

Kolab Server

Technical Description



Erlewein, Frank, Konold & Partner
Beratende Ingenieure und Physiker
Nobelstr. 15
70569 Stuttgart
Germany
<http://www.erfrakon.de>
email: info@erfrakon.de

Kolab Server: Technical Description

by  **erfrakon**

Erlewein, Frank, Konold & Partner
Beratende Ingenieure und Physiker
Nobelstr. 15
70569 Stuttgart
Germany
<http://www.erfrakon.de>
email: info@erfrakon.de

Published February 21th, 2003

This documentation was written in SGML using the DocBook DTD. HTML and Postscript output is generated automatically and depends on the tools used.

Windows, Exchange, and Outlook are registered trademarks of Microsoft Corporation Inc. Insight Connector is a registered trademark of Bynari Inc. HotSync is a registered trademark of Palm Inc. All other herein mentioned trademarks belong to their respective owners.

Revision History

Revision 1.0 February 20th, 2003

Revision 1.0.1 February 21th, 2003

Table of Contents

1. Software Selection	1
2. Reference Platform	3
3. OpenPKG	4
3.1. Why use OpenPKG.....	4
3.2. OpenPKG setup.....	4
3.2.1. Remarks about Bootstrapping OpenPKG.....	4
3.2.2. Installation of prerequisite Software Packages.....	6
3.2.3. Logging of OpenPKG Server Packages.....	7
3.2.4. OpenPKG System Startup Procedure.....	7
4. Kolab Quick Installation	8
4.1. Setup of the Kolab server.....	8
4.1.1. Kroupware CVS.....	8
4.2. Quick Kolab Installation.....	8
5. OpenSSL Certificates	10
6. OpenLDAP2 Directory Server	12
6.1. Configuring SLAPD.....	12
6.2. LDAP Directory Design.....	15
6.2.1. Internet Domain Oriented.....	15
6.2.2. Organisation Oriented.....	15
6.2.3. Location Oriented.....	16
7. Cyrus SASL2 Library	17
8. Postfix Mail Server	22
9. Cyrus IMAP Daemon	23
10. ProFTP Daemon	31
11. Apache Web Server	36
12. Monit Daemon	56
13. Getting Things Together	61
13.1. Kolab Runtime Configuration.....	61
13.2. Kolab Administrative Web Interface.....	63
13.3. Kolab Installation Process.....	63
13.3.1. Services installation.....	63
13.3.2. Kolab Bootstrapping.....	64
13.3.3. Kolab configuration.....	69
13.4. LDAP Directory Schema.....	73
13.4.1. Person.....	73
13.4.2. OrganisationalPerson.....	73
13.5. LDAP Server Configuration.....	73
13.6. SASL.....	75
13.7. Postfix Configuration.....	75
13.8. Cyrus IMAP Daemon.....	77
13.9. Apache Webserver.....	77
13.9.1. Global configuration.....	77
13.9.2. Providing free/busy.....	79

13.9.3. Web administration.....	80
13.10. ProFTPD (Legacy Support)	80
14. Possible Extensions	82
14.1. Virus Checking.....	82
14.2. Spam Filtering	82
14.3. Webmail Interface	82
14.4. LDAP over TLS	82
14.5. LDAP Migration Tool	82
15. Appendix.....	84
15.1. Handling Kolab's LDAP database on the commandline.....	84

Chapter 1. Software Selection

The Kolab server is based on widely available and proven to be stable software components. All software used falls into the category Open Source Software or even Free Software and so is not subject to commercial licensing restrictions.

The major task of the herein described activity is the integration of the Kolab server's components. The experienced difficulties result from the situation that only the latest software versions were chosen and some of them are not very well documented, with regard to interoperation between them.

The Kolab server is build using the following software projects:

1. Carnegie Mellon University Cyrus IMAP Daemon (<http://asg.web.cmu.edu/cyrus/imapd/>) (with Cyrus Sieve) version 2.1.11
2. Carnegie Mellon University Cyrus Simple Authentication and Security Layer (SASL) (<http://asg.web.cmu.edu/cyrus/sasl/>) version 2.1.10 (SASL 2.1+ is needed by Cyrus IMAP Daemon version 2.1+)
3. Berkeley DB (<http://www.sleepycat.com>) version 4.1.25 (DB is needed by Cyrus IMAP Daemon and Postfix)
4. OpenSSL (<http://www.openssl.org>) version 0.9.7 (OpenSSL 0.9.4+ is needed by Cyrus IMAP Daemon version 2.1+)
5. OpenLDAP (<http://www.openldap.org>) version 2.1.12
6. Postfix (<http://www.postfix.org>) version 2.0.3 with TLS extensions version 0.8.11a
7. ProFTP Daemon (<http://www.proftpd.org>) version 1.2.8rc1
8. Apache Webserver (<http://www.apache.org>) version 1.3.27
9. Monit (<http://www.tildeslash.com/monit/>) version 3.1

