The Perfect Server - OpenSUSE 11.1

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>, Till Brehm <t [dot] brehm [at] ispconfig [dot] com>
Last edited 12/18/2008

This is a detailed description about how to set up an **OpenSUSE 11.1** server that offers all services needed by ISPs and hosters: Apache web server (SSL-capable), Postfix mail server with SMTP-AUTH and TLS, BIND DNS server, Proftpd FTP server, MySQL server, Dovecot POP3/IMAP, Quota, Firewall, etc. This tutorial is written for the 32-bit version of OpenSUSE 11.1, but should apply to the 64-bit version with very little modifications as well.

I will use the following software:

- Web Server: Apache 2.2.10 with PHP 5.2.6, Ruby, and Python
- Database Server: MySQL 5.0.67
- Mail Server: Postfix
- DNS Server: BIND9
- FTP Server: proftpd
- POP3/IMAP: I will use Maildir format and therefore install Courier-POP3/Courier-IMAP.
- Webalizer for web site statistics

In the end you should have a system that works reliably, and if you like you can install the free webhosting control panel ISPConfig (i.e., ISPConfig runs on it out of the box).

I want to say first that this is not the only way of setting up such a system. There are many ways of achieving this goal but this is the way I take. I do not issue any guarantee that this will work for you!

1 Requirements

To install such a system you will need the following:

- The OpenSUSE 11.1**DVD**. You can download it here: http://download.opensuse.org/distribution/11.1/iso/openSUSE-11.1-DVD-i586.iso
- A fast internet connection...

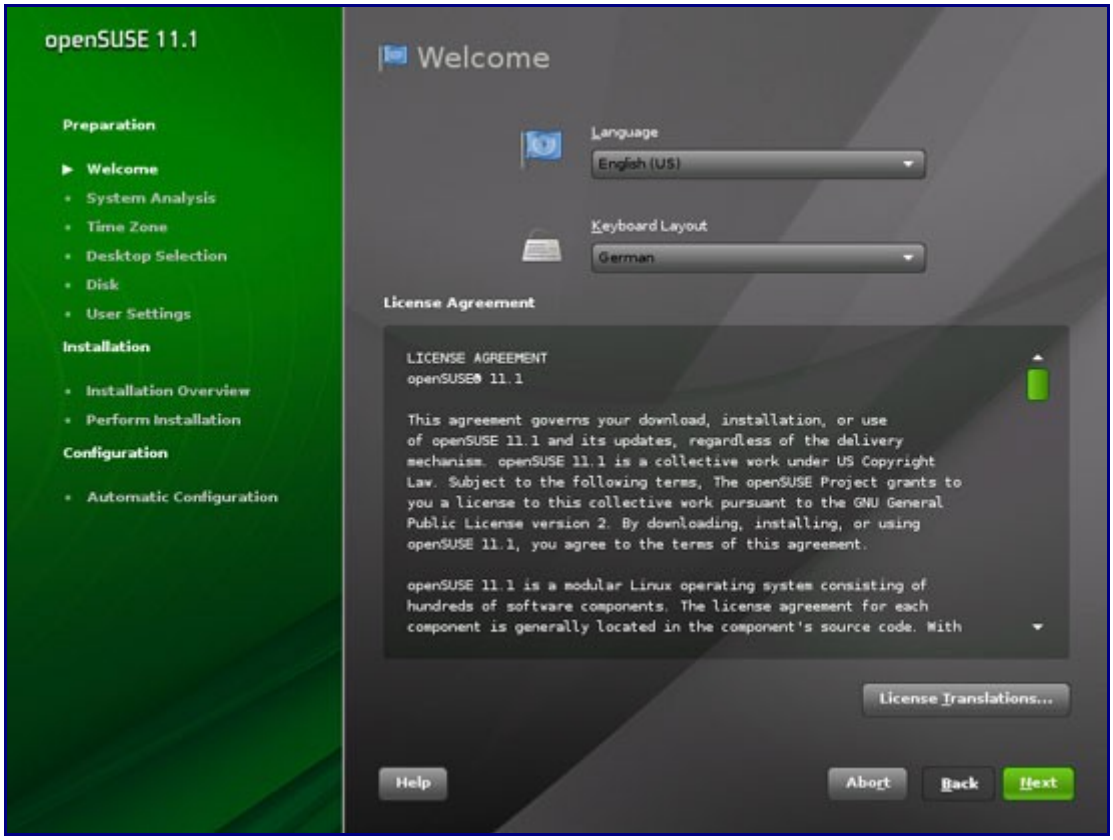
2 Preliminary Note

In this tutorial I use the hostname server1.example.com with the IP address 192.168.0.100 and the gateway 192.168.0.1. These settings might differ for you, so you have to replace them where appropriate.

3 The Base System

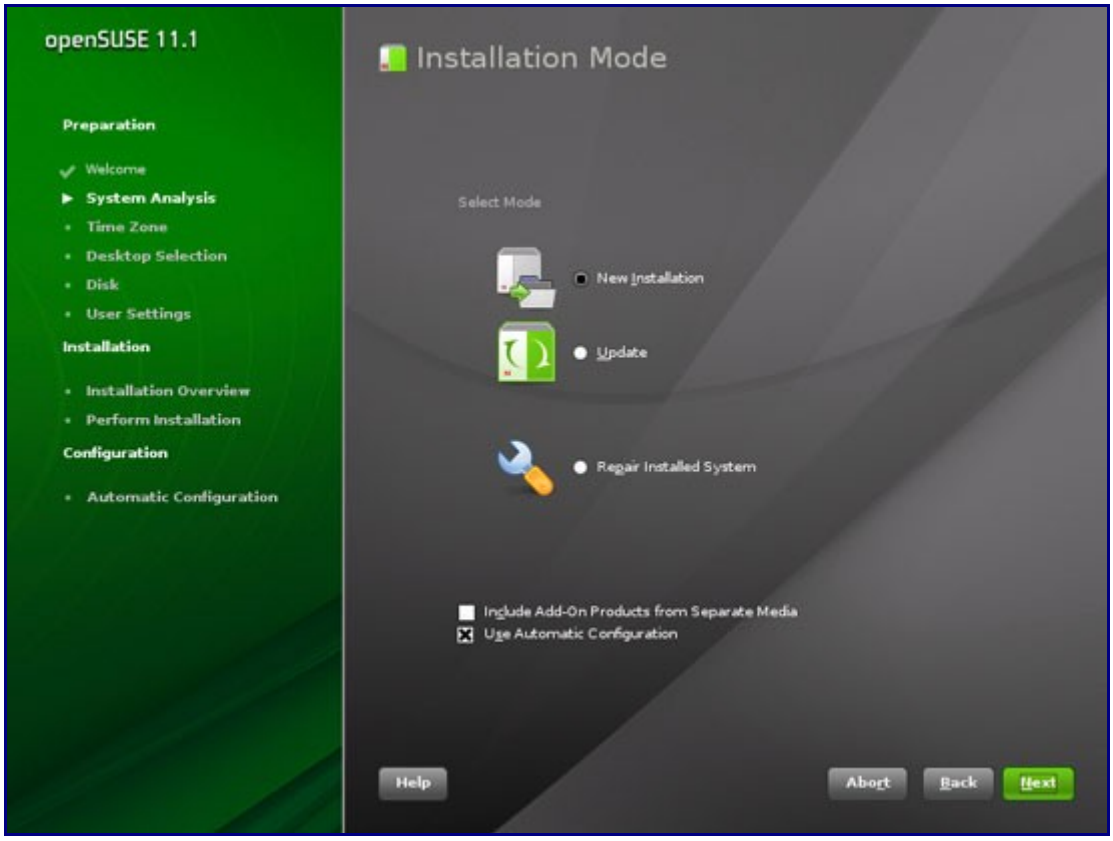
Boot from your OpenSUSE 11.1 DVD and select Installation:

Select your language, keyboard layout and accept the licence terms:

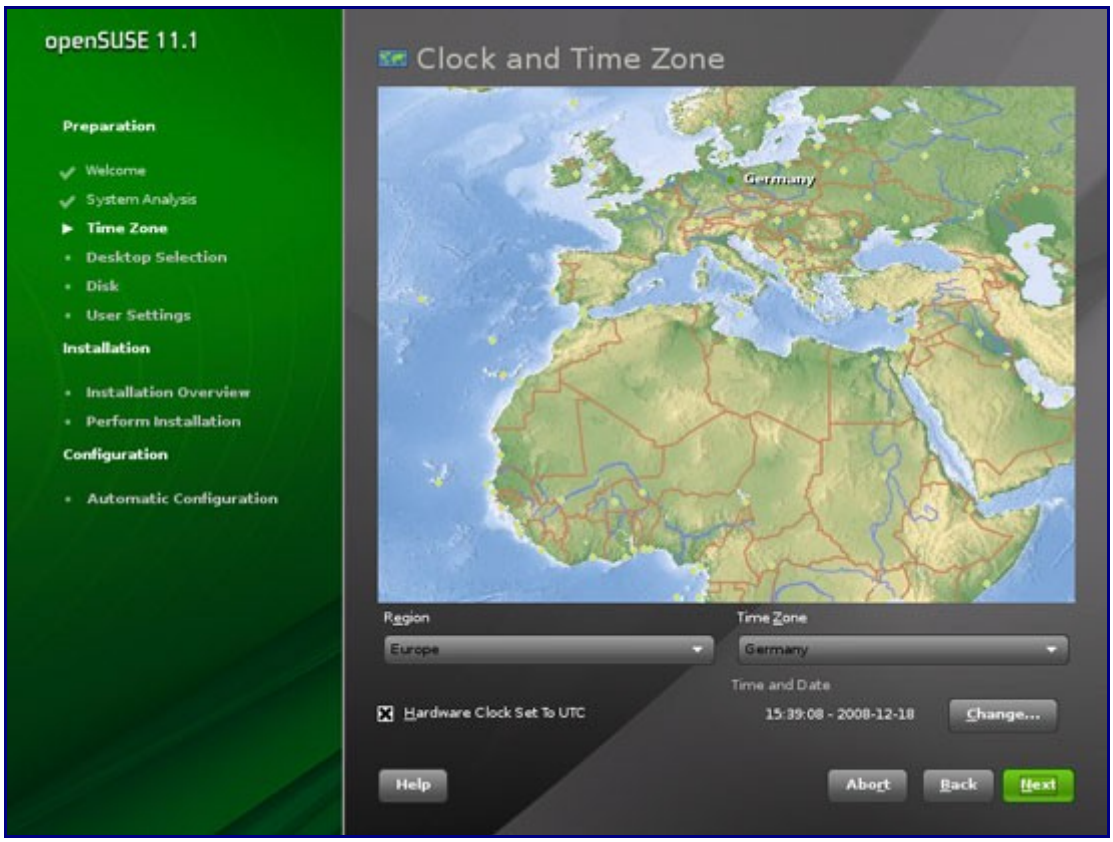


The installer analyzes your hardware and builds the software repository cache:

Select New Installation:

