaarena

<http://forums.viaarena.com/messageview.aspx?catid=28&threadid=67379> hilft ein Update des PXE-BIOS oder der folgende Patch hier weiter.

```
--- linux-2.6.11/drivers/net/via-rhine.c.orig      2005-04-04 13:17:04.202782557 +0
+++ linux-2.6.11/drivers/net/via-rhine.c        2005-04-04 13:19:01.729867587 +0
@@ -1934,9 +1934,6 @@ static void rhine_shutdown (struct devic
        iowrite8(ioread8(ioaddr + StickyHW) | 0x04, ioaddr + StickyHW);
    }

-    /* Hit power state D3 (sleep) */
-    iowrite8(ioread8(ioaddr + StickyHW) | 0x03, ioaddr + StickyHW);
-
    /* TODO: Check use of pci_enable_wake() */
}

```

Es müssen im Prinzip nur die drei Zeilen mit dem „-“ am Zeilenanfang entfernt werden. Danach muss dann der Treiber neu compiliert werden. Dazu erstellt man eine Datei Makefile mit folgendem Inhalt:

```
obj-$(CONFIG_VIA_RHINE) += via-rhine.o
modules:
    make -C /usr/src/linux SUBDIRS=$(PWD) modules
clean:
    make -C /usr/src/linux SUBDIRS=$(PWD) clean

```

und packt sie zusammen mit dem geänderten Treiber in ein Verzeichnis, z.B. /tmp und ruft dort make auf. Danach muss das Modul dann neu in das Client-System integriert werden.

In einem LDC-System nervt der SuSE-Plugger, der bei jedem Start neue Hardware erkennt. Er lässt sich bei Bedarf mittels

```
rm /tftpboot/client/opt/kde3/lib/kde3/susplugger*
rm /tftpboot/client/opt/kde3/lib/libkdeinit_suseplugger*
```

entfernen. Ähnlich geht das mit dem susewatcher.

In dem System läuft Mozilla-Firefox nicht oder maximal einmalig. Die Ursache für das Problem ist nicht ganz klar, aber das entfernen eines Verzeichnisses löst das Problem

```
rmdir /tftpboot/client/opt/MozillaFirefox/lib/extensions/temp/
```

Etwas nervig ist es, dass beim Herunterfahren das Netzwerk gestoppt wird, dann kann sich das System nicht mehr korrekt beenden. Da das Netzwerk von vielen Diensten benutzt wird muss man erzwingen, dass es nicht automatisch gestoppt (gestartet wird es sowieso beim Booten).

```
inserv -r -f network
```

nach einem Wechsel ins Client-System.

## 6. Die Scripten

Die für das System notwendigen Scripte stammen von Bernd Burre, soweit keine weitere Quelle angegeben ist.

### clientprocmount.sh

```
clientdir="/tftpboot/client"
case "$1" in
    start)

```

```

mount -t proc proc $clientdir/proc
mount -t sysfs sysfs $clientdir/sys
;;
stop)
umount $clientdir/proc
umount $clientdir/sys
;;
*)
echo "Usage $0 start/stop"
;;
esac

```

**boot** (ab Zeile 126 ergänzt)

```

# Start unionfs
echo -n "Starte unionfs Dateisystem"
/bin/sh /etc/init.d/union.sh

```

**boot.localfs** (Zeile 125 auskommentieren)

```

# /sbin/update-modules.dep -r

```

### **union.sh**

```

#!/bin/sh
# geschrieben von Bernd Burre 10/2005
# fuer Diskless-Clients
domain="burre.hh.shuttle.de"
hostprefix="client-"
echo "Starte UnionFs"
modprobe unionfs
union=/ram/union
changes=/ram/changes
mount -t tmpfs tmpfs /ram
mkdir $changes $union
mount -t tmpfs tmpfs $changes
ip=$(ifconfig eth0 | head -2 | tail -1 | awk '{ print $2 }' | awk -F: '{ print $2 }')
endip=$(echo $ip | awk -F\. '{ print $4 }')
hostname=${hostprefix}${endip}
mount -n -o remount,ro /
mount -t unionfs -o dirs=$changes=rw:/=ro,copyup=preserve none $union
echo "$ip $hostname.$domain $hostname" >> $union/etc/hosts
echo "$hostname.$domain" > $union/etc/HOSTNAME
mkdir -p $union/changes
mount --move $changes $union/changes
mount --move /lib/klibc/events $union/lib/klibc/events
mount --move /proc $union/proc
mount --move /sys $union/sys
pivot_root $union $union

```

### **boot.soundsetup**

```

#!/bin/sh
#
# adaptation of SuSE runlevel script file for use with
# linux diskless clients v3.2
#
# Copyright (c) 2003 Dirk von Suchodoletz <dirk@goe.net>, 28-01-2005
# All rights reserved.
#
# changed by Bernd Burre
#
version="0.3.6c"
#
# /etc/init.d/boot.hwsetup
#
### BEGIN INIT INFO
# Provides:          boot.soundsetup
# Required-Start:    boot.coldplug boot.udev
# X-UnitedLinux-Should-Start:
# Required-Stop:
# Default-Start:     B
# Default-Stop:
# Description:       start sound hardware autoconfiguration

```

```

### END INIT INFO

# setup preparations
setupprep() {
# load modules (needed eventually for auto detection of hardware components)
modprobe -aq generic_serial ata_piix libata
modprobe -q agpgart || \
    modprobe -q agpgart agp_try_unsupported=1

# end of preparations
}

soundsetup () {
# setting up sound module

# oss and kernel modules named with activation command
hwinfo --sound >/tmp/hwsound.tmp

# prefer alsa sound modules
cmd=`cat /tmp/hwsound.tmp|grep -e "Cmd:" -e "Info:"|grep "snd-" | \
    awk -F : {'print $2'}|sed -e 's,,g'`
# if no one is offered try for the others
if [ -z "$cmd" ] ; then
    cmd=`cat /tmp/hwsound.tmp|grep "Cmd:"| awk -F \" {'print $2'}`
    rm /etc/init.d/alsasound /etc/init.d/rc/*alsasound &>/dev/null
else
    # enable the start of alsa daemon
    for i in rc{3,5}.d/K10alsasound rc{3,5}.d/S12alsasound ; do
        ln -sf /etc/init.d/alsasound /etc/init.d/$i
    done
    # hack because of SuSEs unclever hwinfo --sound
    cmd=`echo $cmd|sed -e "s,via686,via82xx," -e "s,/, /"`
fi
if [ -n "$cmd" ] ; then
    echo "$cmd"|grep "modprobe"&>/dev/null || cmd="modprobe -a $cmd"
    $cmd
# also seem to be named by info line only, if no adaptor is detected use
# the dummy module; load oss compatibility modules afterwards
else
    logwrite "Unable to find an audio device, using snd-dummy.";
    modprobe snd-dummy
fi
grep "snd-" /tmp/hwsound.tmp &>/dev/null && \
    modprobe -a snd-mixer-oss snd-pcm-oss #>> $HWLOG 2>&1
}

# main part of the script
. /etc/rc.status
#. /etc/sysconfig/logfile

rc_reset

case "$1" in
start)
    echo -n "Starting hardware setup "
        echo "as background process ..."
        setupprep
        soundsetup

        rc_status -v1; rc_reset
        #
        ;;
stop)
    rc_failed 3
    rc_status -v
    ;;
status)
    rc_failed 4
    rc_status -v
    ;;
*)
    echo "Usage: $0 {start|stop|status}"
    exit 1
    ;;
esac

rc_exit

```