Key features that lead to the Cyrus IMAP daemon for Kolab:

- the scalability of this IMAP server implementation was shown in numerous earlier applications
- support for access control lists
- support for shared IMAP folders
- decoupling of mail users from system accounts
- maildir formatted mailboxes
- use of the filesystem as a hierarchical database to store variable length emails with an efficient Berkeley DB driven indexing mechanism
- support of IMAP and POP3 over SSL/TLS
- user authentication against LDAP via the SASL2 library
- support of the flexible server scripting language Sieve

- support of email delivery using the LMTP mechanism

The Postfix Mail Server was chosen with the following features in mind:

- easy configurability
- support of smtp over SSL/TLS
- allows SMTP authentication against LDAP via the SASL library
- good security record
- support of email delivery using the LMTP mechanism

Features that lead to the selection of Apache were :

- authentication against LDAP
- HTTP over SSL/TLS
- Webdav support
- server side scripting support via PHP

The Cyrus SASL library is merely a requirement from the IMAP daemon and Postfix. Nevertheless it provides an excellent flexible authentication layer supporting authentication against passwd, shadow, PAM, Kerberos, LDAP, and others

The community process inspired further thoughts which we want to list here.

1. The Courier IMAP daemon seems to be a valid alternative to the Cyrus IMAP daemon. In future research has to be done whether it makes sense to have it in Kolab alternatively.
2. Security aware people tend to avoid PAM for various reasons. Mostly architectural weaknesses were pointed out. We addressed this by using the experimental direct LDAP binding of the SASL authentication daemon. Note that we found this feature to be quite stable and usable and that the variety of the SASL supported mechanisms is still open to the Kolab users.
3. Exim is equally powerful to postfix. It also has an excellent security record. It will surely be desirable to develop a way to have it in Kolab, in future. At this point it was decided to stick with postfix simply for the personal affection of the authors.

Note that replacing parts of the server software will under all circumstances require to modify the administration tools, provided by a later activity and a different paper.

Chapter 2. Reference Platform

The server development is done using an x86 PC with the following specification. Note that the relevant Kolab server software is provided by the OpenPKG system which will be described in the next chapter.

- Hardware: Dual Pentium II, 256 MB
- Software: Debian Sid/Sarge Testing/Unstable
- Kernel: 2.4.18 SMP

Chapter 3. OpenPKG

3.1. Why use OpenPKG

The OpenPKG Project (<http://www.openpkg.org>) "... is a project founded by the Development Team from Cable & Wireless Deutschland's Application Services division. The goal is the creation and maintainance of portable and easy to install software packages for use on the major Unix server platforms" (cite from the project homepage).

The license of OpenPKG is MIT style for the packaging code and a mixture of licenses from all the software packages which are distributed within the OpenPKG environment. All of those licences meet the requirement of the Open Source Initiative (<http://opensource.org/licenses/>).

A major goal for the Kolab server is to not be focused on one single Linux distribution. OpenPKG is designed for having a compatible single software repository for open source and free software over different Unix flavours, FreeBSD, and all Linux distributions. OpenPKG is also designed to be minimal invasive to the underlying operating system.

Within the Kolab project most of the server software needs to be recent. Unfortunately the major Linux distributions provide the needed server software in different and mostly elder releases - for rather good and understandable reasons. To achieve the availability of the Kolab server on different Linux distributions or other Unix flavours and at the same time avoid platform-specific packaging led to a dilemma.

Therefore a decision was made to use the OpenPKG packaging system. OpenPKG uses the RedHat Package Manager version 4. All relevant options for compiling and installing the Kolab software packages are supported by RPM spec files.

We use the version 1.2.0 of OpenPKG and a variety of more recent packages from the OpenPKG Current branch for the Kolab server. The results are to be submitted to the OpenPKG project where applicable and should also apply for forthcoming versions of OpenPKG.

