**Linux-vserver on debian**

**Installation on Debian**

**From Linux-VServer**

Jump to: [navigation](http://linux-vserver.org/Installation_on_Debian#column-one), [search](http://linux-vserver.org/Installation_on_Debian#searchInput)

**Note:** Debian 6.0 is the final version to include precompiled Linux-Vserver kernels. In newer versions (including Debian Testing), you'll have to compile the kernel yourself or [use a pre-packaged kernel](http://linux-vserver.org/Frequently_Asked_Questions#Were_can_I_get_newer_versions_of_VServer_as_ready_made_packages_for_Debian.3F). [[1]](http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=574529)

This guide is written against Debian Etch (4.0) and works on Lenny (5.0) as well. Both releases include kernel **linux-image-vserver-686**, so no manual patching is needed. Hence, Installation on Debian Etch/Lenny is pretty easy and straightforward.

If you need to compile your own kernel, you need to apply the vserver-version.patch. [Details at 2007/May/04](http://www.kwu.hu/vserver.txt)

In lenny and etch the tools are for the 2.2 version of vservers, you can find on beng repository packages for the 2.3 version of util-vserver until it is integrated in debian. See

* [util-vserver:Devdebianpackage](http://linux-vserver.org/util-vserver:Devdebianpackage) - Info about debian v2.3 package from the community
* explanation on how to use this repository from: <http://repo.psand.net/info> How to use the Debian Repository

|  |
| --- |
| **Contents**   [[hide](http://linux-vserver.org/Installation_on_Debian)]   * [1 Packages installation](http://linux-vserver.org/Installation_on_Debian#Packages_installation)   + [1.1 Install util-vserver by source](http://linux-vserver.org/Installation_on_Debian#Install_util-vserver_by_source) * [2 Versions](http://linux-vserver.org/Installation_on_Debian#Versions) * [3 Issues with Squeeze's 2.6.32 Kernel and Util-vserver](http://linux-vserver.org/Installation_on_Debian#Issues_with_Squeeze.27s_2.6.32_Kernel_and_Util-vserver) * [4 Issues with Lenny's 2.6.26 Kernel and Util-vserver](http://linux-vserver.org/Installation_on_Debian#Issues_with_Lenny.27s_2.6.26_Kernel_and_Util-vserver)   + [4.1 Hard CPU scheduling](http://linux-vserver.org/Installation_on_Debian#Hard_CPU_scheduling)   + [4.2 Problems due to Xattrs](http://linux-vserver.org/Installation_on_Debian#Problems_due_to_Xattrs)     - [4.2.1 Chroot Security Problems](http://linux-vserver.org/Installation_on_Debian#Chroot_Security_Problems)     - [4.2.2 Unification Problems](http://linux-vserver.org/Installation_on_Debian#Unification_Problems)   + [4.3 /proc/mounts issue](http://linux-vserver.org/Installation_on_Debian#.2Fproc.2Fmounts_issue)   + [4.4 "Ghosts" guests](http://linux-vserver.org/Installation_on_Debian#.22Ghosts.22_guests)     - [4.4.1 Issue](http://linux-vserver.org/Installation_on_Debian#Issue)     - [4.4.2 Fix](http://linux-vserver.org/Installation_on_Debian#Fix) * [5 Issues with Wheezy's 3.2.55 Kernel and Util-vserver](http://linux-vserver.org/Installation_on_Debian#Issues_with_Wheezy.27s_3.2.55_Kernel_and_Util-vserver) * [6 References](http://linux-vserver.org/Installation_on_Debian#References) |

**Packages installation**

The packages required by Linux-VServer are:

* **linux-image-vserver-686** - This is the current kernel, use **linux-image-vserver-amd64** on 64-bit systems, you can still create 32-bit guests
* **util-vserver** - These are the utilities used to administer the guests
* **ssh** - This is probably already installed, but just in case it isn't

All the packages you need can be obtained via

aptitude install linux-image-vserver-686 util-vserver ssh

so run this as *root* and reboot. To check out wherever everything went fine you may run

uname -r

and check that kernel version contains **vserver**, e.g. **2.6.18-4-vserver-686**. That's it.

Now that the host system is ready, you can proceed with [building guests](http://linux-vserver.org/Building_Guest_Systems).

**Install util-vserver by source**

Occasionally, Debian's util-vserver package can be too old. So, we'll need to compile from [source](http://people.linux-vserver.org/~dhozac/t/uv-testing/).

