German HowTo VServer unter Debian 3.0R1

Version 0.8 (06.11.2003 18:24)

Autoren:

Wolfgang (Shade) Kern
Sven (St%tic) Hummelsberger
Quellen aus Debian VServer HOWTO von Joel Wiesmann (Vielen Dank dafür!)
Http://www.wwip.de

Inhaltsverzeichnis:

1. -> Grundsätzliches
2. -> Systemvoraussetzungen
3. -> Benötigte Packte
4. -> Debian Installation
5. -> Kernel Update 2.4.20
6. -> VServer Installation
6.1 -> Vserver Einweisung
6.2 -> Vserver Referenzsystem erstellen
6.3 -> Vserver Detailkonfiguration
6.4 -> Vserver rebooten & Vserver Autostart
6.5 -> Softwareupdates über Hostsystem
6.6 -> Neuen V Erstellen
6.7 -> Befehlsübersicht

#1. Grundsätzliches:

Mit diesem Dokument möchten wir erläutern wie man eine Virtuelle Maschine unter Debian Woody 3.0r1 erfolgreich installiert. Viele werden schon Dokumentationen im Internet gefunden haben, diese jedoch, so finden wir, sind nicht ausführlich genug. Deshalb meine Dokumentation dazu. An dieser Stelle vielen Dank an Joel Wiesmann der eine gute Basis für diese Dokumentation gelegt hat und natürlich an St%tic der an diesem Projekt eifrig mitarbeitete!

Wir werden von Anfang an Beginnen und uns nach und nach durcharbeiten. Nichts soll zu kurz kommen. Deshalb ist es an dieser Stelle sinnvoll zu erwähnen dass man Debian mittels einer Minimal Boot CD starten sollte um dann von dort aus die Installation zu beginnen. Vorteil hierbei ist das System so schlank wie möglich zu halten. Im Nächsten Abschnitt (2) kommen wir zu den benötigten Paketen. Für die Installation eines Vserver Trägerhosts ist es von Nöten den Linux (Debian) Kernel auf die Version 2.4.20 up zu daten. An dieser Stelle sei gesagt dass eine Kernel Compilierung ausschließlich von erfahrenen Linux Benutzern durchgeführt werden sollte. Wir werden diesen Schritt zwar erklären aber es ist sehr individuell einen Kernel anzupassen, Du solltest also fundierte PC und Systemkenntnisse mitbringen. Anbei noch einige Tipps und Tricks unter Debian die von Nutzen sein könnten.

Packetverwaltung unter Debian:

Packte werden komfortabel über den Debian Packetverwalter nachinstalliert. Du brauchst dazu nur in der Shell

apt-get install PacketName

eintippen und schon wird das Packet, Internetverbindung vorausgesetzt, geladen und installiert. Wenn du schon heruntergeladene Datei installieren

möchtest, dann kannst du das mit

dpkg -i packet.deb

wobei dpkg unter Umständen wegen nicht erfüllter Abhängigkeiten meckern wird. In diesem Fall kann man mit "apt-get -f install" versuchen, die fehlenden Pakete zu installieren.

#2. Systemvoraussetzungen

Die Systemkomponenten sollten mit Bedacht ausgewählt werden da es sich hier um ein Serversystem handelt. Als Grundvoraussetzung sollte man mindestens besitzen:

CPU: Ab 1GHZ (Pentium III oder AMD XP, besser Dual System)
RAM: 512MB (1GB RAM und mehr sollten Performanceprobleme lösen)
HDD: Ab 80GB (Variiert je nach Speicher die ein V bekommen soll)
OS: Debian 3.0R1 (Woody Release)

#3. Benötigte Packete/Dateien:

Die Pakete die unbedingt notwenig sind hier kurz aufgezählt und erläutert.

1. gcc 2.95.3
2. ncursesdev
3. mc
4. ssh
5. make
6. linux-2.4.20.tar.gz
7. patch-2.4.20-vs1.00.diff
8. vserver-0.23.src.tar.gz
9. debian-newvserver.sh
10. debootstrap
11. patch
12. bzip2

Erklärung der einzelnen Pakete:

Punkt 1: Dies ist ein C Compiler zum Compilieren von .c Dateien
Punkt 2: Libary für Grafische Oberflächen in der Shell
Punkt 3: Der Midnight Comander (Ähnlich dem Norton Comander)
Punkt 4: Ein Secure Shell Server (Ähnlich Telnet)
Punkt 5: Der Make Befehl (C Komponente)
Punkt 6: Der Linux Kernel 2.4.20
Punkt 7: Der Vserver Patch für den 2.4.20 Kernel
Punkt 8: Das Vserver Programm
Punkt 9: Ein Shell Script zum anlegen neuer V’s
Punkt 10: Debian Komponente für die Softwareverwaltung
Punkt 11: Der Interpreter für das Patchen des Kernels
Punkt 12: Bzip 2 Komprimierer/Dekomprimierer