3.2. OpenPKG setup

OpenPKG is installed via a bootstrapping mechanism which creates a separate filesystem branch beneath a prefix on the filesystem of the machine - a good choice might be `/kolab`. A quick tutorial about the OpenPKG bootstrapping is available online (<http://www.openpkg.org/doc/quickref/openpkg.txt>).

3.2.1. Remarks about Bootstrapping OpenPKG

Bootstrapping of OpenPKG from source is the preferable option. OpenPKG requires a set of special users and groups in the system. It's important to use the special `openpkg` user when making changes to the OpenPKG installation, to avoid unintended interference with the underlying Unix system. We recommend to bootstrap the OpenPKG environment with the following commands (depending on the version of OpenPKG you are installing):

```
sh openpkg-XXX.src.sh --prefix=/kolab --user=kolab --group=kolab -v
sh openpkg-XXX.ix86-linux2.4-kol.sh
```

The bootstrap process will create the following directory structure under the prefix `/kolab`:

```
RPM
bin
cgi
etc
include
info
lib
libexec
local
man
pub
sbin
share
var
```

All binaries, config directories, logfiles etc. of the OpenPKG software live under the prefix directory `/kolab`.

You are now ready to continue the setup of the OpenPKG system which acts as base of the Kolab server installation. Doing so means using `/kolab/bin/rpm` to get the appropriate packages into the system. Note that since OpenPKG 1.2 you must do this as root. Use caution to not confuse `openpkg`'s `rpm` with an eventual `rpm` of the underlying linux distribution. Packages used by OpenPKG will not fit in other `rpm` based distributions like RedHat or SuSE because of the required basepackage of `openpkg`. This way it can be assured that the OpenPKG RPM database does not interfere with an eventual existing RPM database of the underlying system.

As many of the services used by the Kolab server must be compiled using non-OpenPKG options we supply own RPM spec files.

3.2.2. Installation of prerequired Software Packages

Following the OpenPKG instructions a choice of software packages were built from source and installed into the OpenPKG environment of Kolab. Remember that the order in which you install the packages is also important as there are package dependencies. Therefore we provided a working installation order of the packages. The following packages are the required minimum for the Kolab components:

1. make-3.80-1.2.0
2. binutils-2.13.2.1-1.2.0
3. patch-2.5.8-1.2.0
4. libtool-1.4.3-1.2.0
5. gcc-3.2.1-1.2.0
6. zlib-1.1.4-1.2.0
7. db-4.1.25.0-1.2.0
8. bzip2-1.0.2-1.2.0
9. expat-1.95.5-1.2.0
10. readline-4.3-1.2.0
11. perl-5.8.0-1.2.0
12. m4-1.4o-1.2.0
13. bison-1.35-1.2.0
14. flex-2.5.4a-1.2.0
15. autoconf-2.57-1.2.0
16. openssl-0.9.7-1.2.0
17. pcre-3.9.1-1.2.0
18. perl-crypto-1.2.0-1.2.0
19. perl-conv-1.2.0-1.2.0
20. perl-ds-1.2.0-1.2.0
21. perl-util-1.2.0-1.2.0
22. perl-net-1.2.0-1.2.0
23. perl-term-1.2.0-1.2.0
24. perl-mail-1.2.0-1.2.0
25. perl-time-1.2.0-1.2.0
26. perl-ssl-1.2.0-1.2.0
27. perl-sys-1.2.0-1.2.0

28. perl-www-1.2.0-1.2.0
29. perl-xml-1.2.0-1.2.0
30. perl-ldap-1.2.0-1.2.0
31. gdbm-1.8.3-1.2.0
32. procmail-3.22-1.2.0
33. automake-1.7.2-1.2.0
34. mm-1.2.2-1.2.0

Note that it makes no sense to proceed to later chapters of this documentation unless the bootstrapping has successfully been completed. It is also recommendable to read the OpenPKG documentation and get familiar with its concept and the maintenance, namely the RPM package manager.

Note also that OpenPKG RPM, SPEC files, and RPM packages can not be used with other RPM based Linux distributions like RedHat and SuSE!

3.2.3. Logging of OpenPKG Server Packages

OpenPKG introduces the Fake Syslog Library (<http://www.oss.org/pkg/lib/fsl/>) to separate the server software from the syslog procedure of the underlying Unix system. We had problems with fsl including the new revised version of OpenPKG 1.2. Therefore all services used for the Kolab server uses the syslog functionality of the underlying Linux distribution.