First, install the required packages for util-vserver to compile.

apt-get install vlan dietlibc-dev pkg-config libnss3-dev

Then, we configure util-vserver

./configure --prefix=/usr --enable-release --mandir=/usr/share/man \

--infodir=/usr/share/info --sysconfdir=/etc --enable-dietlibc \

--localstatedir=/var --with-vrootdir=/home

Note: You should change *--with-vrootdir* accordingly

Finally, we run make to finalise the installation

make && make install install-distribution

Running *vserver-info* will show you that the proper util-vserver is installed. :)

Debian likes to be funny, so we need to enable the following,

* echo /usr/lib/util-vserver/vshelper >| /proc/sys/kernel/vshelper
* echo kernel.vshelper = /usr/lib/util-vserver/vshelper >> /etc/sysctl.conf
* update-rc.d vprocunhide defaults
* update-rc.d vservers-default defaults

**Versions**

Debian already contains vservers kernels, so no manual patching and compiling is needed.

|  |  |  |
| --- | --- | --- |
| **Debian release** | **Kernel version** | **VServer version** |
| Squeeze | 2.6.32 | 2.3.0.36.29.6 |
| Lenny | 2.6.26+17 | 2.3.0.35 |
| Etch | 2.6.18+6 | 2.0.2.2-rc9 |

The Vserver versions given above are not completely pure, they have additional patches to fix various issues.

Information on alternative Debian repositories with more functional packages is [contained in this section of the FAQ](http://linux-vserver.org/Frequently_Asked_Questions#Were_can_I_get_newer_versions_of_VServer_as_ready_made_packages_for_Debian.3F).

**Issues with Squeeze's 2.6.32 Kernel and Util-vserver**

* Util-vserver shipping with debian, does not have the symbolic link for squeeze, fixed by

ln -s debian /usr/lib/util-vserver/distributions/squeeze

**Issues with Lenny's 2.6.26 Kernel and Util-vserver**

**Hard CPU scheduling**

This will not work in the Debian 'Lenny' Kernel, the patch used simply does not contain any of this functionality.

**Problems due to Xattrs**

There are two sets of issues within the Lenny kernel caused by the change in value of the Xattrs (extended attributes) applied to file in Vserver setups. The patch used in Debian Lenny uses Xattr flags which are set in positions which differ from the flags set by Debian kernels as well as most of the mainline Vserver patches. This result is that Xattrs of files in a non lenny system appear to have completely different flags in Lenny and vice versa. Since these flags are crucial to vserver hashification and chroot security, they can have devastating effects on Vserver guests and on host system security. If you have recently moved to or away from the stock Lenny Vserver kernel, have look at the symptoms below to see if any match your experiences, and apply the fixes/use another kernel as you see fit.

As of writing these issue has not been corrected within the Debian archive. These fixes must be applied whenever moving vserver guest **from** or **to** the Debian 'Lenny's vserver kernel. For more details and a more concise explanation see [Bertls IRC explanation](http://irc.13thfloor.at/LOG/2009-05/LOG_2009-05-12.txt) .

**Chroot Security Problems**

Linux-Vserver uses file Xattrs to protect guest superusers from being able to view files above their root, preventing access to host file. This creates issues for anyone who:

* has created a guest with a Debian 2.6.26-\*-vserver kernel and wishes to use it with another kernel.
* has created a guest with a different kernel and wishes to use it on a Debian 2.6.26-\*-vserver kernel based host.

In effect, the barrier normally in place for guest servers is not recognised by the kernel (the chroot problem) in the situation above and/or immutable links will not function correctly (the unification problem)failing to break when overwritten) in a unified guest setup. Symptoms suffered may include:

* the possibility of vserver guest processes escaping their chroots and accessing other parts of the filesystem
* guest not starting

To fix the barrier flags for a current kernel, see [these instructions](http://linux-vserver.org/Secure_chroot_Barrier#Solution:_Secure_Barrier). Note that on some setups a barrier flags will appear on all directories under the guest hierarchy, and need to be unset in order to allow the servers to run. Use showattr to reveal the state of play for your guests and fix appropriately.

**Unification Problems**

There is a discrepancy between the immutable-unlink flag used for file unification, the process used in vhashify. This creates considerable issues for anyone who:

* has unified guests with a Debian 2.6.26-\*-vserver kernel and wishes to use them with another kernel.
* has unified guests with a different kernel and wishes to then it on a Debian 2.6.26-\*-vserver kernel based host.

