ISPmail tutorial for Debian Squeeze

- Add new comment
- 242890 reads

This tutorial is for the former stable version "Debian Squeeze". You are likely using newer Debian releases so please choose the right tutorial for your Debian installation.

What this tutorial is *NOT* about

If you just need a no-cost mail server quickly and don't care about the technical mumbo-jumbo then this is the wrong place for you. There are ready solutions like iRedMail if you are in a hurry. Don't expect to learn much though if you chose the dark side. I personally recommend you understand what you are doing - that's what the ISPmail tutorials are for. Experience from giving support in the #postfix channel on IRC shows that most hobby sysadmins fail because some detail in a complex setup goes wrong and they have no idea how to track it down. Actually I have a feeling that other projects use the ISPmail tutorials to get started, too. :)

What this tutorial is about

You are interested in learning all the basics about running your own mail server using Debian Squeeze? Be my guest. All you need is a computer running Debian (Squeeze) and an internet connection. This tutorial explains step-by-step how to set up such a server and give you lots of background information in the process. Depending on the internet connection and server hardware you will be able to run a reliable mail service for thousands of domains and users.

Soon you will be proud operator of a mail server that can:

- receive and store email for your users from other mail servers on the internet
- let your users retrieve the email through IMAP and POP3 - even using SSL-encrypted connections
- receive and forward ("relay") email for your users using SMTP authentication
- offer a webmail interface to read emails in a web browser
- detect most spam emails and filter them out or tag them

License/Copyright

Things you will need

The server setup described here is totally standard work for a professional system administrator. Depending on your level of knowledge and experience you may walk through this document easily or curse the author and fail miserably. You will need to know or learn about different topics regarding basic system administration tasks, DNS, SMTP, MySQL and POP3/IMAP. What you will need:

- A computer to run Debian on (e.g. a dedicated server)
- A Debian installation medium (I prefer a Debian netinstall CD)
- An internet connection
- If your internet connection has a dynamic IP address then you will need an SMTP relay (most decent ISPs offer that) because you usually can't send out emails directly to other mail servers.
- An internet domain and control over its MX record
- Control over your firewall (if you have one) to open the needed ports for SMTP, IMAP and POP3.
- 1-8 hours of time depending on your knowledge.

About this tutorial

Many years ago I wanted to turn my Debian server into a mail server with virus scanning, spam detection, email forwarding, POP3 and IMAP access and a webmail service. All the components were available but it took a while until they worked properly together. So I summed up my desk full of scrawly notes into a tutorial that has become pretty famous. According to my web server statistics it's still the most frequently read article at workaround.org.

This document is not a simple copy-and-paste tutorial where you just copy the commands from the web site and run it on your server. Instead it will make you understand the different components that you are setting up. In the end you will be skilled enough to debug problems yourself. If you feel you need help with your setup then try the hints in the Troubleshooting section or ask on the mailing list. Unlike many other Postfix tutorials on the internet this is already the sixth edition. Writing this tutorial took a lot of work so these are not just quick draft notes thrown together but a consistent document guiding you.

The whole tutorial is split into several chapters. Please use the links on the right side or below to navigate through the tutorial. If you prefer all content on a single page (e.g. for printing the tutorial) then use the "*printer-friendly*" link below. You are also invited to comment on the pages - just click on the "*Add new comment*" link at the bottom.

If you like the tutorial then a tiny donation is appreciated to run the servers and pay for the internet connection.

Big picture

- Add new comment
- 126269 reads

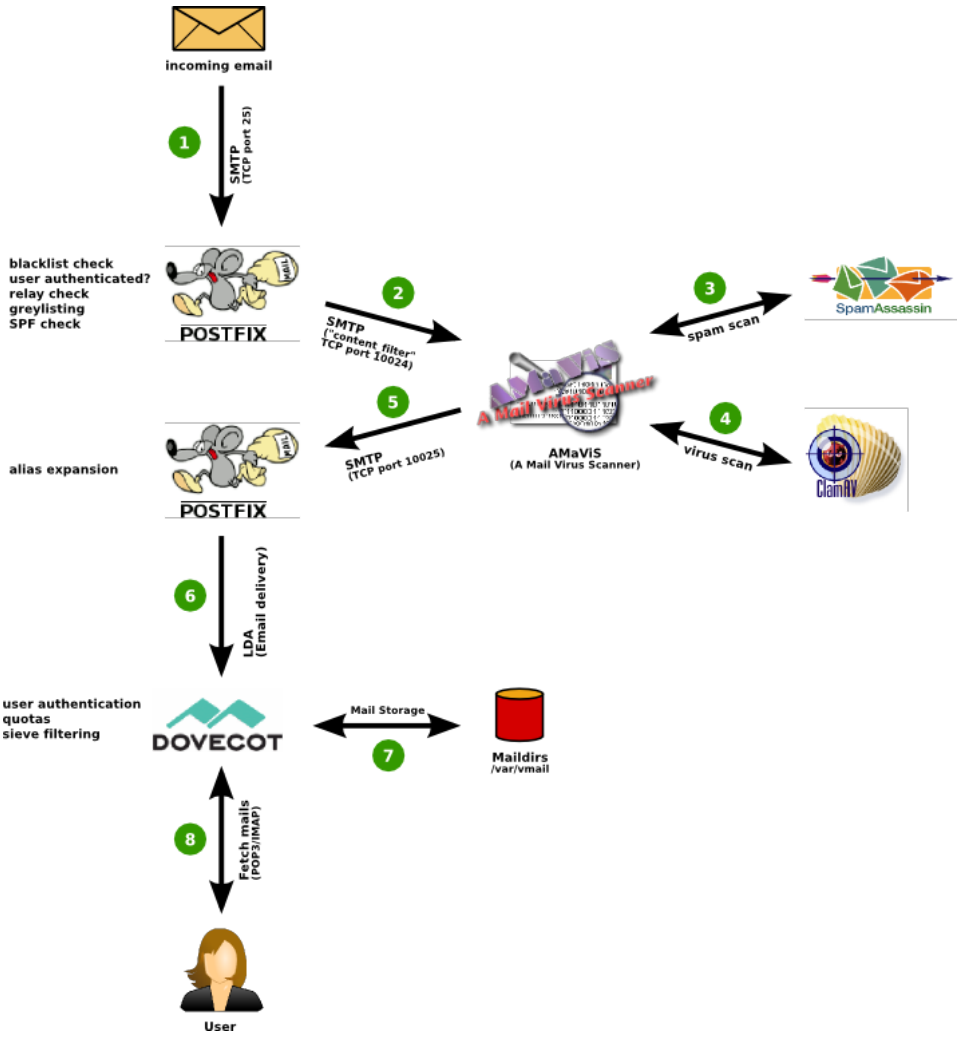
The Software we will use

The configuration described here uses these software components (the versions are Debian Squeeze's default versions):

- Postfix (2.7.1) for receiving incoming emails from the internet and doing basic checks
- Dovecot (1.2.15) to store emails on hard disk and allow users to access their emails using POP3 and IMAP
- Roundcube (0.3.1) as a webmail interface so users can read their emails using a web browser
- MySQL (5.1.49) as the database backend storing information about domains, user accounts and email forwardings
- AMaViS (2.6.4) for content scanning incoming emails using ClamAV and SpamAssassin
- Clam Antivirus (0.97) for virus checking
- SpamAssassin (3.3.1) for spam checking

The wonderous ways of an Email

Before going into the details let's see the big picture:



1. An email is sent to your server via the SMTP protocol on TCP port 25. Postfix accepts the connection, reads the email and does some basic checks. Is the sender blacklisted on a realtime blacklist? Is the email from an authenticated user so we bypass relay checks? Or is the recipient of the email a valid user on our system? If we don't trust the remote system yet we apply greylisting. At this stage Postfix can reject the email or accept it.
2. Postfix forwards the email via the SMTP protocol on the TCP port 10024 to AMaViS for content checking. Notice that at this stage the email can't be rejected any more. So AMaVis can either accept it or throw it away. Commonly AMaViS is configured to add a certain email header so the user can see that AMaViS thinks it is spam.
3. AMaViS lets SpamAssassin check the email for spam. SpamAssassin will be taught which emails are spam to increase its detection accuracy.
4. AMaViS also runs the email through ClamAV to see if it contains any viruses.
5. After these checks AMaViS returns the email to the Postfix process but on TCP port 10025. Postfix is configured to trust emails sent to this port so further content checks are skipped.
6. Postfix forwards the email to Dovecot. Dovecot can optionally apply per-user filters so that emails can be stored in certain email folders automatically if desired.
7. Dovecot writes the email to the hard disk in maildir format.
8. The user's email client can now fetch the new emails from Dovecot using the POP3 or IMAP protocol.

Migrating from the Lenny tutorial

- Add new comment
- 46102 reads

The ISPmail tutorial is maintained since 2002. You may have followed former versions of the tutorial and now want to know how to upgrade your mail server to Squeeze properly. It is hard to provide exact instructions on what steps to go through. Here are some general hints:

Upgrade the operating system

First you should read the Debian Squeeze release notes before attempting to upgrade your system from Lenny to Squeeze.

Changing the Dovecot configuration

Squeeze uses a newer version of the Dovecot software. In the new version the "cmusieve" plugin has been renamed to "sieve" and needs to be renamed in the /etc/dovecot/dovecot.conf. Please consult /usr/share/doc/dovecot-common/README.Debian after upgrading Dovecot.

Roundcube instead of Squirrelmail (optional)

You can continue to use Squirrelmail as a webmail user interface. RoundCube is just a more modern interface so if you prefer something more fancy then you may want to read the page on using RoundCube.

Virtual Domains

- Add new comment
- 67894 reads

Postfix distinguishes between three kinds of domains. This is a very important concept that you need to understand. Probably half of the support request of desperate readers is caused by misunderstandings here. This page is just for learning - there is nothing to be done on your server yet. See also the documentation on virtual domains on the Postfix website.

Local domains

Postfix is the software component that speaks SMTP and sends and receives emails from the internet. Typically Postfix knows about *local domains* and *local users*. A local user is just a normal system user - one that is listed in the /etc/passwd file. This means that *all* system users will get emails for *any* local domain. The "mydestination" configuration setting lists all local domains. Example:

```
mydestination = example.org, example.com, example.net
```

Let's say you created a system user "johndoe" (e.g. using the "adduser" command). This simple setup will make Postfix receive emails for

- johndoe@example.org
- johndoe@example.com
- johndoe@example.net

You can't make johndoe's account just work in one domain. So this is not feasible for different users in different domains. Neither will it work well with many users as you had to create system accounts for each of them. Is it still a good idea to set up at least one local domain in case of configuration or operation problems with other types of domains. If you don't feel creative then "mydestination = localhost" is a safe choice. Postfix automatically receives emails for these users and saves them under /var/mail/USERNAME.

Virtual mailbox domains

This type of domains is the most important type in this tutorial. A *virtual mailbox domain* is also a domain used to receive email. But you do not use system users (/etc/passwd) to specify valid email addresses in that domain. Instead you explicitly tell Postfix which addresses are valid in a domain. A simple way to configure such domains and users is using text files. Consider the following mapping of recipient email addresses to mailboxes on the disk:

Virtual user	Virtual mailbox location on disk
john@example.org	/var/mail/example.org/john/Maildir
jack@example.org	/var/mail/example.org/jack/Maildir
jack@example.com	/var/mail/example.com/jack/Maildir

You have two domains: foo.org and bar.org. So first you will have to tell Postfix about these domains. This is done by setting

```
virtual_mailbox_domains = example.org example.com
```

in your Postfix configuration. Next you need to tell Postfix which email addresses you are ready to receive email for and where to store the received emails on disk. The respective text file could be stored in /etc/postfix/virtual_mailbox_users and would look like this:

```
john@example.org  /var/vmail/example.org/john/Maildir
jack@example.org  /var/vmail/example.org/jack/Maildir
jack@example.com  /var/vmail/example.com/jack/Maildir
```

As you can see the valid email addresses are specified in the left column. And the place on disk where the emails for each recipient address are stored is specified in the right column. In most Postfix literature you may also find the acronym LHS for "left hand side" - this means the left column. Equally the RHS is the "right hand side" - the right column. Such a table with two columns is also called a *mapping* or *hash table.*

In the above example I have just hardcoded the virtual domains in the Postfix configuration file ("virtual_mailbox_domains = example.org example.com"). Obviously with many domains this is not feasable any more. So you can also use a mapping file to configure the domains. Let's assume you saved it to /etc/postfix/virtual_mailbox_domains and it looked like this:

```
example.org  OK
example.com  OK
```

You may wonder why we can' t just list the domains one-per-line in this file. The reason is that a mapping file always has two columns. In such a "one-dimensional" mapping (list of domains) Postfix does not care about your second column. It does not even have to be "OK" - it can be any string.

If you decided that such text files are okay for you then you will still have to compile these files by running the "postmap" command on them. Example:

```
postmap /etc/postfix/virtual_mailbox_domains
postmap /etc/postfix/virtual_mailbox_users
```

postmap will create create additional files based on the above file names but with a ".db" suffix. Postfix will not do that automatically - this is a common caveat. And it will only obey the *.db files. So do not forget to run postmap after you changed a mapping file. To make these mappings known to Postfix you would add these lines to your main.cf configuration file:

```
virtual_mailbox_domains = hash:/etc/postfix/virtual_mailbox_domains
virtual_mailbox_maps = hash:/etc/postfix/virtual_mailbox_users
```

Now you have the tools you need to set up thousands of domains and email accounts in two text files. That's nice. But actually I promised that we will store such data in a MySQL database. Fortunately this is not much harder than using the above text files. Remember: a mapping is simply a way to assign one value to another . So all you have to do is tell Postfix how to access the two columns of a mapping from a database table. This is done using '.cf' configuration files (see also http://www.postfix.org/MYSQL_README.html or run "man 5 mysql_table" in the shell).

Example virtual_mailbox_maps.cf file:

```
# Information on how to connect to your MySQL server
user = someone
password = some_password
hosts = 127.0.0.1

# The database name on the MySQL server
dbname = mailserver

# The SQL query string
query = SELECT mailbox_path FROM virtual_users WHERE email_address='%s'
```

Imagine that you have a database table for virtual users with two columns. The left-hand side is the "email_address" column in that table. And the right-hand side is the "mailbox_path" column in that table. So this SQL query gets the right-hand side (mailbox path) for a given email address (email_address). The "%s" is the placeholder for the left-hand side and is filled by Postfix on every lookup.

Note that a lookup here must only return just one row from the database. Postfix must uniquely know where the mailbox path for a given user is. There are other mappings though where it's allowed to have multiple right-hand side items for one left-hand side item - for example in virtual aliases as shown in the next section.

To use the above configuration file you have to configure it in Postfix's main.cf file:

```
virtual_mailbox_maps = mysql:virtual_mailbox_maps.cf
```

If later you find that this mapping is not doing what you intended then the "postmap -q" command can be used to ask Postfix what the right-hand side value for a given left-side value would be. Say that you are interested in the mailbox_path for the email_address "john@foo.org":

```
postmap -q john@example.org mysql:virtual_mailbox_maps.cf
```

Postfix will then run the above SQL query with your "john@example.org" argument:

```
SELECT mailbox_path FROM virtual_users WHERE email_address='john@example.org'
```

The result should be:

```
/var/mail/example.org/john/Maildir
```

(Note: In this tutorial we will not let Postfix deliver the email directly. Rather it hands over incoming email to Dovecot. So we won't use the above virtual_mailbox_maps in this tutorial. It is still important to understand how Postfix deals with virtual users.)

Virtual alias domains

Virtual alias domains are used for forwarding email from an email address to one or more other email addresses. Virtual alias domains can't receive email though. They only forward mail somewhere else. The *virtual_alias_maps* mapping contains forwardings (source, destination) of users or domains to other email addresses or entire domains. Incidentally *virtual_alias_maps* are obeyed for any received email. So in most cases you do not really need virtual alias domains as you can declare all domains as *virtual mailbox domains* and use *virtual alias maps* for forwarding purposes. Technically defining a domain as a *virtual alias domain* makes Postfix accept email for that domain but you still need an entry in the *virtual_alias_maps* mapping to tell Postfix where to forward the email.

A note on the *virtual_alias_maps*: they can return multiple right-hand side destinations (to forward to) for one left-hand side source (the original recipient). You can use that to forward an email to several recipients and to control whether you want to keep a copy.

Example 1: forward all email for john@example.org to jeff@example.com

```
john@example.org   jeff@example.com
```

This one is simple. You have the source (john@example.org) and the destination (jeff@example.com) or the forwarding. John will never see the email.

Example 2: forward all email for john@example.org to jeff@example.com but also receive a copy

```
john@example.org   john@example.org
john@example.org   jeff@example.com
```

This is a bit trickier. If Postfix queries this mapping for john@foo.org it will get two results. (Postfix is smart enough not to create a loop but to understand that you want to get a copy of the emaill.) This is the same as one row with the recipients seperated by commas:

```
john@example.org   john@example.org, jeff@example.com
```

Example 3: forward all email for any domain in the example.org domain to joe@example.com

```
@example.org   joe@example.com
```

This is called a *catch-all* alias. It will accept email for any user in the *example.org* domain and forward it to joe@example.com. If jill@example.org

would not be an explicitly defined virtual user then her email would be caught by the catch-all alias and forwarded to joe@baz.org.

Beware: Catch-all aliases catch spam. Lots of spam. They may look comfortable because they forward all email to one person without the need for creating aliases. But spammers often try to guess email addresses at a known domain. And with a catch-all alias you will receive spam for any of those guessed email addresses. Try to avoid them and rather define the existing email addresses. Even if it seems to be more work.

Installing Debian

- Add new comment
- 43554 reads

If you already have a server with Debian on it and cannot or don't want to reinstall then feel free to skip this page. This may be the case if you are using a (virtual) root server from a hosting company or you do not have console access.

Dimensioning the server

So you can choose the actual server yourself? Great. This gives you the choice on the hardware, the disk partitioning scheme and choice of file systems. As a general guideline a decent mail server should have at least 2 GB of RAM and lots of disk space living on a hardware-based RAID controller. Users will collect a lot of mails with a lot of useless attachments. And don't expect them to clean up their inbox - especially when offering IMAP. Project 5 GB of mailbox space per user. Regarding the CPU load there is mainly spam and virus scanning that needs raw computing power. A common server will handle dozens of emails per second without spam checking. If you don't need spam checking or just get smaller amounts of email then even an old desktop PC will do well.

Basic installation and Partitioning

It's rumoured that even a chicken can install Debian if you just put enough grains on the enter key. And that depicts pretty much how easy it is to install Debian. Insert your boot medium and start the installation. Even if your native language is not english I still suggest you choose english as a system language. If later you have trouble with the server you will more likely find help when google'ing for english error messages.

Next I recommend that you take a little time though when it comes to partitioning your disk. When the installation asks how to partition choose "*Manual*". Throughout this tutorial you will store your users' emails in the /var/vmail partition and will hold the most files which also sum up to a lot of used space. The MySQL database will live in /var/lib/mysql. Log files will live in /var/log.

So my recommendation is:

- / (10 GB, ext4)
- /boot (500 MB - don't forget to mark it as "*bootable*", ext3)
- /var (5 GB, ext4)
- /var/vmail (the more the better, ext4)
- /tmp (1 GB, ext4, optional but recommended)
- swap (1 GB)

The file system can also be ext3, XFS or ReiserFS if you like. Choosing the right file system is a religious issue. XFS is almost always a safe choice for servers but in some conditions may become very slow. ext3 is okay if you don't want to reformat your /var/vmail partition but the file system check after a reboot can take hours. ReiserFS is also a decent choice but the future of the file system is not entirely clear.

If at all possible choose to use the *logical volume manager* (LVM) which gives you great flexibility about growing or (depending on the file system you chose) shrinking partitions or even making consistent backups using the *snaphot* feature. If you haven't dealt with LVM yet then don't fear it. Even if you have never used LVM you will quickly learn how it works and will never again want to install a server without it. (Unfortunately the article I once wrote about it has gone. But rest assured that I will redo it.)

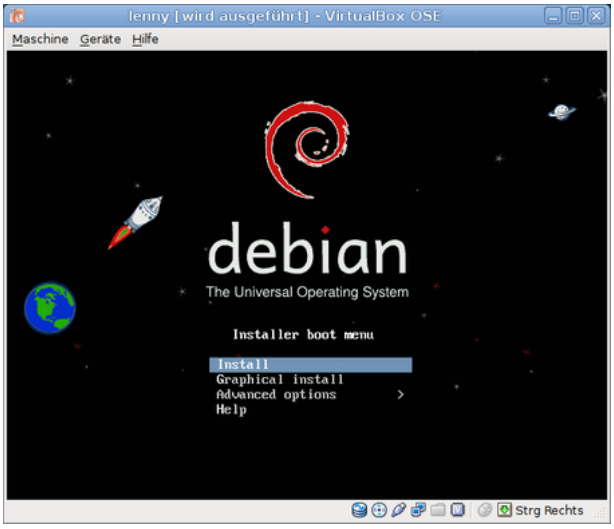
Installation wallpaper

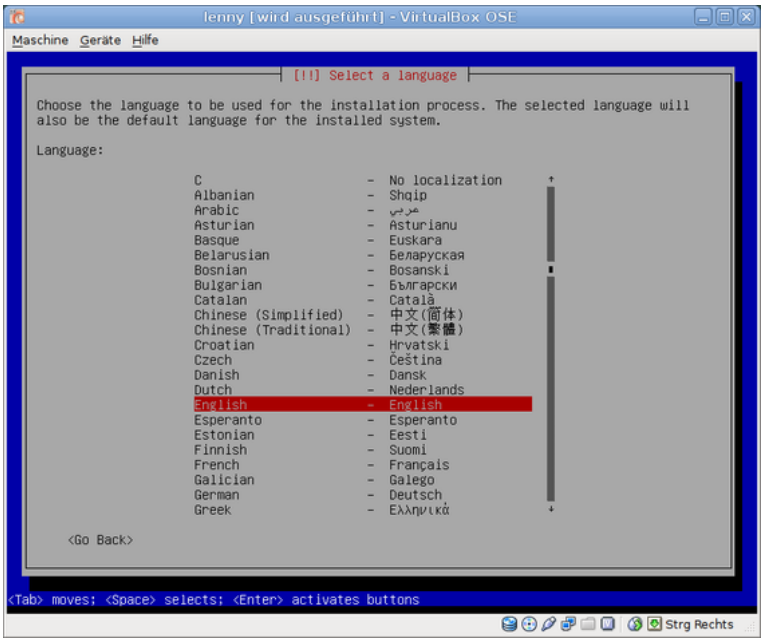
If you are unsure how the installation procedure would look like (or you don't have a chicken handy) then feel free to follow this set of screenshots documenting the installation.

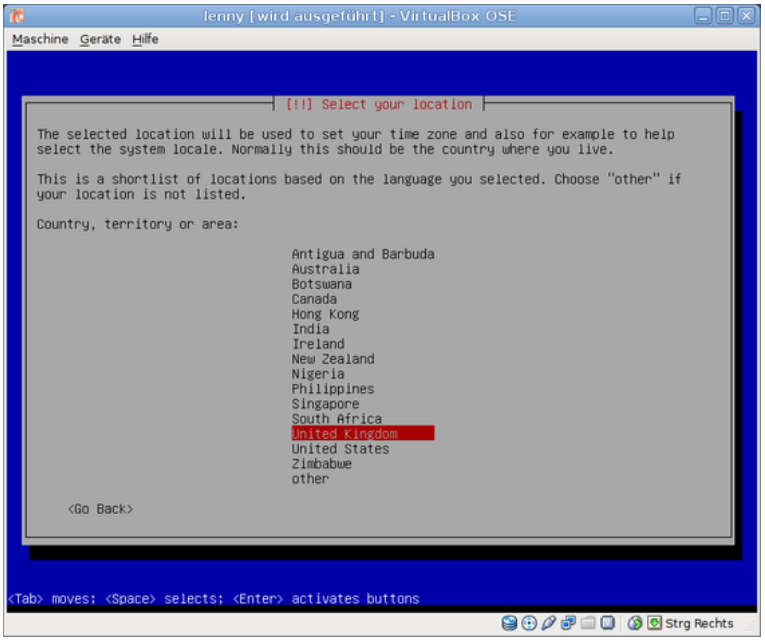
Installation screenshot wallpaper

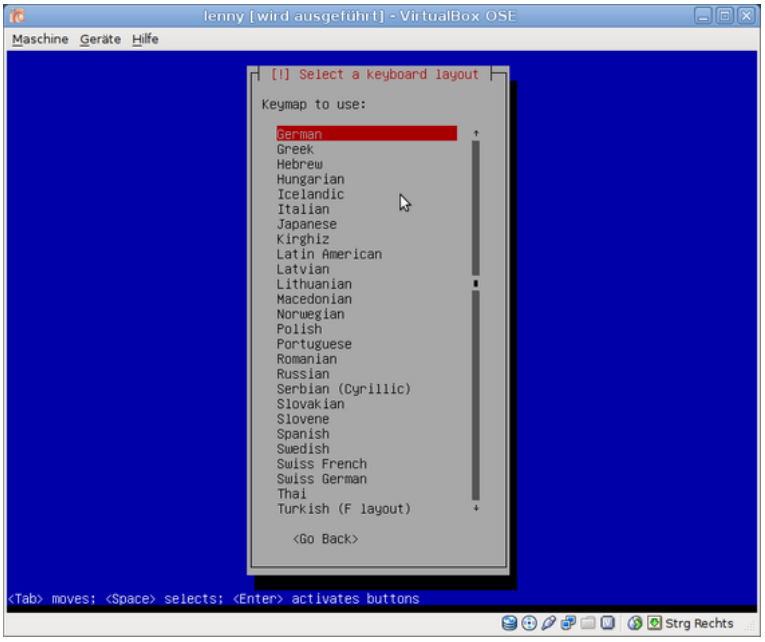
- Add new comment
- 24525 reads

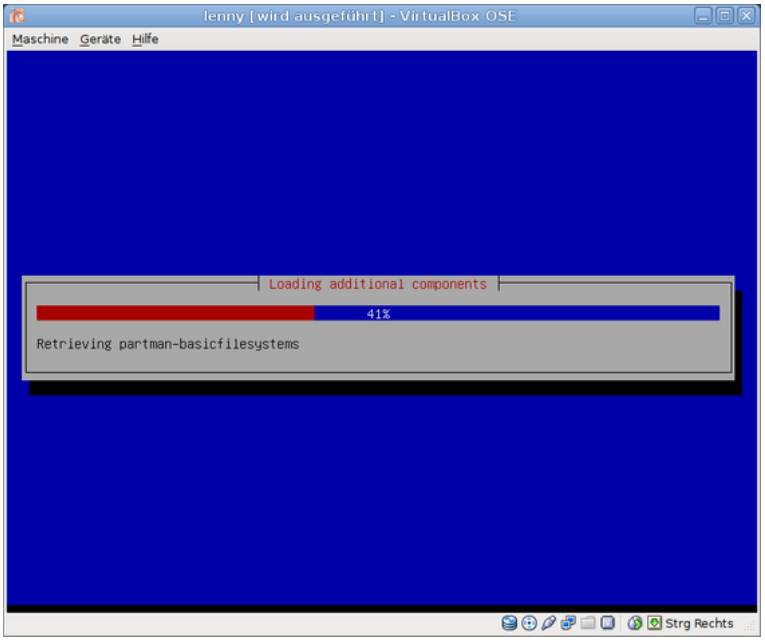
This wallpaper documents a Debian Squeeze installation with partitioning the system using the logical volume manager (LVM).