3.2.4. OpenPKG System Startup Procedure

OpenPKG places a single SystemV startup script inside the systems `/etc/init.d` directory. From there it rolls out an own single runlevel structure under `/kolab/etc/rc.d`

We describe later how the Kolab services fit in there.

Chapter 4. Kolab Quick Installation

4.1. Setup of the Kolab server

If you succeeded in installing the OpenPKG environment described above you can now install the core services of the Kolab server.

4.1.1. Kroupware CVS

It is recommended that you use the CVS that is provided by Intevation. This is the place where bugfixes will be first checked in and provided to the public. Please set the following environment variable. This is done via `export` if you use the bash shell.

```
CVSROOT=:pserver:anonymous@intevation.de:/home/kroupware/  
jail/kolabrepository
```

A web interface to the CVS is available at <http://intevation.de/cgi-bin/viewcvs-kolab.cgi/server/> (<http://intevation.de/cgi-bin/viewcvs-kolab.cgi/server/>).

To download the necessary files used for the Kolab server issue the following commands:

1. (as kolab user) `cd /kolab/RPM/SRC`
2. `cvs co -d apache server/apache`
3. `cvs co -d imapd server/imapd`
4. `cvs co -d kolab server/kolab`
5. `cvs co -d monit server/monit`
6. `cvs co -d openldap server/openldap`
7. `cvs co -d postfix server/postfix`
8. `cvs co -d proftpd server/proftpd`
9. `cvs co -d sasl server/sasl`

4.2. Quick Kolab Installation

The installation order of the packages is important as there are dependencies between them. A list of packages as of the time of writing this document is listed here.

1. `openldap-2.1.12`

2. sasl-2.1.10
3. proftpd-1.2.8rc1
4. postfix-2.0.3
5. cyrus-imapd-2.1.11
6. monit-3.1
7. apache-1.3.27
8. kolab-1.0beta2

As most of those packages need to be modified we provide specfiles and a Makefile along with some needed patches. So you should not download the OpenPKG RPM files provided for that open source projects and use instead our CVS Versions of the specfiles. All data needed for compilation and installation of the packages can be downloaded via anonymous CVS. Of course we do not provide the source packages of all the different services. But the rpm build mechanism automatically downloads the needed tarballs from their originating site before compilation by using the wget mechanism.

There are pre-build source and binary RPM packages available. Refer to KDE FTP Server (<ftp://ftp.kde.org/pub/kde/unstable/server/kolab/>) for the latest version.

Installing the packets should end with the installation of the kolab RPM. At this point kolab does bootstraps itself. Thereby creating a random password used for manager access of the server. You should note that password in order to be able to log into the webinterface as manager. At any time you can lookup the password in the created configuration file `/kolab/etc/openldap/slapd.conf`

Chapter 5. OpenSSL Certificates

OpenSSL certificates are required at several services on the Kolab server. As you install the kolab RPM (see next chapter) the bootstrapping procedure of the Kolab server automatically creates certificates that can be used with the Kolab server. Three keys are provided by the bootstrapping procedure:

- /kolab/etc/kolab/cert.pem
- /kolab/etc/kolab/key.pem
- /kolab/etc/kolab/CAcert.pem

If you want to use your own keys feel free to do so. Of course you should check with the configuration files of Cyrus IMAPd, Postfix and Apache in order to change the certificate settings. Here are the relevant configuration options which have to be adapted if you want to change the kolab created keys:

```
/kolab/etc/imapd/imapd.conf:
...
tls_cert_file:           /kolab/etc/kolab/cert.pem
tls_key_file:           /kolab/etc/kolab/key.pem
...

/kolab/etc/apache/apache.conf:
...
SSLCACertificateFile    /kolab/etc/kolab/CAcert.pem
...
SSLCertificateFile      /kolab/etc/kolab/cert.pem
SSLCertificateKeyFile   /kolab/etc/kolab/key.pem
...

/kolab/etc/postfix/main.cf:
...
smtpd_tls_CAfile = /kolab/etc/kolab/CAcert.pem
smtpd_tls_cert_file = /kolab/etc/kolab/cert.pem
smtpd_tls_key_file = /kolab/etc/kolab/key.pem
...
```

Note that by default the CA and the keys will have one year maximum lifetime. The lifetime of a CA can be up to ten years if one sets the used key length to 2048 bit. The TLS keys for the services can be set to five years, the key length does not need to be longer than 1536 bits in order to stay secure.