Symptoms suffered may include:

* file that cannot be deleted
* any process involving the writing of files in guests not working
* files not being unlinked on write

To fix the problem each file must be unlinked then the unification re-applied, or one could try this script submitted to [bugs.debian.org](http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=508523).

**/proc/mounts issue**

The vserver's /proc/mounts let appear the vserver path on the host. lsof (for example) is able to print it.

**"Ghosts" guests**

**Issue**

Sometimes a guests loose it's name in vserver-stats and is acting like a zombie. It's impossible to restart or kill it. Stopping all the guests with the util-vserver init.d script doesn't solve the issue. vkill --xid $CTX doesn't either.

**Fix**

echo 50 > /var/run/vservers/$vserver

Seems to fix the issue

**Issues with Wheezy's 3.2.55 Kernel and Util-vserver**

* Util-vserver 0.30.216-pre3060-1 from the APT repository (repo.psand.net) moves vshelper to a new location. If you attempt to manage guests (restart, start, etc) without restarting the util-vserver service, vserver will display "vshelper functionality" errors. The util-vserver service will notify the kernel of the new location when the service is restarted. However, restarting util-vserver will also restart all of your guests. You can immediately notify the kernel of the new vshelper location (without restarting util-vserver or your guests) by running the following command:

echo /usr/lib/x86\_64-linux-gnu/util-vserver/vshelper >| /proc/sys/kernel/vshelper

**References**

* Linux-VServer HOWTO by Daniel15: <http://howtoforge.com/linux_vserver_debian_etch>

Retrieved from "<http://linux-vserver.org/index.php?title=Installation_on_Debian&oldid=38565>"

**Building Guest Systems**

**From Linux-VServer**

Jump to: [navigation](http://linux-vserver.org/Building_Guest_Systems#column-one), [search](http://linux-vserver.org/Building_Guest_Systems#searchInput)

|  |
| --- |
| **Contents**   [[hide](http://linux-vserver.org/Building_Guest_Systems)]   * [1 Building a guest](http://linux-vserver.org/Building_Guest_Systems#Building_a_guest)   + [1.1 Building guests using the debootstrap build method](http://linux-vserver.org/Building_Guest_Systems#Building_guests_using_the_debootstrap_build_method)   + [1.2 Building guests using the yum build method](http://linux-vserver.org/Building_Guest_Systems#Building_guests_using_the_yum_build_method)   + [1.3 Building guests using the template build method](http://linux-vserver.org/Building_Guest_Systems#Building_guests_using_the_template_build_method)   + [1.4 Building guests using the rsync build method](http://linux-vserver.org/Building_Guest_Systems#Building_guests_using_the_rsync_build_method)   + [1.5 Building guests using the clone build method](http://linux-vserver.org/Building_Guest_Systems#Building_guests_using_the_clone_build_method) * [2 Post-build customization](http://linux-vserver.org/Building_Guest_Systems#Post-build_customization) * [3 Verifying guest functionality](http://linux-vserver.org/Building_Guest_Systems#Verifying_guest_functionality) * [4 How to remove a screwed up vserver](http://linux-vserver.org/Building_Guest_Systems#How_to_remove_a_screwed_up_vserver) * [5 See also](http://linux-vserver.org/Building_Guest_Systems#See_also) |

**Building a guest**

There are three parts to the vserver ... build command. Each part is separated by --. The first part are generic vserver options, such as hostname, available IP-addresses, and context id. The second part is specific to the chosen build method (by -m <method> in the first part). The third part is completely optional and only implemented for a few build methods. These are options passed directly to whatever application is used to build guest.

There are a lot of options not covered here (yet). Use vserver - build --help to see them all.

**Building guests using the debootstrap build method**

***(applies to Debian, Ubuntu,...)***

* Build a Debian lenny guest using the ftp.de.debian.org mirror.

vserver vserver1 build \

-m debootstrap --[context](http://linux-vserver.org/Context) 42 \

--hostname vserver1.mydomain.com \

--interface eth0:192.168.1.10/24 \

-- -d lenny -m <http://ftp.de.debian.org/debian>

lenny can be replaced by e.g. etch for Debian etch, edgy for Ubuntu edgy, or sid for Debian sid. Anything your debootstrap version can handle will be fine.