Punkte 1 – 5 & 10 - 12 per

apt-get install gcc ncursesdev mc ssh make debootstrap patch bzip2

Punkt 6 über die off. Kernel Homepage [2]
Punkte 7 & 8 über [3]
Punkt 9 über [5]

#4. Debian Installation:

Die Installation von Debian sollte über ein System mit Netzwerkkarte, Internet und wenn möglich hinter einem Router (Oder Ähnlichem was den Internetzugang mittels Gateway und DNS Eintrag ermöglicht).

Installiert sollte alles von einer 6MB großen CD, deren Image man unter [1] findet, werden. Das ermöglicht eine große Flexibilität in der Installation da man von einem minimalen Grundsystem ausgeht. Die in Punkt 3 erwähnten Pakete sollten dann auf jeden Fall mittels dem im Installationsprogramm enthaltenen Routinen installiert werden, falls nicht können Sie später nachinstalliert werden (Punkt 1 Grundsätzliches). Zu beachten ist bei der Festplattenpartitionierung, dass die VServer möglichst eine eigene Partition kriegen sollten. In meinem Fall hat das Hostsystem eine Kapazität von 3GB und eine 2. Platte 30GB (Dies ist ein Testsystem ;) auf der 2. Platte habe ich einige Partitionen angelegt, 1GB, 2GB, 5GB und 10GB welche in Verzeichnisse àla VM1-1, VM2-2, VM3-5 und VM4-10 bekommen haben.

Es ist von Vorteil beim tasksel keine, bzw. gerade mal die Pakete für C und C++ Development anzuwählen. dselect kann übersprungen werden. Ich habe im Nachhinein dselect aufgerufen, gecheckt welche Software mir noch untergejubelt werden soll und diese nochmals durchgeforscht und die unnötigen Packages entfernt.

Fazit: Kleines Hostsystem - ~200mb!

#5. Kernel 2.4.20 Installation

Um den Kernel zu installieren sind einige Schritte von Bedeutung die ungeübte Benutzer besser an einen Bekannten, Freund oder Fachmann vergeben sollten. In unserer Version wird der Kernel auf unser System zugeschnitten. Bitte beachte an dieser Stelle dass er unter Umständen auf anderen Systemen (Unterschied AMD – Intel) nicht lauffähig sein wird!

Zuerst sollten wir den Kernel linux-2.4.20.tar.gz mit dem Befehl:

```
Tar -xfz linux-2.4.20.tar.gz
```

entpacken. Das ganze machen wir am besten in einem erstellen Verzeichnis namens

```
/install/linux
```

In das erhaltene Verzeichnis /install/linux/linux-2.4.20/ entpacken wir dann auch das Kernelpacket linux-vserver-1.00.tar.bz2 mit dem Befehl

```
Gzip –d linux-vserver-1.00.tar.bz2
```

Wir erhalten dann eine Datei namens patch-2.4.20-vs1.00.diff die wir mit dem Befehl

```
patch –p1 patch-2.4.20-vs1.00.diff
```

ausführen und somit den Kernel damit auf die Tätigkeit des Vserver-Trägerhosts vorbereiten. Nach dem das geschehen ist sollten wir den Kernel konfigurieren. Das geschieht mit dem Befehl

```
make menuconfig
```

jetzt sollten die ersten Übersetzungsschritte passieren und nach einiger Zeit ein Grafisches Menü erscheinen in dem wir unseren Kernel anpassen. Die dortigen Anpassungen sollten mit Bedacht geschehen und mit äußerster

Sorgfalt durchgeführt werden, passiert hier ein Fehler ist es vorprogrammiert dass das System nach dem Reboot nicht wieder hochkommt. Lass dir hier unter Umständen von einer Erfahrenen Person helfen! An dieser Stelle wichtig:

Wir verlassen uns nicht auf Module! Bitte keine der im Konfigurationsmenü vorgegebenen Methoden als Modul nutzen sondern ausschließlich im Kernel fest eingebaut!!