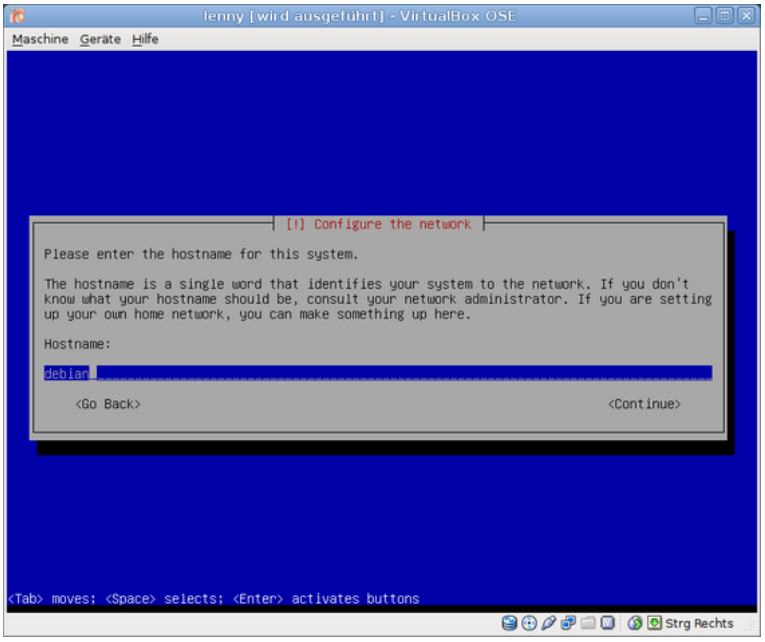

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
debian
The Universal Operating System
Installer boot menu
Install
Graphical install
Advanced options
Help
Strg Rechts

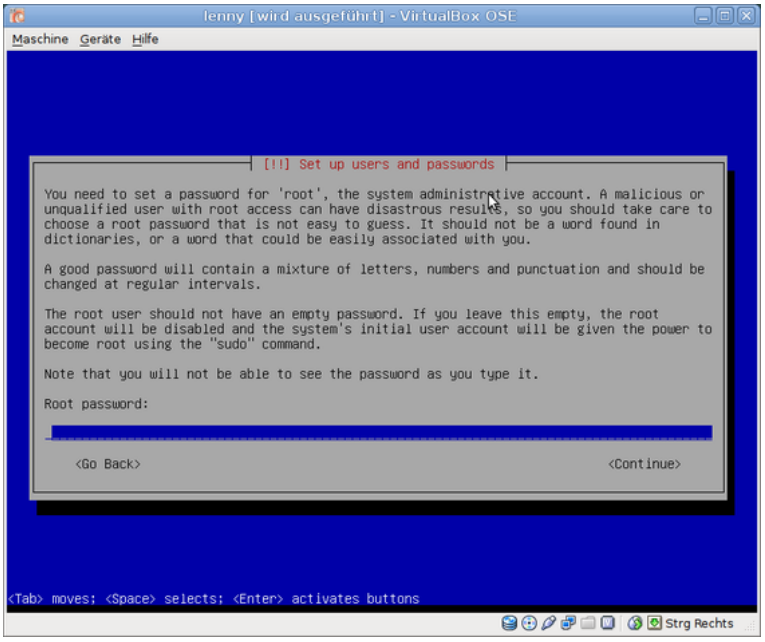

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Select a language
Choose the language to be used for the installation process. The selected language will also be the default language for the installed system.
Language:
C - No localization
Albanian - Shqip
Arabic - عربي
Asturian - Asturianu
Basque - Euskara
Belarusian - Беларуская
Bosnian - Bosanski
Bulgarian - Български
Catalan - Català
Chinese (Simplified) - 中文(简体)
Chinese (Traditional) - 中文(繁體)
Croatian - Hrvatski
Czech - Čeština
Danish - Dansk
Dutch - Nederlands
English - English
Esperanto - Esperanto
Estonian - Eesti
Finnish - Suomi
French - Français
Galician - Galego
German - Deutsch
Greek - Ελληνικά
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

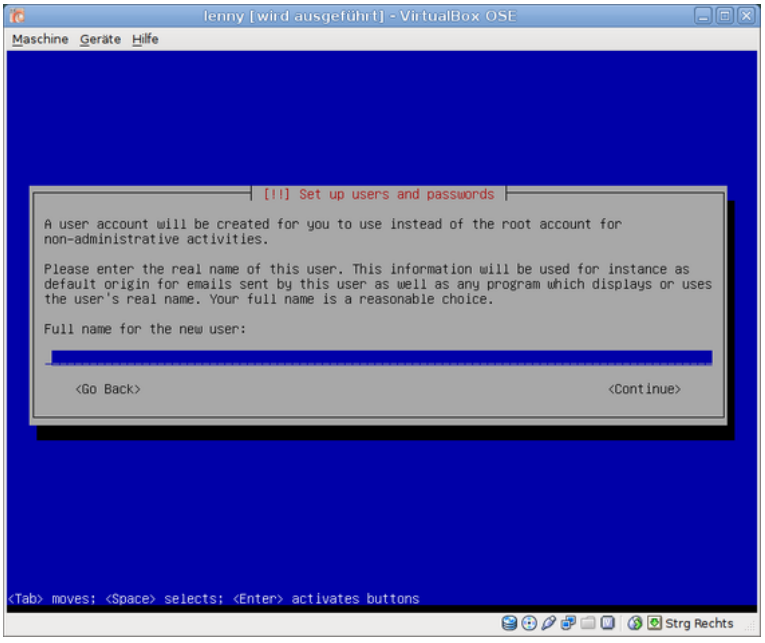

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Select your location
The selected location will be used to set your time zone and also for example to help select the system locale. Normally this should be the country where you live.
This is a shortlist of locations based on the language you selected. Choose "other" if your location is not listed.
Country, territory or area:
Antigua and Barbuda
Australia
Botswana
Canada
Hong Kong
India
Ireland
New Zealand
Nigeria
Philippines
Singapore
South Africa
United Kingdom
United States
Zimbabwe
other
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

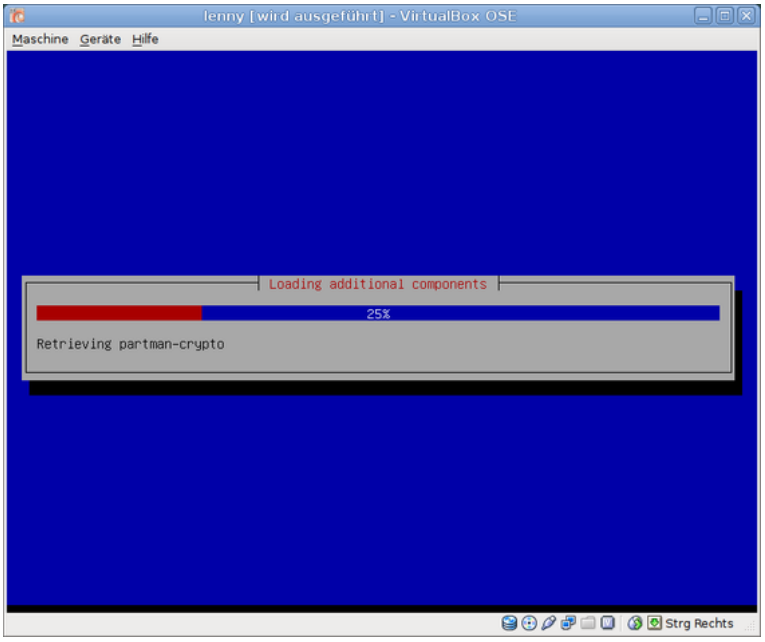

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Select a keyboard layout
Keymap to use:
German
Greek
Hebrew
Hungarian
Icelandic
Italian
Japanese
Kirghiz
Latin American
Latvian
Lithuanian
Macedonian
Norwegian
Polish
Portuguese
Romanian
Russian
Serbian (Cyrillic)
Slovakian
Slovene
Spanish
Swedish
Swiss French
Swiss German
Thai
Turkish (F layout)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

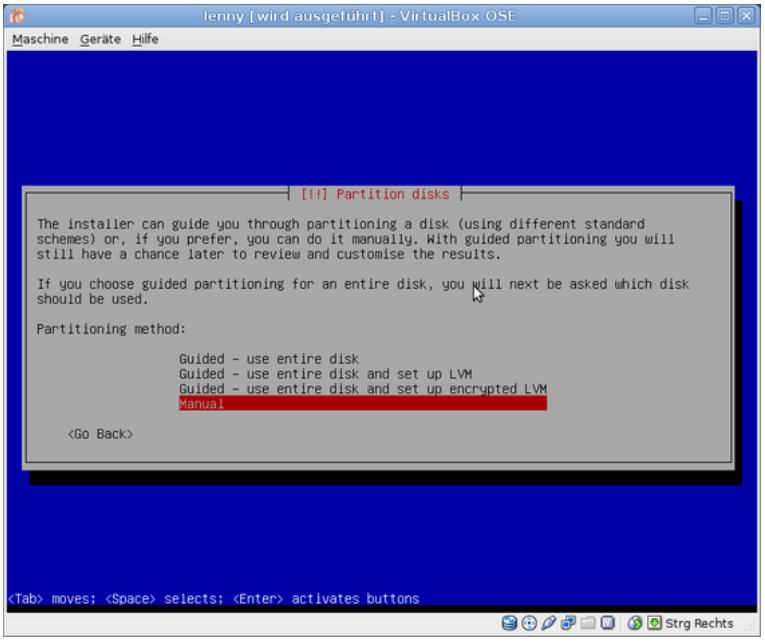

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Loading additional components
41%
Retrieving partman-basicfilesystems
Strg Rechts

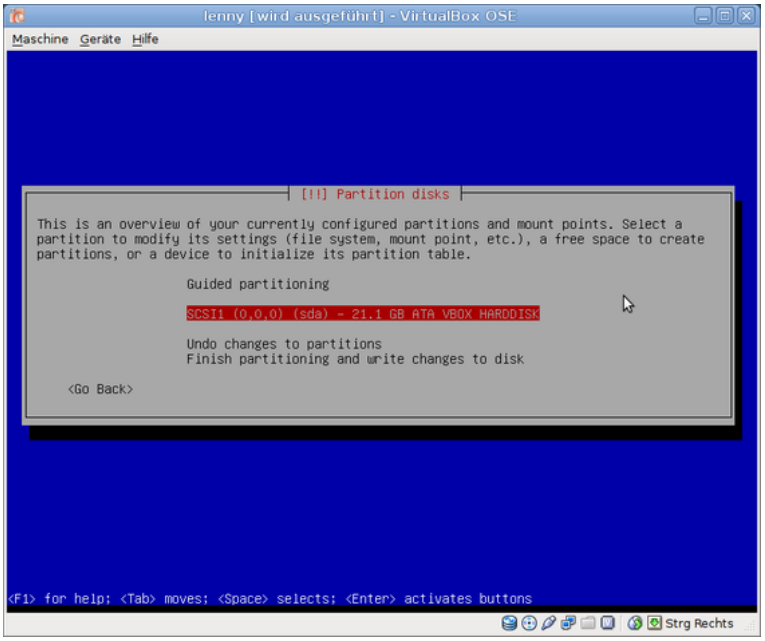

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Configure the network
Please enter the hostname for this system.
The hostname is a single word that identifies your system to the network. If you don't know what your hostname should be, consult your network administrator. If you are setting up your own home network, you can make something up here.
Hostname:
debian
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

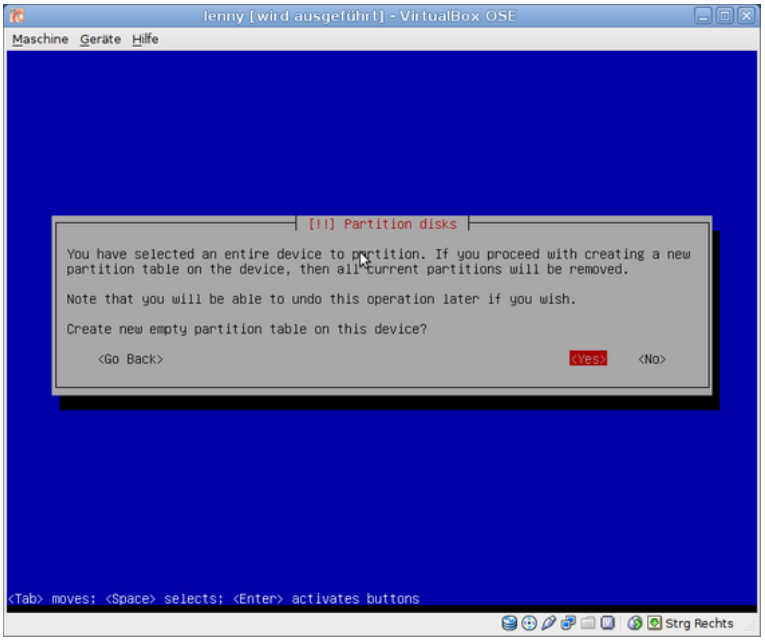

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Set up users and passwords
You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.
A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.
Note that you will not be able to see the password as you type it.
Root password:
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

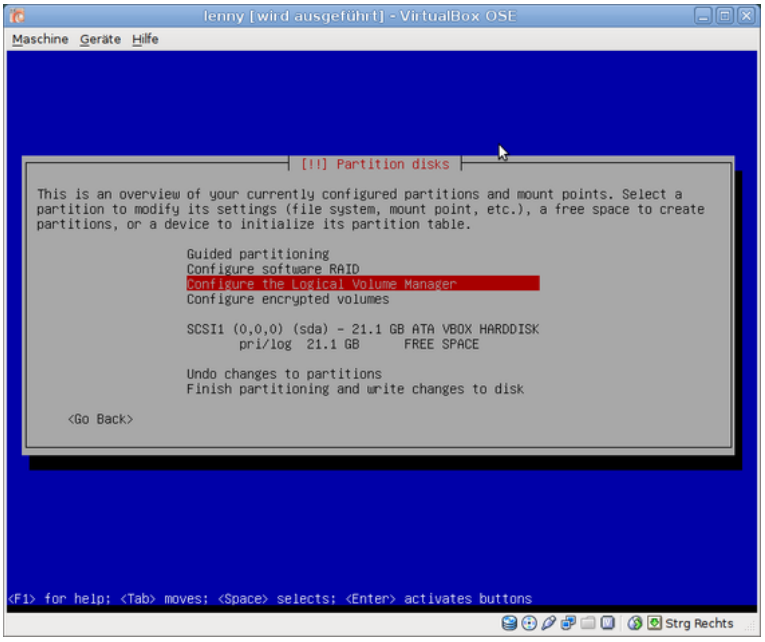

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Set up users and passwords
A user account will be created for you to use instead of the root account for non-administrative activities.
Please enter the real name of this user. This information will be used for instance as default origin for emails sent by this user as well as any program which displays or uses the user's real name. Your full name is a reasonable choice.
Full name for the new user:
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

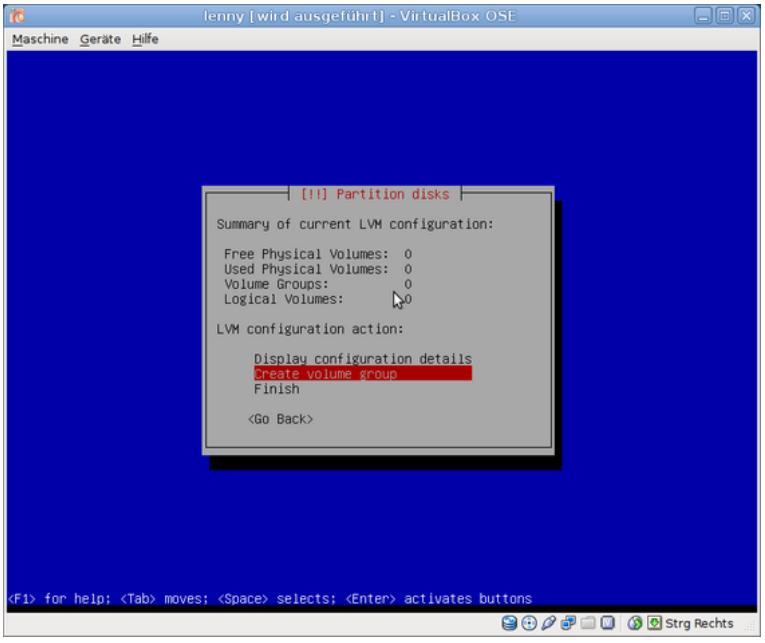

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Loading additional components
25%
Retrieving partman-crypto
Strg Rechts

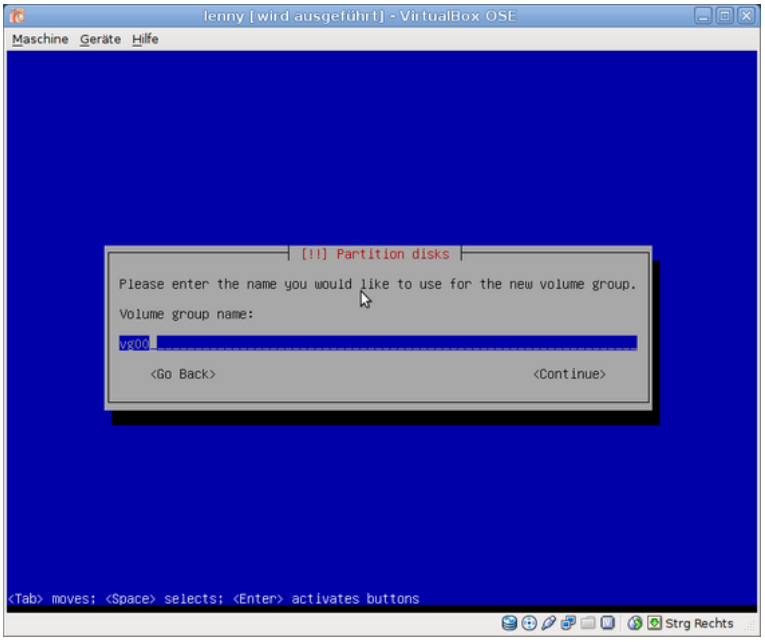

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.
If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.
Partitioning method:
Guided - use entire disk
Guided - use entire disk and set up LVM
Guided - use entire disk and set up encrypted LVM
Manual
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

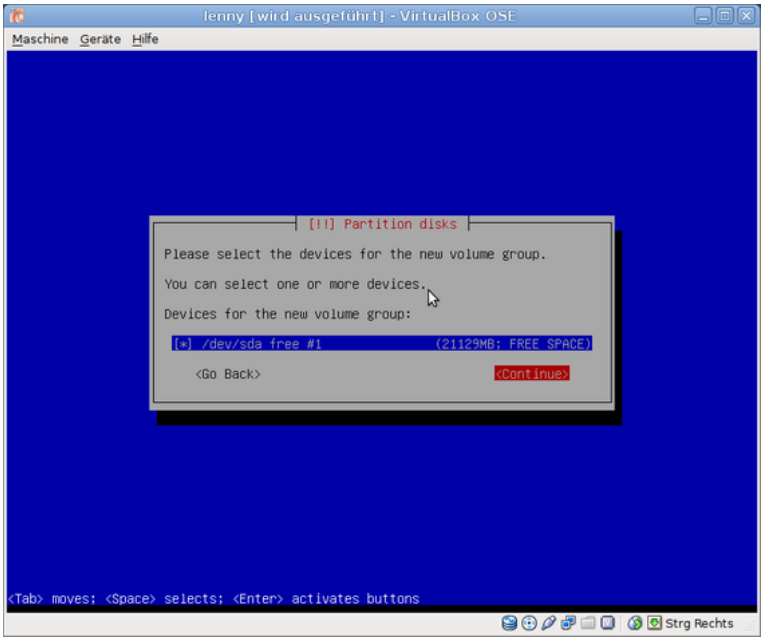

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
SCSI1 (0,0,0) (sda) - 21.1 GB ATA VBOX HARDDISK
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

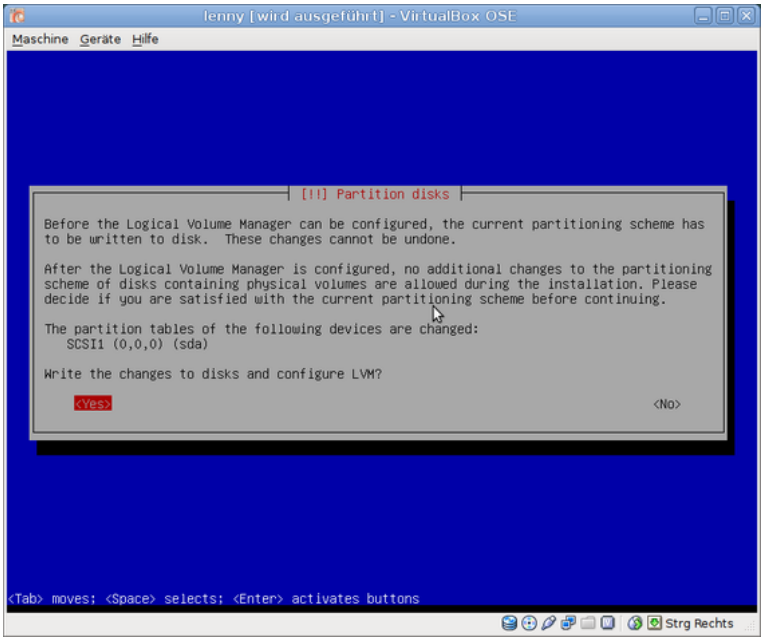

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You have selected an entire device to partition. If you proceed with creating a new partition table on the device, then all current partitions will be removed.
Note that you will be able to undo this operation later if you wish.
Create new empty partition table on this device?
<Go Back>
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

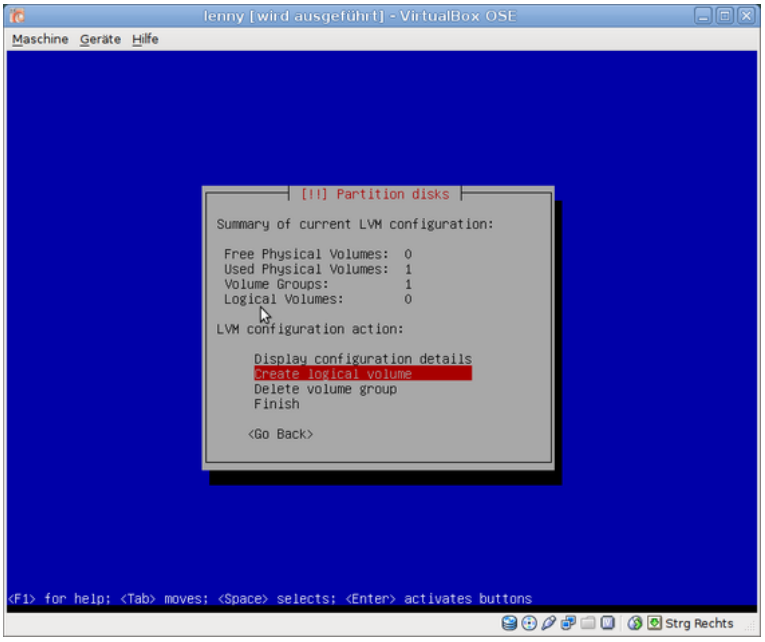

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
Configure software RAID
Configure the Logical Volume Manager
Configure encrypted volumes
SCSI1 (0,0,0) (sda) - 21.1 GB ATA VBOX HARDDISK
pri/log 21.1 GB FREE SPACE
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

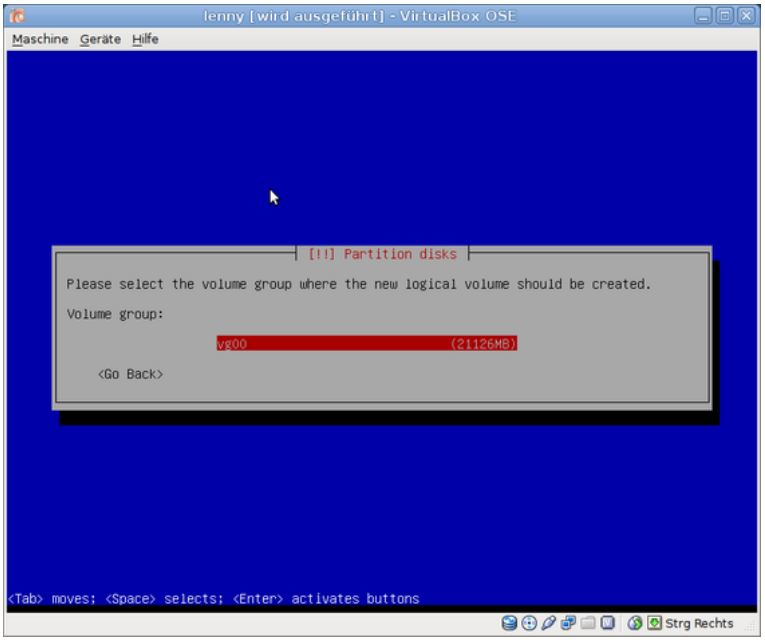

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 0
Volume Groups: 0
Logical Volumes: 0
LVM configuration action:
Display configuration details
Create volume group
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

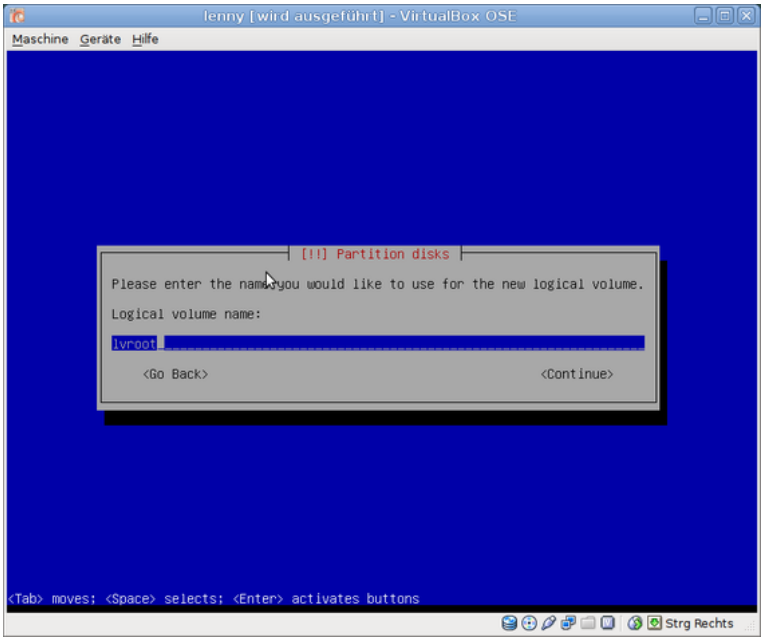

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new volume group.
Volume group name:
vg00
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