If you want to build a 32-bit guest on a 64-bit host, append -- --arch i386 to the above command line. The same applies to other additional debootstrap options: place them after a second --. Likewise, in case your host architecture does not match one available in Debian by default, add your architecture in the same way. E.g. you might need to add -- --arch amd64 to create a 64-bit Debian guest on a Gentoo host.

Please note that vserver...pkgmgmt has has no use on Debian guests: package management is internalized by default and cannot be externalized. To externalize package management on Debian vserver guests, you have to create the guest using the newvserver command from the vserver-debiantools Debian package (see the --pkgcache option). Note that newvserver is considered obsolete and only suited for those who want to foot themselves in shoot.

**Building guests using the yum build method**

***(applies to Fedora, Centos, RHEL,...)***

* Make sure that your host system already has yum installed, preferably one that has been patched to work better with chroots.
* Build a CentOS 5-based guest using a minimal set of packages.

vserver vserver2 build -m yum --[context](http://linux-vserver.org/Context) 42 \

--hostname vserver2.mydomain.com \

--interface eth0:192.168.1.11/24 -- -d centos5

centos5 can be replaced by e.g. f8 for Fedora 8, or fc6 for Fedora 6.

If you want to build a 32-bit guest on a 64-bit, prepend linux32 to this and subsequent yum commands.

* If you want to manage the packages inside the guest, you will have to install some package management program(s) as well as internalize the RPM database. This can be achieved by

vyum vserver2 -- install yum

vserver vserver2 pkgmgmt internalize

Internal package management means that commands such as rpm, yum and rpmbuild can be used from inside the guest, as opposed to requiring the host administrator to run vrpm or vyum. If you use rpmbuild, you'll need internal package management, or use --nodeps (but that's *strongly* discouraged).

* Depending on your host's/guest's distribution, you may also need to do

vserver vserver2 exec bash -c "rm -f /var/lib/rpm/\_\_db\*; rpm --rebuilddb"

This is required if every rpm operation, for example *rpm -qa*, complains about a database version mismatch.

**Building guests using the template build method**

***(applies to Gentoo,...)***

A template is a file containing a complete guest filesystem. This can be a tar(1)ball, a cpio(1)-archive, or a dump(8). It can be compressed using either gzip or bzip2. Multiple templates can be used, to do e.g. guest-specific modifications.

* Build a guest using a single template named stage4-i686-20070905.tar.bz2 located in /vservers/.templates.

vserver vserver3 build -m template \

--[context](http://linux-vserver.org/Context) 42 --hostname vserver3.mydomain.com \

--interface eth0:192.168.1.12/24 \

--initstyle gentoo -- -d gentoo \

-t /vservers/.templates/stage4-i686-20070905.tar.bz2

**OR**

* Build a guest using multiple templates, one named stage4-i686-20070905.tar.bz2 and one named httpd.tar.bz2.

vserver vserver3 build -m template \

--[context](http://linux-vserver.org/Context) 42 --hostname vserver3.mydomain.com \

--interface eth0:192.168.1.12/24 \

--initstyle gentoo -- -d gentoo \

-t /vservers/.templates/stage4-i686-20070905.tar.bz2 \

-t /vservers/.templates/httpd.tar.bz2

**Building guests using the rsync build method**

The rsync build method can be used to move a guest from one system to another. It is preferable for most guest distributions if the source guest is stopped when you create a one based on it, but it's not strictly required.

* Build a guest by rsync'ing from vserver3 on host1.

RSYNC\_RSH=ssh

vserver vserver4 build -m rsync --[context](http://linux-vserver.org/Context) 42 \

--hostname vserver4.mydomain.com \

--interface eth0:192.168.1.13/24 \

-- --source root@host1:/vservers/vserver3

**OR**

* Build a guest by rsync'ing from vserver1 on the same host.

vserver vserver4 build -m rsync --[context](http://linux-vserver.org/Context) 42 --hostname vserver4.mydomain.com --interface eth0:192.168.1.13/24 -- --source vserver1

**Building guests using the clone build method**

The clone build method copies the filesystem from one guest to another, much like the rsync build method, but the thing that separates it is that it knows about unified/hashified files. This means that it only creates new links for such files, and copies the rest, which can lead to significantly speedier builds.