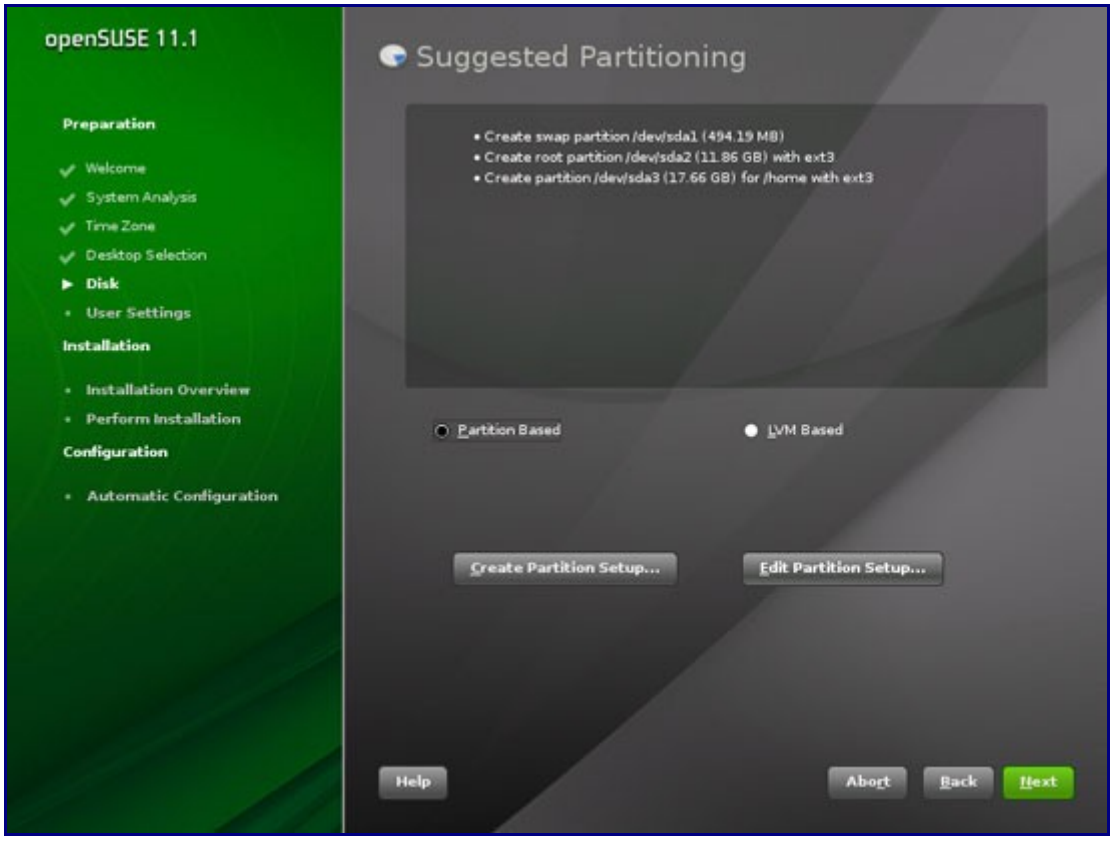


Select the region and timezone:

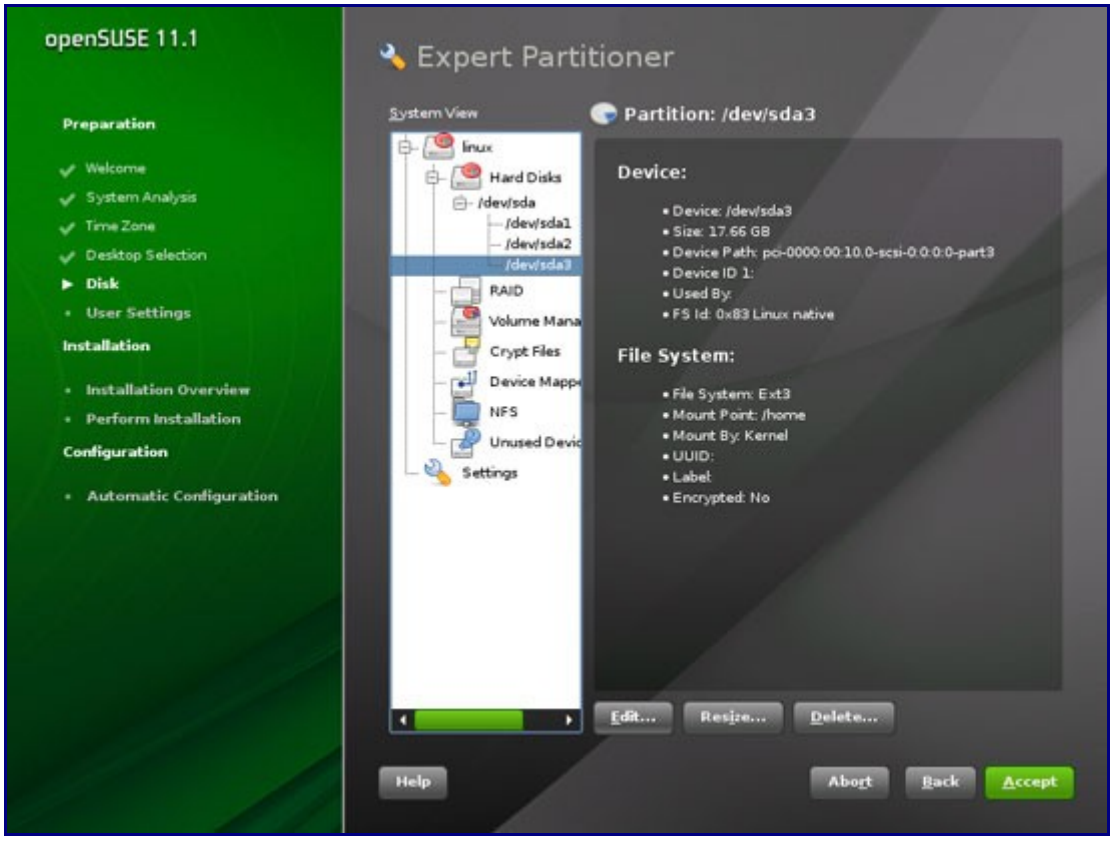


We select Other > Minimal Server Selection (Text Mode) here as we want to install a server without X-Window desktop. The X-Window system is not nescessary to run the server and would slow down the system. We will do all administration tasks on the shell or trough an SSH connection, e.g. via PuTTY from a remote desktop.

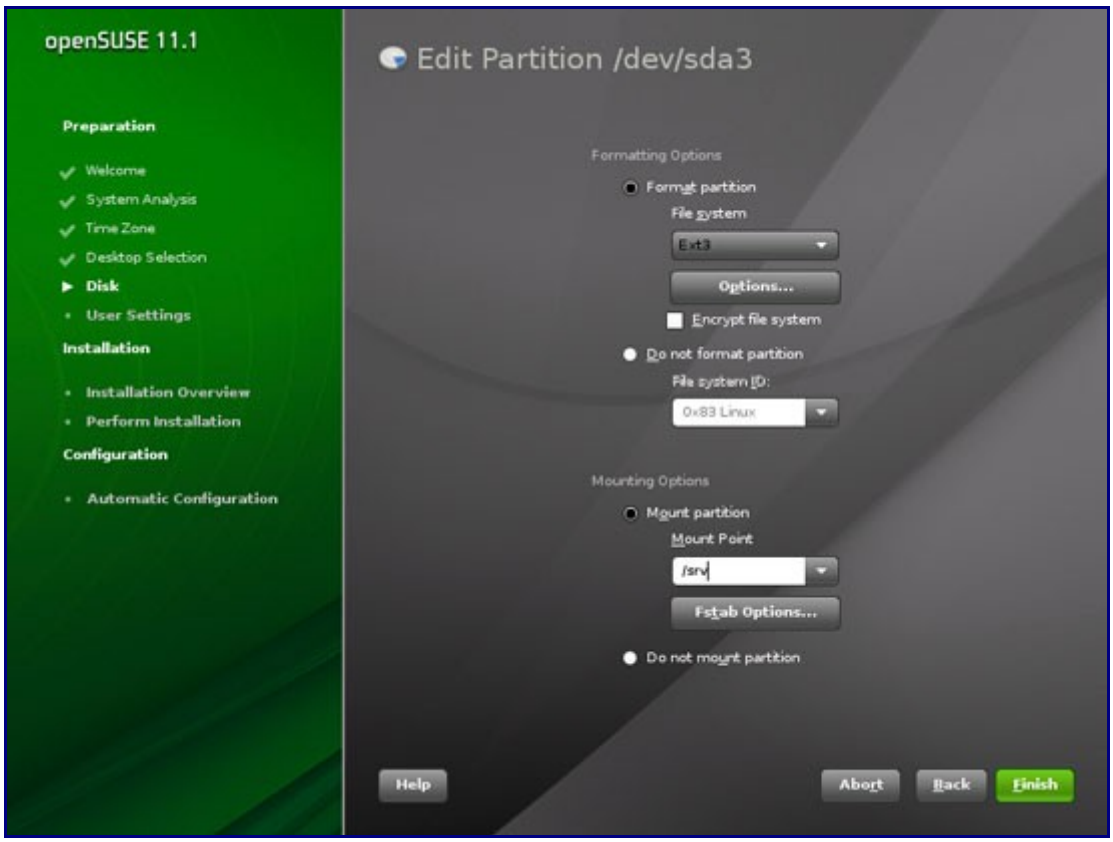
Click on Edit partition setup... to change the proposed partitions. As this is a server setup, we need a large /srv partition instead of the /home partition:



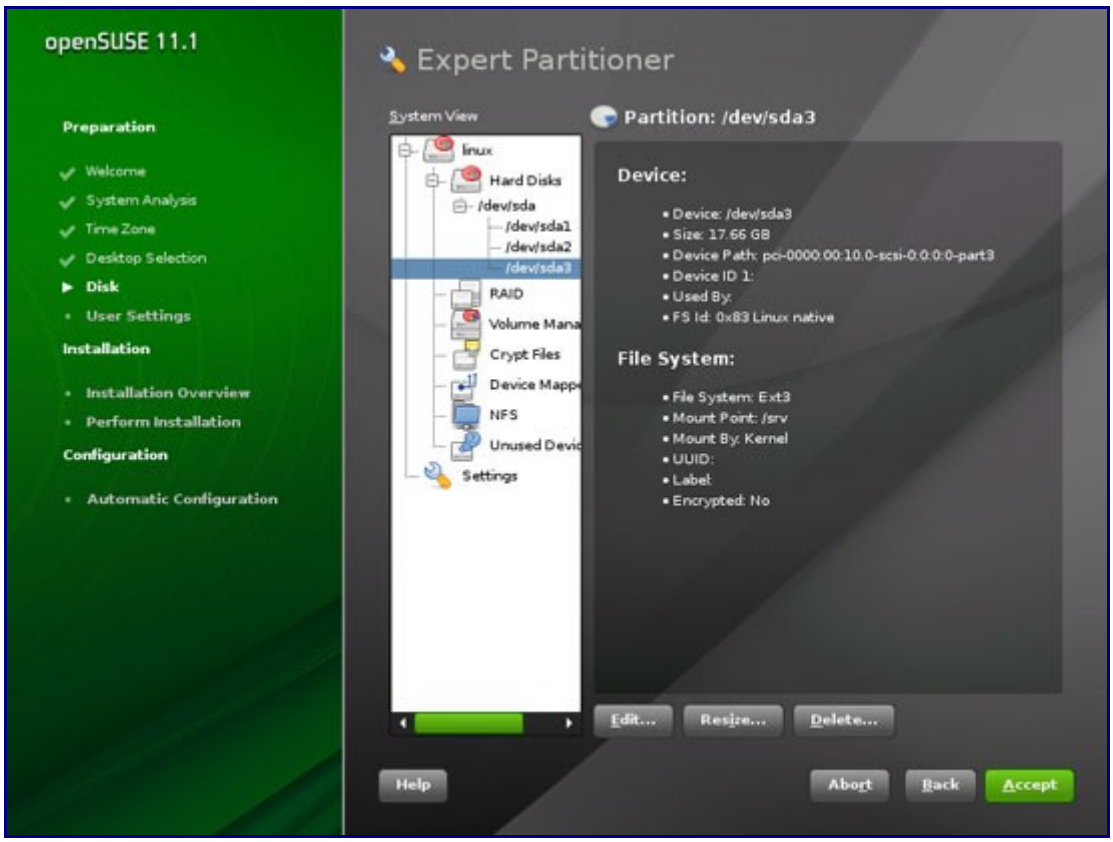
Select /dev/sda3 and click on Edit...:



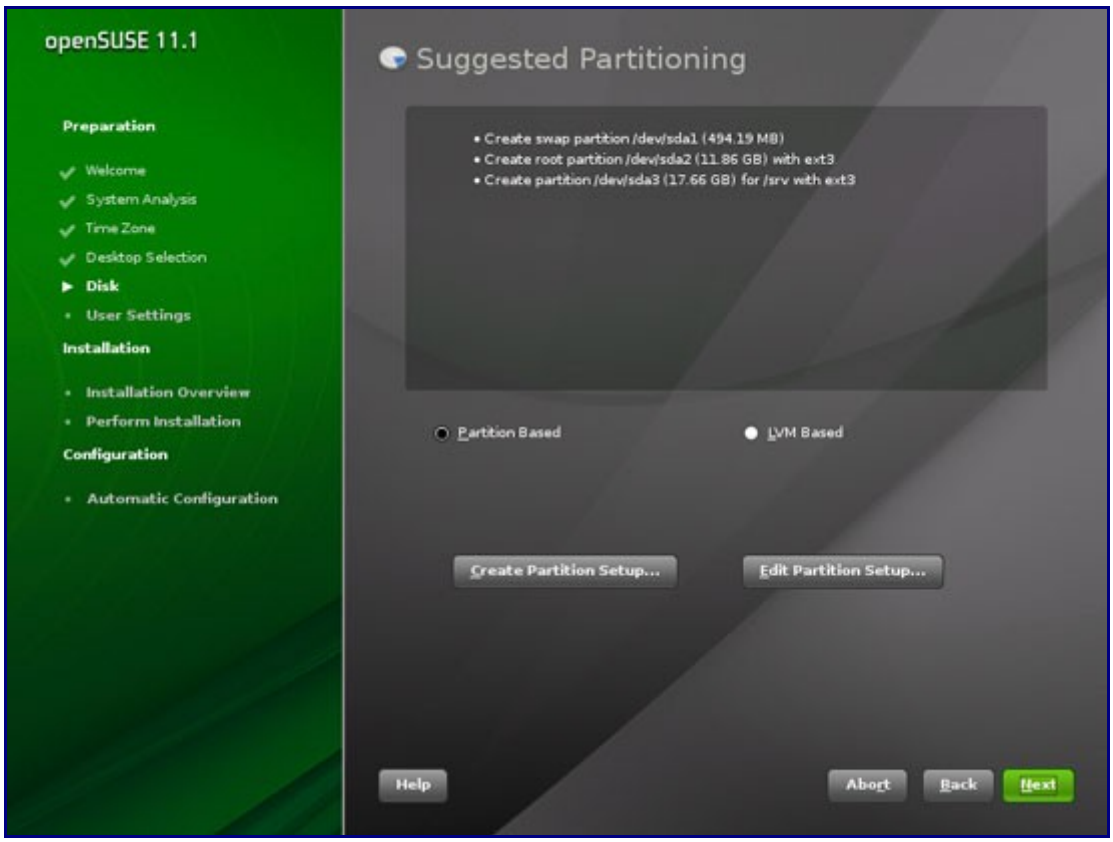
Change the Mount Point to /srv and click on Finish:



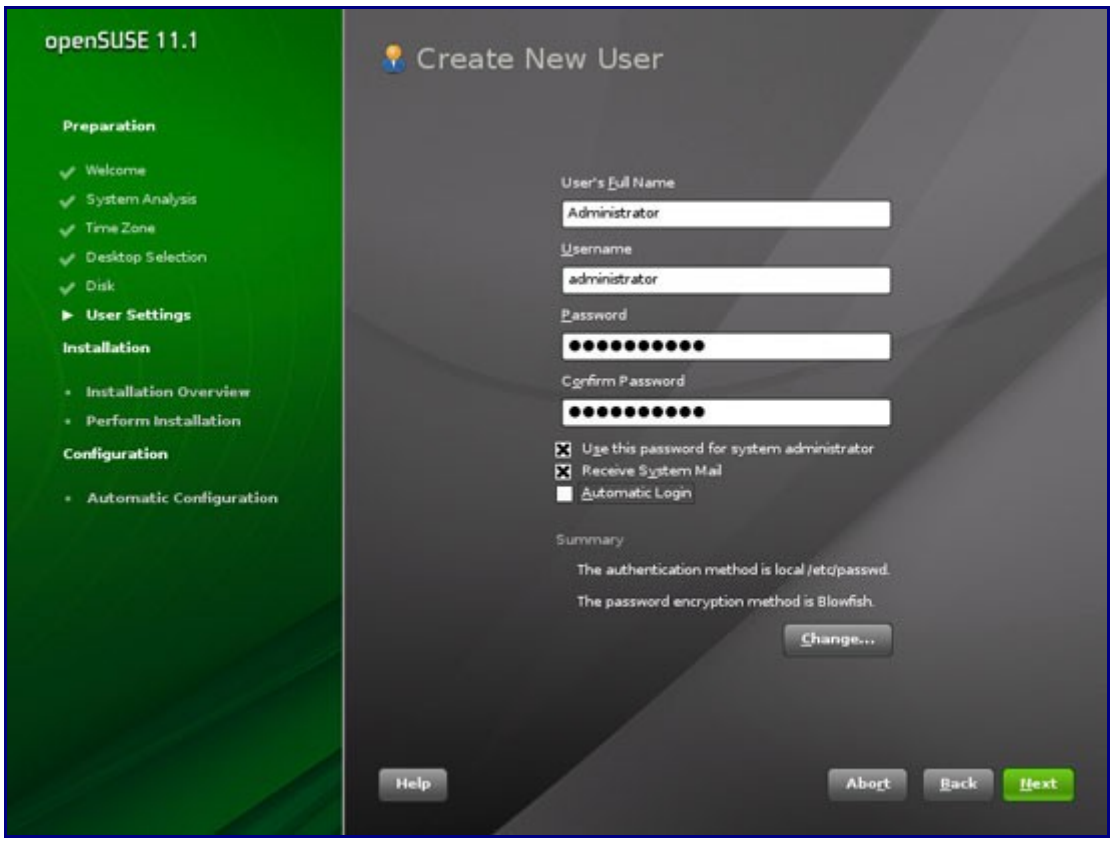
Click on Accept:



The resulting setup should look like this. Click on Next:

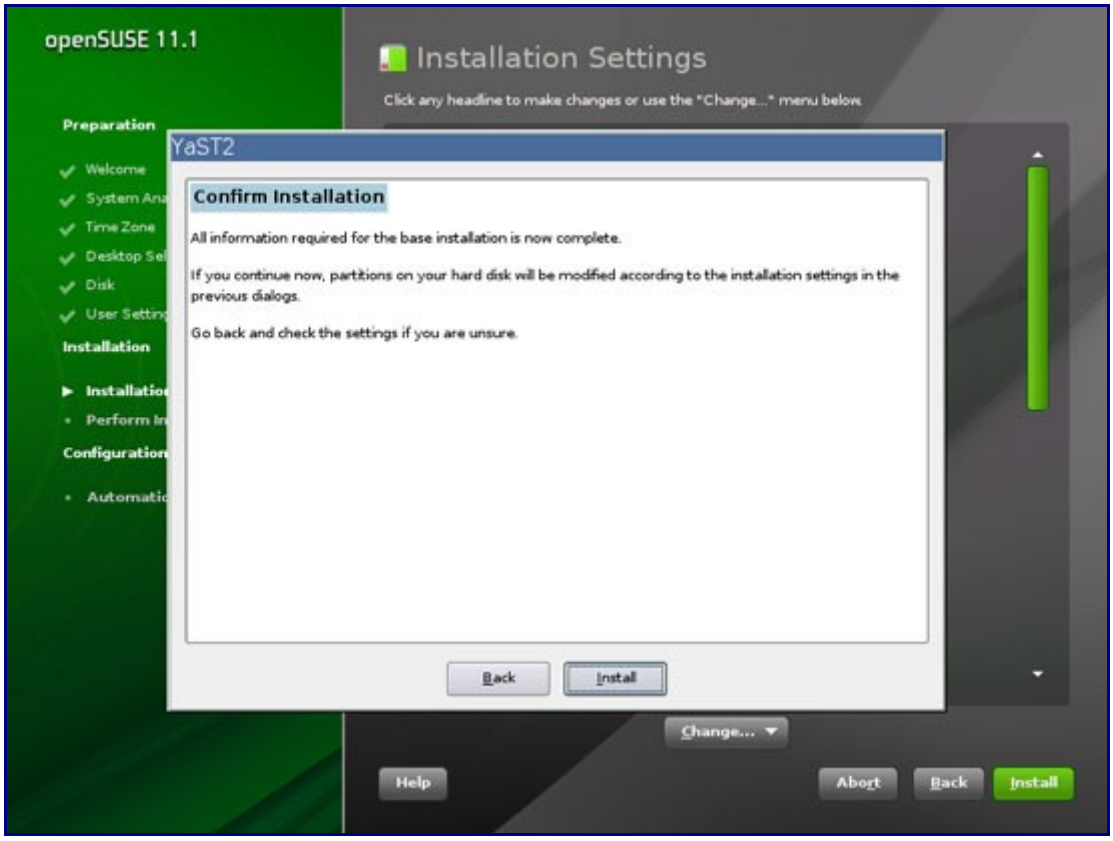


Now I create a user named administrator. You may use any username you like. Make sure that you disable the Automatic Login checkbox for this user. The password that you enter here will be used as the root password:

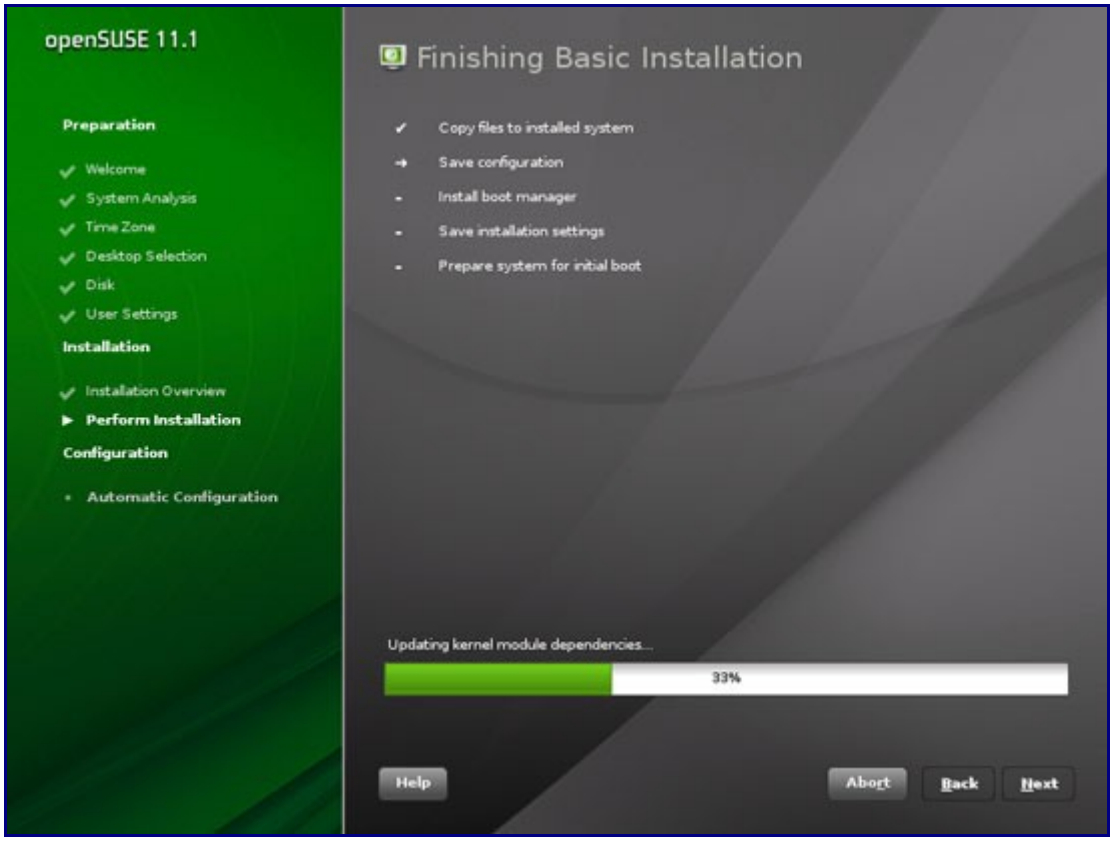


The installer shows an overview of the selected install options. Click on Install to start the installation process.

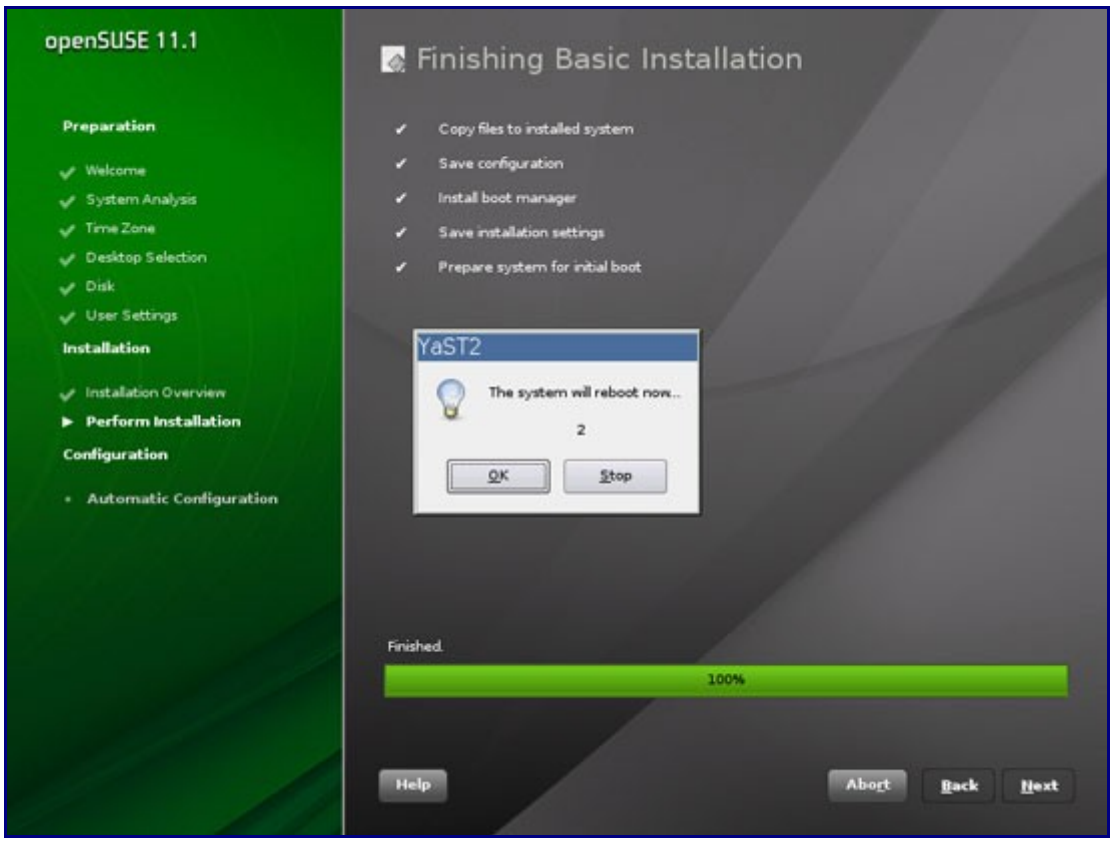
Confirm that you want to start the installation:



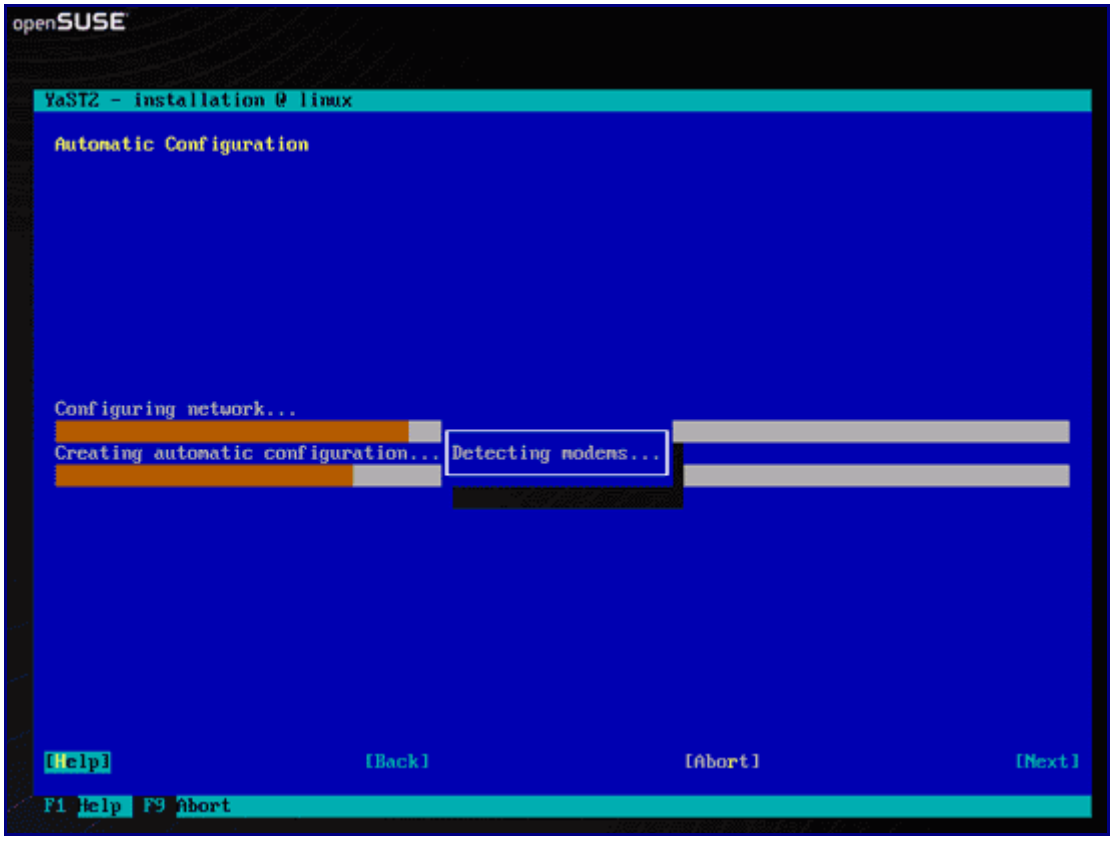
The installer formats the hard disk, installs the software packages and prepares the system configuration for the first boot:



After the basic installation is finished, the system will do an automatic reboot:



The automatic configuration starts right after the system has rebooted:



Now log in with the username root and the password that you selected during the installation.

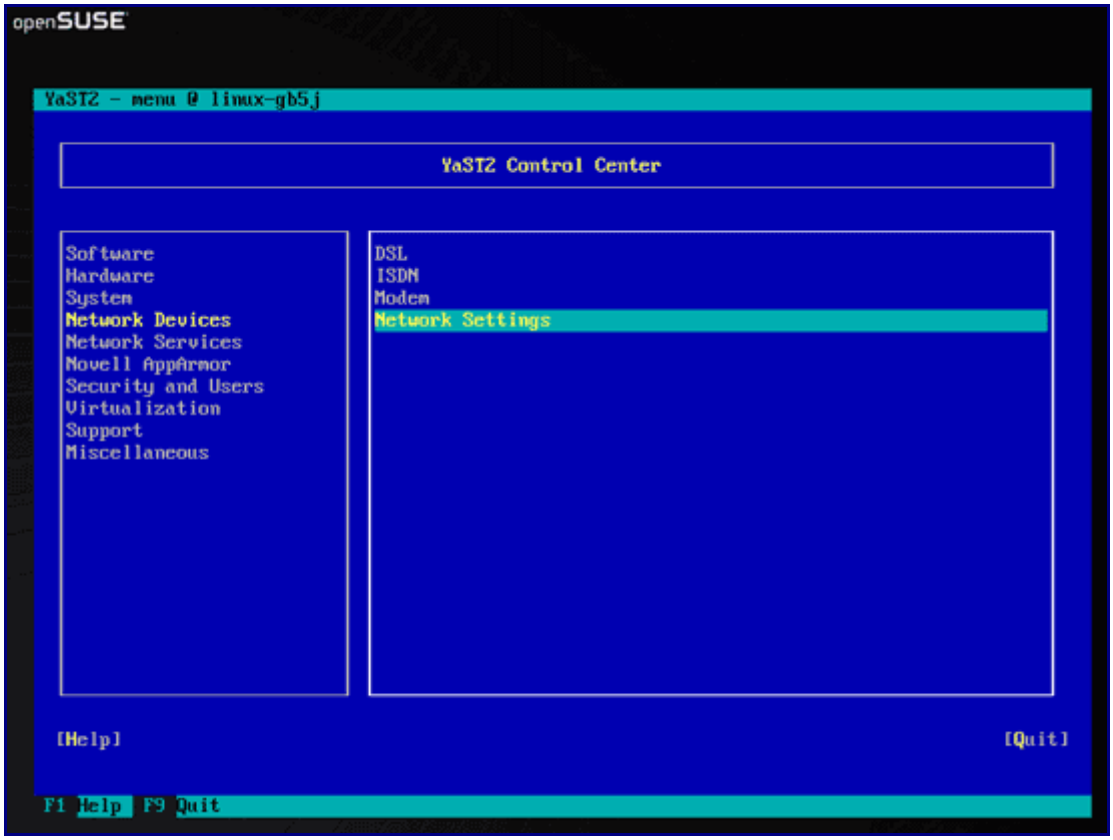
4 Configure The Network settings

We use Yast, the OpenSuSE system management tool to reconfigure the network card settings. After the first boot, the system is configured to get the IP address with DHCP. For a server we will switch it to a static IP address.