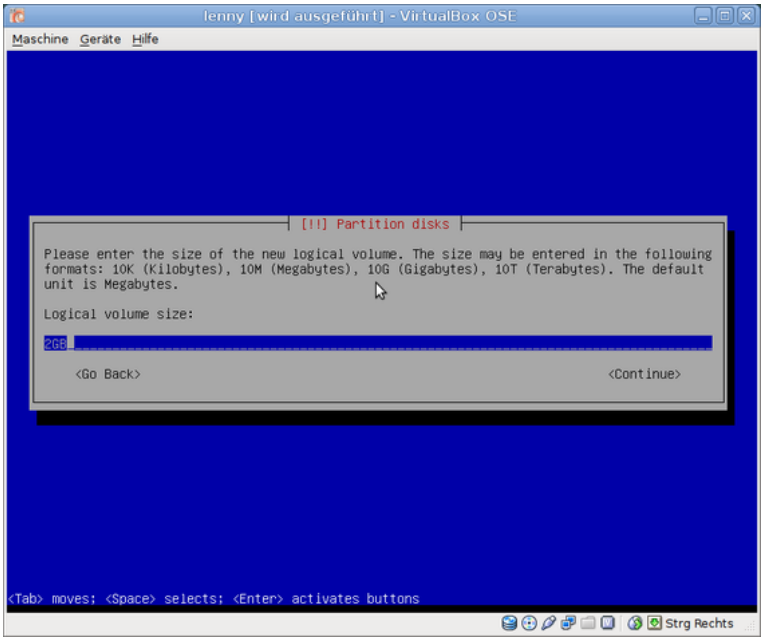

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the devices for the new volume group.
You can select one or more devices.
Devices for the new volume group:
[*] /dev/sda free #1 (21129MB; FREE SPACE)
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

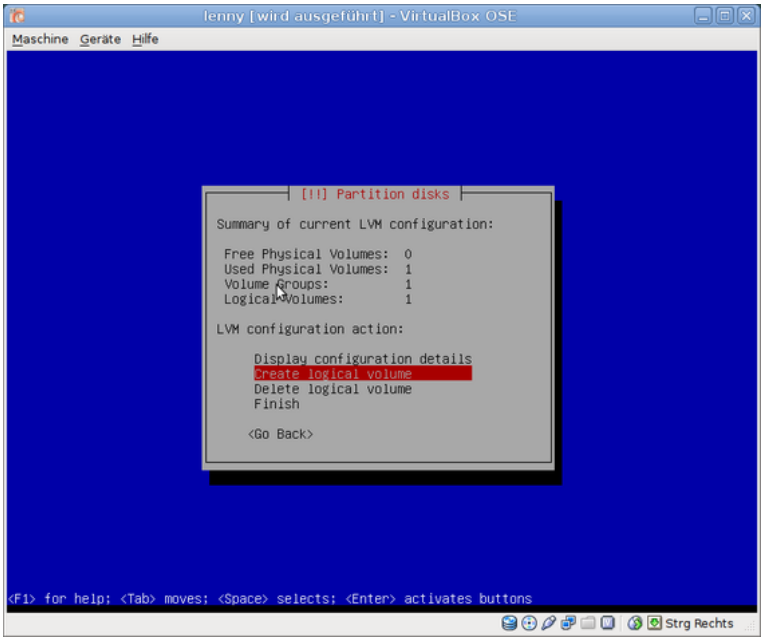

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Before the Logical Volume Manager can be configured, the current partitioning scheme has to be written to disk. These changes cannot be undone.
After the Logical Volume Manager is configured, no additional changes to the partitioning scheme of disks containing physical volumes are allowed during the installation. Please decide if you are satisfied with the current partitioning scheme before continuing.
The partition tables of the following devices are changed:
SCSI1 (0,0,0) (sda)
Write the changes to disks and configure LVM?
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

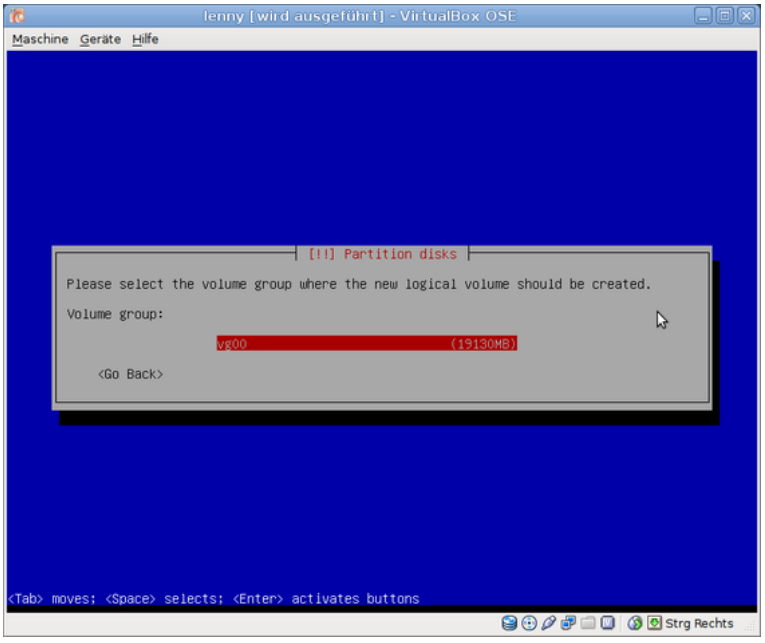

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 0
LVM configuration action:
Display configuration details
Create logical volume
Delete volume group
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

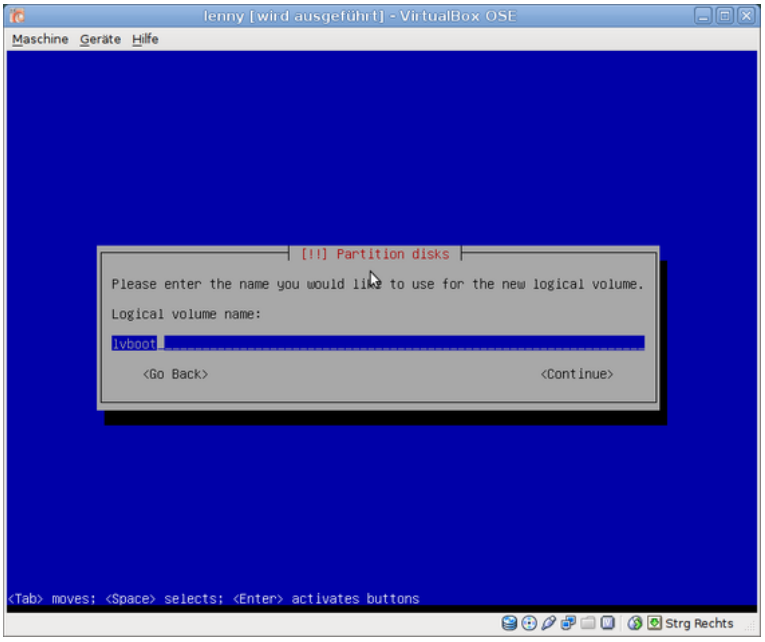

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (21126MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

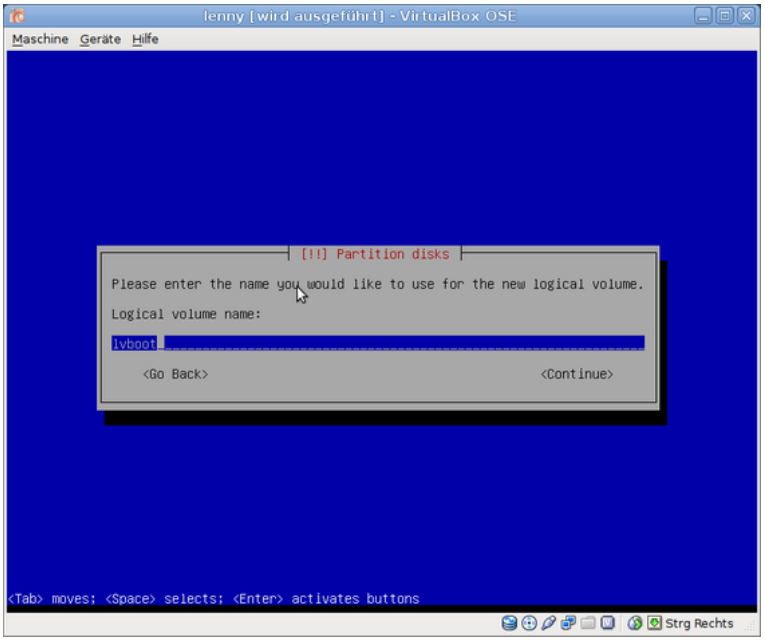

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvroot
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

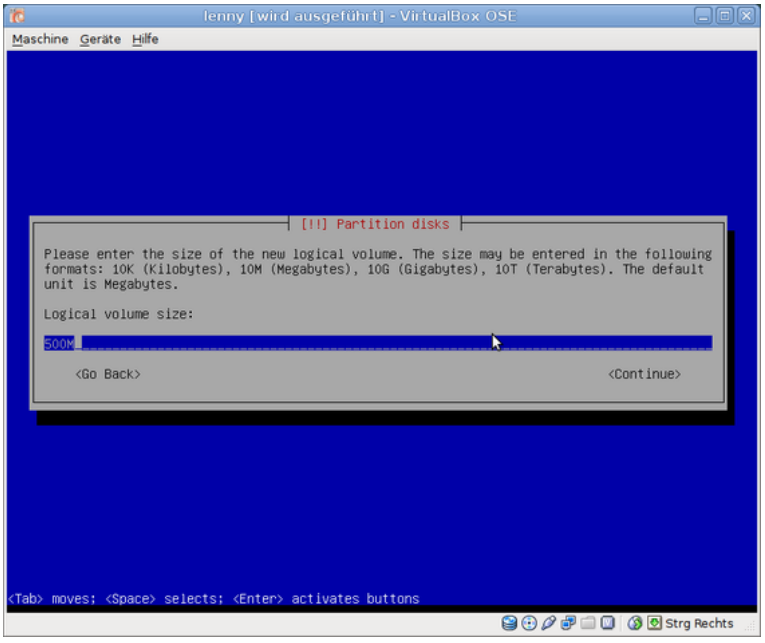

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
2GB
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

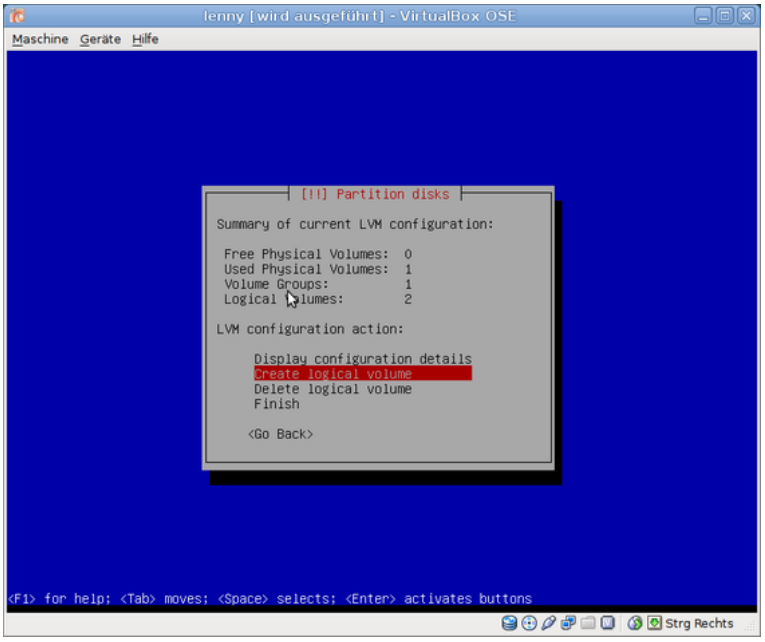

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 1
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

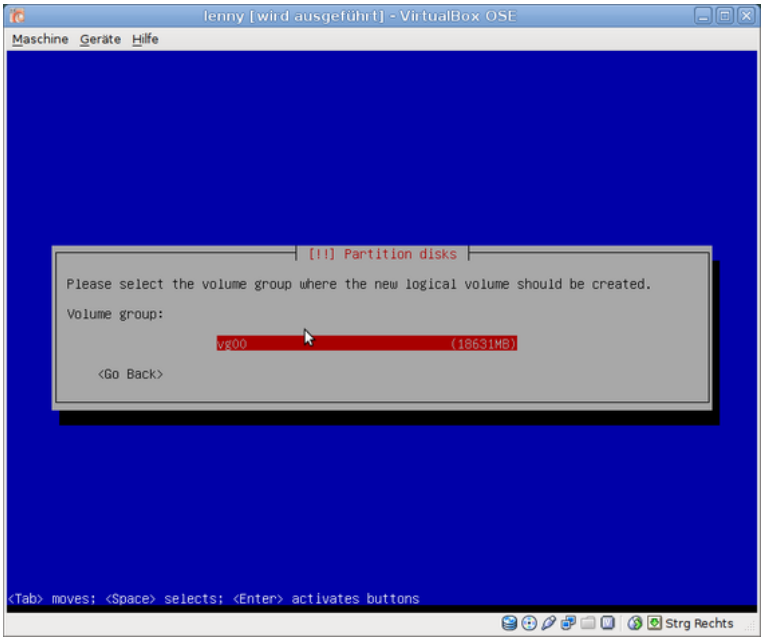

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (19130MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

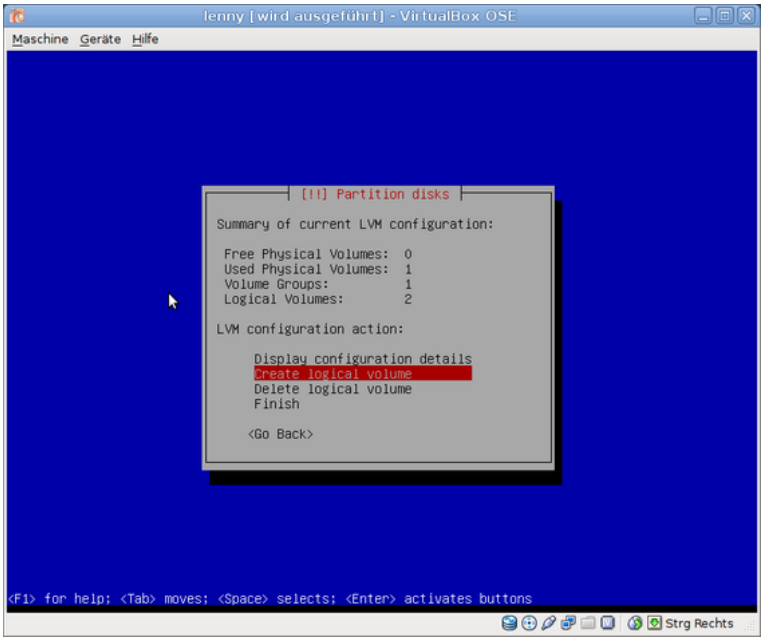

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvboot
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

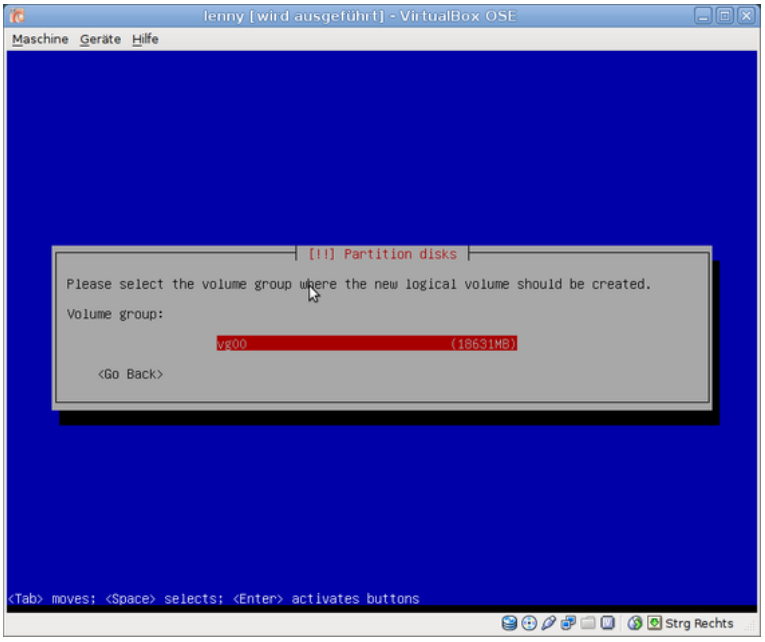

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvboot
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

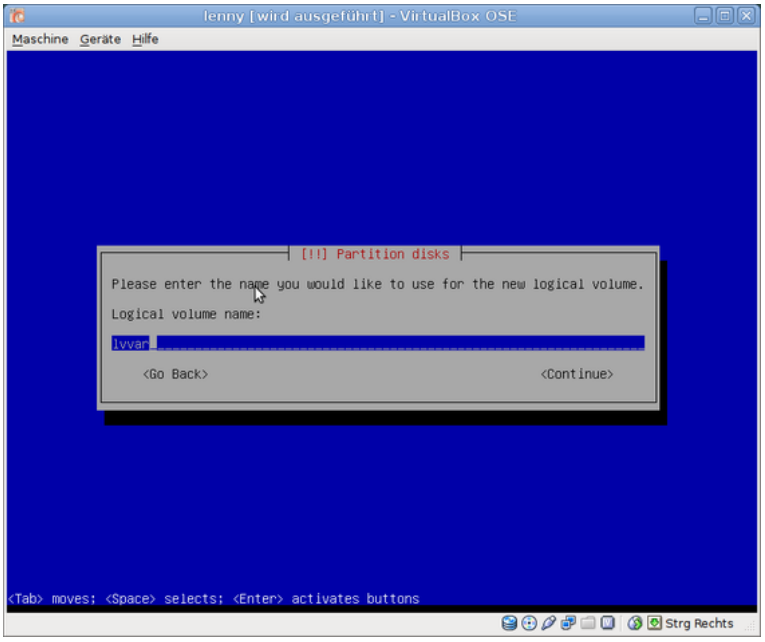

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
500M
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

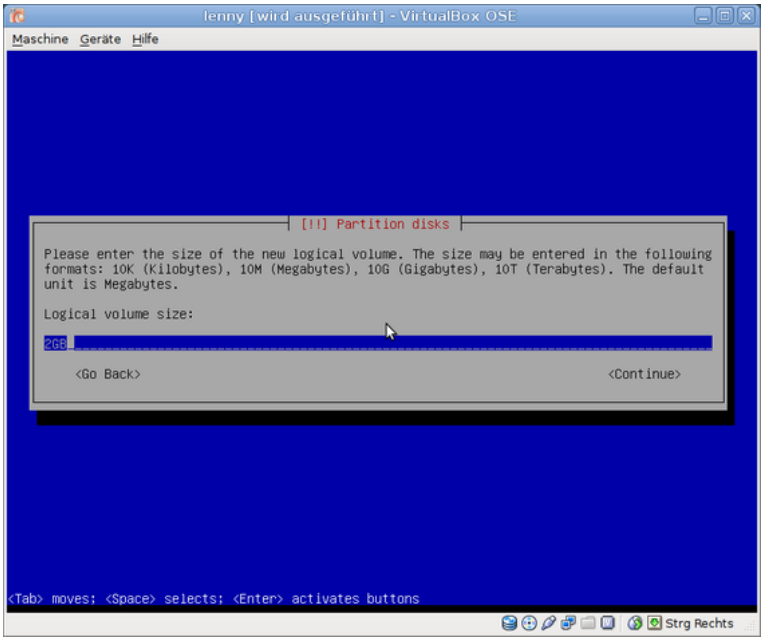

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 2
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

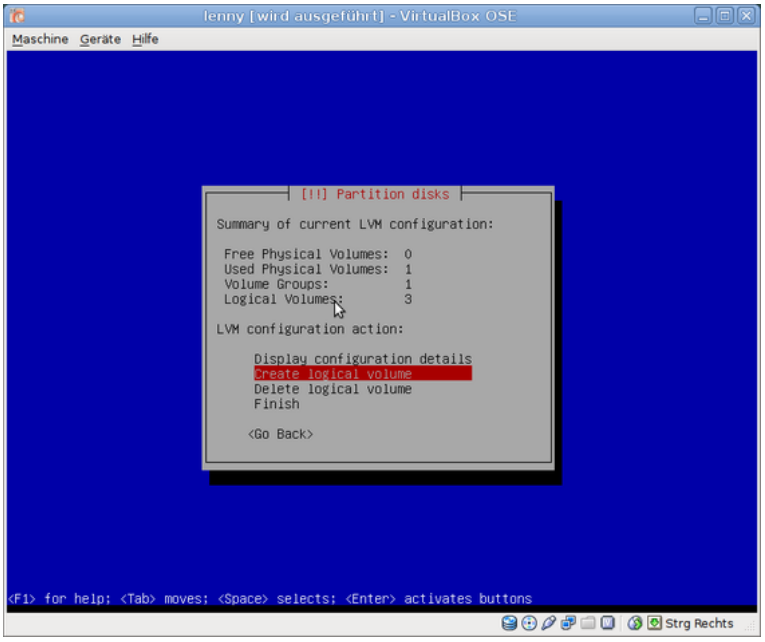

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (18631MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

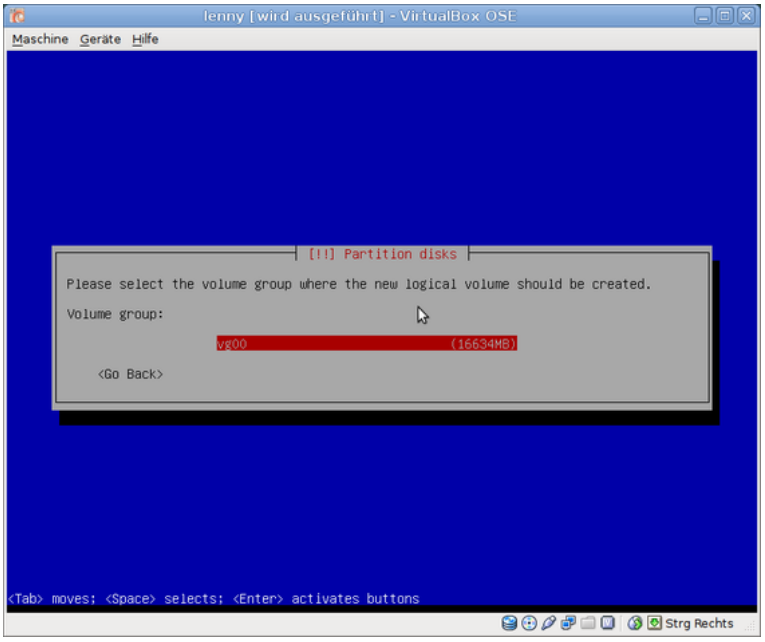

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 2
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

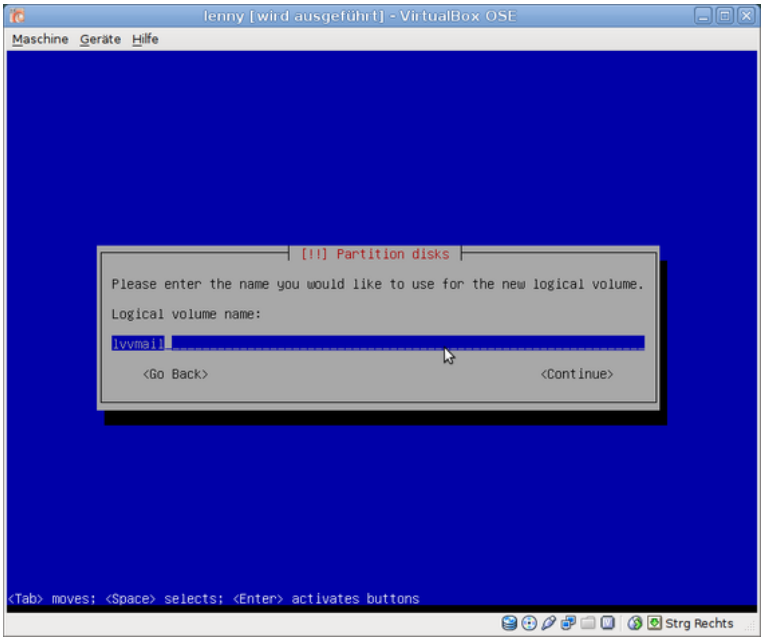

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (18631MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

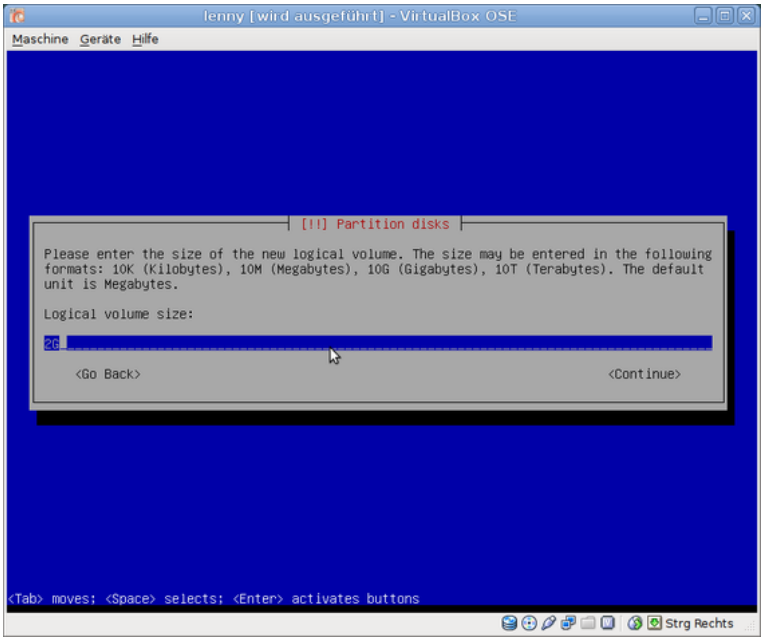

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvvar
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

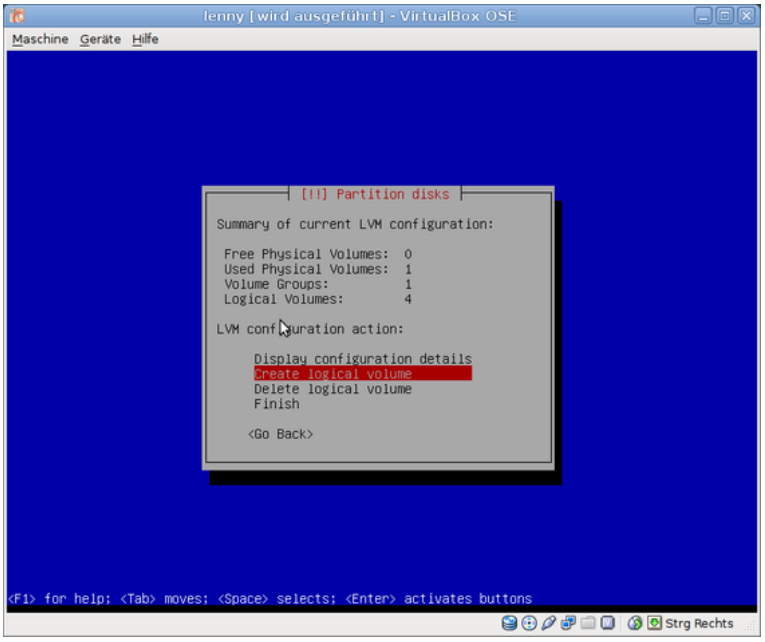

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
2GB
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