Nach der erfolgreichen Beendigung der Konfiguration drücke man ESC und die Konfiguration wird gesichert. Sobald dieser Vorgang abgeschlossen ist kann man durch den Befehl

Make dep

Alle Abhängigkeiten prüfen und den Kernel somit so gut wie fertig stellen. Dieser Vorgang dürfte etwas Zeit in Anspruch nehmen, keine Panik! Sobald dieser Prozess abgeschlossen ist müssen wir den Kernel noch komprimieren, dazu dient dieser Befehl

Make bzImage

Bitte beachtet hierbei auf die Groß und Kleinschreibung! Nun wird ein Gepacktes Kernel Packet erstellt welches später gebootet werden kann. Sobald dieser Prozess abgeschlossen ist muss der Kernel anstelle des alten Kernels kopiert werden. Meist befindet sich dieser in einem der folgenden Verzeichnisse

/vmlinuz, /boot/vmlinuz, /bzImage oder /boot/bzImage

Der alte Kernel, meißt vmlinuz muss nun umbenannt werden. In meinem Fall habe ich das einfach vmlinuz_old getan. Der neue Kernel sollte gemäß den Spezifikationen in vmlinuz-2.4.20-ctx umbenannt werden. Der Link in / sollte ebenfalls mit dem Befehl

mv /vmlinuz /vmlinuz.old

umbenannt werde. Um einen neuen Link in / für den neuen Kernel zu erstellen geben wir folgenden Befehl ein

ln -s /boot/vmlinuz-2.4.20-ctx /vmlinuz

dies erstellt einen Link in unser Dokument Root / um den neuen Kernel booten zu können muss in

/etc/lilo.conf

Der Eintrag

```
# Boot up Linux by default.
#
Default=Linux

Image=/vmlinuz
        Label=Linux
        Read-only
```

Eingefügt bzw überprüft werden. Nach dem das geschehen ist muss der Lilo ausgeführt werden. Das geht mit dem einfachen Befehl

/sbin/lilo

Nun sollte eine Meldung kommen die besagt er habe etwas hinzugefügt (Added *). Nach dieser Prozedur muss der Rechner neu gestartet werden. Jetzt wird sich zeigen ob die Übersetzung geklappt hat. An diesem Punkt angelangt sollte man alle angeschlossenen Geräte testen. Bitte auch auf Fehler und Warnungen während des Bootens achten. Schwerwiegende Fehler sollten behoben werden! Sobald alles in Ordnung ist beginnen wir mit der Vserver Installation.

#6.1 Vserver Installation:

Anbei eine kurze Begriffserklärung:

-> Hostsystem:
System welches die VServer hostet.

-> Referenzsystem:
Ein Referenz-VServer. Die Quelle aller kommenden V's

-> V:
Ein produktiver VServer.

Nun entpacken wir die Datei vserver-0.23.src.tar.gz mit folgendem Befehl

Tar xvf vserver-0.23.src.tar.gz

Dies sollte ein Verzeichnis namen vserver-0.23 erstellt haben. Wir wechseln in das Verzeichnis und führen folgenden Befehl aus

Make && make install

Anschließend finden sich folgende neuen Dateien unter /usr/sbin:

```
vtop            (?)
vserver-stat    (Informationen ueber aktive VServer Instanzen)
vserver         (VServer Kontrollscript -> stop, start etc.)
vrpm            (unbenutzt)
vpstree         (?)
vps             (ps mit zusaetzlicher "CONTEXT"-Spalte)
vkill           (?)
vfiles          (?)
vdu             (unbenutzt)
reducecap       (Reduzieren der Capabilities)
rebootmgr       (Reboot-Daemon fuer V's)
newvserver      (unbenutzt)
chcontext       (Context change)
chbind          (Bindet Prozesse an spezifische IP)
```

Startscripte in /etc/init.d:

```
vservers        (Startet alle vserver welche ON_BOOT=yes haben in Config)
v_xinetd        (Startet xinetd gebunden an IP)
v_sshd          (Selbe mit SSHD)
v_smb           (Selbe mit Samba)
v_sendmail      (Selbe mit Sendmail)
v_portmap       (Selbe mit Portmap)
v_named         (Selbe mit named)
v_httpd         (Selbe mit httpd)
rebootmgr       (Reboot-Manager fuer V's und vreboot)
```

Ausserdem unter /usr/lib/vserver:

```
capchroot       (?)
fakerunlevel    (?)
filetime        (?)
```

```
ifspec          (?)
install-<...>   (unbenutzt)
listdevip       (?)
readlink        (?)
sample.conf     (Beispiel V-Config)
sample.sh       (unbenutzt)
save_s_context  (?)
showattr        (unbenutzt, nur bei Hardlinks -> spezielles Attribut)
showperm        (Permission Bits fuer Dateien Ausgabe)
vbuild          (unbenutzt)
vcheck          (unbenutzt)
vreboot         (fuer reboots in VServern)
vserverkillall (Killall in VServern)
vservers.grabinfo.sh   (VServer infos und Status in XML)
vsysvwrapper    (?)
vunify          (unbenutzt)
```

Einige dieser Tools wie vrpm und vunify sind allerdings leider für uns nicht von Interesse, da sie leider für den RPM gedacht sind.