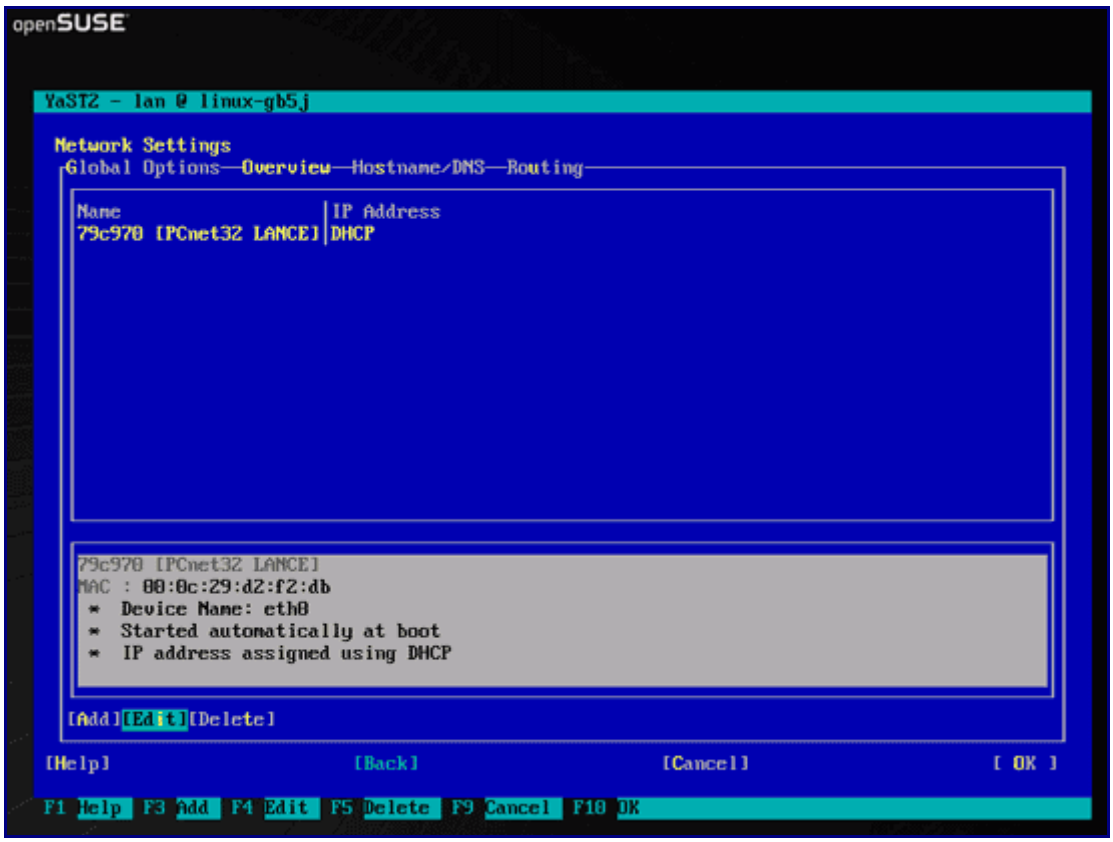
Run

yast2

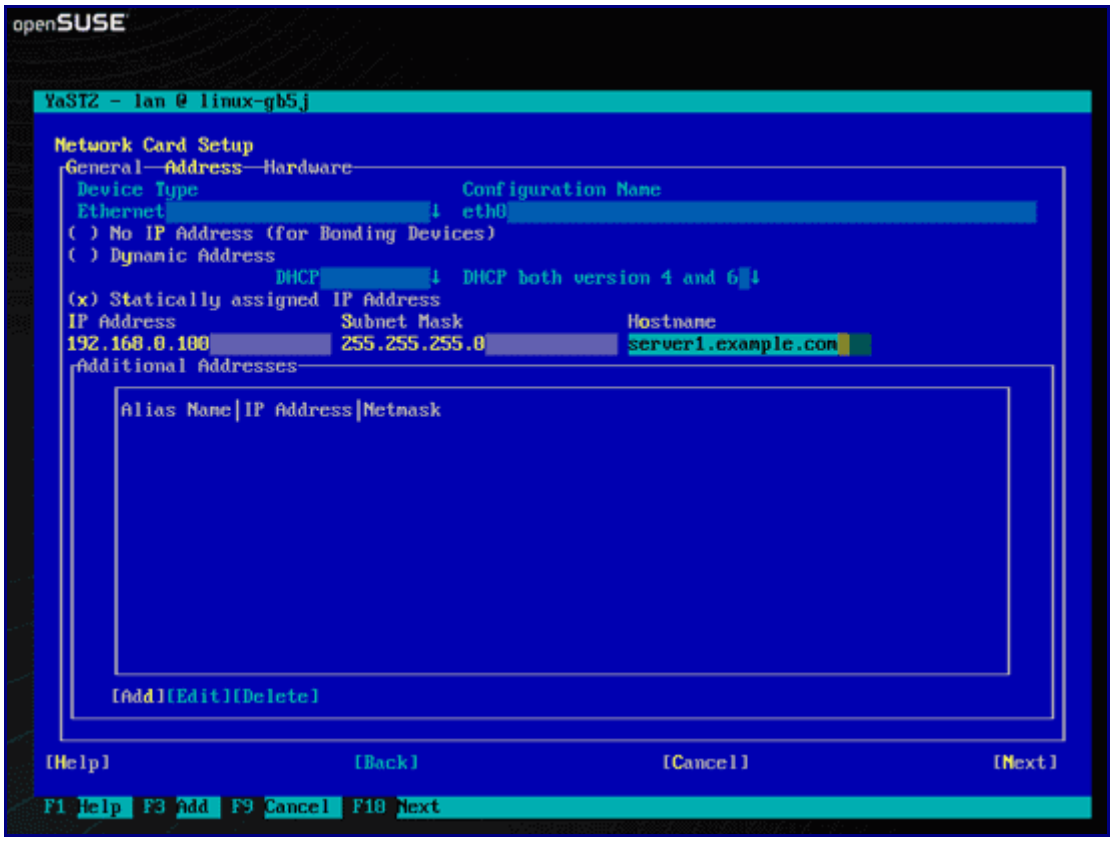
Select Network Devices > Network Settings:



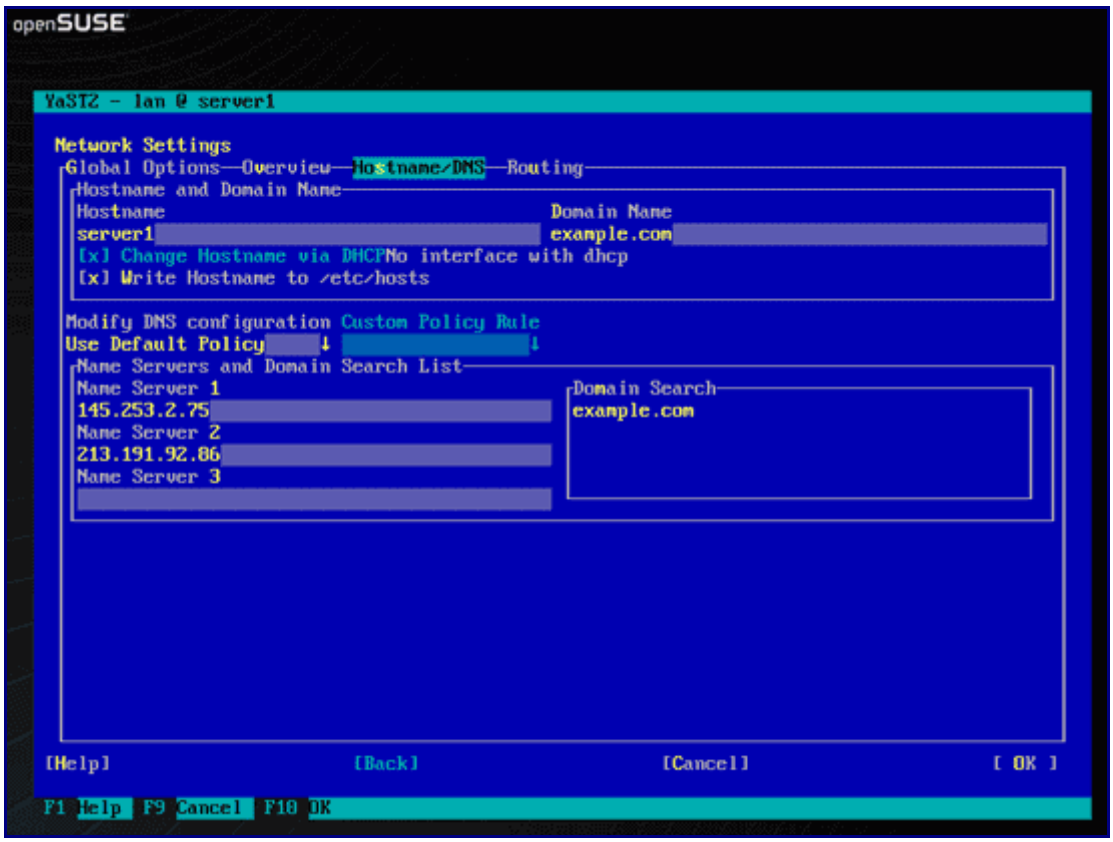
Select your network card and then Edit:



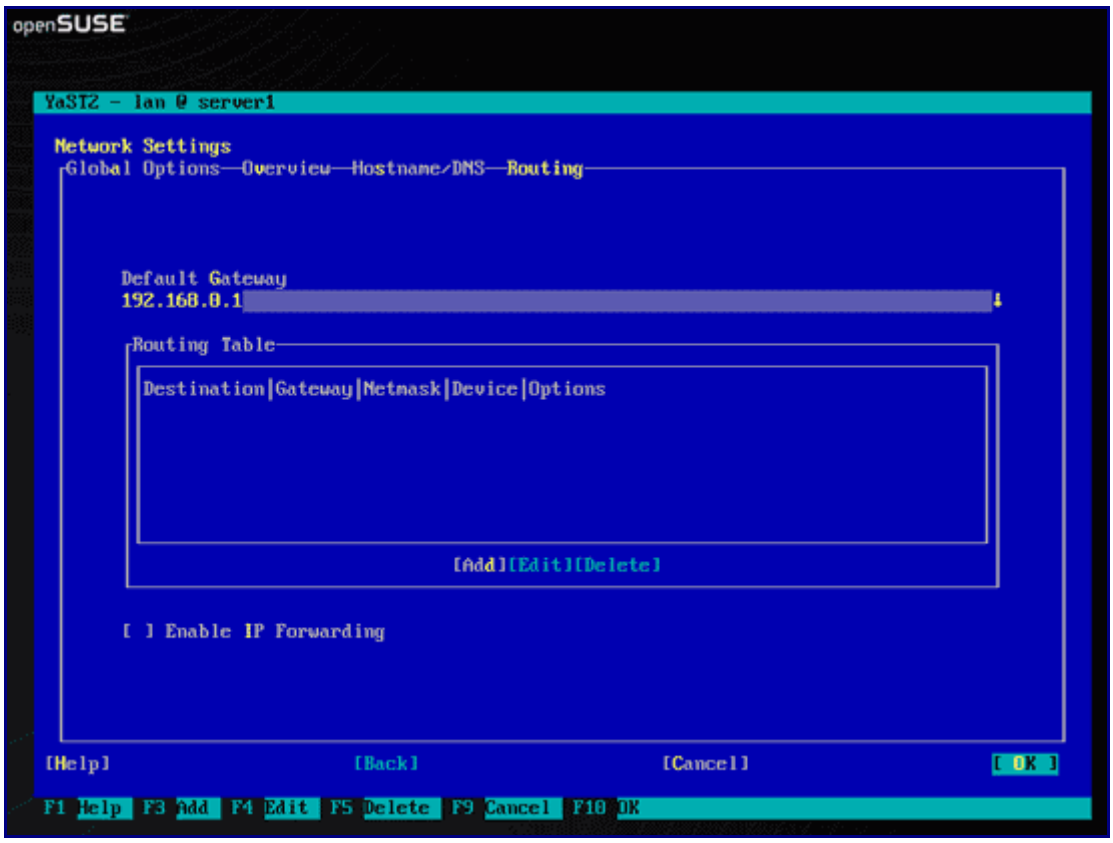
Select Statically assigned IP Address and enter the IP address, subnet mask and hostname and save the changes by selecting Next:



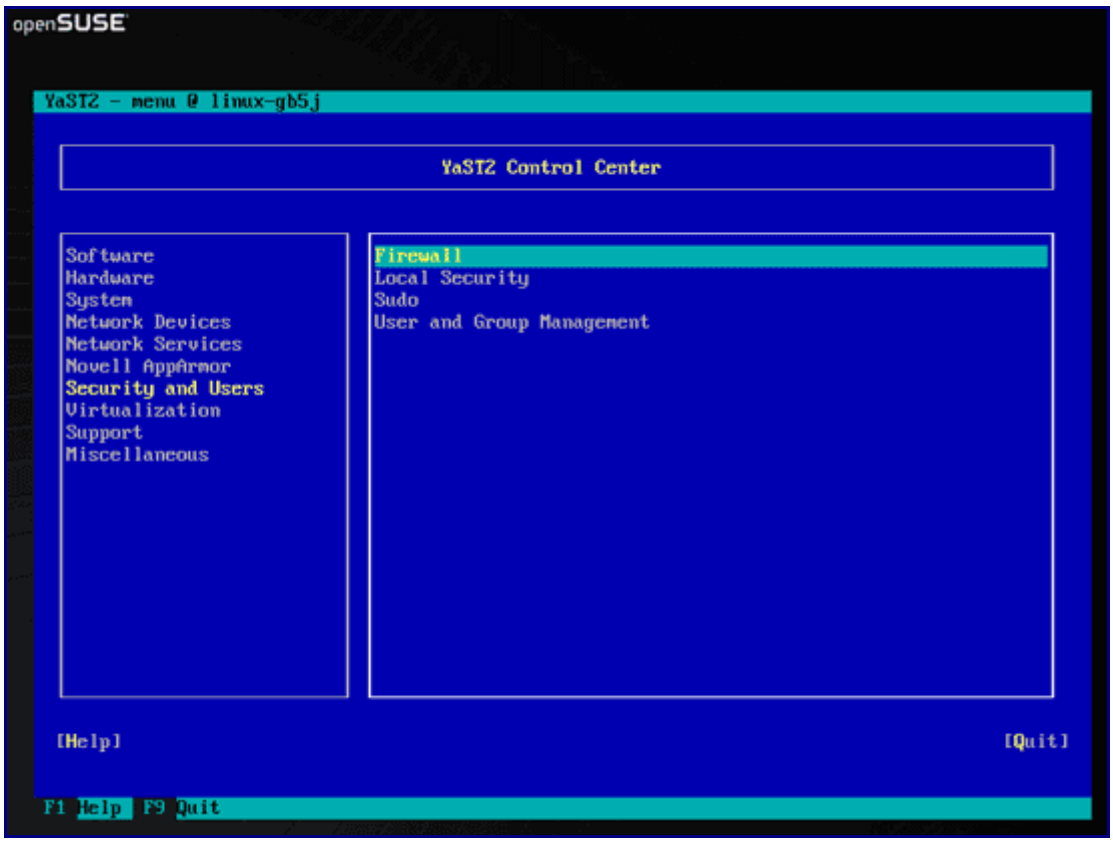
Now select Hostname/DNS and enter the hostname (e.g. server1.example.com) and nameservers (e.g. 145.253.2.75 and 213.191.92.86):



Now select Routing and enter the default gateway and hit OK:

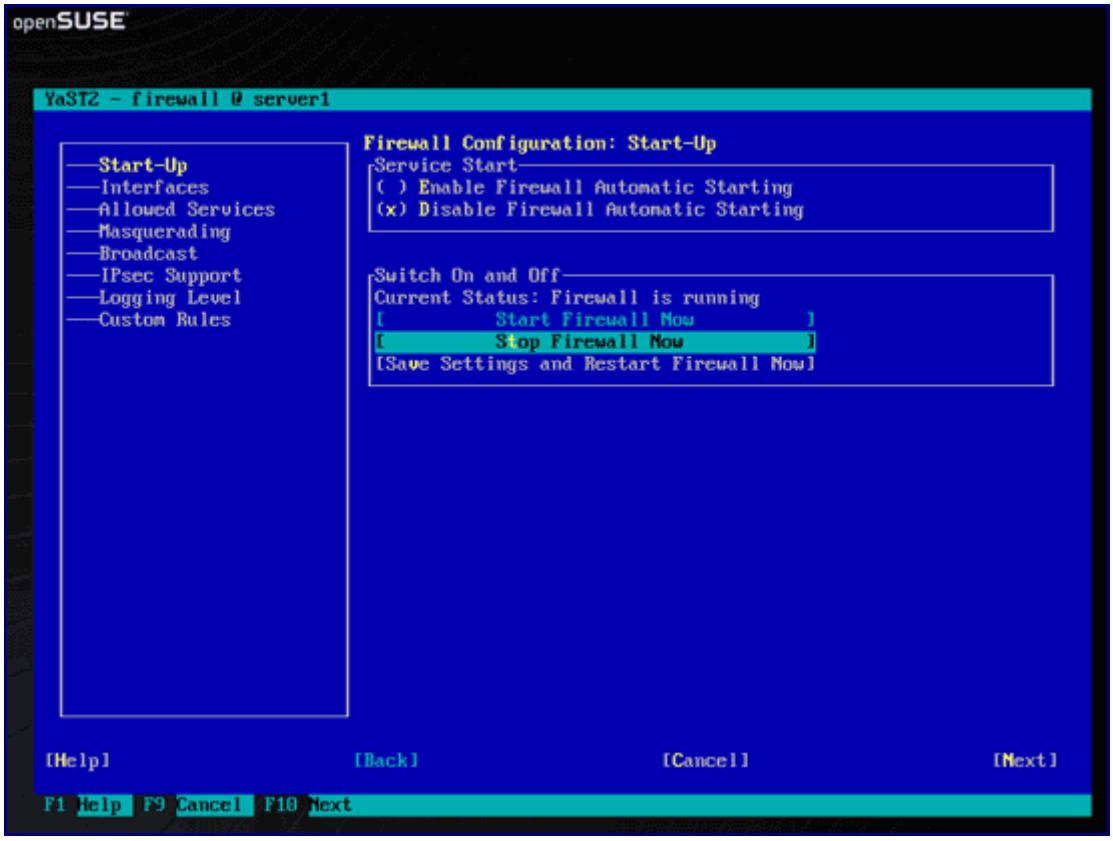


To configure the firewall, select Security and Users > Firewall in Yast:

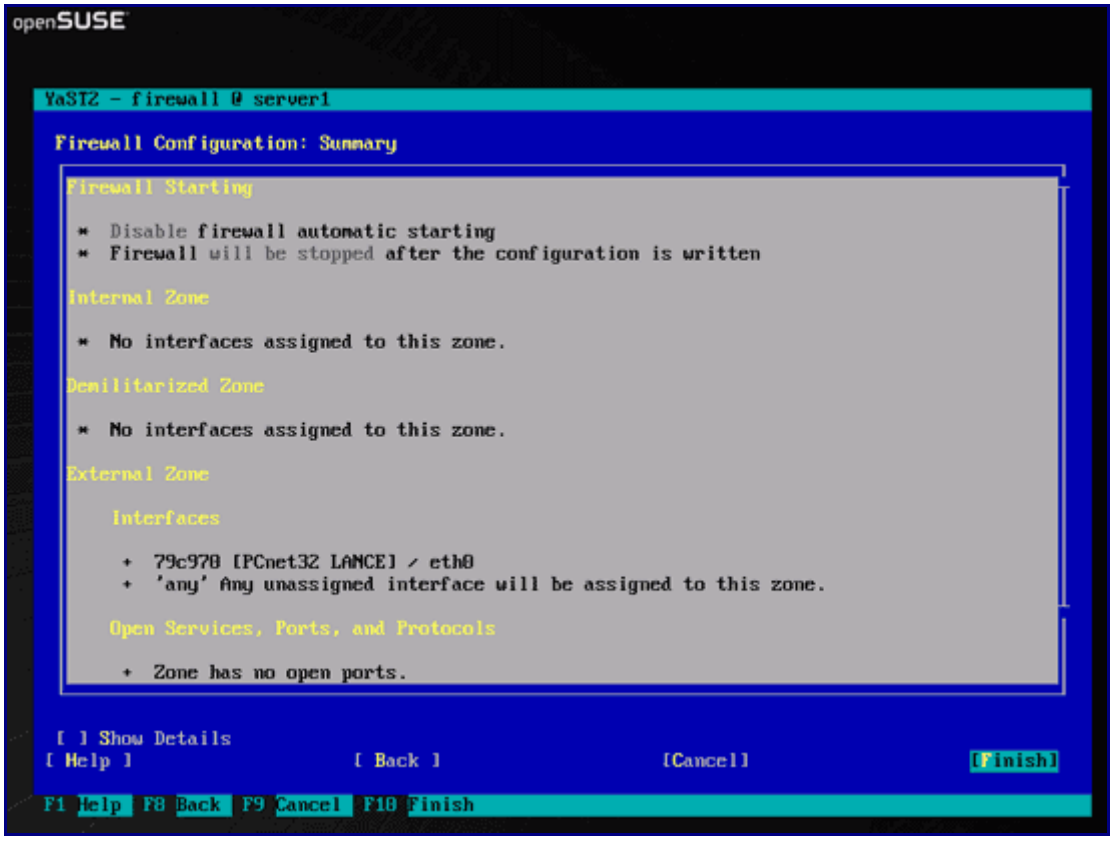


I want to install ISPConfig at the end of this tutorial which comes with its own firewall. That's why I disable the default OpenSUSE firewall now. Of course, you are free to leave it on and configure it to your needs (but then you shouldn't use any other firewall later on as it will most probably interfere with the OpenSUSE firewall).