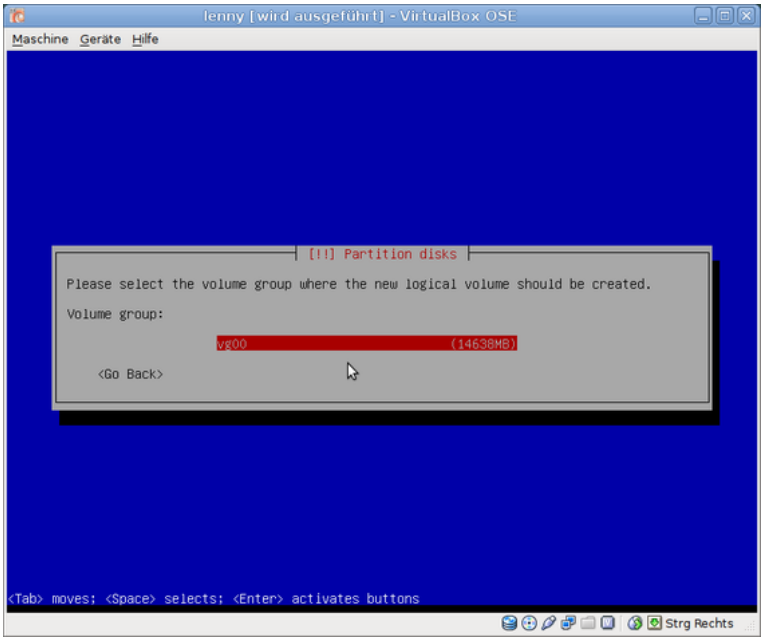

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 3
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

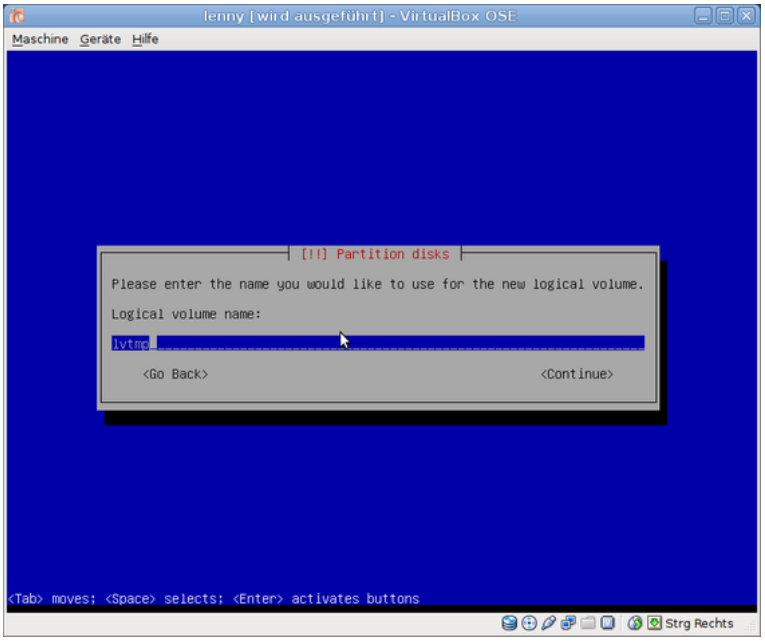

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (16634MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

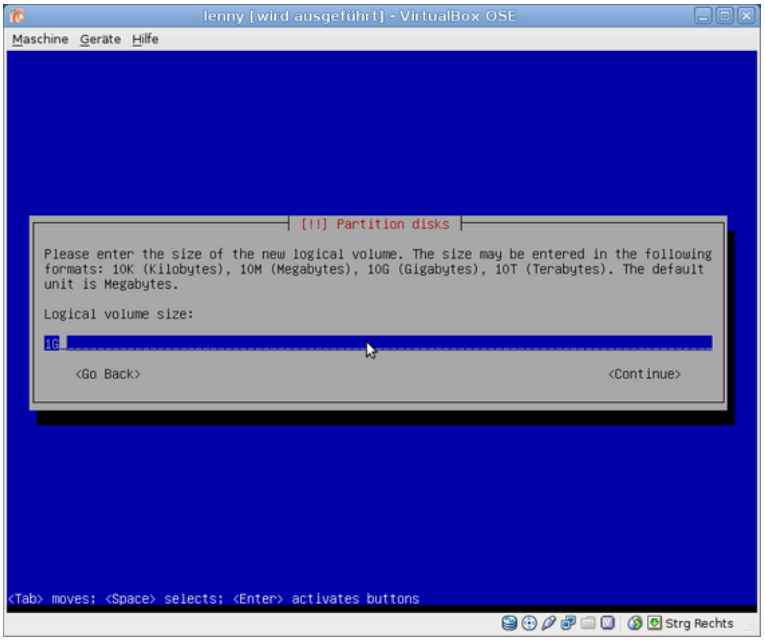

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvvmail
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

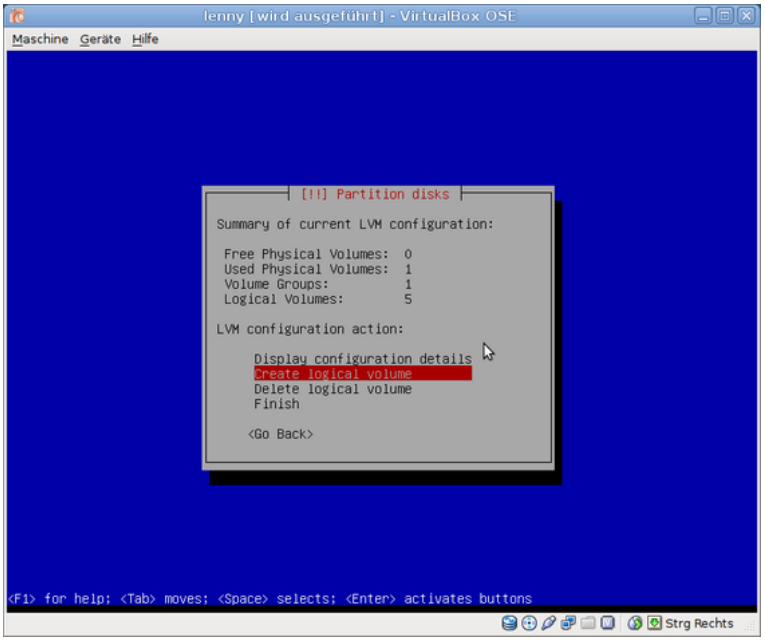

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
2G
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

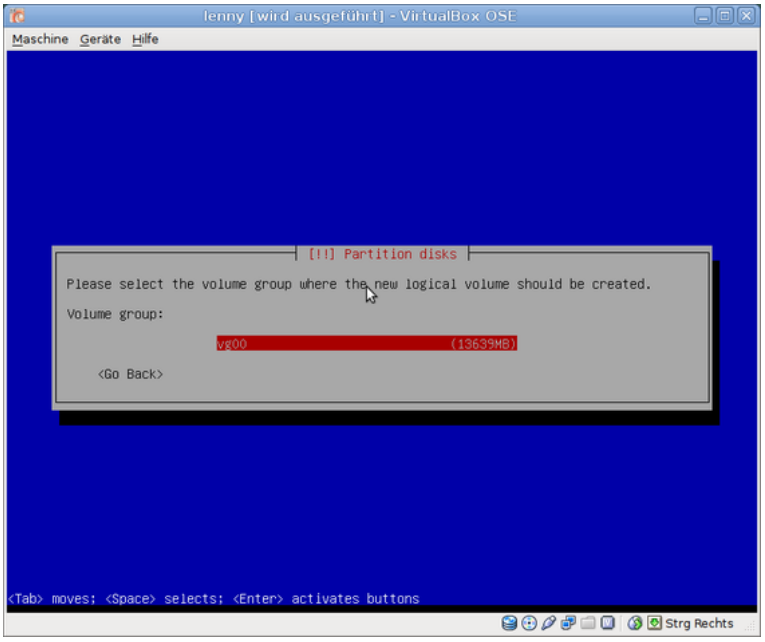

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 4
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

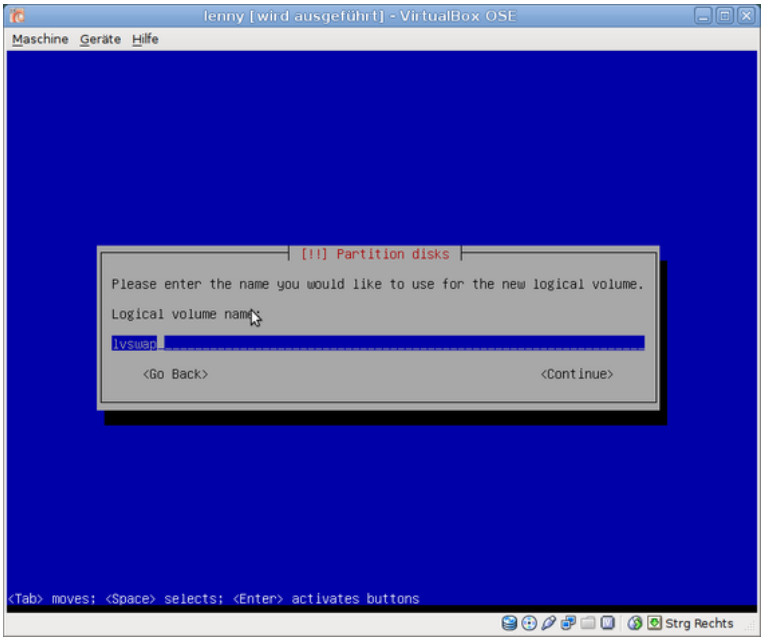

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (14638MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

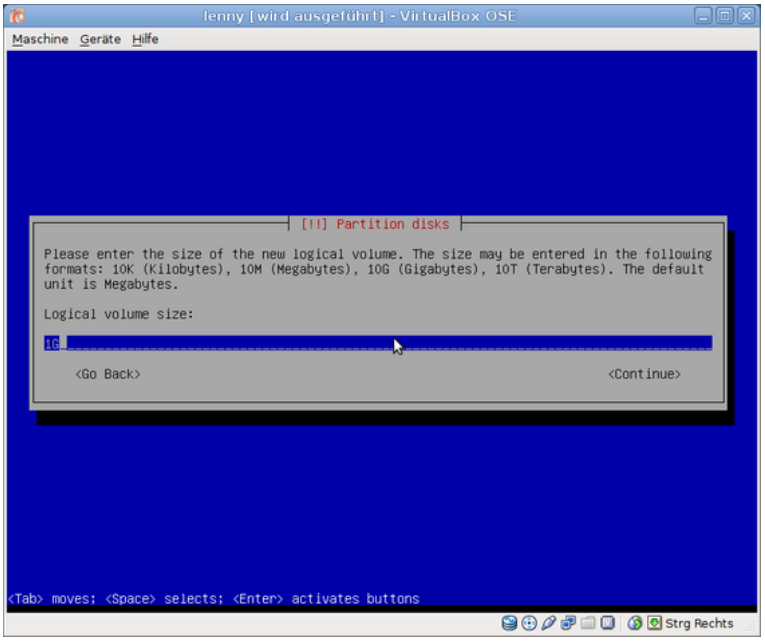

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvtmp
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

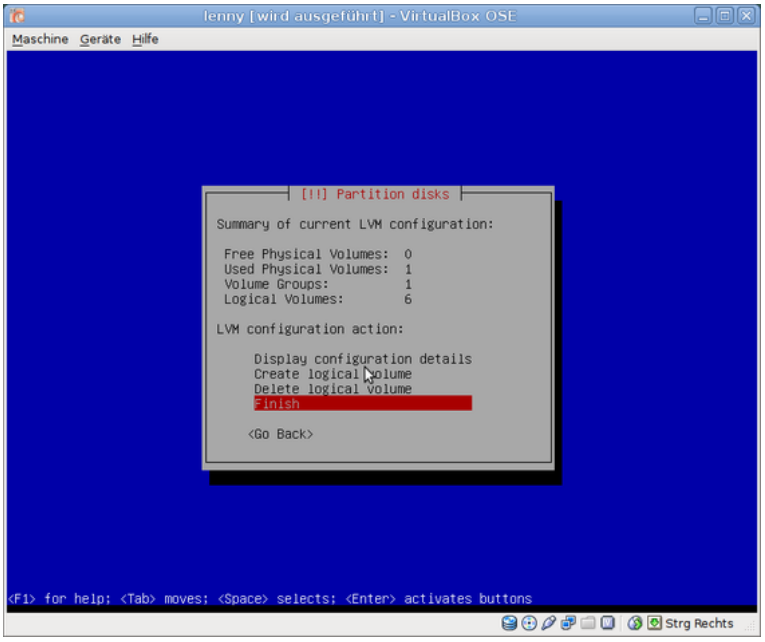

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
1G
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

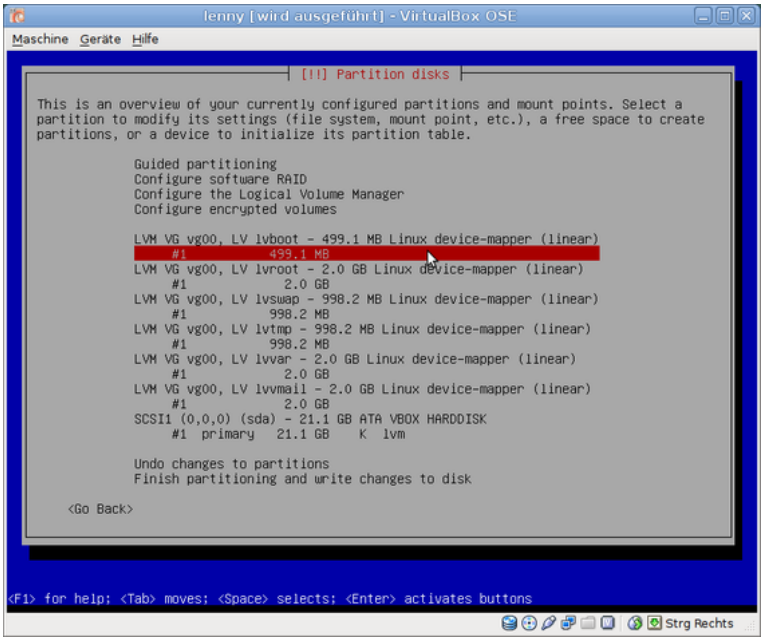

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 5
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

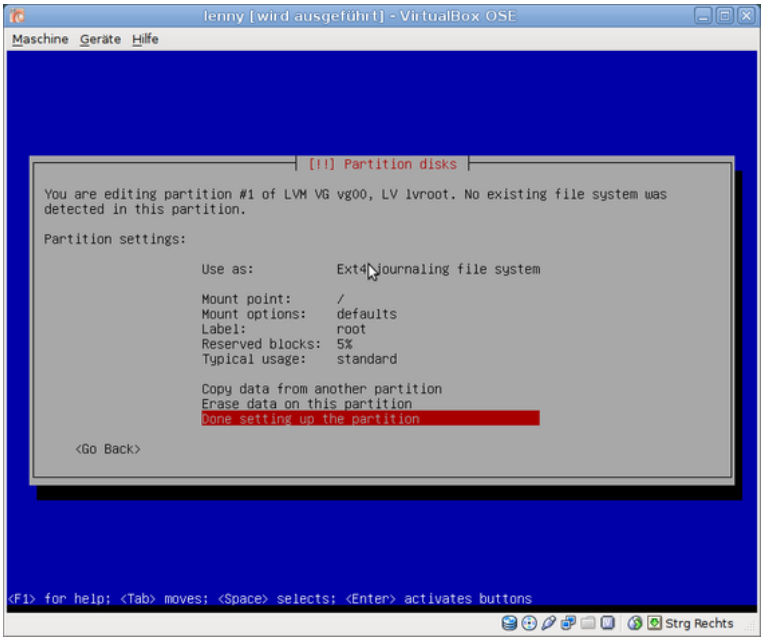

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please select the volume group where the new logical volume should be created.
Volume group:
vg00 (13639MB)
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

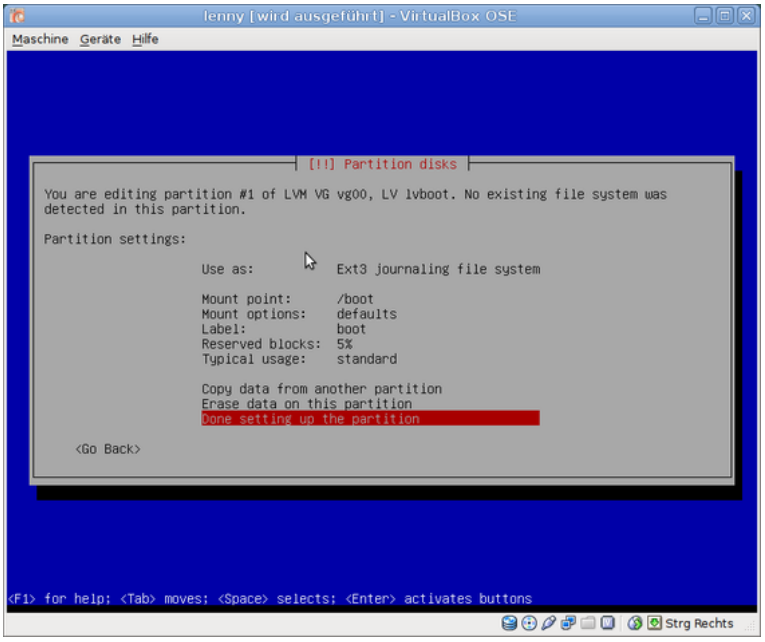

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the name you would like to use for the new logical volume.
Logical volume name:
lvswap
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

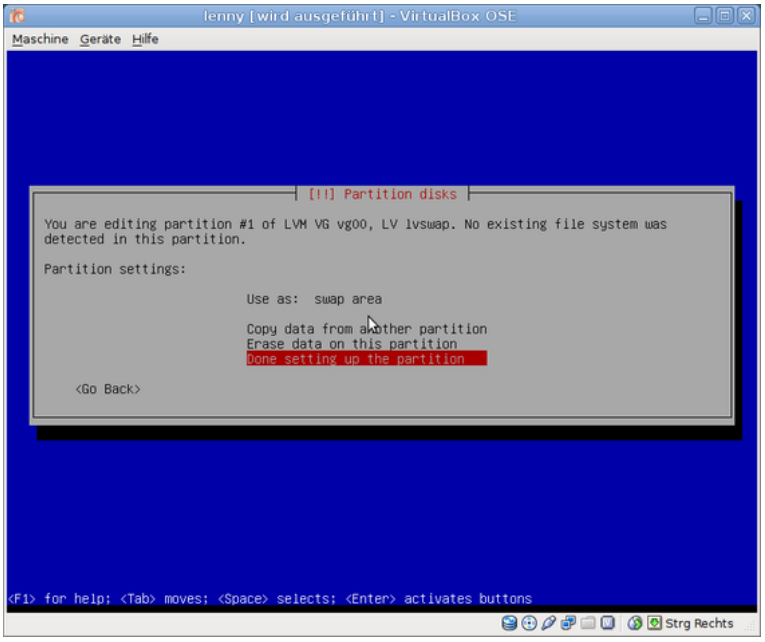

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Please enter the size of the new logical volume. The size may be entered in the following formats: 10K (Kilobytes), 10M (Megabytes), 10G (Gigabytes), 10T (Terabytes). The default unit is Megabytes.
Logical volume size:
1G
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

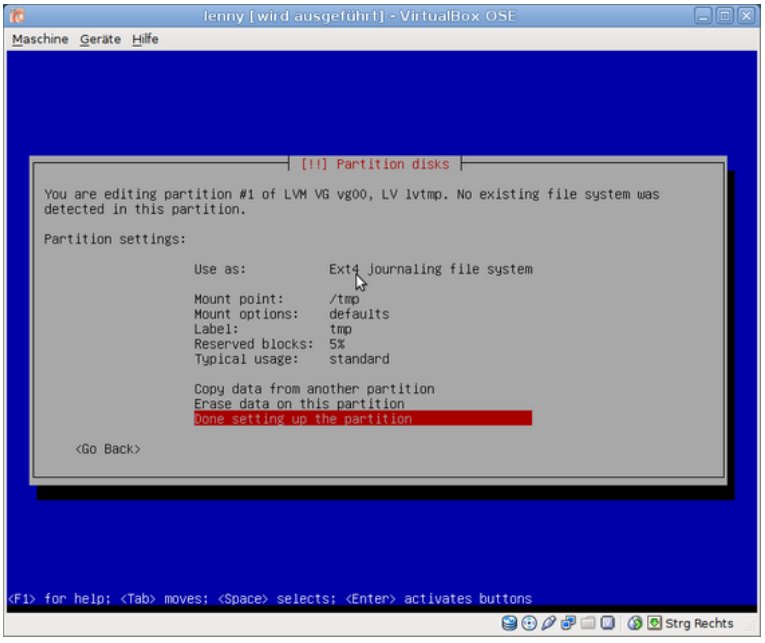

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
Summary of current LVM configuration:
Free Physical Volumes: 0
Used Physical Volumes: 1
Volume Groups: 1
Logical Volumes: 6
LVM configuration action:
Display configuration details
Create logical volume
Delete logical volume
Finish
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

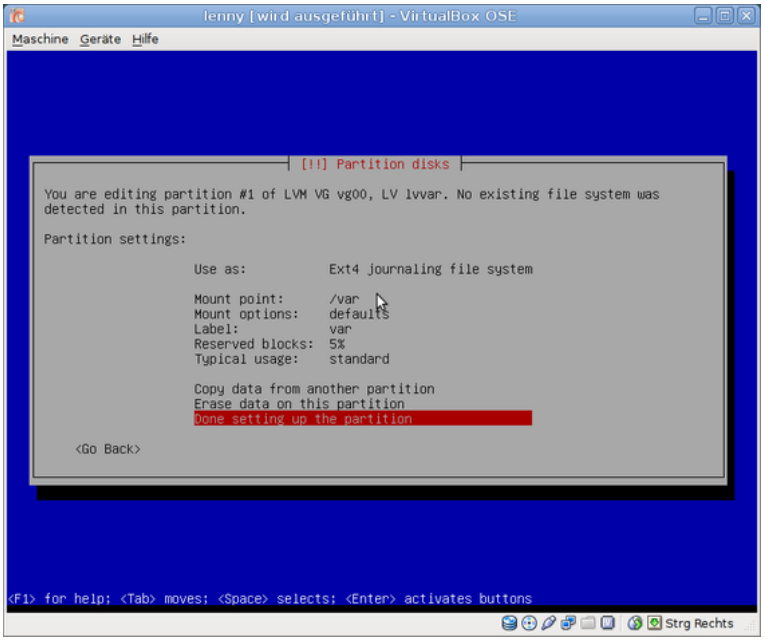

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
Configure software RAID
Configure the Logical Volume Manager
Configure encrypted volumes
LVM VG vg00, LV lvboot - 499.1 MB Linux device-mapper (linear)
#1 499.1 MB
LVM VG vg00, LV lvroot - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB
LVM VG vg00, LV lvswap - 998.2 MB Linux device-mapper (linear)
#1 998.2 MB
LVM VG vg00, LV lvtmp - 998.2 MB Linux device-mapper (linear)
#1 998.2 MB
LVM VG vg00, LV lvvar - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB
LVM VG vg00, LV lvvmail - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB
SCSI1 (0,0,0) (sda) - 21.1 GB ATA VBOX HARDDISK
#1 primary 21.1 GB K lvm
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

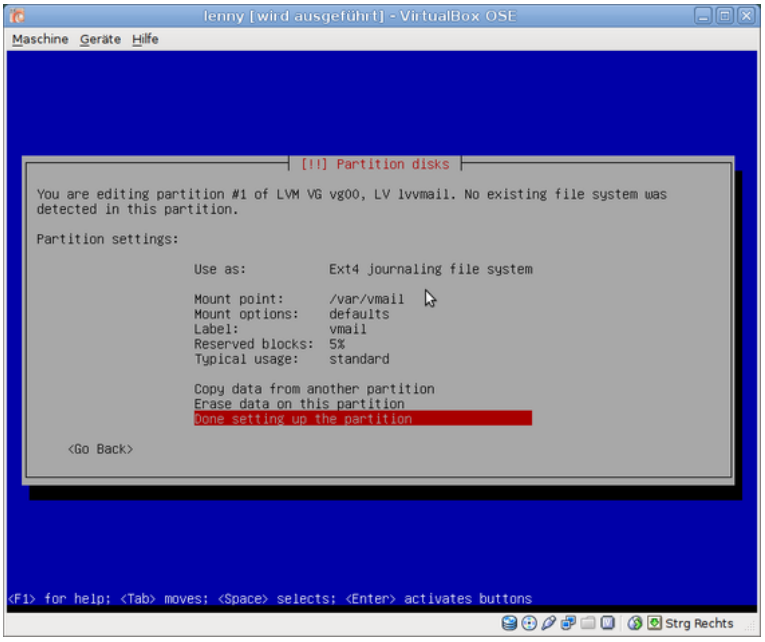

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvroot. No existing file system was detected in this partition.
Partition settings:
Use as: Ext4 journaling file system
Mount point: /
Mount options: defaults
Label: root
Reserved blocks: 5%
Typical usage: standard
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

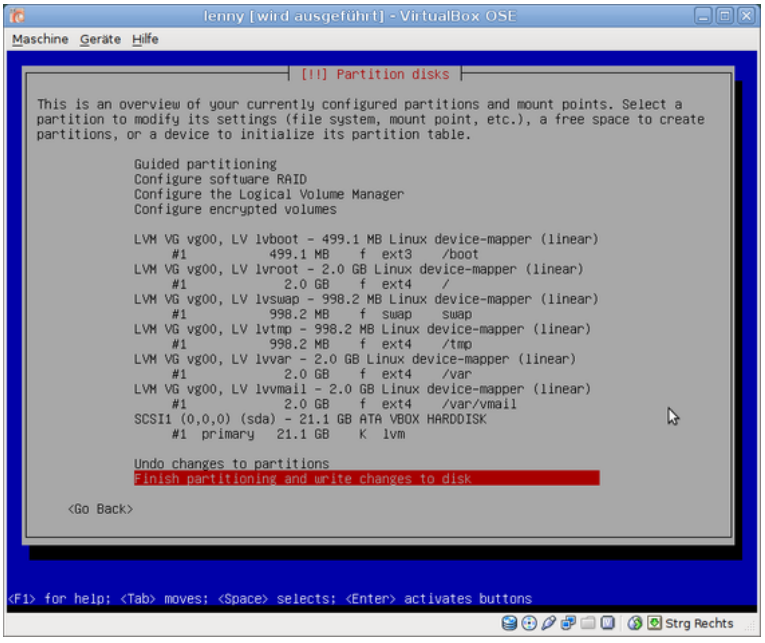

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvboot. No existing file system was detected in this partition.
Partition settings:
Use as: Ext3 journaling file system
Mount point: /boot
Mount options: defaults
Label: boot
Reserved blocks: 5%
Typical usage: standard
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