The kolab bootstrapping procedure creates the certificates using the following script (it is called kolab_sslcert.sh):

```
#!/bin/sh

echo "Generating kolab's SSL/TLS certificates"

PWD=`pwd`
```

```

TMPDIR="@@@kolab_prefix@@@/etc/kolab/tmp"
mkdir $TMPDIR
mkdir -p $TMPDIR/demoCA/private/
mkdir -p $TMPDIR/demoCA/newcerts
mkdir -p $TMPDIR/demoCA/certs
mkdir -p $TMPDIR/demoCA/crl
cd $TMPDIR
touch demoCA/index.txt
echo "01" > demoCA/serial

echo -n "generate self-signed CA ... "
echo -e ".\n.\n.\n.\n.\n'\n'hostname'\n.\n" | \
    @@@kolab_prefix@@@/bin/openssl req -new -x509 -nodes \
        -keyout demoCA/private/akey.pem \
        -out demoCA/cacert.pem -days 3650 2>/dev/null
echo "done"

echo -n "generate certificate and sign request ... "
echo -e ".\n.\n.\n.\n.\nkolab\n.\n\n" | \
    @@@kolab_prefix@@@/bin/openssl req -new -nodes \
        -keyout key.pem -out newreq.pem \
        -days 3650 2>/dev/null
cat newreq.pem key.pem > new.pem
echo "done"

echo -n "sign certificate with newly created CA ... "
echo -e "y\n" | @@@kolab_prefix@@@/bin/openssl ca \
    -policy policy_anything \
    -out cert.pem -infiles new.pem 2>/dev/null 1>&2
sleep 2
echo "done"

cp demoCA/cacert.pem @@@kolab_prefix@@@/etc/kolab/CAcert.pem
cp key.pem @@@kolab_prefix@@@/etc/kolab/key.pem
cp cert.pem @@@kolab_prefix@@@/etc/kolab/cert.pem
cd $PWD
rm -rf $TMPDIR

echo "New certificates have been installed under \
    @@@kolab_prefix@@@/etc/kolab/"

```

Chapter 6. OpenLDAP2 Directory Server

6.1. Configuring SLAPD

The OpenPKG OpenLDAP package does by default include SASL support and does not support the fsl library. OpenLDAP with SASL is not needed by the Kolab server setup and creates an unfortunate bi-directional dependency to the SASL package. The SPEC file with some small additions follows.

```
# package information
Name:          openldap
Summary:       Lightweight Directory Access Protocol (LDAP) Toolkit
URL:           http://www.openldap.org/
Vendor:        OpenLDAP Project
Packager:      The OpenPKG Project
Distribution:  OpenPKG [PLUS]
Group:         Database
License:       GPL
Version:       2.1.12
Release:       1.2.0

%option        with_fsl          no
%option        with_sasl         no

# list of sources
%define        url                ftp://ftp.openldap.org/pub/openldap/
                                openldap-release/openldap-%{version}.tgz
Source0:       %( [ ! -f %{SOURCE openldap-%{version}.tgz} ] \
                && wget -c %url; echo %url )
Source1:       rc.openldap
Source2:       Makefile

# build information
Prefix:        %{l_prefix}
BuildRoot:     %{l_buildroot}
BuildPreReq:   OpenPKG, openpkg >= 1.2.0, openssl,
                db >= 4.1.24, make, gcc
PreReq:        OpenPKG, openpkg >= 1.2.0
%if "%{with_fsl}" == "yes"
BuildPreReq:   fsl
PreReq:        fsl
%endif
%if "%{with_sasl}" == "yes"
BuildPreReq:   sasl
PreReq:        sasl
%endif
AutoReq:       no
AutoReqProv:   no
```

```

%description
    OpenLDAP is an open source implementation of the Lightweight
    Directory Access Protocol. The suite includes: slapd: stand-alone
    LDAP server; slurpd:- stand-alone LDAP replication server; libraries
    implementing the LDAP protocol, and utilities, tools, and sample
    clients.

%prep
    %setup -q

%build
    ln -sf %{l_prefix}/lib/libdb.a %{l_prefix}/lib/libdb-4.a
    ln -sf %{l_prefix}/lib/libdb.a %{l_prefix}/lib/libdb4.a
    CC="%{l_cc}" \
    CFLAGS="%{l_cflags -O} -static" \
    CPPFLAGS="%{l_cppflags sasl}" \
    LDFLAGS="%{l_ldflags}" \
    ./configure \
        --prefix=%{l_prefix} \
        --localstatedir=%{l_prefix}/var/openldap \
        --enable-ldb \
        --with-ldb-api=berkeley \
        --with-ldb-module=static \
        --with-ldb-type=btree \
        --enable-slurpd \
    %if "%{with_sasl}" == "no"
        --without-cyrus-sasl \
    %endif
        --disable-shared
    %{l_make} %{l_mflags}

%install
    rm -rf $RPM_BUILD_ROOT
    %{l_shtool} subst -v -s \
        -e "s;^\((prefix[^=]*=\\).*\);\\1 $RPM_BUILD_ROOT%{l_prefix};g" \
        -e "s;^\((exec_prefix[^=]*=\\).*\);\\1 \
            $RPM_BUILD_ROOT%{l_prefix};g" \
        -e "s;^\((localstatedir[^=]*=\\).*\);\\1 \
            $RPM_BUILD_ROOT%{l_prefix}/var/openldap;g" \
        `find . -name Makefile -print`
    %{l_make} %{l_mflags} install
    rm -f $RPM_BUILD_ROOT%{l_prefix}/etc/openldap/*.default
    rm -f $RPM_BUILD_ROOT%{l_prefix}/etc/openldap/*/*.default
    %{l_shtool} mkdir -f -p -m 755 \
        $RPM_BUILD_ROOT%{l_prefix}/var/openldap
    %{l_shtool} mkdir -f -p -m 755 \
        $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d
    %{l_shtool} install -c -m 755 -e 's;@l_prefix@;%{l_prefix};g' \
        %{SOURCE rc.openldap} $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/
    %{l_rpmttool} files -v -ofiles -r$RPM_BUILD_ROOT \
        %{l_files_std} \
        '%config %{l_prefix}/etc/openldap/ldap*'