Select Disable Firewall Automatic Starting and Stop Firewall Now, then hit Next:



Hit Finish and leave Yast:



5 Install Some Software

Now we install a few packages that are needed later on. Run

yast2 -i findutils readline libgcc glibc-devel findutils-locate gcc flex lynx compat-readline4 db-devel wget gcc-c++ make vim telnet cron iptables iputils man man-pages

If you're on a 64-bit system (only then!), you must also do this:

cd /usr/lib
ln -s /usr/lib64/libssl.a libssl.a
ln -s /usr/lib64/libssl.so libssl.so

6 Quota

To install quota, run

yast2 -i quota

Edit /etc/fstab to look like this (I added ,usrquota,grpquota to the mountpoints / and /srv):

vi /etc/fstab

```
/dev/sda1     swap      swap        defaults                0 0
/dev/sda2     /         ext3        acl,user_xattr,usrquota,grpquota        1 1
/dev/sda3     /srv      ext3        acl,user_xattr,usrquota,grpquota        1 2
proc          /proc     proc        defaults                0 0
sysfs         /sys      sysfs       noauto                  0 0
debugfs       /sys/kernel/debug     debugfs     noauto                  0 0
```

```
devpts       /dev/pts devpts     mode=0620,gid=5       0 0
```

Then run:

touch /aquota.user /aquota.group
chmod 600 /aquota.*
touch /srv/aquota.user /srv/aquota.group
chmod 600 /srv/aquota.*

mount -o remount /
mount -o remount /srv

quotacheck -avugm
quotaon -avug

Dont be worried if you see these error messages - they are normal when you run quotacheck for the first time:

quotacheck: WARNING - Quotafile //aquota.user was probably truncated. Cannot save quota settings...
quotacheck: WARNING - Quotafile //aquota.group was probably truncated. Cannot save quota settings...
quotacheck: Scanning /dev/sda2 [/] done
quotacheck: Checked 5286 directories and 45399 files
quotacheck: WARNING - Quotafile /srv/aquota.user was probably truncated. Cannot save quota settings...
quotacheck: WARNING - Quotafile /srv/aquota.group was probably truncated. Cannot save quota settings...
quotacheck: Scanning /dev/sda3 [/srv] done
quotacheck: Checked 7 directories and 4 files

7 DNS Server

Run

yast2 -i bind bind-chrootenv bind-devel bind-utils

Then we add the system startup links for BIND and start it:

chkconfig --add named
/etc/init.d/named start

Bind will run in a chroot jail under /var/lib/named.

8 MySQL

In order to install MySQL, we run

yast2 -i mysql mysql-client mysql-shared perl-DBD-mysql perl-DBI perl-Data-ShowTable libmysqlclient-devel

Then we add the system startup links for MySQL and start it:

chkconfig --add mysql
/etc/init.d/mysql start

Now check that networking is enabled. Run

netstat -tap | grep mysql

In the output you should see something like this:

```
server1:~ # netstat -tap | grep mysql
tcp        0      0 *:mysql                 *:*                     LISTEN      8566/mysqld
server1:~ #
```

If you don't see a line like this, edit /etc/my.cnf, comment out the option skip-networking:

vi /etc/my.cnf

```
[...]
#skip-networking
[...]
```

and restart your MySQL server:

/etc/init.d/mysql restart

To secure the MySQL installation, run:

mysql_secure_installation

Now you will be asked several questions:

```
server1:~ # mysql_secure_installation


NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MySQL to secure it, we'll need the current
password for the root user.  If you've just installed MySQL, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MySQL
root user without the proper authorisation.

Set root password? [Y/n] <-- Y
New password: <-- fill in your desired MySQL root password
Re-enter new password: <-- confirm that password
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MySQL installation has an anonymous user, allowing anyone
to log into MySQL without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] <-- Y
 ... Success!
```

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] <-- Y
... Success!

By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] <-- Y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] <-- Y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MySQL installation should now be secure.

Thanks for using MySQL!

Server1:~ #

Now your MySQL setup should be secured.

9 Postfix With SMTP-AUTH And TLS

Now let's install Postfix and Cyrus-SASL:

yast2 -i postfix cyrus-sasl cyrus-sasl-crammd5 cyrus-sasl-digestmd5 cyrus-sasl-gssapi cyrus-sasl-otp cyrus-sasl-plain cyrus-sasl-saslauthd procmail

Then we add the system startup links for Postfix and saslauthd and start them:

chkconfig --add postfix
/etc/init.d/postfix start

chkconfig --add saslauthd
/etc/init.d/saslauthd start

Afterwards we create the certificates for TLS:

mkdir /etc/postfix/ssl
cd /etc/postfix/ssl/
openssl genrsa -des3 -rand /etc/hosts -out smtpd.key 1024

chmod 600 smtpd.key
openssl req -new -key smtpd.key -out smtpd.csr

openssl x509 -req -days 3650 -in smtpd.csr -signkey smtpd.key -out smtpd.crt

openssl rsa -in smtpd.key -out smtpd.key.unencrypted

mv -f smtpd.key.unencrypted smtpd.key
openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650

Next we configure Postfix for SMTP-AUTH and TLS:

postconf -e 'mydomain = example.com'
postconf -e 'myhostname = server1.$mydomain'
postconf -e 'mynetworks = 127.0.0.0/8'
postconf -e 'smtpd_sasl_local_domain ='
postconf -e 'smtpd_sasl_auth_enable = yes'
postconf -e 'smtpd_sasl_security_options = noanonymous'
postconf -e 'broken_sasl_auth_clients = yes'
postconf -e 'smtpd_sasl_authenticated_header = yes'
postconf -e 'smtpd_recipient_restrictions =
permit_sasl_authenticated,permit_mynetworks,check_relay_domains'
postconf -e 'inet_interfaces = all'
postconf -e 'alias_maps = hash:/etc/aliases'
postconf -e 'smtpd_tls_auth_only = no'
postconf -e 'smtp_use_tls = yes'
postconf -e 'smtpd_use_tls = yes'
postconf -e 'smtp_tls_note_starttls_offer = yes'
postconf -e 'smtpd_tls_key_file = /etc/postfix/ssl/smtpd.key'
postconf -e 'smtpd_tls_cert_file = /etc/postfix/ssl/smtpd.crt'
postconf -e 'smtpd_tls_CAfile = /etc/postfix/ssl/cacert.pem'
postconf -e 'smtpd_tls_loglevel = 1'
postconf -e 'smtpd_tls_received_header = yes'
postconf -e 'smtpd_tls_session_cache_timeout = 3600s'
postconf -e 'tls_random_source = dev:/dev/urandom'

(Make sure you use the right hostname/domain in mydomain and myhostname!)

To enable TLS connections in Postfix, edit /etc/postfix/master.cf and uncomment the tlsmgr line so that it looks like this one:

vi /etc/postfix/master.cf

```
[...]
tlsmgr    unix  -       -       n       1000?   1       tlsmgr
[...]
```

Now restart Postfix:

/etc/init.d/postfix restart

To see if SMTP-AUTH and TLS work properly now run the following command:

telnet localhost 25

After you have established the connection to your Postfix mail server type

ehlo localhost

If you see the lines

250-STARTTLS

and

250-AUTH PLAIN LOGIN

then everything is fine.

On my system the output looks like this:

```
server1:/etc/postfix/ssl # telnet localhost 25
Trying ::1...
Connected to localhost.
Escape character is '^]'.
220 server1.example.com ESMTP Postfix
ehlo localhost
250-server1.example.com
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-AUTH PLAIN LOGIN
250-AUTH=PLAIN LOGIN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
quit
221 2.0.0 Bye
Connection closed by foreign host.
server1:/etc/postfix/ssl #
```

Type

quit

to return to the system's shell.

10 Courier-IMAP/Courier-POP3

I want to use a POP3/IMAP daemon that has Maildir support. That's why I use Courier-IMAP and Courier-POP3.

yast2 -i courier-imap fam-server courier-authlib expect tcl

Afterwards we add the system startup links and start POP3, IMAP, POP3s and IMAPs:

```
chkconfig --add fam
chkconfig --add courier-authdaemon
chkconfig --add courier-pop
chkconfig --add courier-imap
/etc/init.d/courier-pop start
/etc/init.d/courier-imap start
chkconfig --add courier-pop-ssl
```

chkconfig --add courier-imap-ssl
/etc/init.d/courier-pop-ssl start
/etc/init.d/courier-imap-ssl start

If you do not want to use ISPConfig, configure Postfix to deliver emails to a user's Maildir*:

postconf -e 'home_mailbox = Maildir/'
postconf -e 'mailbox_command ='
/etc/init.d/postfix restart

***Please note:** You do not have to do this (but it does not hurt ;-)) if you intend to use ISPConfig on your system as ISPConfig does the necessary configuration using procmail recipes. But please go sure to enable Maildir under Management -> Server -> Settings -> EMail in the ISPConfig web interface.

11 Apache/PHP5

Now we install Apache with PHP5:

yast2 -i apache2 apache2-devel apache2-mod_perl apache2-mod_php5 apache2-prefork perl-HTML-Parser perl-HTML-Tagset perl-Tie-IxHash perl-URI perl-libwww-perl php5 php5-devel zlib zlib-devel

Then we install some PHP5 modules:

yast2 -i php5-bcmath php5-bz2 php5-calendar php5-ctype php5-curl php5-dbase php5-dom php5-ftp php5-gd php5-gettext php5-gmp php5-iconv php5-imap php5-ldap php5-mbstring php5-mcrypt php5-mysql php5-ncurses php5-odbc php5-openssl php5-pcntl php5-pgsql php5-posix php5-shmop php5-snmp php5-soap php5-sockets php5-sqlite php5-sysvsem php5-tokenizer php5-wddx php5-xmlrpc php5-xsl php5-zlib php5-exif php5-fastcgi php5-pear php5-sysvmsg php5-sysvshm ImageMagick curl

Next we edit /etc/apache2/httpd.conf:

vi /etc/apache2/httpd.conf

and change DirectoryIndex to

```
[...]
DirectoryIndex index.html index.htm index.shtml index.cgi index.php index.php5
index.php4 index.php3 index.pl index.html.var index.aspx default.aspx
[...]
```

Edit /etc/sysconfig/apache2 and add rewrite to the APACHE_MODULES line:

vi /etc/sysconfig/apache2

```
[...]
APACHE_MODULES="actions alias auth_basic authn_file authz_host authz_groupfile
authz_default authz_user authn_dbm autoindex cgi dir env expires include
log_config mime negotiation setenvif ssl suexec userdir php5 rewrite"
[...]
```

Also add SSL to the APACHE_SERVER_FLAGS line:

```
[...]
APACHE_SERVER_FLAGS="SSL"
[...]
```

Now configure your system to start Apache at boot time:

chkconfig --add apache2

Then run

SuSEconfig
/etc/init.d/apache2 start

11.1 Disable PHP And Perl Globally

(If you do not plan to install ISPConfig on this server, please skip this section!)