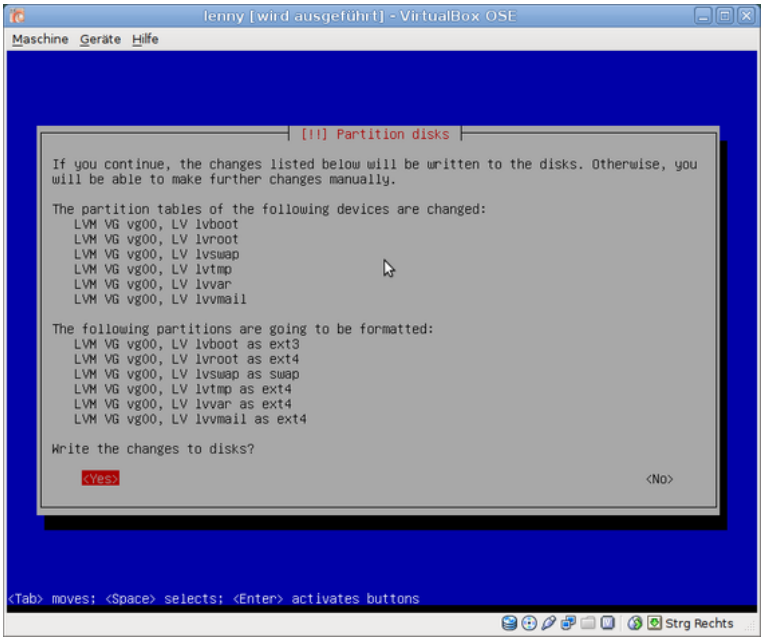

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvswap. No existing file system was detected in this partition.
Partition settings:
Use as: swap area
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

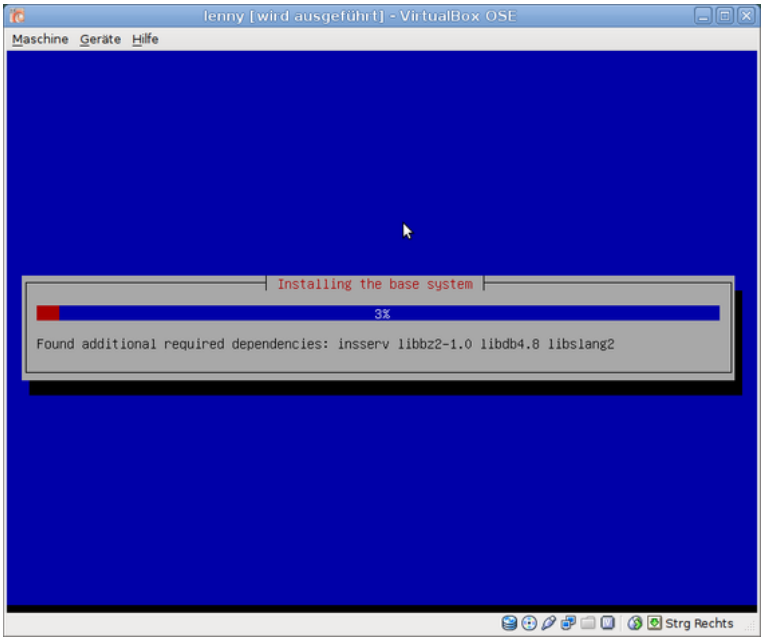

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvtmp. No existing file system was detected in this partition.
Partition settings:
Use as: Ext4 journaling file system
Mount point: /tmp
Mount options: defaults
Label: tmp
Reserved blocks: 5%
Typical usage: standard
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

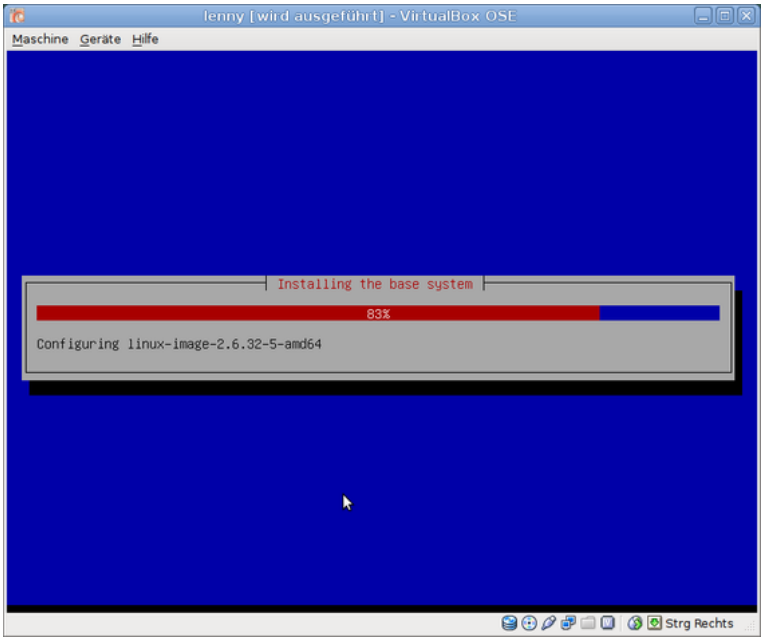

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvvar. No existing file system was detected in this partition.
Partition settings:
Use as: Ext4 journaling file system
Mount point: /var
Mount options: defaults
Label: var
Reserved blocks: 5%
Typical usage: standard
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

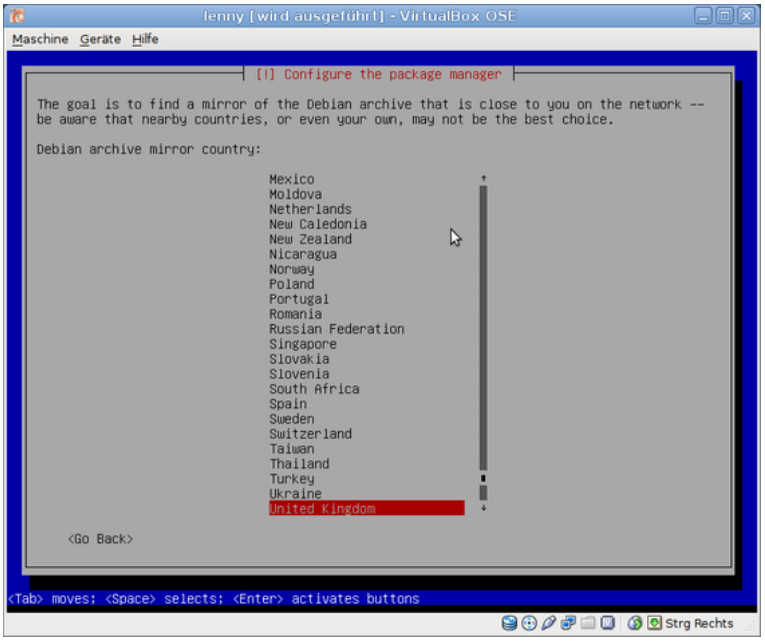

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvvmail. No existing file system was detected in this partition.
Partition settings:
Use as: Ext4 journaling file system
Mount point: /var/vmail
Mount options: defaults
Label: vmail
Reserved blocks: 5%
Typical usage: standard
Copy data from another partition
Erase data on this partition
Done setting up the partition
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

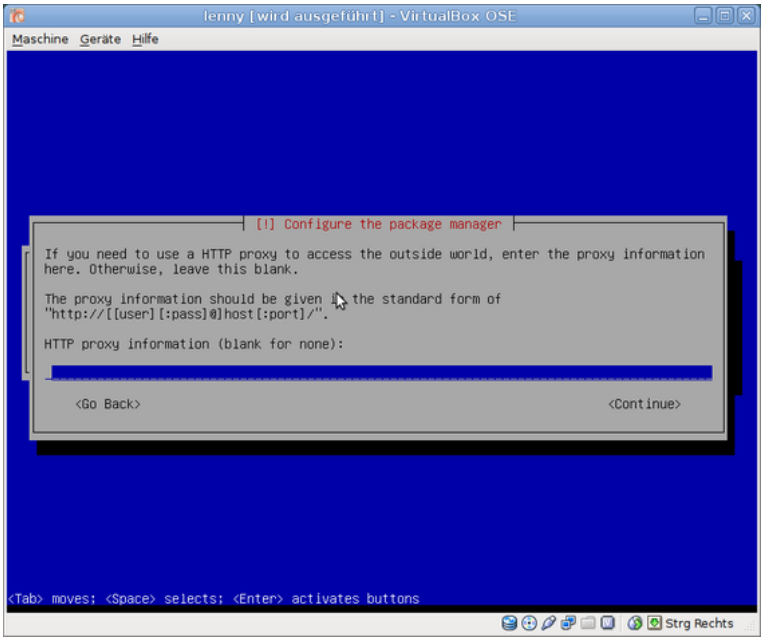

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.
Guided partitioning
Configure software RAID
Configure the Logical Volume Manager
Configure encrypted volumes
LVM VG vg00, LV lvboot - 499.1 MB Linux device-mapper (linear)
#1 499.1 MB f ext3 /boot
LVM VG vg00, LV lvroot - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB f ext4 /
LVM VG vg00, LV lvswap - 998.2 MB Linux device-mapper (linear)
#1 998.2 MB f swap swap
LVM VG vg00, LV lvtmp - 998.2 MB Linux device-mapper (linear)
#1 998.2 MB f ext4 /tmp
LVM VG vg00, LV lvvar - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB f ext4 /var
LVM VG vg00, LV lvvmail - 2.0 GB Linux device-mapper (linear)
#1 2.0 GB f ext4 /var/vmail
SCSI1 (0,0,0) (sda) - 21.1 GB ATA VBOX HARDDISK
#1 primary 21.1 GB K lvm
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>
<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

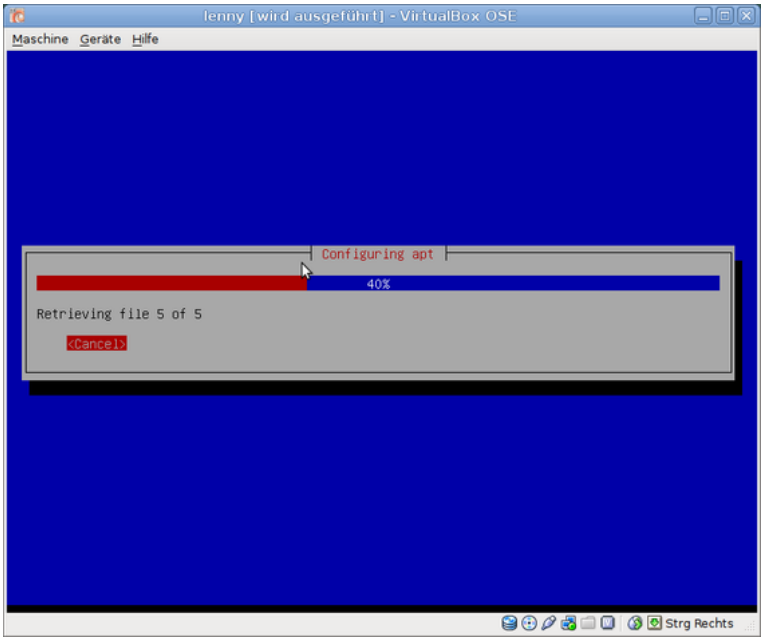

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Partition disks
If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.
The partition tables of the following devices are changed:
LVM VG vg00, LV lvboot
LVM VG vg00, LV lvroot
LVM VG vg00, LV lvswap
LVM VG vg00, LV lvtmp
LVM VG vg00, LV lvvar
LVM VG vg00, LV lvvmail
The following partitions are going to be formatted:
LVM VG vg00, LV lvboot as ext3
LVM VG vg00, LV lvroot as ext4
LVM VG vg00, LV lvswap as swap
LVM VG vg00, LV lvtmp as ext4
LVM VG vg00, LV lvvar as ext4
LVM VG vg00, LV lvvmail as ext4
Write the changes to disks?
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

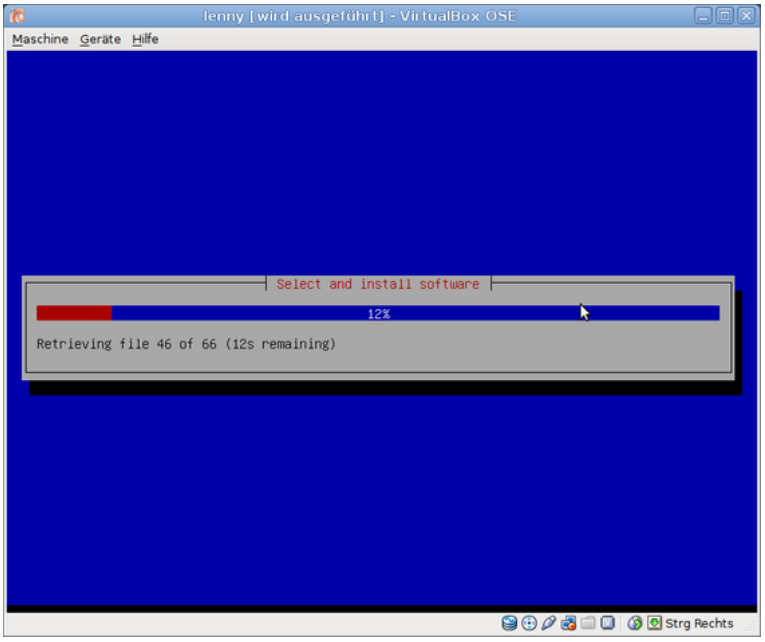

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Installing the base system
3%
Found additional required dependencies: insserv libbz2-1.0 libdb4.8 libslang2
Strg Rechts

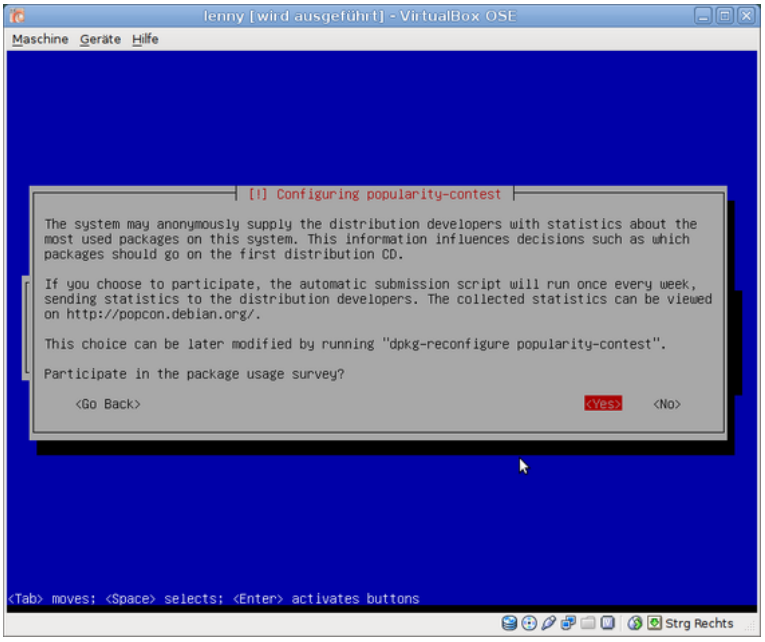

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Installing the base system
83%
Configuring linux-image-2.6.32-5-amd64
Strg Rechts

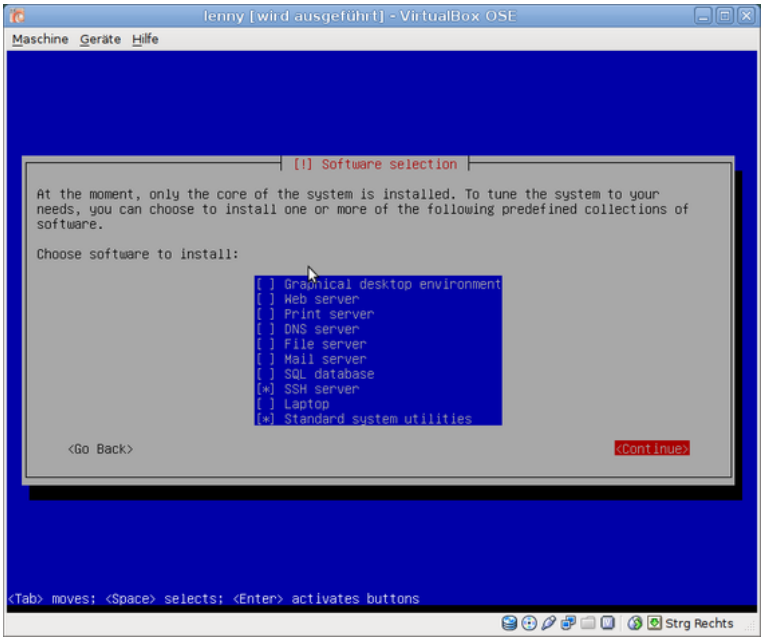

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Configure the package manager
The goal is to find a mirror of the Debian archive that is close to you on the network --
be aware that nearby countries, or even your own, may not be the best choice.
Debian archive mirror country:
Mexico
Moldova
Netherlands
New Caledonia
New Zealand
Nicaragua
Norway
Poland
Portugal
Romania
Russian Federation
Singapore
Slovakia
Slovenia
South Africa
Spain
Sweden
Switzerland
Taiwan
Thailand
Turkey
Ukraine
United Kingdom
<Go Back>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

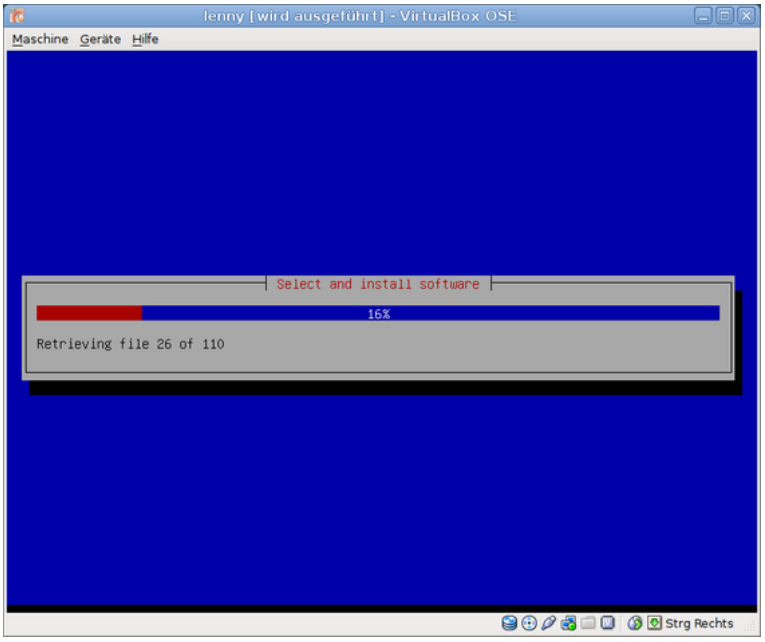

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Configure the package manager
If you need to use a HTTP proxy to access the outside world, enter the proxy information here. Otherwise, leave this blank.
The proxy information should be given in the standard form of "http://[[user][:pass]@]host[:port]/".
HTTP proxy information (blank for none):
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

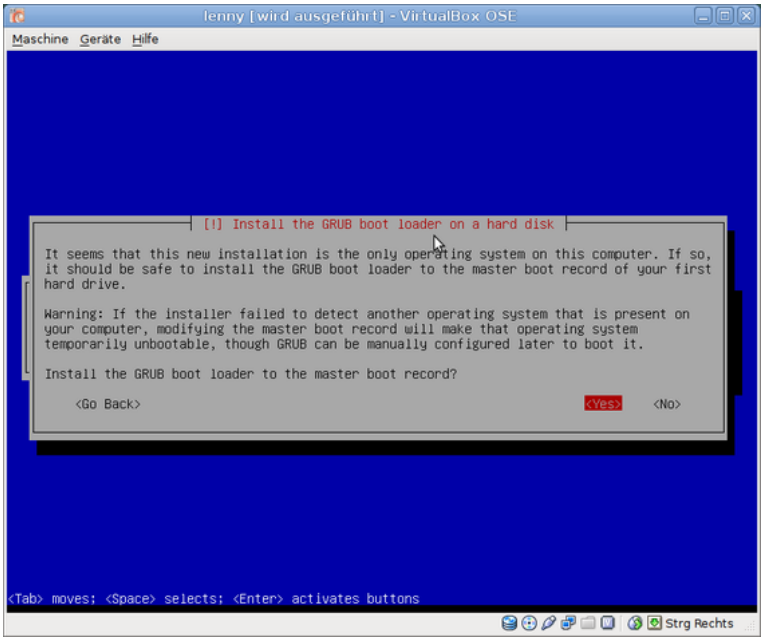

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Configuring apt
40%
Retrieving file 5 of 5
<Cancel>
Strg Rechts

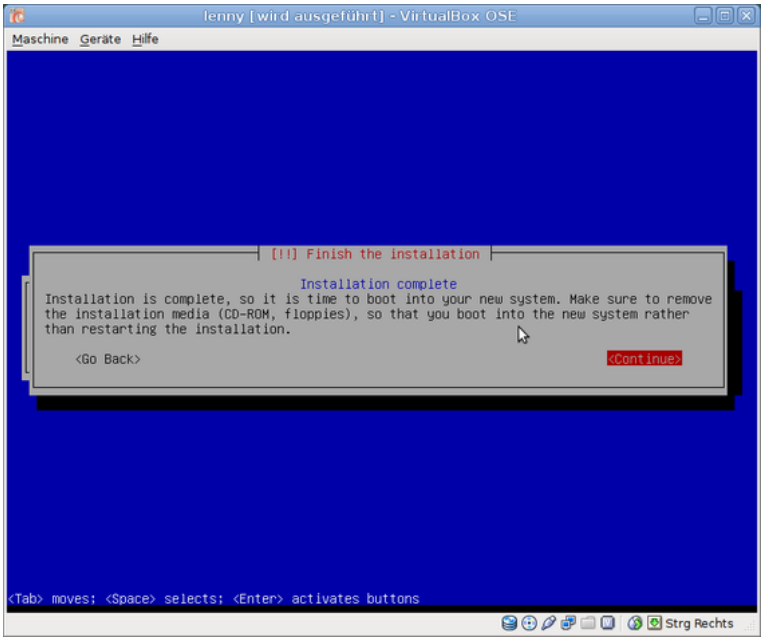

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Select and install software
12%
Retrieving file 46 of 66 (12s remaining)
Strg Rechts

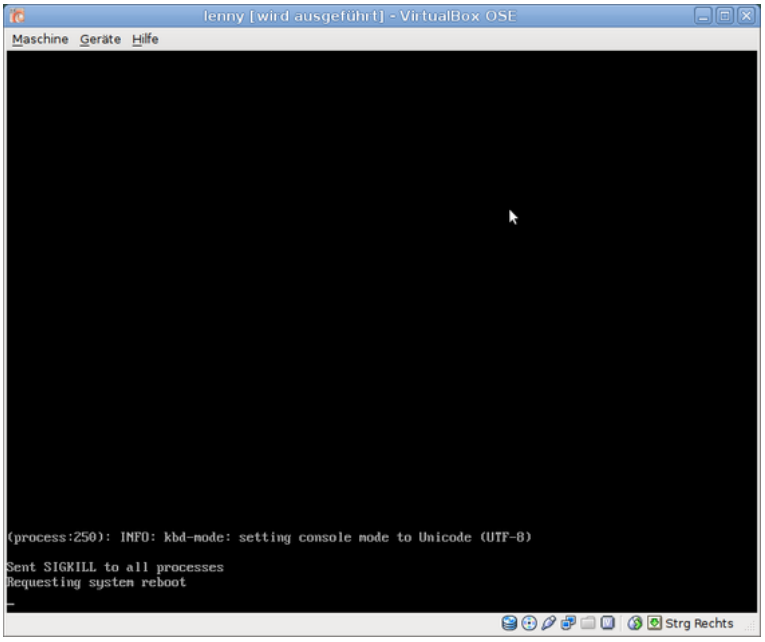

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Configuring popularity-contest
The system may anonymously supply the distribution developers with statistics about the most used packages on this system. This information influences decisions such as which packages should go on the first distribution CD.
If you choose to participate, the automatic submission script will run once every week, sending statistics to the distribution developers. The collected statistics can be viewed on http://popcon.debian.org/.
This choice can be later modified by running "dpkg-reconfigure popularity-contest".
Participate in the package usage survey?
<Go Back>
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts

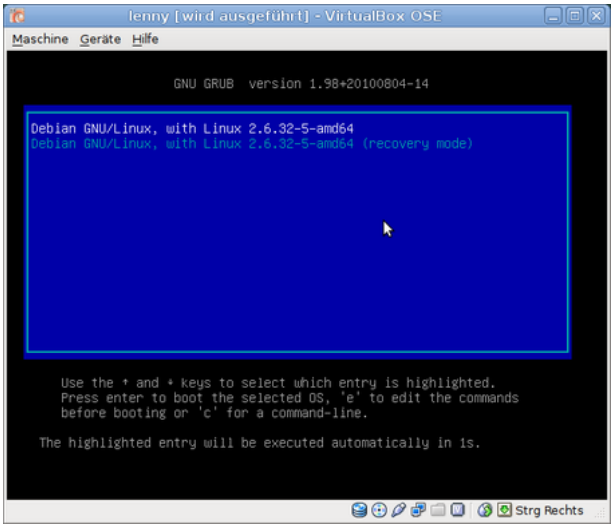

lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Software selection
At the moment, only the core of the system is installed. To tune the system to your needs, you can choose to install one or more of the following predefined collections of software.
Choose software to install:
[ ] Graphical desktop environment
[ ] Web server
[ ] Print server
[ ] DNS server
[ ] File server
[ ] Mail server
[ ] SQL database
[*] SSH server
[ ] Laptop
[*] Standard system utilities
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts


lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
Select and install software
16%
Retrieving file 26 of 110
Strg Rechts


lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!] Install the GRUB boot loader on a hard disk
It seems that this new installation is the only operating system on this computer. If so, it should be safe to install the GRUB boot loader to the master boot record of your first hard drive.
Warning: If the installer failed to detect another operating system that is present on your computer, modifying the master boot record will make that operating system temporarily unbootable, though GRUB can be manually configured later to boot it.
Install the GRUB boot loader to the master boot record?
<Go Back>
<Yes>
<No>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts


lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
[!!] Finish the installation
Installation complete
Installation is complete, so it is time to boot into your new system. Make sure to remove the installation media (CD-ROM, floppies), so that you boot into the new system rather than restarting the installation.
<Go Back>
<Continue>
<Tab> moves; <Space> selects; <Enter> activates buttons
Strg Rechts


lenny [wird ausgeführt] - VirtualBox OSE
Maschine Geräte Hilfe
(process:250): INFO: kbd-mode: setting console mode to Unicode (UTF-8)
Sent SIGKILL to all processes
Requesting system reboot
Strg Rechts



Installing necessary Debian packages

- Add new comment
- 61693 reads

By now your Debian Squeeze server should be installed correctly. Now it's time to install the necessary Debian packages to make it an actual mail server.

To get a clean system with all security updates installed you may want to run:

apt-get update

apt-get upgrade

Debian Squeeze does not install the SSH server by default. So to spare you an extra walk to the server room I suggest:

apt-get install ssh

Then let's install Postfix with MySQL backend support:

apt-get install postfix postfix-mysql

When you get asked for the mail server configuration type please choose "*Internet site*". Enter your own mail server name (the fully qualified domain name).

As by default Debian installs exim as a mail service you should remove its remains:

apt-get --purge remove 'exim4*'

You also need a MySQL server:

apt-get install mysql-server

Note: You can run MySQL server on the same system as your actual mail service - but don't have to. The mail server can communicate with the MySQL server via TCP networking. Maybe you even have a MySQL server in your network already that you can use.

During the installation you will get asked for the password of a newly created "root" user. This is not your system's "root" login but a special administrative user used to access the MySQL server. Choose a secure password you want and write it down. And another warning: during the installation a MySQL user account called "debian-sys-maint" will be created with a random password. The password is stored in the `/etc/mysql/debian.cnf` file. Do not touch this file or change the user's password in the database or you won't be able to stop or start the MySQL service any more.