6.2 Referenzsystem mit debian-newvserver.sh

Zuerst sollten wir uns im Klaren sein wohin wir die VServer wollen. Per "default" wird überall in der Dokumentation /vservers angegeben. Also machen wir es auf dieselbe art:

```
mkdir /vservers
```

```
debian-newvserver.sh --copy-vreboot --dist woody --fakeinit --mirror
<mirror> -v --vsroot /vservers --hostname base --domain <domain> --ip <ip
des vhost>
```

Dies erstellt den Referenz-VServer. Das Hostsystem sollte nun vom gewählten Debian Spiegel[6] herunterladen. Bei einigen Systemen ist es notwendig diesen Befehl mit einem sh vorangesetzt zu starten. Man kann vservers/ARCHIVE nach der Referenzsystem-Installation löschen. Es ist von Vorteil bei tasksel nur die C/C++ Development Pakete anzuwählen und den Rest dediziert auf die vserver zu laden.

Es schadet nicht trotz dem installieren per Script einen Blick in Kapitel 7 zu werfen da dieses auch noch einige Fragen abdeckt.

6.3. Detailkonfigurationen

Nun sind wir immer nur local eingeloggt. Wenn wir nun per ssh versuchen auf die zugewiesene IP einzuloggen kommen wir aber noch immer auf das Hostsystem. Um dies zu vermeiden müssen wir auf dem Hostsystem in der Datei

```
/etc/ssh/sshd_config
```

in der Zeile

```
ListenAddress 0.0.0.0"
```

die 0.0.0.0 zu der IP des Hostsystemes ändern. Danach ein:

```
/etc/init.d/ssh stop && /etc/init.d/ssh start
```

und:

```
vserver base stop && vserver base start
```

Nun sollte es funktionieren. Jetzt sollten wir von unserer Installationsorgie noch ein paar Leichen auf dem V (!) aufräumen (_NICHT_ Hostsystem!):

```
cp /usr/lib/vserver/vreboot /sbin/reboot
cp /sbin/reboot /sbin/halt
cp /sbin/halt /sbin/shutdown
rm -r /usr/src/linux* /usr/lib/vserver /etc/vservers #
```

VServer und Linux-Sourcen brauchen wir nicht auf diesem System

```
cd /usr/sbin
```

```
rm vtop vserver vserver-stat vrpm vpstree vps vkill vfiles vdu reducecap
rebootmgr newvserver chcontext chbind
```

6.4. VServer rebooten & VServer start bei Bootup

Wie soll man nun einen VServer wenn man denn mal eingeloggt ist rebooten? Dazu brauchen wir die Tools vreboot (welches wir zu /sbin/reboot, /sbin/halt und /sbin/shutdown in dem Referenzsystem kopiert haben) und auf dem Hostsystem den rebootmgr-Daemon. Um den rebootmgr zu starten können wir den bereits vorhandenen Startscript

```
/etc/init.d/rebootmgr
```

in rc2 linken.

```
ln -s /etc/init.d/rebootmgr /etc/rc2.d/S90rebootmgr
ln -s /etc/init.d/rebootmgr /etc/rc2.d/K90rebootmgr
```

Um alle VServer automatisch zu starten welche in der Konfigurationsdatei

```
ONBOOT=yes
```

haben, linken wir die bereits existierende

```
/etc/init.d/vservers
```

ebenfalls nach rc2 aber vor den rebootmgr:

```
ln -s /etc/init.d/vservers /etc/rc2.d/S89vservers
ln -s /etc/init.d/vservers /etc/rc2.d/K89vservers
```

Wer jetzt mutig ist, darf nun das Hauptsystem rebooten und prüfen ob der Base-Vserver wieder oben ist und rebootmgr läuft.