* Build a guest by using vserver4 as a reference.

vserver vserver5 build -m clone --hostname vserver5.mydomain.com --interface eth0:192.168.1.14/24 -- --source /vservers/vserver4

**Post-build customization**

**Verifying guest functionality**

* Once the guest is built, it's time to start it.

vserver vserverX start

* At this point, you can use both

vserver vserverX enter

and

vserver vserverX exec ...

to do things inside the guest.

* To stop it, simply use

vserver vserverX stop

**How to remove a screwed up vserver**

* To to remove a screwed up vserver

vserver vserverX delete

**See also**

* [Installing Ubuntu on Debian](http://linux-vserver.org/Installing_Ubuntu_on_Debian)
* [Installing 32-bit Fedora on 64-bit Debian](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian)

Retrieved from "<http://linux-vserver.org/index.php?title=Building_Guest_Systems&oldid=5009>"

**Installing Ubuntu on Debian**

**From Linux-VServer**

Jump to: [navigation](http://linux-vserver.org/Installing_Ubuntu_on_Debian#column-one), [search](http://linux-vserver.org/Installing_Ubuntu_on_Debian#searchInput)

How to install an Ubuntu vserver guest on Debian.

|  |
| --- |
| **Contents**   [[hide](http://linux-vserver.org/Installing_Ubuntu_on_Debian)]   * [1 Prepare debootstrap](http://linux-vserver.org/Installing_Ubuntu_on_Debian#Prepare_debootstrap)   + [1.1 What is a Debootstrap?](http://linux-vserver.org/Installing_Ubuntu_on_Debian#What_is_a_Debootstrap.3F)   + [1.2 Preparing](http://linux-vserver.org/Installing_Ubuntu_on_Debian#Preparing) * [2 Build](http://linux-vserver.org/Installing_Ubuntu_on_Debian#Build)   + [2.1 Build with guest architecture the same as host (eg. 32-bit guest on 32-bit host, 64-bit guest on 64-bit host etc.)](http://linux-vserver.org/Installing_Ubuntu_on_Debian#Build_with_guest_architecture_the_same_as_host_.28eg._32-bit_guest_on_32-bit_host.2C_64-bit_guest_on_64-bit_host_etc..29)   + [2.2 Build 32-bit guest on 64-bit host](http://linux-vserver.org/Installing_Ubuntu_on_Debian#Build_32-bit_guest_on_64-bit_host) * [3 See also](http://linux-vserver.org/Installing_Ubuntu_on_Debian#See_also) |

**Prepare debootstrap**

**What is a Debootstrap?**

**From Ubuntu Packages:**

*"debootstrap is used to create a Debian base system from scratch, without requiring the availability of dpkg or apt. It does this by downloading .deb files from a mirror site, and carefully unpacking them into a directory which can eventually be chrooted into."*

So a debootstrap is basically something that will let you build your guest system.

**Preparing**

You will need the debootstrap script for your desired Ubuntu release (eg feisty, gusty, hardy).

Check that you have the debootstrap script in /usr/share/debootstrap/scripts/, eg /usr/share/debootstrap/scripts/gusty.

If you do not have the debootstrap script for your desired Ubuntu release, the easiest way is to download and install the latest debootstrap deb from [Debian sid](http://packages.debian.org/sid/debootstrap).

If the script is not available in Debian sid, download Ubuntu's latest bootstrap release from [here](http://packages.ubuntu.com/search?keywords=debootstrap). (eg, as of August 2008, download the debootstrap deb for Hardy Heron). You want the debootstrap from the latest release, even if you want to install a previous release Ubuntu vserver guest.

Extract the Ubuntu scripts from the deb and place them in /usr/share/debootstrap/scripts/.

For example, do something like:

$ cd /tmp

$ wget <http://mirrors.kernel.org/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.13.deb>

$ dpkg --extract debootstrap\_1.0.13\_all.deb debootstrap-intrepid

$ sudo cp -P debootstrap-intrepid/usr/share/debootstrap/scripts/{warty,warty.buildd,hoary,hoary.buildd,breezy,dapper,edgy,feisty,intrepid} /usr/share/debootstrap/scripts/

**Build**

**Build with guest architecture the same as host (eg. 32-bit guest on 32-bit host, 64-bit guest on 64-bit host etc.)**

# vserver hardy-guest build -m debootstrap --context 42 --hostname hardy-guest --interface dummy0=eth0:10.0.0.123/8 --rootdir /var/lib/vservers -- -d hardy -m <http://us.archive.ubuntu.com/ubuntu/>