You will want to offer POP3 and IMAP services to your users so you need to install Dovecot's services:

apt-get install dovecot-pop3d dovecot-imapd

If you intend to offer a webmail service I can recommend the Roundcube package. Previous versions of the tutorial recommended Squirrelmail but honestly Roundcube is way more comfortable. Type:

apt-get install roundcube

This will also install an Apache server and PHP packages. You will get asked if you want to maintain the RoundCube package using "dbconfig-common". Basically this means that Debian will set up the database for you and in future releases upgrade the database. I suggest you say "Yes" here. The database type is "MySQL". When asked for the "database's administrative user's password" enter the password you entered for the MySQL "root" user when you installed the MySQL server. Then you will have to think of a safe password that RoundCube will use to access its database and enter that twice.

If you believe you are not a MySQL ninja on the command line then I suggest you install the PHPMyAdmin software which allows you to manage your MySQL database in the web browser:

apt-get install phpmyadmin

Again you are asked if you want to manage the phpmyadmin database using "dbconfig-common". Answer with "yes", enter the MySQL "root" user's password and think of a new password used for PHPMyAdmin to connect to its database. As a web server to be used select "apache2".

And finally the console-based *mutt* email client lets you read mail from mailboxes directly from the hard disk. It will be helpful for testing the configuration. And it's even a very powerful IMAP email if you like using the console. Maybe you'll start to like it, too. You should install it at least for testing:

```
apt-get install mutt
```

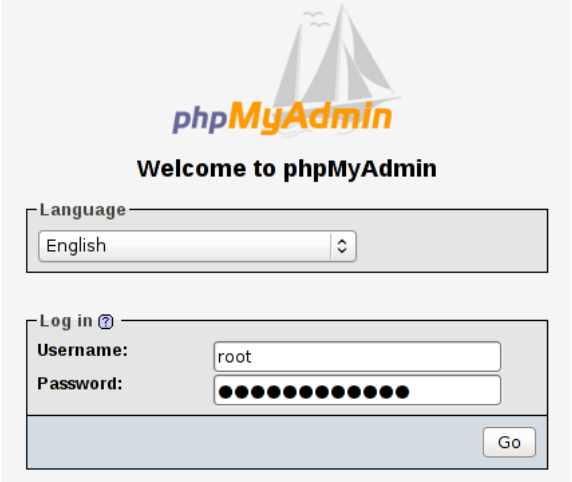
Now all basic packages are installed and it's time to prepare the database in the next chapter.

Preparing the database

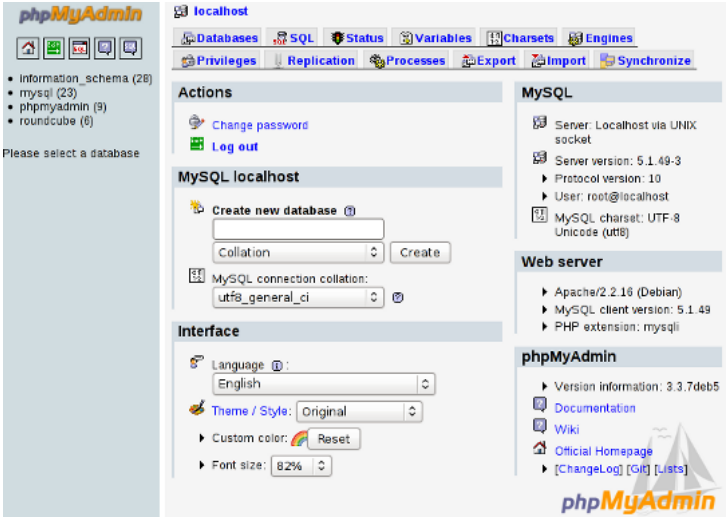
- Add new comment
- 100309 reads

Preparing the database

Now it's time to prepare the MySQL database that will store information that controls your mail server. In the process you will have to enter SQL queries. You can enter them on the 'mysql' command line. But if you are less experienced with MySQL I suggest you start easy with "phpMyAdmin" by pointing your web browser at this URL: http://YOUR-MAIL-SERVER/phpmyadmin. You should see a web page like:



Login as "root" with your MySQL administrative password. Then you will find yourself on the main screen:



This will help you manage your databases. But in the course of this tutorial I will just document how to work with MySQL using the console. SQL statements will be marked in blue.

Create the database

Your first task is to create a new database in MySQL. Let's call it '*mailserver*'. In a root shell enter this command:

```
mysqladmin -p create mailserver
```

You will be asked for the MySQL "root" password that you entered when you installed the MySQL package.

Add a less privileged MySQL user

For security reasons you should create another MySQL user account with fewer privileges. Postfix just needs to read from the database so it does not need write access.

Connect to your database:

```
mysql -p mailserver
```

When you see the "mysql>" prompt enter the following SQL statement to grant the appropriate privileges but instead of the dummy password 'mailuser2011' please choose a more secure password. The "pwgen" or "apg" tools will generate random passwords for you if you don't feel creative. I will keep using 'mailuser2011' in this tutorial though.

```
GRANT SELECT ON mailserver.*
TO 'mailuser'@'127.0.0.1'
IDENTIFIED BY 'mailuser2011';
```

This will create a user called 'mailuser' that has only the privilege to select/read data from the database but not to alter it. If you want to add or alter data in the database either use the 'root' account or create another account for that purpose.

Create the database tables

In the newly created database you will have to create tables that store information about domains, forwardings and the users' mailboxes. First create a table for the list of virtual domains that you want to host:

```
CREATE TABLE `virtual_domains` (
  `id` int(11) NOT NULL auto_increment,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

The next table contains information on the actual user accounts. Every user has a username and password. It is used for accessing the mailbox by POP3 or IMAP, logging into the webmail service or to send mail ("relay") if they are not in your local network. As users tend to easily forget things the user's email address is also used as the login username. Let's create the users table:

```
CREATE TABLE `virtual_users` (
  `id` int(11) NOT NULL auto_increment,
  `domain_id` int(11) NOT NULL,
  `password` varchar(32) NOT NULL,
  `email` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `email` (`email`),
  FOREIGN KEY (domain_id) REFERENCES virtual_domains(id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

The **email** field will contain the email address/username. And the **password** field will contain an MD5 hash of the user's password. The *unique key* on the **email** field makes sure that there are no two users in a domain accidentally.

And finally a table is needed for *aliases* (email forwardings from one account to another):

```
CREATE TABLE `virtual_aliases` (
  `id` int(11) NOT NULL auto_increment,
  `domain_id` int(11) NOT NULL,
  `source` varchar(100) NOT NULL,
  `destination` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (domain_id) REFERENCES virtual_domains(id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Here the **source** column contains the email address of the user who wants to forward their mail. In case of *catchall* addresses the source looks like "@domain". The **destination** column contains the target email address. As described in the section on virtual domains there can be several rows for a source address designating multiple destinations who will get copies of an email.

You wonder about the *foreign keys*? They express that entries in the virtual_aliases and virtual_users tables are connected to entries in the virtual_domains table. This will keep the data in your database consistent because you cannot create virtual aliases or virtual users that are not connected to a virtual domain. The suffix 'ON DELETE CASCADE' means that if you delete a row from the referenced table that the deletion will also be done on the current table automatically. So you do not leave orphaned entries accidentally. Imagine that you do not host a certain domain any longer. You can remove the domain entry from the virtual_domains table and all dependent/referenced entries in the other tables will also be removed. (Note however that this would not remove the physical mail directories from the hard disk automatically.)

An example of the data in the tables:

virtual_domains

id	*name*
1	example.org
2	example.net

virtual_users

id	*domain_id*	*email*	*password*
1	1	john@example.org	14cbfb845af1f030e372b1cb9275e6dd
2	1	steve@example.org	a57d8c77e922bf756ed80141fc77a658
3	2	kerstin@example.net	5d6423c4ccddcbbdf0fcfaf9234a72d0

Let us add a simple alias:

virtual_aliases

id	*domain_id*	*source*	*destination*

1	1	steve@example.org	devnull@workaround.org
2	2	kerstin@example.net	kerstin42@yahoo.com
3	2	kerstin@example.net	kerstin@mycompany.com

This will make the mail for steve@example.org be redirected to devnull@workaround.org. And the mail for kerstin@example.net is redirected to both kerstin42@yahoo.com and kerstin@mycompany.com. Neither Steve nor Kerstin receive a copy of the email.

Test data

Let's populate the database with the example.org domain, a john@example.org email account and a forwarding of jack@example.org to john@example.org. Open a MySQL shell and issue the following SQL queries:

```
INSERT INTO `mailserver`.`virtual_domains` (
  `id` ,
  `name`
)
VALUES (
  '1', 'example.org'
);

INSERT INTO `mailserver`.`virtual_users` (
  `id` ,
  `domain_id` ,
  `password` ,
  `email`
)
VALUES (
  '1', '1', MD5( 'summersun' ) , 'john@example.org'
);

INSERT INTO `mailserver`.`virtual_aliases` (
  `id`,
  `domain_id`,
  `source`,
  `destination`
)
VALUES (
  '1', '1', 'jack@example.org', 'john@example.org'
);
```

Postfix/Database configuration

- Add new comment
- 132382 reads

In the previous chapter you have fed the MySQL database. Now it's time to make use of it. The entry point for all email on your system is Postfix. So we need to tell Postfix where to find the database-stored information. Let's start by telling it which virtual domains you have.

virtual_mailbox_domains

As described earlier a mapping in Postfix is just a table that contains a left-hand side (LHS) and a right-hand side (RHS). To make Postfix use MySQL to define a mapping we need a 'cf' file (configuration file). Start by creating a file called `/etc/postfix/mysql-virtual-mailbox-domains.cf` for the virtual_mailbox_domains mapping that contains:

```
user = mailuser
password = mailuser2011
hosts = 127.0.0.1
dbname = mailserver
query = SELECT 1 FROM virtual_domains WHERE name='%s'
```

Imagine that Postfix received an email for somebody@example.org and wants to find out if *example.org* is a virtual mailbox domain. It will run the above SQL query and replace '%s' by 'example.org'. If it finds such an entry in the virtual_domains table it will return a '1'. Actually it does not matter what exactly is returned as long as there is a result.

Note: You may be tempted to write "localhost" instead of "127.0.0.1". Don't do that because there is indeed a difference in this context. "localhost" will make Postfix look for the MySQL socket file and it can't find it within it's chroot jail at /var/spool/postfix because it is at /var/run/mysqld/mysqld.sock by default. But if you tell Postfix to use 127.0.0.1 as described here you make Postfix use a TCP connection to port 3306 on localhost which is working even if Postfix is jailed.

And you need to make Postfix use this database mapping:

```
postconf -e virtual_mailbox_domains=mysql:/etc/postfix/mysql-virtual-mailbox-domains.cf
```

(The `postconf -e` command conveniently adds configuration lines to your `/etc/postfix/main.cf` file. It also activates the new setting instantly so you do not have to reload the Postfix process.)

Postfix will now search your virtual_domains table to find out if "example.org" is a virtual mailbox domain. Let us see if this works. You have set up the "example.org" domain in the previous chapter already. So we can query Postfix now to see if it will find the domain in the database:

```
postmap -q example.org mysql:/etc/postfix/mysql-virtual-mailbox-domains.cf
```

You should get '1' as a result. Your first mapping is working. Great. Get straight to the second one.

virtual_mailbox_maps

You will now define the *virtual_mailbox_maps* which is mapping email addresses (left-hand side) to the location of the user's mailbox on your harddisk (right-hand side). If you saved incoming email to the hard disk using Postfix's built-in *virtual delivery agent* then it would be queried to find out the mailbox path. But in our setup the actual delivery is done by Dovecot's LDA (*local delivery agent*) so Postfix does not really care about the path. Postfix just needs to check if a certain email address is valid. Similar to the above you need an SQL query that searches for an email address and returns "1".

Next you will need to create a *".cf"* file to tell Postfix about the SQL query for this table. In addition to the email address it is also important to get the user's password later on. As the path of the user's mailbox is fixed it is not important to get that information from the database. The directory structure will always be `/var/vmail/$DOMAIN/$USER`. So for John's example it would be `/var/vmail/example.org/john`.

Now things are a bit simpler and you can finally create a "*.cf*" file at `/etc/postfix/mysql-virtual-mailbox-maps.cf` that is as simple as:

```
user = mailuser
password = mailuser2011
hosts = 127.0.0.1
dbname = mailserver
query = SELECT 1 FROM virtual_users WHERE email='%s'
```

Tell Postfix that this mapping file is supposed to be used for the *virtual_mailbox_maps* mapping:

```
postconf -e virtual_mailbox_maps=mysql:/etc/postfix/mysql-virtual-mailbox-maps.cf
```

Test if Postfix is happy with this mapping by asking it where the mailbox directory of our john@example.org user would be:

```
postmap -q john@example.org mysql:/etc/postfix/mysql-virtual-mailbox-maps.cf
```

You should get `"1"` back which means that `john@example.org` is an existing virtual mailbox user on your server. Later in the Dovecot configuration part you will also use the *email* and *password* fields but Postfix does not need them here. Great, there is just one mapping left to define:

virtual_alias_maps

The *virtual_alias_maps* mapping is used for forwarding emails from one email address to another. It is possible to name multiple destinations. In the database this is achieved by using different rows. See the page on virtual domains if you need details.

Create another ".cf" file at `/etc/postfix/mysql-virtual-alias-maps.cf`:

```
user = mailuser
password = mailuser2011
hosts = 127.0.0.1
dbname = mailserver
query = SELECT destination FROM virtual_aliases WHERE source='%s'
```

Make Postfix use this database mapping:

```
postconf -e virtual_alias_maps=mysql:/etc/postfix/mysql-virtual-alias-maps.cf
```

Test if the mapping file works as expected:

```
postmap -q jack@example.org mysql:/etc/postfix/mysql-virtual-alias-maps.cf
```

You should see the expected destination:

```
john@example.org
```

Optional: Black magic for if you need catch-all aliases

As explained earlier in the tutorial there is way to alias all email for a domain to a certain destination email address. This is called a "catchall" alias. Catchalls catch all emails for a domain if there is no specific virtual user for that email address. A catchall alias looks like "@example.org" and forwards email for the whole domain to one account. We have created the 'john@example.org' user and would like to forward all *other* email on the domain to 'kerstin@example.com'. So we would add a catchall alias like:

source	destination
@example.org	kerstin@example.com

Now imagine what happens when Postfix receives an email for 'john@example.org'. Postfix will first check if there are any aliases in the *virtual_alias_maps* table. (It does not look at the *virtual_mailbox_maps* table at the moment.) It finds the catchall entry as above and since there is no more specific alias the catchall account matches and the email is redirected to 'kerstin@example.com'. This is probably not what you wanted. So you need to make the table rather look like this:

email	destination
@example.org	kerstin@example.com
john@example.org	john@example.org

More specific aliases have precedence over general catchall aliases. Postfix will lookup all these mappings for each of:

- john@example.org (most specific)
- john (only works if "example.org" is the $myorigin domain)
- @example.org (catchall - least specific)

Postfix will find an entry for 'john@example.org' first and sees that email should be "forwarded" to 'john@example.org' - the same email address. This trickery may sound weird but it is needed if you plan to use catchall accounts. So the *virtual_alias_maps* mapping must obey both the

"view_aliases" view and this "john-to-himself" mapping. This is outlined in the virtual(5) man page in the *TABLE SEARCH ORDER* section.

Create a "*.cf*" file `/etc/postfix/mysql-email2email.cf` for the latter mapping:

```
user = mailuser
password = mailuser2011
hosts = 127.0.0.1
dbname = mailserver
query = SELECT email FROM virtual_users WHERE email='%s'
```

Check that you get John's email address back when you ask Postfix if there are any aliases for him:

```
postmap -q john@example.org mysql:/etc/postfix/mysql-email2email.cf
```

The result should be the same address:

```
john@example.org
```

Now you need to tell Postfix that these two mappings should be searched by adding this line to your `main.cf`:

```
postconf -e virtual_alias_maps=mysql:/etc/postfix/mysql-virtual-alias-maps.cf,mysql:/etc/postfix/mysql-email2email.cf
```

The order of the two mappings is not important here.

You did it! All mappings are set up and the database is generally ready to be filled with domains and users. Make sure that only 'root' and the 'postfix' user can read the "*.cf*" files - after all your database password is stored there:

```
chgrp postfix /etc/postfix/mysql-*.cf

chmod u=rw,g=r,o= /etc/postfix/mysql-*.cf
```

Setting up Dovecot

- Add new comment
- 196741 reads

Let us now configure Dovecot which will do several things for us:

- get emails from Postfix and save them to disk
- execute user-based "sieve" filter rules (can be used to put away emails to different folders)
- allow the user to fetch emails using POP3 or IMAP

Before we get to the actual configuration for security reasons I recommend that you create a new system user that will own all virtual mailboxes. The following shell commands will create a system group "vmail" with GID (group ID) 5000 and a system user "vmail" with UID (user ID) 5000. (Make sure that UID and GID are not yet used or choose another - the number can be anything between 1000 and 65000 that is not yet used):

```
groupadd -g 5000 vmail
useradd -g vmail -u 5000 vmail -d /var/vmail -m
```

Also make sure that this directory has the proper permissions:

```
chown -R vmail:vmail /var/vmail
chmod u+w /var/vmail
```

The configuration files for Dovecot are found under `/etc/dovecot`. Start editing the main file...

/etc/dovecot/dovecot.conf

See the line `protocols` and define the protocols you want to offer. By default this line reads:

```
protocols = imap imaps pop3 pop3s
```

so that Dovecot starts the IMAP and POP3 services and also its equivalents that work over an encrypted SSL (*secure socket layer*) connection. If you want to be strict about not allowing insecure connections then leave out the "imap" and "pop3" keywords here.

Although this is a less secure setting you will probably still need it:

```
disable_plaintext_auth = no
```

This will allow plaintext passwords over an unsecured (non-SSL) connection. By default it is set to 'yes' for security reasons. Setting it to 'no' will mean less security but may help users of a "certain" Microsoft email software that is problematic in many ways.

An important setting is:

```
mail_location = maildir:/var/vmail/%d/%n/Maildir
```

which will tell that the users' mailboxes are always found at `/var/vmail/DOMAIN/USER/Maildir` and that it should be in *maildir* format.

There is already a section "namespace private" in your dovecot.conf which is commented out by "#" characters. The "private" namespace is the personal mailbox of a certain user. You can leave this section disabled and get a maildir directory schema like:

/var/vmail/christoph.haas/email/Maildir/.spam

If you followed previous ISPmail tutorials then your directories may be different. If you rather have:

/var/vmail/christoph.haas/email/Maildir/.INBOX.spam

then you need to declare that in the "namespace private" section as follows. Enable this section and make sure these variables are set:

```
namespace private {
   separator = .
   inbox = yes
}
```

Next look for a section called "auth default". First define the allowed authentication mechanisms:

```
 mechanisms = plain login
```

Usually "plain" is used but a certain Micros*oft email client insists on using "login". Both mechanism use plain text so it is strongly recommended that your users use IMAPS and POP3S which are the SSL/TLS encrypted equivalents to IMAP and POP3.

As you browse through the section you see many backends that Dovecot can access to get the email users' data. We are using SQL lookups for the passdb (=password) but static information to get the users's (because all users follow the same scheme). Inside this section you need to set:

```
passdb sql {
   args = /etc/dovecot/dovecot-sql.conf
}
```

which tells Dovecot that the passwords are stored in an SQL database and:

```
userdb static {
   args = uid=5000 gid=5000 home=/var/vmail/%d/%n allow_all_users=yes
}
```

to tell Dovecot where the mailboxes are located. This is similar to the *mail_location* setting. The user gets authenticated in the "passdb sql" section. So the "userdb static" section defined where the mail folders are located. Using "userdb sql" is not needed as all mailboxes follow a fixed directory schema. This saves an SQL query for each access. The "allow_all_users=yes" setting means that it is not necessary for Dovecot to check if a certain user exists. We can do that because Postfix has already ensured (in the virtual_mailbox_maps query) that the users existed before their email was handed over to Dovecot's "deliver" agent.

You will want to comment out the section called `"passdb pam` that deals with system users. Otherwise Dovecot will also look for system users when someone fetches emails which leads to warnings in your log file.

Now look for another section called `socket listen`. Here you define socket files that are used to interact with Dovecot's authentication mechanism. Make the section read:

```
 socket listen {
   master {
      path = /var/run/dovecot/auth-master
      mode = 0600
      user = vmail
   }

   client {
      path = /var/spool/postfix/private/auth
      mode = 0660
      user = postfix
      group = postfix
   }
}
```

The *"master"* section is needed to give Dovecot's delivery agent (the program that saves a new mail to the user's mailbox) access to the userdb information. The "*client*" section creates a socket inside the "chroot" directory of Postfix. This socket file will be used by Postfix for SMTP authentication when users send their email through your mail server as a relay.

(*chroot* means that parts of Postfix are jailed into `/var/spool/postfix` and can only access files in that directory or its subdirectories. It is a good security measure so that even if Postfix had bugs and were hacked then the attacker would not be able to access /etc/passwd for example because it's outside of /var/spool/postfix.)

And the "*`protocol lda`*" section needs to be customized. The LDA (*local delivery agent*) is more capable than Postfix's built-in virtual delivery agent. It allows for quotas and Sieve (ships with the *dovecot-common* package) filtering. Let the section be:

```
 protocol lda {
   auth_socket_path = /var/run/dovecot/auth-master
   postmaster_address = postmaster@example.com
   mail_plugins = sieve
   log_path =
}
```

Please change the above *postmaster* email address to a valid address where a real human administrator can be reached.

The log_path setting is optional but may help you figure out why a certain server-side filter is not doing what you expect. Leaving it empty as shown above will log delivery details in your normal /var/log/mail.log but you can also use a file name here to create a seperate logfile.

Finally edit the `/etc/dovecot/dovecot-sql.conf` and change these settings:

```
driver = mysql
connect = host=127.0.0.1 dbname=mailserver user=mailuser password=mailuser2011
default_pass_scheme = PLAIN-MD5
password_query = SELECT email as user, password FROM virtual_users WHERE email='%u';
```

Whenever Dovecot needs to check an email user's password if will run the above query. It will create an MD5 hash of the user's password and look for that in the "virtual_users" database table.

Restart Dovecot:

```
/etc/init.d/dovecot restart
```

Now look at your `/var/log/mail.log` logfile. You should see:

... dovecot: Dovecot v1.2.15 starting up (core dumps disabled)
... dovecot: auth-worker(default): mysql: Connected to 127.0.0.1 (mailserver)

Before you send a first test email you will need to fix file system permissions for the `/etc/dovecot/dovecot.conf` file so that the *vmail* user can access the Dovecot configuration. The reason is that Postfix starts the delivery agent with *vmail* permissions:

```
chgrp vmail /etc/dovecot/dovecot.conf
chmod g+r /etc/dovecot/dovecot.conf
```

We should also make sure that only root can access the SQL configuration file so nobody else is reading your database access passwords:

```
chown root:root /etc/dovecot/dovecot-sql.conf
chmod go= /etc/dovecot/dovecot-sql.conf
```

Make Postfix talk to Dovecot

- Add new comment
- 137090 reads

In a previous chapter we made sure that Postfix knows which emails it's allowed to receive. Now what to do with the email? It has to be stored to disk. Usually that's done by Postfix itself which comes with a very basic mail delivery agent (MDA) called "virtual" that just saves incoming emails to virtual mailboxes on your hard disk. But as we will use Dovecot (for IMAP and POP3 access) anyway we can use its more featureful "*local delivery agent*" (also known as "Dovecot LDA"). It even allows you to use rules to run different automatic actions on incoming email. To make Postfix use that agent you will have to add a service to your `/etc/postfix/master.cf`:

```
dovecot unix - n n - - pipe
  flags=DRhu user=vmail:vmail argv=/usr/lib/dovecot/deliver -f ${sender} -d ${recipient}
```

(Note: the second line has to be indented by spaces!)

Restart Postfix:

```
postfix reload
```

Also make Postfix use that service for virtual delivery by running these two commands in the shell:

```
postconf -e virtual_transport=dovecot
postconf -e dovecot_destination_recipient_limit=1
```

So now Postfix will pass on incoming emails to virtual users to the `/usr/lib/dovecot/deliver` program.

Testing email delivery

- Add new comment
- 117066 reads

At this point the /var/vmail directory should be empty yet. You can get a list of all files and directories within by running:

```
find /var/vmail
```

There is probably nothing except perhaps a "lost+found" directory if /var/vmail is on a seperate partition.

Let's now try to send an email to the user john@example.org:

```
echo test | mail john@example.org
```

If everything worked as expected Postfix has accepted the email and forwarded it to Dovecot which in turn wrote the email in John's maildir. Look again:

```
find /var/vmail
```

You should see something like:

```
/var/vmail/
/var/vmail/example.org
/var/vmail/example.org/john
/var/vmail/example.org/john/Maildir
/var/vmail/example.org/john/Maildir/dovecot-uidvalidity
/var/vmail/example.org/john/Maildir/tmp
/var/vmail/example.org/john/Maildir/dovecot.index.log
/var/vmail/example.org/john/Maildir/dovecot-uidlist
/var/vmail/example.org/john/Maildir/dovecot-uidvalidity.4daa0b32
/var/vmail/example.org/john/Maildir/new
/var/vmail/example.org/john/Maildir/new/1302989618.M190758P1940.debian,S=376,W=386
/var/vmail/example.org/john/Maildir/cur
```

Your files may have slightly different numbers. So John has a new email in his inbox. Your /var/log/mail.log will look something like:

```
postfix/pickup[1788]: 2A907A8C: uid=0 from=<root>

postfix/cleanup[1936]: 2A907A8C: message-id=<20110416213338.2A907A8C@myserver>

postfix/qmgr[1789]: 2A907A8C: from=<root@myserver>, size=306, nrcpt=1 (queue active)
```

dovecot: deliver(john@example.org): msgid=<20110416213338.2A907A8C@myserver>: saved mail to INBOX

postfix/pipe[1939]: 2A907A8C: to=<john@example.org>, relay=dovecot, delay=0.03, delays=0.02/0/0/0.01, dsn=2.0.0, status=sent (delivered via dovecot service)

postfix/qmgr[1789]: 2A907A8C: removed

If anything went wrong then carefully check the last lines of your /var/log/mail.log. It will very likely point you to the problem.

(If you are curious how to send an email to your mail server using a manual SMTP session then read the respective section in the Lenny tutorial. It's also good as an additional test because the "mail" command bypasses a few of Postfix's security features.)

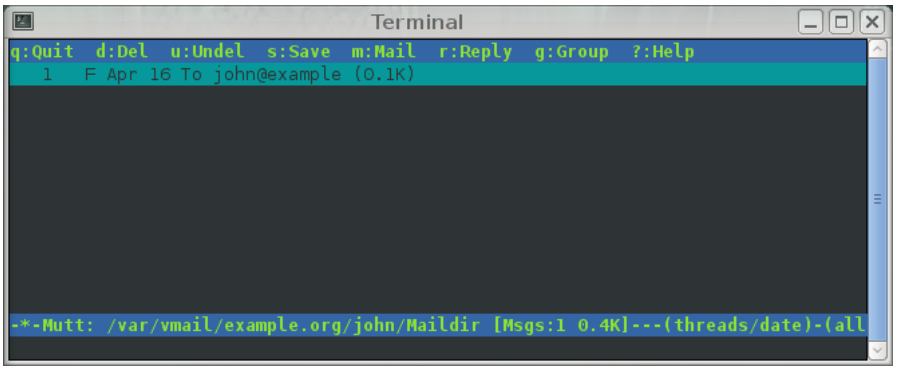
Access the email on disk

Apparently everything went well. To read the email from John's inbox you can use either POP3, IMAP or access the maildir directly. The latter can be done using the console-based "mutt" email client:

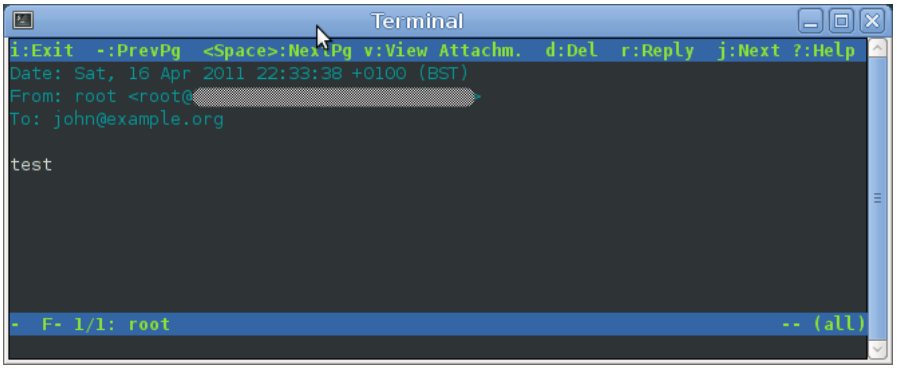
mutt -f /var/vmail/example.org/john/Maildir

(You may get asked to create /root/Mail - this is standard procedure. Just answer "yes" or press Enter.)

What you see now are the contents of John's mailbox:



Press enter and you can read the email number 1:



So there is your test email. Press "q" to quit "mutt".

Access the email using IMAP

Actually we just cheated a little as we have accessed John's inbox directly on disk. A better test is to use POP3 or IMAP. And fortunately "mutt" supports IMAP:

mutt -f imap://john@example.org@localhost

You may be prompted to confirm that you are connecting to a mail server with an untrusted SSL certificate. That's okay. In the end you should see the index and email just like in the screenshots above. That worked? Great. Otherwise check your /var/log/mail.log for error messages.

If you still can't get enough and want to run a manual POP3 and IMAP session using TELNET then check out the Lenny tutorial on a complete example.

POP3 versus IMAP

If you wonder what the difference between POP3 and IMAP is:

- *POP*3 (Post Office Protocol) is a simple protocol that lets you fetch email from a single mailbox. It is usually used to collect all emails, though you can also leave them on the server but this is a bit of a hack and you can't create multiple folders on the server to sort your mail. It saves space on the mail server because the email gets moved to the user's hard disk on their computer. But they won't be able to access the same email from another computer. Besides you cannot create multiple folders on the server to sort your mail. There is just the inbox. This variant is antiquated and not exactly user-friendly.
- *IMAP* (Internet Messaging Application Protocol) is predominantly focused upon leaving your mail on the server but you can also collect it like POP3. The inbox is where your incoming emails are stored but users can also maintain folders and move emails to them. But users can move emails to different directories. IMAP is useful when you want to access your email from different locations without losing mail because you fetched it from another location. The drawback is that lazy users leave their mail on the server thus filling up your server's hard disk (unless you use quotas).

Authenticated SMTP

- Add new comment
- 103805 reads

Relaying

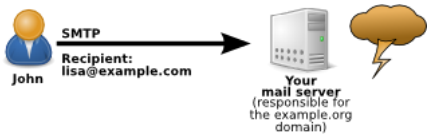
Before we dive into SMTP authentication I want you to understand what *relaying* actually means. When Postfix receives an email and needs to forward it to another server this is called *relaying*.

Incoming email

When someone on the internet sends an email to john@example.org some other mail server will deliver the email using SMTP to your mail server. Postfix will determine that it's responsible for email addresses in the example.org domain and accept the email. John can then use POP3 or IMAP to fetch the email from your server.

Outgoing email (without authentication)

John is on the internet somewhere and wants to send an email to lisa@example.com. As your mail server is not responsible for the "example.com" domain it would have to forward the email to the responsible mail server. So your server receives John's email and forwards (relays) it to the mail server that is responsible for ...@example.com email addresses. This may seem like a harmless case your mail server will deny that:



Why? Because anyone can claim to be John and make your mail server forward mail. If an attacker (like a spammer) would send millions of spam emails through your server then other organisations will accuse *you* of spamming. Your mail server would be what people call an "open relay". This is not what you want because your mail server's IP address would get blacklisted and you will have serious trouble ever sending out mail again. So without any proof that John is actually John your server will reject the email.

Outgoing email (with authentication)

So how does John send his email? He needs to use *authenticated* SMTP. This is similar to the previous case but John's email program will send his username and password first.

mynetworks

In addition to using SMTP authentication you can tell Postfix to always relay email for certain IP addresses. The mynetworks setting contains the list of IP networks or IP addresses that you trust. Usually you define your own local network here. The reason John had to authenticate in the above example is because he is not sending the email from your local network (usually).

Enabling SMTP authentication in Postfix

Authenticated SMTP with Postfix has been a hassle in the past. It was done through the SASL (*Simple Authentication and Security Layer*) library that was once part of the Cyrus mail server. It was nearly impossible to debug and threw error messages that were gibberish and misleading. Fortunately nowadays we can make Postfix ask the Dovecot server to verify the username and password. And as you already configured Dovecot's authentication part this is really easy now. Postfix just needs some extra configuration.

```
postconf -e smtpd_sasl_type=dovecot
postconf -e smtpd_sasl_path=private/auth
postconf -e smtpd_sasl_auth_enable=yes
postconf -e smtpd_recipient_restrictions=" \
  permit_mynetworks \
  permit_sasl_authenticated \
  reject_unauth_destination"
```

`smtpd_sasl_auth_enable` enables SMTP authentication altogether. And the `smtpd_recipient_restrictions` define rules that are checked after the remote user sends the `RCPT TO:` line during the SMTP dialog. In this case relaying is allowed if:

- `permit_mynetworks`: the user is in the local network (`mynetworks`) or
- `permit_sasl_authenticated`: if the user is authenticated or
- `reject_unauth_destination`: the mail is destined to a user of a domain that is a local or virtual domain on this system (`mydestination`, `virtual_alias_domains` or `virtual_mailbox_domains`).

There are further restrictions (`smtpd_client_restrictions`, `smtpd_helo_restrictions`, `smtpd_sender_restrictions`) that get checked during the different states of the SMTP dialog (IP connection, HELO/EHLO command, MAIL FROM command) but for now you should put all restrictions into the `smtpd_recipient_restrictions`.

If you are curious to see how an authenticated SMTP session works on a TCP level then the Lenny tutorial has information on that.

Proper SSL certificates for Postfix and Dovecot

- Add new comment
- 187546 reads

So far you will have received warning on the SSL certificates you use for Postfix, Dovecot and the RoundCube email web interface. SSL/TLS is a great way to automatically encrypt the passwords between the email user and your mail server. So you want to have proper certificates. There are three ways you can handle your certificate:

Either: Leave it like it is

The users will receive a warning that the certificate is invalid and likely does not even match the name of the mail server.

- Advantage: lazy, no costs
- Disadvantage: some email clients will refuse the certficate because not even the server name matches
- Conclusion: only do this if you don't care

Or: Create a self-signed certificate

This will at least give your users a certificate with with the proper name of the mail server.

- Advantage: little work, no costs
- Disadvantage: you train your users to accept any certificate (even a malicious one)
- Conclusion: do this if you have a very small group of users who you can tell about the certificate

Creating SSL certificates can be tricky due to the syntax of the "openssl" command line tool that often reminds me of text adventures.

Dovecot

Here comes the command to create a Dovecot certificate:

```
openssl req -new -x509 -days 3650 -nodes -out /etc/ssl/certs/dovecot.pem -keyout /etc/ssl/private/dovecot.pem
```

What you enter in the fields is entirely your choice. The only notable exception is the "Common Name" which has to be exactly the name of your server in the way that users will access it. So if you tell your users to access your mail server at "mail.example.org" then this has to be entered here. This certificate will be valid for 10 years (10 times 365 days).

Do not forget to set the permissions on the private key so that no unauthorized people can read it:

```
chmod o= /etc/ssl/private/dovecot.pem
```

And you will have to restart Dovecot to make it read your new certificate:

```
/etc/init.d/dovecot restart
```

Postfix

To create a certificate to be used by Postfix use:

```
openssl req -new -x509 -days 3650 -nodes -out /etc/ssl/certs/postfix.pem -keyout /etc/ssl/private/postfix.pem
```

Do not forget to set the permissions on the private key so that no unauthorized people can read it:

```
chmod o= /etc/ssl/private/postfix.pem
```

You will have to tell Postfix where to find your certificate and private key because by default it will look for a dummy certificate file called "ssl-cert-snakeoil":

```
postconf -e smtpd_tls_cert_file=/etc/ssl/certs/postfix.pem
postconf -e smtpd_tls_key_file=/etc/ssl/private/postfix.pem
```

Or: Use a free certification authority

- Advantage: a little work, no costs
- Disadvantage: these certification authorities are not included in all email clients
- Conclusion: if you don't want to spend money then this is your best chance

There are barely any free services like that. I have had good experience with StartSSL although their web site is sometimes confusing. Once you have created a key file and certificate then use these files as shown in the previous section about creating self-signed certificates.

Or: Buy an SSL certificate

- Advantage: certificate will be accepted automatically by the user's email program
- Disadvantage: you throw a lot of money at the certification mafia that doesn't deserve it
- Conclusion: do this if you will run a professional public mail server

Honestly I dislike any commercial certification authority I have been in contact with. So choose the lesser evil yourself.

DNS - to make mail servers find you

- Add new comment
- 90938 reads

About MX entries

So now you have your working mail server. But how do emails find you? The answer lies in the most important service on the internet: DNS. Assume that you are the owner of the gmail.com domain. And a mail server somewhere on the other end of the internet wants to send an email to john@gmail.com. What the other mail server needs to do is find out which server on the internet it will have to establish an SMTP connection to in order to deliver the email. It does it by querying the DNS name server responsible for the "gmail.com" domain for an MX or alternatively an A record. MX stands for "mail exchanger" and is a set of records telling which mail server(s) to use. Let's run a query:

```
$> host -t MX gmail.com
gmail.com mail is handled by 10 alt1.gmail-smtp-in.l.google.com.
gmail.com mail is handled by 20 alt2.gmail-smtp-in.l.google.com.
gmail.com mail is handled by 30 alt3.gmail-smtp-in.l.google.com.
gmail.com mail is handled by 40 alt4.gmail-smtp-in.l.google.com.
gmail.com mail is handled by 5 gmail-smtp-in.l.google.com.
```

So as a result we get 5 different MX records. Each of them consists of a priority and the host name of the mail server. A mail server would pick the entry with the highest priority (=the lowest number) and establish an SMTP connection. In this example that would be the priority 5 server gmail-smtp-in.l.google.com. If that server could not be reached then the next best server with priority 10 would be used and so on. So all you have to do in your own DNS zone is add an MX entry pointing to your mail server. If you want to run a backup mail server (which is outside of the scope of this tutorial) then you can add a second entry with a lower priority.

A mistake some people make is using an IP address in MX records. That is not allowed. An MX record always points to a host name. You will have to add an A record to point to the actual IP address of your mail server.

(In the above example it is very unlikely that a mail server will ever have to use the server with priority 40. Adventurous system administrators can add such a low-priority entry and see who connects to it. Because an interesting fact is that spammers often try these servers first - hoping that it is just for backup purposes and less restrictive than the main server. If you see someone connecting to the lowest-priority address first without having tried a higher-priority mail server then you can be pretty certain that it's not a friend who's knocking at your door.)

Fallback to A entries

It's always best to explicitly name mail servers in the MX records. If you can't do that for whatever reason then the remote mail server will just do an A record lookup for the IP address and then send email there. If you just run one server for both the web service and the email service then you can do that. But if the web server for your domain is located at another IP address than your mail server then this won't work.

Dynamic DNS for your home mail server

What - you don't have a domain? No problem. With DynDNS you can get a server name on the internet for free. Get an account there, set up a hostname and make it point to your IP address. If you are on a dynamic IP address you can have your DNS record updated automatically by using programs like "ddclient" and still receive email on that server/domain name. You can even use an MX record that is different from the A record if you like. So there is no excuse for not having your own mail server at home.

SMTPd restrictions, SPF, DKIM and greylisting

- Add new comment
- 97043 reads

Postfix provides the ability to apply filters during the SMTP session. When an email arrives at your mail server the following steps are usually run through:

- *CLIENT*: Incoming TCP connection on port 25
- *HELO*
  The sending mail server tells you its name.
- *MAIL FROM*
  The sender's name and email address of the email
- *RCPT TO*
  The recipient the mail is intended for
- *DATA*
  In this phase of the SMTP session the actual email is transmitted
- QUIT
  End of SMTP session

With the appropriate settings of SMTPd restrictions you can control what checks Postfix will run when an email is received. Those restrictions are defined in the /etc/postfix/main.cf configuration file and are called smtpd_..._restrictions. You can for example restrict certain IP addresses to even open an SMTP (TCP/25) connection to your mail server during the "client" phase. Or you can filter what recipients are allowed during the "rcpt to" phase. According to the above list of phases these restrictions apply:

- smtpd_client_restrictions (default: empty)
- smtpd_helo_restrictions (default: empty)
- smtpd_sender_restrictions (default: empty)
- smtpd_recipient_restrictions (default: permit_mynetworks, reject_unauth_destination)
- smtpd_data_restrictions (default: empty)

So by default Postfix only applies checks during the "RCPT TO" phase to check if the email is either coming from your local network (the sending server's IP address is part of the "mynetworks" setting) or if the recipient has a valid account on your mail server. Otherwise the email would get rejected. It is important that you understand that the restrictions are checked in the respective phase. So it's pointless to apply checks on the sender's email address (MAIL FROM) in the HELO phase because that information is not yet known at that phase. Many mail server administrators just put all the checks into the smtpd_recipient_restrictions. The following sections give you some advice on how to configure your restrictions

properly.

Realtime blacklists (RBL)

The most important technique nowaways to fight spammers are IP blacklists. There exist dozens of these lists that you can use freely. The operators of such lists have different policies regarding who is getting blacklisted. But most of them have spam probes on the internet that spammers fall for. The technique of spam traps means that otherwise unused email addresses are put on a web site in a hidden place or use them on mailing lists. Spammers then automatically spider (=scan) the internet for email addresses and then send their spam to the found addresses. If such a trap account gets email then that information is used by the operators of an IP blacklist to build a list of server that send spam. It is potentially dangerous to use IP blacklists though because you might block email from a legit server if that server was inadvertently blacklisted. My favorite blacklist "SORBS" for example has let me down several times by blacklistting entire internet providers for a period of hours or even days. So it's a good idea to check their policies and see if you agree with it. In the end the risk is yours. Once you reject an email you cannot get it back. But I still strongly recommend you use blacklists or you will literaly drown in spam.

IP blacklists are specified by a domain name like "bl.spamcop.net". To check if a certain IP address is listed in that zone you need to check if an entry for the reversed IP address exists in that zone. Assume that a mail server with the IP address 217.217.34.231 wants to send you an email. If you told Postfix to use the "bl.spamcop.net" blacklist then Postfix will reverse this IP address and check the DNS entry of 231.34.217.217.bl.spamcop.net. I just did that and got a result:

```
$> host 231.34.217.217.bl.spamcop.net
231.34.217.217.bl.spamcop.net has address 127.0.0.2
```

The fact that a result (127.0.0.2) was found means that the IP is blacklisted and that I should not accept emails from it. Postfix will then tell the sender that it triggered a blacklist and cancels the connection. The result returned almost always is 127.0.0.x where the X stands for the reason of the blacklisting. You will have to read the policy of the respective blacklist to see the reason. Let's take another IP address that is not blacklisted - for example 85.214.93.191:

```
$> host 191.93.214.85.bl.spamcop.net
Host 191.93.214.85.bl.spamcop.net not found: 3(NXDOMAIN)
```

I did not get a result. So the IP is not blacklisted and I should accept email from it.

There are a few major blacklists that you should check out:

- SORBS (dnsbl.sorbs.net)
  SORBS is very efficient but every few months tends to blacklist even large ISPs accidentally. I currently do not use it as I had too many user complaints.
- SpamCop (bl.spamcop.net)
  Pretty reliable. On a sample day it blocked 29 mails on my server.
- SpamHaus (zen.spamhaus.org)
  Also pretty reliable. On a sample day it blocked 147 mails on my server.
- UCEprotect (dnsbl-1.uceprotect.net)
  Careful - the Class-2 and Class-3 lists are too strict and block even legit senders. On a sample day it blocked 33 mails on my server.

If you used these blacklists then the restrictions in your /etc/postfix/main.cf would be these:

```
smtpd_recipient_restrictions =
  permit_mynetworks
  reject_rbl_client dnsbl.sorbs.net
  reject_rbl_client bl.spamcop.net
  reject_rbl_client zen.spamhaus.org
  reject_rbl_client dnsbl-1.uceprotect.net
  reject_unauth_destination
```

(Note: You can also list the restrictions all in one line and seperated by commas. But I prefer the way to use multiple lines and indent the subsequent lines. It makes the main.cf file more readable.)

With these restrictions in place Postfix will run each of these checks after the "RCP TO" phase during the SMTP dialog.

1. Is the sending IP address in your network range?
2. Is it blacklisted by SORBS?
3. Is it blacklisted by SpamHaus?
4. Is it blacklisted by UCEprotect?
5. Is your mail server responsible for the recipient's domain? Or is the sender using SMTP authentication for relaying?

Further restrictions to consider

Postfix offers quite a lot of checks you can use in your smtpd_…_restrictions. Just to name a few that I use on my mail servers:

- reject_unknown_client_hostname
  Runs a two-way DNS check of the IP address and hostname of the sending server. This can be potentially dangerous and reject legit email from misconfigured mail servers but nowadays every decent mail server is supposed to take care of their DNS settings.
- reject_unknown_sender_domain
  Checks if the domain used in the sender's email address actually exists. This runs a check for an MX and A record of the sending domain.
- reject_unauth_pipelining
  Some spam bots just try to deliver emails as fast as possible. But formally the sending server must wait for the recipient server's response in each SMTP step. If the recipient server allows pipelining then fast sending is allowed. This setting drops the connection if pipelining is attempted before it was offered by the destination server.

Further settings can be found in the Postfix documentation.

Are you blacklisted?

Of course you want to send out emails through your mail server. So you need to make sure that your server IP address is not blacklisted either. I suggest you check your server's IP address through one of the blacklist test services:

- http://www.anti-abuse.org/multi-rbl-check/
- http://rbl-check.org/

(These services also give you an idea what other blacklists are available.)

If you are listed on any of these lists then you will want to get off the lists. How you do that depends on each blacklist's policy. Some don't even support manual removal from these lists. If your ISP assigned you a dynamic IP address then you are in a so called "dialup IP range" and are most likely blacklisted and it's pointless to get off the list - read here for a solution. A lot of spam is coming from dynamic IP addresses. Either spammers send email from their ISP account. Or the large amount of virus-infested Windows PCs in the world is forwarding spam. So dynamic IP addresses are frowned upon by mail server administrators.

Source spoof prevention

The above section on IP blacklists helped you fight spam by rejecting it. Now let's add another security measure to prevent *source address spoofing*. *Spoofing* means that someone fakes information. And the *source address* is the email address that appears to be the sender of an email. So it means that somebody pretends to be someone else by faking the email address of the sender. Maybe you haven't given it much thought but basically everyone can send an email and pretend to be someone else. Perhaps you are even using multiple sender addresses yourself. What prevents someone from pretending he is you? There are two common options:

1. SPF (Sender Policy Framework): Using SPF anyone can find out if an email was sent from an IP address (=mail server) that is allowed to act as a mail server for that sender domain. Technically with SPF you create a DNS entry for your domain and list the valid IP addresses there that are allowed to send email on your domain's behalf.
2. DKIM (DomainKeys Identified Mail): DKIM uses a private/public key pair to sign all outgoing emails automatically. By making your domain's public key available through DNS every mail server on the internet can check if the signature is valid for your domain and nobody spoofed the email. Although SPF is more widespread than DKIM it has a major design flaw because it fails when you forward email (as the sending IP address changes and may not be a valid address listed in the SPF entry any more).

Setting your own SPF entry

The content of the SPF entry follows a certain syntax. To make things easy you can use the service at OpenSPF to set up your SPF entry. Enter your domain name there and fill out the fields. The web site will tell you how your SPF entry will have to look like. Add this to your DNS zone and you will both help others to fight spam as well as help yourself to prevent that anyone is sending unwanted emails on your behalf. Formerly a TXT resource record in the DNS zone was used but nowadays there is a distict SPF record type used for that purpose.

Checking other SPF entries

Many domains already have SPF entries. So you are encouraged to use that information for your own spam and phishing protection. Read the next section on greylisting to learn how to use the "tumgreyspf" service to use both the power of SPF and greylisting.

Forwarding breaks SPF

SPF has a major design flaw. It fails if someone forwards email to another server. Then the forwarding server is not listed in the SPF entry and the destination mail server will (correctly) reject the email. Less technically-savy users will not understand this limitation. So many mail administrators just take the SPF checks into account when looking for spam but they don't ultimately reject the email. It's up to you. SPF reliably fights spoofing and phishing for major domains like PayPal, eBay, Amazon or other large organisations that are often a target of phishers. In some cases it will just be overly careful.

Checking DKIM signatures of incoming email

DKIM was invented for a similar reason as SPF. It provides email authentication - to allow a receiving mail server to check that the email is authentic for a given sender domain. SPF just tells which IP are allowed to send email for a sender domain. But DKIM is working way more reliably because it adds a cryptographic signature to every outgoing email. To verify if an email is authentic a mail server can use the email's signature and match it with the sender domain's public key that can be obtained by DNS. Sounds too complicated? Actually it's pretty simple if you understand the basics of public-key cryptography. Have you ever used PGP? Then you know that you had to create a keypair consisting of a private key you can use to sign emails and a public key that you can give others to encrypt emails for you or check your signatures. DKIM uses exactly that technique. The sending mail server has a keypair for each domain. When it sends out an email it will use the private key for the sender domain, take the email that is to be sent out and sign it with a cryptographic signature. That signature is added to the email (as an additional mail header) and the receiving mail server can take the email and the signature from the mail header and the public key of the sender domain to determine whether the signature was valid.

Adding automatic verification of other domains' DKIM signatures is pretty easy. Just install the DKIM filter software:

```
apt-get install dkim-filter
```

Actually this software is called a "milter". Milters are "mail filters" that were introduced as additional pieces of software for the ancient Sendmail mail server software. Milters speak a certain protocol that mail server can use to communicate with it and Postfix fortunately knows how to use milters, too. Right after the installation the "dkim-filter" process is already started and running in the background. But as Postfix is running in a chroot jail in /var/spool/postfix it cannot use the DKIM filter's communication socket at its default location in /var/run/dkim-filter/dkim-filter.sock. You can fix the permissions and make DKIM run a socket in /var/spool/postfix instead but I prefer to rather run dkim-filter as a TCP service rather than a socket. Edit your /etc/default/dkim-filter file that contains its default settings and add

```
SOCKET="inet:54321@localhost"
```

to it. Restart the dkim-filter process:

```
/etc/init.d/dkim-filter restart
```

Now dkim-filter is listening to TCP port 54321 on the localhost interface. To make Postfix use this milter run these commands to add the appropriate milter definitions to your /etc/postfix/main.cf file:

```
postconf -e smtpd_milters=inet:127.0.0.1:54321
postconf -e non_smtpd_milters=inet:127.0.0.1:54321
```

Now Postfix will run all emails (both incoming and outgoing) through your DKIM filter and check DKIM signatures (if found) on incoming emails. It will not sign outgoing emails though - this is covered in the next section.

A side note: you can decide what should happen if there is a problem with the DKIM filter. By default Postfix will temporarily reject emails until you fixed the issue. It's up to you. But I prefer to just accept emails if there is a milter problem. So I set:

```
postconf -e milter_default_action=accept
```

For more information on milters please consult the Postfix documentation on milters.

Setting up DKIM for sending emails

The actual idea of DKIM is to make a domain's mail server sign outgoing emails automatically. To do that you need to create a cryptographic keypair first. Fortunately you do not have to buy a certificate because publishing the key in your DNS zone is enough. The tool you need is "dkim-genkey" and it requires the bit-length of the key and the domain name. You can optionally use a selector to use multiple keys. By default the selector is called "default". I suggest you use the domain name as your selector which will name your key files correctly automatically.

It's best to create a seperate directory to store the keys. I keep mine in /etc/postfix/dkim:

```
mkdir /etc/postfix/dkim
cd /etc/postfix/dkim
```

Then create the keypair there:

```
dkim-genkey -b 1024 -d example.org -s example.org
```

In your current directory you will now find two files.

- "example.org.private" is the private key that the DKIM milter will use to sign all outgoing emails
- "example.org.txt" is the public key that you are supposed to add to your domain's DNS zone

The configuration file for the DKIM milter is /etc/dkim-filter.conf. Edit that file. The only change you need is to add a line for the key list file like:

```
KeyList     /etc/dkim-keys.conf
```

Create a new file /etc/dkim-keys.conf that you just pointed to. Its contents define which key is used for which sender address. Or to be specific the syntax is:

```
sender-pattern:signing-domain:keypath
```

as pointed out in the "man dkim-filter.conf" man page. So if your domain were example.org you would write:

```
*@example.org:example.org:/etc/postfix/dkim/example.org
```

Add one line per (sender) domain that you wish to sign outgoing emails for.

Caveat: dkim-filter adds ".private" to the keypath you specify. So the above line is correct. Do not specify /etc/postfix/dkim/example.org.private or signing will fail.

Restart your DKIM milter:

```
/etc/init.d/dkim-filter restart
```

Now outgoing emails should get a DKIM signature automatically. Send an email to another mail server and check the headers of the email. You should see a header like this:

```
DKIM-Signature: v=1; a=rsa-sha256; c=simple/simple; d=example.org; s=example.org t=1307571770;
bh=OtZOIOSkkhL1t+kR5KOE6HvvjUjLkDE+70agcKmcjJg=; h=Message-ID:Date:From:MIME-Version:To:Subject:Content-Type:
Content-Transfer-Encoding;
b=RlxE9IRiOJrOzM4JbJFIp/YmK4XBRJFti79rL0vzGkppAC1AJi2CdtwGFT/sTovt/iKikrdyQ7M7JhdlC9u3bh8rrvhLA53HZCmu6WvHvv059ysdpjUl
```

The meaning of each parameter can be found on the Wikipedia page. or in the less human-friendly format definition in the RFC 4871.

However the recipient cannot tell if this is a correct signature because they don't have the public key to verify this signature with. Edit your DNS zone for your domain and just add the TXT record that "dkim-genkey" created for you in /etc/postfix/dkim/example.org.txt. On my test server it looked like this:

```
example.org._domainkey IN TXT "v=DKIM1; g=*; k=rsa;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDKPFhCADCGcxDchGHFqvSoQYuW0epBT0gJuCwDMkX0uPejeeMdBNDcaGo8AyZ
; ----- DKIM example.org for example.org
```

Verify that your DNS zone for your domain really contains the right entry by querying for a TXT record with your domain name and the selector you used:

```
dig +short example.org._domainkey.example.org txt
```

And it should return your public key:

```
"v=DKIM1\; g=*\; k=rsa\;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDKPFhCADCGcxDchGHFqvSoQYuW0epBT0gJuCwDMkX0uPejeeMdBNDcaGo8AyZ
```

Now all outgoing emails for …@example.org will get a DKIM signature added automatically and recipients can verify them. Any mail server relying on DKIM signature checks can now check whether emails from your domain are legit.

I assume that you have set up the DKIM milter in Postfix as described in the previous section. Setting "smtpd_milters" and "non_smtpd_milters" in your /etc/postfix/main.cf will now do both the verification of DKIM signatures as well as the signing.

Greylisting

Another way to help you reduce the amount of spam you receive is using *greylisting*. Whitelisting means that a certain kind of email is always accepted (delivery status 2.x.x). Blacklisting is the opposite and always blocks certain emails (delivery status 5.x.x). Greylisting is something in between and temporarily refuses to accept the email for a couple of minutes. The idea is to tell virus infested computers trying to spam you from

actual mail servers. Viruses (or rather: spam worms) just try to deliver a spam email once. But real mail servers have a queue of outgoing emails and upon a temporary error they will try to re-deliver the email. If the sending mail server has never sent us an email then greylisting will send back a temporary error code (delivery status 4.x.x) which signals the sending server to try delivering the email again in a few minutes. Your server will take a note of the IP address of the sending server and after a set period (a common value is 10 minutes) the blockade is lifted and upon the next delivery attempt the email will be accepted. Also the greylisting service will take a note in its cache and further delivery attempts will be allowed instantly.

The obvious drawback is that emails are often delayed by several minutes - especially if you just start using greylisting and the cache of trusted sending servers is empty. It is your decision whether your users will accept that. In my personal experience greylisting is not as effective nowadays as it has been in the beginning. It appears like malware now just tries to deliver spam emails several times. But it's one measure to maybe block some spam emails.

There are several ways to implement greylisting. The way I do it is using the "tumgreyspf" software. Start by installing it:

```
apt-get install tumgreyspf
```

To make Postfix use tumgreyspf you have to define a service for it in the /etc/postfix/master.cf:

```
tumgreyspf unix  -       n       n       -       -       spawn
   user=tumgreyspf argv=/usr/bin/tumgreyspf
```

You also need to add it to your smtpd_recipient_restrictions of your /etc/postfix/main.cf file. Example:

```
smtpd_recipient_restrictions =
   permit_mynetworks
   permit_sasl_authenticated
   reject_rbl_client bl.spamcop.net
   check_policy_service unix:private/tumgreyspf
   reject_unauth_destination
```

These are example restrictions. Do not just copy/paste it. The example is just meant to show you a good place to add the policy check in your smtpd_recipient_restrictions.

Finally restart Postfix:

```
postfix reload
```

To customize the behavior of tumgreyspf you can alter the files in /etc/tumgreyspf.

(This procedure is also documented in the /usr/share/doc/tumgreyspf/README.Debian file.)

Final relay test

Do you remember what I said earlier about relaying? You must not become an open relay or else you would be considered a supporter of spammers and be blacklisted. There is a pretty simple way to run a couple of checks. Log into your mail server and run:

```
telnet relay-test.mail-abuse.org
```

Wait a few seconds. The remote service will connect back to your server and run a few sanity checks. If after running through all the check you see

```
System appeared to reject relay attempts
Connection closed by foreign host.
```

then you configured your mail server safely. Well done.

Optional: Content scanning with AMaViS

- Add new comment
- 72892 reads

In another chapter we already dealt with restrictions during the SMTP dialog. You learned about real-time black lists (RBLs) which help you block most of the incoming spam. If you are still getting too much spam and want to do more then content-scanning can help you. A content scanner looks into the actual email instead of just doing simple checks during the SMTP session. Unfortunately content-scanning with Postfix is rather complicated. However the software of choice here is AMaViS - short for "a mail virus scanner". It's a software written in Perl that uses additional software like SpamAssassin (for spam detection) and ClamAV (virus scanning) as well as many other virus scanners.

Advantages:

- can detect spam and block the email or mark it as spam
- can detect viruses and block infested emails
- supports automatic DKIM signing

Disadvantages:

- uses a lot of CPU (lowers your delivery rate by factor 5 to 10)
- spam scanning only works globally (an individual user cannot train the spam filter to match their habits)
- using content filters with Postfix works but is error prone

To understand how Postfix and AMaViS work together please take a look at the big picture again. When Postfix passes an email to AMaViS it uses SMTP on a non-standard port - 10024 by default. AMaViS scans the email and if it's successful it will then pass it back to Postfix on another port - 10025 by default.

Installation

Install AMaViS and its suggested packages:

```
apt-get install amavisd-new spamassassin clamav clamav-daemon arj zoo nomarch cpio lzop cabextract apt-listchanges libnet-ldap-perl
libauthen-sasl-perl  libdbi-perl libmail-dkim-perl p7zip rpm unrar-free libsnmp-perl
```

This setup is complex and can go wrong. So if you are adding content filtering to your live server I strongly suggest you enable the *soft bounce* feature in Postfix. If Postfix intends to reject or bounce an email then this feature will rather keep the mail in the queue and try again later. Whenever you do major changes in your mail server setup I recommend you enable it:

```
postconf -e soft_bounce=yes
```

As explained above AMaViS will accept emails on TCP/10024 and re-inject them to Postfix on TCP/10025. By default Postfix doesn't listen on port 10025. Neither does it know of AMaViS. So first you need to edit /etc/postfix/master.cf and add these two services:

```
smtp-amavis unix -      -       n     -       2  smtp
   -o smtp_data_done_timeout=1200
   -o smtp_send_xforward_command=yes
   -o disable_dns_lookups=yes
   -o max_use=20

127.0.0.1:10025 inet n    -       n       -       -     smtpd
   -o content_filter=
   -o smtpd_delay_reject=no
   -o smtpd_client_restrictions=permit_mynetworks,reject
   -o smtpd_helo_restrictions=
   -o smtpd_sender_restrictions=
   -o smtpd_recipient_restrictions=permit_mynetworks,reject
   -o smtpd_data_restrictions=reject_unauth_pipelining
   -o smtpd_end_of_data_restrictions=
   -o smtpd_restriction_classes=
   -o mynetworks=127.0.0.0/8
   -o smtpd_error_sleep_time=0
   -o smtpd_soft_error_limit=1001
   -o smtpd_hard_error_limit=1000
   -o smtpd_client_connection_count_limit=0
   -o smtpd_client_connection_rate_limit=0
   -o receive_override_options=no_header_body_checks,no_unknown_recipient_checks,no_milters
   -o local_header_rewrite_clients=
```

(Note that all lines but the first lines of each service need to be indented by spaces.)

Restart Postfix to make it pick up these two new services:

```
postfix reload
```

And you need to tell Postfix to use the "smtp-amavis" service as a content filter. Making Postfix forward emails to AMaViS is done by setting the content_filter setting. Also set the "receive_override_options" setting that will be explained later by running these shell commands:

```
postconf -e content_filter=smtp-amavis:[127.0.0.1]:10024
postconf -e receive_override_options=no_address_mappings
```

The first line is probably obvious. All email is forwarded to the "smtp-amavis" service defined in the /etc/postfix/master.cf using TCP port 10024 on the IP address 127.0.0.1 (localhost).

The second line is less obvious. But it's important or you risk delivering email duplicates to your users. The reason is that Postfix processes an incoming email twice. First when it comes from the internet. And second when it comes back from AMaViS after being content-scanned. Without disabling address mappings here the aliases (forwardings) would be checked twice. Of course an alias is supposed to be checked just once to avoid duplicate delivery. That's why you use this setting here.

Configuring AMaViS

AMaViS is configured using different files in /etc/amavis/conf.d. You may want to check what is set there. Above all you need to change the 15-content_filter_mode file to enable virus and spam scanning. Confusingly the default settings are:

```
#@bypass_virus_checks_maps = (
#   \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);

#@bypass_spam_checks_maps = (
#   \%bypass_spam_checks, \@bypass_spam_checks_acl, \$bypass_spam_checks_re);
```

To enable scanning you have to remove the "#" signs in front of these four lines. It sounds like it would then bypass the scanning but instead it enables it.

The default scanner is ClamAV which is a free and decent anti-virus software. To use it you will have to get a group membership right:

```
adduser clamav amavis
/etc/init.d/clamav-daemon restart
```

The configuration files in /etc/amavis/conf.d are run in sorted order. A setting in "50-user" will override the same setting in "01-debian". So I recommend you only change the existing (but empty) "50-user" file there. A couple of settings that you should consider:

$sa_spam_subject_tag

If AMaViS believes that an email is spam then it will change the subject to begin with this string. By default that string is "***SPAM***" which I found annoying. If you do not want to alter the subject you should set it to an empty string like this:

```
$sa_spam_subject_tag=undef;
```

Users can still find out if AMaViS flagged the email as spam by checking the X-Spam-Status header.

$sa_tag_level_deflt

Emails with a spam detection score greater or equal to this level will get spam headers added. Until you are happy with your AMaViS configuration I suggest you set it to a very low score ("undef" is the smallest possible value) so that headers are always added. Recommendation:

```
$sa_tag_level_deflt=undef;
```

$sa_tag2_level_deflt

Emails with a spam score greater or equal to this level will be marked as spam. Usually the default (6.31) is reasonable. Lower it if too much spam gets through. Raise it if you get too many regular mails caught as spam.

$sa_kill_level_deflt

If an email scores this level then AMaViS will act on the email according to the *$final_spam_destiny setting*. It should be set to the same value as $sa_tag2_level_deflt

$final_spam_destiny

I do not like the default D_BOUNCE setting here. The usual approach with spam is to flag an email as spam by adding email headers. Your users can then use these headers to decide if they want to sort an email into an extra folder or discard it. But if you bounced them you would hit some innocent person because spam mails never contain the correct sender address. Just let the user decide what to do with spam. So it is strongly recommended that you set this to D_PASS:

```
$final_spam_destiny=D_PASS;
```

There are a lot of further settings that you can use to tweak AMaViS to your own preferences.

Make sure that your 50-user file ends with "1;" or else AMaViS will not start up properly.

Restart AMaViS if you have made changes to the config files:

```
/etc/init.d/amavis restart
```

Make sure that AMaViS is listening on TCP port 10024:

```
netstat -nap | grep 10024
```

You should get this output:

```
tcp  0   0 127.0.0.1:10024     0.0.0.0:*    LISTEN   12345/amavisd
```

If you get such a line then AMaViS is running and waiting for incoming SMTP sessions. Otherwise check your /var/log/mail.log file - perhaps you have made a mistake in the configuration files.

This should get you started. For a more detailed documentation please see the README.Postfix that provided by AMaViS.

Why filter outbound email?

In former tutorials I used SQL queries to determine whether an email is incoming or outgoing. Many system administrators do not want to scan outgoing email for spam because:

- your own newsletter could be sent to your customers flagged as spam
- you are wasting CPU resources

But nowadays I would rather recommend to allow scanning any email because

- it helps determine a virus infestation of a local PC
- it helps AMaViS learn both ham (good email) and spam (unwanted email)

Ultimately you decide. There are different approaches described in the AMaViS documentation.

Testing

Now that everything is set up you will want to test your spam scanning process. Fortunately SpamAssassin comes with both a ham (good) and spam (bad) email for you to test with. You can send these emails into your mail system and watch the /var/log/mail.log file to see if AMaViS is doing the right thing.

First let's try the spam email:

```
sendmail john@example.com < /usr/share/doc/spamassassin/examples/sample-spam.txt
```

I addition to many lines in your /var/log/mail.log file dealing with "postfix" you should see one line dealing with "amavis". It will look like this:

```
May  7 12:42:33 debian amavis[1834]: (01834-01) Passed SPAM, <root@...> -> <john@example.org>, quarantine: T/spam-TrRe46tTm2+r.gz, Message-ID: <GTUBE1.1010101@example.net>, mail_id: TrRe46tTm2+r, Hits: 1002.448, size: 944, queued_as: 22DC9D40, 5022 ms
```

This shows that the email was accepted ("Passed") but flagged as spam ("SPAM"). The spam score is 1002.448 ("Hits") which is way above the limit I have set as $sa_tag2_level_deflt. Also the email was copied to /var/lib/amavis/virusmails/T/spam-TrRe46tTm2+r.gz in GZIPped format. To learn how to deal with quarantined emails please see the AMaViS documentation.

Nearly done

Did you enable *soft bounce* earlier? Everything works? If the

```
mailq
```

command shows that there are still emails in the queue that need to be delivered then re-queue them first:

```
postsuper -r ALL
```

Re-scheduling means that Postfix reconsiders what to do with every email in the queue. Transport (routing) and content filtering information stick to an email. So even if you reconfigure Postfix then emails in the queue would not pick that up until you re-queue them.

Now flush the queue

```
postfix flush
```

and the emails should get delivered. The /var/log/mail.log will give you information on what happens.

If everything works as you expect then switch off soft bounce mode again and you are done:

```
postconf -e soft_bounce=no
```

Want more?

If you seriously want to run AMaViS on a production mail server then please spend time with its documentation. It's a complex piece of software that can integrate a lot of third-party software like virus scanners. It can be tweaked a lot. It can automatically add DKIM signatures on outgoing email. It has a quarantine system so you can moderate spam emails. It just offers many additional features outside of the scope of this tutorial. If you believe you have a valuable addition to this tutorial then please send a comment on this page. Thank you.

Just one word on spam scanning. SpamAssassin (as being used by AMaViS) uses different rules to look for spam but can also learn to distinguish ham from spam emails. To do that properly it needs to scan an equal amount of several hundred ham and spam emails before it can really do its job. And as far as I know there is no way to learn spam per-user. So whatever you teach SpamAssassin is good or bad will apply for all your users.

Sending email from a dynamic IP address

- Add new comment
- 51973 reads

If you want to run a mail server on your Debian server at home then you are probably on a dynamic IP address (one that changes once a day) from your internet provider. The problem with dynamic IP addresses is that they are blacklisted by most mail servers on the internet. You cannot send email directly to other mail servers and need to send your email through a server that is not blacklisted. In this case you just need two things:

- You need to send out emails through your ISPs mail relay. If they don't offer such a service you can still try to find a commercial SMTP relay elsewhere on the internet.
- Your ISP must not block SMTP (TCP port 25) connections to your IP address. If they do you cannot be sent emails from other mail servers.

Without these prerequisites I suggest you get an inexpensive VPS (virtual private server) and run the mail server from there.

Sending out email through your ISP is not that hard. Your ISP very likely offers an SMTP server that you can use with a username and a password. If you use their SMTP server from your mail client or from your mail server is technically no difference. So if you can send emails from your mail client then you are set. In your /etc/postfix/main.cf you will have to set:

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
relayhost = [name.of.your.isp.relay.server]
```

(In case you wonder: configuration statements like smtp_* stand for outgoing SMTP while smtpd_* statements are for incoming SMTP connections. So these settings apply to outgoing SMTP connections.)

This sets "name.of.your.isp.relay.server" as the hostname of your ISP's relay server. Of course you need to set the right name here. The brackets means that no DNS lookup for MX records is done. The "relayhost" is the name of the server that Postfix will send all outgoing emails to. Without a "relayhost" Postfix would use DNS lookups to determine which mail server is responsible for a certain domain. The sasl_passwd file is where the username and password will be set that you have to use for authenticating at your ISP's relay server. The file looks like this.

```
[name.of.your.isp.relay.server]     herbert.ludkins:Ialenuv2
```

Pretty easy. You name the relay server exactly as in "relayhost", insert a space and then write the username, a colon and the password. Afterwords you need to compile this file as you want to use is as a "hash":

```
postmap /etc/postfix/sasl_passwd
```

This will give you an sasl_passwd.db file that Postfix can use.

You can also read about these settings in the Postfix documentation on SMTP authentication.

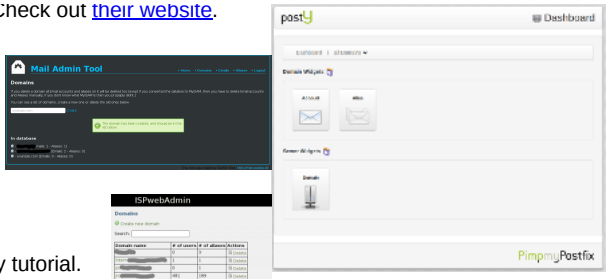
Managing domains, accounts and forwardings

- Add new comment
- 32173 reads

Managing control information of a mail server in a databaes gives a lot of flexibility compared to raw text files. But let's be honest for a moment. Most reader of this tutorial won't set up a mail server with thousands of users to make money with. They are just curious what it's like to run a mail server and wanted to learn a great deal. Are you like them? Then you probably just want to manage a domain and a handful of users and don't like to edit database rows through SQL statements, right? Don't despair - there are two ready web interfaces to manage your mail server.

posty

Lukas Weis and his team have created a very nifty web interface they call "posty". Check out their website.



Mail Admin Tool

Steffan Slot created a PHP based mangement interface that you can find at http://mat.ssdata.dk/

ISPwebAdmin

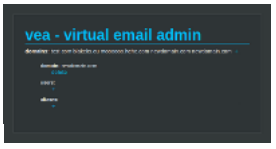
I created this web interface in Python using the Pylons web framework for the Lenny tutorial. And it also works on Debian Squeeze. It may take half an hour to set it up because it's not just a bunch of PHP files to throw on a web server. But it has served me well for two years on a production mail server with 10 domains and 1000 users.

MailAdm

Robert Kuntz is working on MailAdm to replace all other tools. Check out a demo version.

VEA

Felix (last name unknown has written a fancy and lightweight web interface. See his web site that even contains a live demonstration.



Scott Moody's management interface

Scott Moody has contributed another simple web interface consisting of just one PHP file that helps you manage your mail server. Visit his web site if you are interested.

An unnamed contributor claims that Scott's script does not work with Squeeze and provides a fixed script.

mailadmin

Ben Clarke also wrote a free and open source web administration interface. You can find his work at GitHub.

Troubleshooting

- Add new comment
- 30563 reads

General troubleshooting tips

- Run "postfix check" to make Postfix look for obvious configuration errors. If it returns no output then no problem was found.
- Read your /var/log/mail.log and look for warnings and errors.

Common problems and solutions

ClamAV fails to scan for viruses

May 7 12:42:28 debian amavis[1834]: (01834-01) (!)run_av (ClamAV-clamd) FAILED - unexpected , output="/var/lib/amavis/tmp/amavis-20110507T124228-01834/parts: lstat() failed: Permission denied. ERROR\n"

You forgot to run

```
adduser clamav amavis
```

Missing indentation in the master.cf

The /etc/postfix/master.cf file needs proper indentation. The first line of each service starts in the first column. Additional lines of the same service need to be indented by spaces.

Postfix keeps unwanted emails in the queue

Check that you do not keep soft_bounce enabled. If "postconf soft_bounce" shows "yes" then run "postconf soft_bounce=no".

I get "Permission denied" from Dovecot in the mail.log file

You have the permissions wrong. Run:

```
chmod -R vmail:vmail /var/vmail
```

General questions

Why is the database not normalized (email addresses with domain in virtual_users table)?

It's possible to normalize the database and use the JOIN syntax to get the domain name from the virtual_domains table. But that would lead to string operations when Postfix and Dovecot look up a certain email address. This quickly becomes a performance penalty when you have many users. So in this case performance wsas rated higher than strict normalisation.

Sysadmin niceties

- Add new comment
- 36174 reads

This page deals with a few miscellaneous issues that system administrators should consider.

Logrotate

On a busy mail server your /var/log/mail.log will grow quickly. That file not grow indefinitely thanks to the *logrotate* software that gets installed by default. Every day (controlled by cron - see the /etc/cron.daily/logrotate file) log files will get *rotated*. That means the old file "mail.log" gets renamed to "mail.log.1" and the Syslog daemon is restarted so it opens a new empty mail.log file. Upon the next rotation cycle the "mail.log.1" gets "mail.log.2" which will get compressed with GZip so it becomes a much smaller "mail.log.2.gz" file.

This is controlled by the /etc/logrotate.d/rsyslog file. By default is defines the rotation cycle a "weekly" and "rotate 4". So your mail.log is rotated once a week and after four rotation cycles the oldest file is deleted. Busy mail servers may need a daily rotation instead - so change the "weekly" to "daily". And if you have to provide log files for more than four rotation cycles then increase the number from "rotate 4" to e.g. "rotate 90" to make it 90 days.

Message queue and queue IDs

Understanding the Postfix mail queue is very useful. Postfix will put every accepted email into its queue before another Postfix process picks it up and tries to deliver it. You can see what emails are currently in your queue by running

```
mailq
```

in the shell. On a busy mail server you may have hundreds of email being stuck for various reasons. Maybe an ISP is throttling your mail server because you are sending unusual amounts of emails (easily happens when sending newsletters) or the destination mail server is currently unreachable. An example entry in the queue may look like this:

```
04D8CA8C442     2399 Tue May 31 09:49:11  MAILER-DAEMON
            (connect to mx.example.com[146.44.5.213]:25: Connection timed out)
                                    info@example.com
```

The "04D8CA8C442" is the queue ID. To see what happened with this email you can *grep* through your /var/log/mail.log file:

```
grep 04D8CA8C442 /var/log/mail.log
```

The output could be:

```
May 31 09:49:11 mx1 postfix/smtpd[4428]: 04D8CA8C442: client=web.localnet[10.10.41.3]
May 31 09:49:11 mx1 postfix/cleanup[4473]: 04D8CA8C442: message-id=<20110531094911.0A73631F59D@newsletter.example.net>
May 31 09:49:11 mx1 postfix/qmgr[25512]: 04D8CA8C442: from=<>, size=2399, nrcpt=1 (queue active)
May 31 09:49:41 mx1 postfix/smtp[4945]: 04D8CA8C442: to=<info@example.com>, relay=none, delay=30, delays=0.05/0.06/30/0,
dsn=4.4.1, status=deferred (connect to mx.example.com[146.44.5.213]:25: Connection timed out)
```

The different Postfix processes that handled the email are shown after the "postfix/...". The "smtpd" received the email from the web.localnet server. Then the "cleanup" process put it into the mail queue. Next the "qmgr" moved the email into the active queue and the "smtp" tried to deliver it. The remote mail server mx.example.com could not be connected to so Postfix kept the email in the queue. The DSN (delivery status notification) code was 4.4.1. All codes starting with 4 are temporary errors - Postfix will retry to deliver the email. Codes starting with 5 are permanent errors and Postfix will instantly bounce the email and inform the sender of the delivery failure. Codes starting with 2 are successes.

If the delivery was successful the "postfix/smtp" line will look like this:

```
Jun  5 12:41:10 mail postfix/smtp[12044]: 2171CA860E0: to=<jonathan@example.com>, relay=mx.example.com[191.13.14.2]:25,
delay=0.32, delays=0.04/0/0.17/0.1, dsn=2.0.0,
 status=sent (250 2.0.0 Ok: queued as B0BAC736372)
```

The last part of the message contains a string that the remove mail server returns after the successful delivery. So if you see that an email got lost between your and the remote (mx.example.com) mail server this log line helps a lot. You see that the email was once handled in your queue as ID 2171CA860E0 and then delivered to mx.example.com which put it into its own queue as queue ID B0BAC736372. So you can try to contact the postmaster of the remote server and ask what happend to this email.

Getting statistics

Reading your mail log file barely gives you interesting information at a glance. Especially on a busy mail server you will barely get an idea what is going on except for counting the lines of the log file. I prefer to run "pflogsumm" (Postfix Log Summary) on my mail logs. You can easily install this software:

```
apt-get install pflogsumm
```

The just run this program on your log file:

```
pflogsumm /var/log/mail.log
```

What you get is general counts like:

```
messages

  129   received
  331   delivered
```

```
    1  forwarded
    8  deferred  (59  deferrals)
    3  bounced
  586  rejected (63%)
    0  reject warnings
    0  held
    0  discarded (0%)

 3108k  bytes received
 3967k  bytes delivered
   36  senders
   28  sending hosts/domains
  229  recipients
  147  recipient hosts/domains
```

as well as per-hour traffic summaries (how many emails were sent and received during what hours). You also get a sorted list of sender and recipient domains. Aside from general statistics pflogsumm also extracts errors and warnings so you get a quick overview of why emails were deferred or rejected/bounced and the effectiveness of the RBLs (real-time blacklists) that you use.

Take a look at the example cron entries in /usr/share/doc/pflogsumm/examples on your system. They help you get an automatic daily and weekly report.

Blocking script kiddies with fail2ban

You are surely aware that the internet is not the friendly place it once used to be. Fortunately most of the attackers you will face at your mail server are stupid and adventurous script kiddies. They will use dumb scripts to try a few username/password combinations to get access to mailboxes. In your /var/log/mail.log this will look like this:

```
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<mp3>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<oscar>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<webpage>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<webpage>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<bbuser>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<telecom>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<root>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
dovecot: pop3-login: Disconnected (auth failed, 1 attempts): user=<dbadmin>, method=PLAIN, rip=61.144.24.114, lip=87.152.157.104
```

We are not using user names like "oscar" but rather "oscar@example.com" so this embarrassing attempt will not work anyway. But why take the chance? We can use the "fail2ban" software to deal with brute force attacks like this easily. First install the fail2ban software on your server:

```
apt-get install fail2ban
```

Fail2ban starts a daemon process that can watch various log files, look for certain attack patterns there (defined by regular expressions) and act accordingly. We can teach fail2ban to look for "auth failed" lines like above and add a firewall rule (using iptables by default) to block access from the kid's IP address. By default fail2ban will run such a blocking action it it finds three attack attempts within ten minutes and will then block the attacker for ten minutes. These values are configurable.

Add a file "/etc/fail2ban/filter.d/dovecot-pop3imap.conf" containing:

```
[Definition]
failregex = (?: pop3-login|imap-login): .*(?:Authentication failure|Aborted login \(auth failed|Aborted login \(tried to use
disabled|Disconnected \(auth failed).*rip=(?P<host>\S*),.*
ignoreregex =
```

This adds a regular expression telling fail2ban what a failed login attempt looks like. The P<host> part at "rip" (remote IP) will contain the respective IP address of the attacker. fail2ban passes it to the blocking action to ban this very IP address.

Then edit the "/etc/fail2ban/jail.conf" file. First enable the "postfix" part that is disabled by default:

```
[postfix]
enabled  = true
port     = smtp,ssmtp
filter   = postfix
logpath  = /var/log/mail.log
```

Also add a section for the Dovecot configuration you added earlier.

```
[dovecot-pop3imap]
enabled = true
port = pop3,pop3s,imap,imaps
filter = dovecot-pop3imap
logpath = /var/log/mail.log
```

Restart fail2ban:

```
/etc/init.d/fail2ban restart
```

Now you can wait for kids to try their luck. But to check that everything works you can also fake an attack by sending manually crafted log entries:

```
logger -p mail.info -t dovecot "imap-login: Aborted login (auth failed, 2 attempts): user=<hanswaltergeorgfoo>, method=PLAIN,
```

```
rip=10.20.30.40, lip=1.2.3.4, TLS"
```

Run the above command three times so that fail2ban triggers its action. If everything went as desires then your /var/log/fail2ban.log will read:

```
fail2ban.actions: WARNING [dovecot-pop3imap] Ban 10.20.30.40
```

Now attackers can do no more than three attacks within ten minutes. Nice, isn't it?