Anschließend loggen wir uns auf dem base ein und probieren gleich mal zu rebooten. Um uns auf den base einzulogen geben wir ein

```
Vserver base enter
```

Der Reboot sollten dann wie folgt aussehen

```
base:~# reboot
base:~# Connection to 192.168.1.20 closed by remote host.
Connection to 192.168.1.3 closed.
```

Falls ihr via "vserver base enter" rebootet, habt ihr möglicherweise wenn es euch auf euer Hauptsystem rausschmeißt ein defektes Terminal (ihr sehr nicht mehr was ihr schreibt). Um dies zu beheben einfach "reset" eingeben.

6.5. Smartes Softwareupgrade

Nun sind wir an dem Punkt angelangt wo wir über DPKG nachdenken müssen. DPKG funktioniert ohne Probleme auf unserem Referenz-VServer. Aber wenn wir nun noch 10 weitere V's aus dem base Referenzsystem machen wird's langsam ein bisschen viel wenn alle V Systeme ein

"apt-get update && apt-get upgrade" fahren.

Die Lösung ist eigentlich einfach: Wir wissen unser Referenzsystem "base" ist die Grundlage aller V's. Wenn wir also das base-System upgraden (was wir sowieso müssen um die V's die daraus noch entstehen aktuell zu halten), müssen wir dieselben Patches auch auf den bereits existierenden V's installieren. Zusätzlich kommen bei den V's noch eigene, auf den VServer dedizierte Software hinzu, um welchen sich aber der VServer selbst kümmern sollte.

Wir lassen das Referenzsystem also ein "apt-get update" fahren und anschließend statt einem

"apt-get upgrade" ein "apt-get -dy upgrade".

Dies lädt die Packages nur runter welche upzugraden sind. Diese können nun mittels einem Script und Cronjob vom Hostsystem aus kontrolliert werden.

Wieso vom Hostsystem aus? Ganz einfach Per default sollten wir den Referenzserver deaktiviert halten (vserver base stop - sonst könnte man mit default-Passwort einloggen) - somit ist Cron nicht aktiv. Ein weiterer Grund ist, dass wir vom Hostsystem auf alle VServer zugreifen können und mit

vserver base exec <kommando>

"remote" vom Hostsystem auf den VServern Befehle ausführen können und ebenso die herunter geladenen Packages so einfach verteilen.

Das fertige Script kann von [4] bezogen werden!

6.6. Neuen V erstellen

Um nun einen neuen V zu erstellen und es mit dem [4]-Script zu benutzen muss man eine Kopie des Referenzsystemes erstellen. Das Referenzsystem selbst darf somit nicht produktiv genutzt werden und sollte _immer_ deaktiviert sein.

mkdir /vservers/<V>

chmod 000 /vservers/<V>/..

cp -Rpv /vservers/base/bin /vservers/base/dev /vservers/base/etc /vservers/base/home /vservers/base/initrd /vservers/base/lib /vservers/base/mnt /vservers/base/opt /vservers/base/root /vservers/base/sbin /vservers/base/tmp /vservers/base/usr /vservers/base/var /vservers/<V>

cp /etc/vservers/base.conf /etc/vservers/<V>.conf

Anpassen im V:

/etc/hostname
/etc/hosts
/etc/motd

Anpassen auf dem Hostsystem:

/etc/vservers/<V>.conf

Ihr habt es geschafft! Ihr könnt nun Vserver erstellen, rebooten, stoppen, starten und mit aktueller Software versorgen! Versucht nach einem Start eines V's eurer wahl nun einmal einen V mittels Putty (unter Windows) oder mittels ssh unter Linuc über SSH zu erreichen. Akzeptiert die Fingerprint Nachricht und logt euch ein ;)

6.7 Befehlsreferenzen

vserver <V> enter	V Betreten
vserver <V> stopp	V Stoppen
vserver <V> start	V Starten
vserver-stat	V Systeme überprüfen
vserver base exec <kommando>	Einen Befehl vom Host auf dem V Ausführen
apt-get install <Packet>	Lädt & Installiert Packete Von Debian
apt-get upgrade	Updatet Debian Packete
./configure	Sourcen Konfiguration
make	Befehl um Packete zu kompilieren
Make install	Kompilierte Sourcen installieren

Links

[1] BootCD DownLink:
http://people.debian.org/~dwhedon/boot-floppies/bootcompact.iso

[2] Debian Installation:
http://www.debian.org/releases/stable/installmanual

[2] Linux Kernel Sourcen
http://www.kernel.org

[3] VServer Homepage:
http://www.linux-vserver.org

[4] Smart Upgrade Script (smart-upgrade.sh) Danke an http://secuserv.ch
http://www.world-of-pit.de/downloads/shade/smart-upgrade.sh

[5]
debian-newvserver.sh
http://www.paul.sladen.org/vserver/debian/