In ISPConfig you will configure PHP and Perl on a per-website basis, i.e. you can specify which website can run PHP and Perl scripts and which one cannot. This can only work if PHP and Perl are disabled globally because otherwise all websites would be able to run PHP/Perl scripts, no matter what you specify in ISPConfig.

To disable PHP and Perl globally, we edit /etc/mime.types and comment out the application/x-perl and application/x-php lines:

vi /etc/mime.types

```
[...]
#application/x-perl pl pm al perl
#application/x-php php php3 php4
[...]
```

Then edit /etc/apache2/conf.d/php5.conf and comment out all AddHandler lines:

vi /etc/apache2/conf.d/php5.conf

```
<IfModule mod_php5.c>
        #AddHandler application/x-httpd-php .php4
        #AddHandler application/x-httpd-php .php5
        #AddHandler application/x-httpd-php .php
        #AddHandler application/x-httpd-php-source .php4s
        #AddHandler application/x-httpd-php-source .php5s
        #AddHandler application/x-httpd-php-source .phps
        DirectoryIndex index.php4
        DirectoryIndex index.php5
        DirectoryIndex index.php
</IfModule>
```

Afterwards we restart Apache: /etc/init.d/apache2 restart

11.2 mod_ruby

OpenSUSE 11.1 doesn't have a mod_ruby package, therefore we must compile it manually. First we install the prerequisites:

yast2 -i apache2-devel ruby ruby-devel

Afterwards we build mod_ruby as follows:

cd /tmp
wget http://www.modruby.net/archive/mod_ruby-1.3.0.tar.gz
tar zxvf mod_ruby-1.3.0.tar.gz
cd mod_ruby-1.3.0/
./configure.rb --with-apr-includes=/usr/include/apr-1

make
make install

To enable mod_ruby, we open /etc/sysconfig/apache2 and add ruby to the APACHE_MODULES line, e.g. like this:

vi /etc/sysconfig/apache2

```
[...]
APACHE_MODULES="actions alias auth_basic authn_file authz_host authz_groupfile
authz_default authz_user authn_dbm autoindex cgi dir env expires include
log_config mime negotiation setenvif ssl suexec userdir php5 rewrite ruby"
[...]
```

Afterwards we run

SuSEconfig

and restart Apache:

/etc/init.d/apache2 restart

11.3 mod_python

To install mod_python, we simply run:

yast2 -i apache2-mod_python

To enable mod_python, open /etc/sysconfig/apache2 and add python to the APACHE_MODULES line, e.g. like this:

vi /etc/sysconfig/apache2

```
[...]
APACHE_MODULES="actions alias auth_basic authn_file authz_host authz_groupfile
authz_default authz_user authn_dbm autoindex cgi dir env expires include
log_config mime negotiation setenvif ssl suexec userdir php5 rewrite ruby
python"
[...]
```

Afterwards, run

SuSEconfig

and restart Apache:

/etc/init.d/apache2 restart

12 Proftpd

I want to use ProFTPd instead of vsftpd which is SUSE's default FTP server because the control panel software I am going to install on this server (ISPConfig) works better with ProFTPd on OpenSUSE 11.1. Since there are no OpenSUSE packages for ProFTPd I have to compile it manually.

First we install some prerequisites:

yast2 -i libcap libcap-devel

Then we build ProFTPd as follows:

cd /tmp/
wget --passive-ftp ftp://ftp.proftpd.org/distrib/source/proftpd-1.3.2rc3.tar.gz
tar xvfz proftpd-1.3.2rc3.tar.gz
cd proftpd-1.3.2rc3/
./configure --sysconfdir=/etc
make
make install
cd ..
rm -fr proftpd-1.3.2rc3*

Now create the file /etc/init.d/proftpd:

vi /etc/init.d/proftpd

```
#! /bin/sh
# Copyright (c) 2000-2001 SuSE GmbH Nuernberg, Germany.
# All rights reserved.
#
# Original author: Marius Tomaschewski <mt@suse.de>
#
# Slightly modified in 2003 for use with SuSE Linux 8.1,
# by http://www.learnlinux.co.uk/
#
# Slightly modified in 2005 for use with SuSE Linux 9.2,
# by Falko Timme
#
# /etc/init.d/proftpd
#
### BEGIN INIT INFO
# Provides:               proftpd
# Required-Start:        $network $remote_fs $syslog $named
# Required-Stop:
# Default-Start:        3 5
# Default-Stop:        0 1 2 6
# Description:                Starts ProFTPD server
### END INIT INFO
# Determine the base and follow a runlevel link name.
base=${0##*/}
link=${base#*[SK][0-9][0-9]}
# Force execution if not called by a runlevel directory.
test $link = $base && START_PROFTPD=yes  # Modified by learnlinux.co.uk
test "$START_PROFTPD" = yes || exit 0    # Modified by learnlinux.co.uk
# Return values acc. to LSB for all commands but
# status (see below):
#
# 0 - success
# 1 - generic or unspecified error
# 2 - invalid or excess argument(s)
# 3 - unimplemented feature (e.g. "reload")
# 4 - insufficient privilege
# 5 - program is not installed
# 6 - program is not configured
# 7 - program is not running
proftpd_cfg="/etc/proftpd.conf"
proftpd_bin="/usr/local/sbin/proftpd"
proftpd_pid="/usr/local/var/proftpd.pid"
[ -r $proftpd_cfg ] || exit 6
[ -x $proftpd_bin ] || exit 5
# Source status functions
. /etc/rc.status
# First reset status of this service
rc_reset
```

```
case "$1" in
    start)
  echo -n "Starting ProFTPD Server: "
  test -f /etc/shutmsg && rm -f /etc/shutmsg
  /sbin/startproc $proftpd_bin
  rc_status -v
  ;;
    stop)
  echo -n "Shutting down ProFTPD Server: "
  test -x /usr/local/sbin/ftpshut && /usr/local/sbin/ftpshut now && sleep 1
  /sbin/killproc -TERM $proftpd_bin
  test -f /etc/shutmsg && rm -f /etc/shutmsg
  rc_status -v
  ;;
    restart)
  ## If first returns OK call the second, if first or
  ## second command fails, set echo return value.
  $0 stop
  $0 start
  rc_status
  ;;
    try-restart)
  ## Stop the service and if this succeeds (i.e. the
  ## service was running before), start it again.
  ## Note: not (yet) part of LSB (as of 0.7.5)
  $0 status >/dev/null &&  $0 restart
  rc_status
  ;;
    reload|force-reload)
  ## Exclusive possibility: Some services must be stopped
  ## and started to force a new load of the configuration.
  echo -n "Reload ProFTPD Server: "
  /sbin/killproc -HUP $proftpd_bin
  rc_status -v
  ;;
    status)
  # Status has a slightly different for the status command:
  # 0 - service running
  # 1 - service dead, but /var/run/  pid  file exists
  # 2 - service dead, but /var/lock/ lock file exists
  # 3 - service not running
  echo -n "Checking for ProFTPD Server: "
  checkproc $proftpd_bin
  rc_status -v
  ;;
    probe)
  ## Optional: Probe for the necessity of a reload,
  ## give out the argument which is required for a reload.
  [ $proftpd_cfg -nt $proftpd_pid ] && echo reload
  ;;
    *)
  echo "Usage: $0 {start|stop|status|restart|reload|try-restart|probe}"
  exit 1
  ;;
esac
# Set an exit status.
rc_exit
```

Then run

chmod 755 /etc/init.d/proftpd
chkconfig --add proftpd

Start ProFTPd:

/etc/init.d/proftpd start

If you get the following error...

Starting ProFTPD Server: - Fatal: UseIPv6: Use of the UseIPv6 directive requires IPv6 support (--enable-ipv6) on line 14 of '/etc/proftpd.conf'
startproc: exit status of parent of /usr/local/sbin/proftpd: 1

... open /etc/proftpd.conf and comment out or remove the UseIPv6 line:

vi /etc/proftpd.conf

```
[...]
# Don't use IPv6 support by default.
#UseIPv6                                off
[...]
```

For security reasons you can add the following lines to /etc/proftpd.conf:

vi /etc/proftpd.conf

```
[...]
DefaultRoot ~
IdentLookups off
ServerIdent on "FTP Server ready."
[...]
```

Be sure to comment out the following lines in order to allow ftp users to CHMOD:

```
[...]
# Bar use of SITE CHMOD by default
#<Limit SITE_CHMOD>
#  DenyAll
#</Limit>
[...]
```

and restart ProFTPd:

/etc/init.d/proftpd restart

13 Webalizer

To install webalizer, just run

yast2 -i webalizer

14 Synchronize the System Clock

If you want to have the system clock synchronized with an NTP server do the following:

yast2 -i xntp

Then add system startup links for ntp and start ntp:

chkconfig --add ntp
/etc/init.d/ntp start

15 Install some Perl Modules needed by SpamAssassin (comes with ISPConfig)

Run

yast2 -i perl-HTML-Parser perl-Net-DNS perl-Digest-SHA1

16 Disable AppArmor

AppArmor is a security extension of SUSE (similar to Fedora's SELinux) that should provide extended security. In my opinion you don't need it to configure a secure system, and it usually causes more problems than advantages (think of it after you have done a week of trouble-shooting because some service wasn't working as expected, and then you find out that everything was ok, only AppArmor was causing the problem). Therefore I disable it (this is a must if you want to install ISPConfig later on).

We can disable it like this:

/etc/init.d/boot.apparmor stop
chkconfig -d boot.apparmor

17 The End

The configuration of the server is now finished, and if you wish you can now install ISPConfig on it, following these instructions: http://www.ispconfig.org/manual_installation.htm

Make sure you check out the ISPConfig 2.x - First Steps guide after the installations. One absolutely necessary step to make PHP work with ISPConfig on OpenSUSE is described in chapter 2.4.3 of that guide:

Open /home/admispconfig/ispconfig/lib/config.inc.php...

vi /home/admispconfig/ispconfig/lib/config.inc.php

... and change $go_info["server"]["apache2_php"] to addhandler:

```
[...]
$go_info["server"]["apache2_php"] = 'addhandler';
[...]
```

17.1 A Note On SuExec

If you want to run CGI scripts under suExec, you should specify /srv/www as the web root for websites created by ISPConfig as SUSE's suExec is compiled with /srv/www as Doc_Root. Run

/usr/sbin/suexec2 -V

and the output should look like this:

server1:~ # /usr/sbin/suexec2 -V
 -D AP_DOC_ROOT="/srv/www"
 -D AP_GID_MIN=96
 -D AP_HTTPD_USER="wwwrun"
 -D AP_LOG_EXEC="/var/log/apache2/suexec.log"
 -D AP_SAFE_PATH="/usr/local/bin:/usr/bin:/bin"
 -D AP_UID_MIN=96

```
 -D AP_USERDIR_SUFFIX="public_html"
server1:~ #
```

So if you want to use suExec with ISPconfig, don't change the default web root (which is /srv/www) if you use expert mode during the ISPConfig installation (in standard mode you can't change the web root anyway so you'll be able to use suExec in any case).

18 Links

- OpenSUSE: http://www.opensuse.org
- ISPConfig: http://www.ispconfig.org