**Build 32-bit guest on 64-bit host**

# vserver hardy-guest-32 build -m debootstrap --context 42 --hostname hardy-guest-32 --interface dummy0=eth0:10.0.0.123/8 --rootdir /var/lib/vservers -- -d hardy -m <http://us.archive.ubuntu.com/ubuntu/> -- --arch i386

Verify 32-bitness of guest:

host:# file /var/lib/vservers/hardy-guest-32/bin/ls

/var/lib/vservers/hardy-guest-32/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.6.0, dynamically linked (uses shared libs), for GNU/Linux 2.6.0, stripped

**See also**

* [Building Guest Systems](http://linux-vserver.org/Building_Guest_Systems)
* [Frequently\_Asked\_Questions#I\_run\_a\_Debian\_host\_and\_want\_to\_build\_an\_Ubuntu\_guest.\_Howto.3F](http://linux-vserver.org/Frequently_Asked_Questions#I_run_a_Debian_host_and_want_to_build_an_Ubuntu_guest._Howto.3F)
* [Installing 32-bit Fedora on 64-bit Debian](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian)
* [Upstart issues](http://linux-vserver.org/Upstart_issues)

Retrieved from "<http://linux-vserver.org/index.php?title=Installing_Ubuntu_on_Debian&oldid=5004>"

**Installing 32-bit Fedora on 64-bit Debian**

**From Linux-VServer**

Jump to: [navigation](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#column-one), [search](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#searchInput)

Guide to install a 32-bit Fedora guest into a 64-bit Debian host.

|  |
| --- |
| **Contents**   [[hide](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian)]   * [1 Set up yum repository](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#Set_up_yum_repository)   + [1.1 Proxy](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#Proxy) * [2 Build vserver](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#Build_vserver) * [3 Verify](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#Verify) * [4 See also](http://linux-vserver.org/Installing_32-bit_Fedora_on_64-bit_Debian#See_also) |

**Set up yum repository**

host# mkdir /etc/rpm

host# mkdir /var/lib/rpm

host# apt-get install yum linux32

host# cp -a /usr/lib/util-vserver/distributions/fc6/{yum,yum.repos.d} /etc/vservers/.distributions/fc6

vserver build does not use /etc/yum\*, though you will see an ignorable error message about it if it doesn't exist.

Edit the files /etc/vservers/.distributions/fc6/yum.repos.d/fedora-{core,updates,extras}.repo

* Uncomment the first baseurl= line of each file (by removing the leading # character)

**Proxy**

If you use a proxy, append to /etc/vservers/.distributions/fc6/yum/yum.conf (in the [main] section) a line of the form:

proxy=<http://proxy.yourcompany.com:3128>

**Build vserver**

host# export ARCH=i386

host# export http\_proxy=<http://proxy.yourcompany.com:3128>

host# linux32 vserver fedora-zod-32-gustname build -m yum --context 42 --hostname=fedora-zod-32-guestname --interface dummy0=eth0:10.0.0.123/8 --rootdir /var/lib/vservers -- -d fc6 -- --arch i386

host# unset ARCH

you may also create **/etc/rpm/platform** file:

guest# vi /etc/rpm/platform

i386-redhat-linux-gnu

this way yum and rpm will use i386 packages

NOTE: You may get an error "AttributeError: 'module' object has no attribute 'Context'" when executing this (Debian Lenny 5.0.3) because the gpgme package is too old. You can simply edit the misc.py file and early out that function that fails and the process will continue just fine. See <http://openvz.org/pipermail/users/2009-January/002554.html> for details.

**Verify**

To observe the 32-bitness of the guest's applications, do:

host# linux32 vyum fedora-zod-32-guestname -- install file

host# vserver fedora-zod-32-guestname start

host# vserver fedora-zod-32-guestname enter

fedora-zod-32-guestname# file /bin/ls

/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.9, stripped

or

host# file /var/lib/vservers/fedora-zod-32-guestname/bin/ls

/var/lib/vservers/fedora-zod-32-guestname/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared libs), for GNU/Linux 2.6.9, stripped

**See also**

* [Building Guest Systems](http://linux-vserver.org/Building_Guest_Systems)
* [Installing Ubuntu on Debian](http://linux-vserver.org/Installing_Ubuntu_on_Debian)

Retrieved from "<http://linux-vserver.org/index.php?title=Installing_32-bit_Fedora_on_64-bit_Debian&oldid=5001>"