```

```
%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT
```

The Slapd LDAP server is parameterized by the config file `slapd.conf` and the LDAP schema definition which we present later.

We make use of `openldap`'s `slurpd` daemon and have modified the startup script as follows.

```
#!/l_prefix@/lib/openpkg/bash @l_prefix@/etc/rc
##
## rc.openldap -- Run-Commands for OpenLDAP Daemon
##

%config
    openldap_enable="yes"

%start -p 200 -u root
    opServiceEnabled openldap || exit 0
    if [ -f @l_prefix@/var/openldap/slapd.pid ]; then
        PID=`cat @l_prefix@/var/openldap/slapd.pid | awk '{print $1}'`
        SLAPDS=`ps -p $PID 2>/dev/null | grep -c slapd | awk '{print $1}'`
        if [ $SLAPDS -gt 0 ]; then
            echo "Warning: slapd is already running under pid $PID!"
        else
            @l_prefix@/libexec/slapd -f @l_prefix@/etc/openldap/slapd.conf
        fi
    else
        @l_prefix@/libexec/slapd -f @l_prefix@/etc/openldap/slapd.conf
    fi
    killall -9 slurpd 2>/dev/null
    sleep 1
    @l_prefix@/libexec/slurpd -f @l_prefix@/etc/openldap/slapd.conf

%stop -p 200 -u root
    opServiceEnabled openldap || exit 0
    if [ -f @l_prefix@/var/openldap/slapd.pid ]; then
        kill -INT `cat @l_prefix@/var/openldap/slapd.pid`
    fi
    killall -9 slurpd 2>/dev/null
    exit 0

%restart -u root
    opServiceEnabled openldap || exit 0
    if [ -f @l_prefix@/var/openldap/slapd.pid ]; then
        kill -INT `cat @l_prefix@/var/openldap/slapd.pid`
    fi
```

```
killall -9 slurpd 2>/dev/null
sleep 2
@l_prefix@/libexec/slapd -f @l_prefix@/etc/openldap/slapd.conf
@l_prefix@/libexec/slurpd -f @l_prefix@/etc/openldap/slapd.conf
```

6.2. LDAP Directory Design

As the community process rised various questions concerning the actual LDAP layout, a short overview is given.

Directories can be layed out quite differently. We only present here three approaches commonly used. The exact layout does not have an actual implication on the authentication or the address book of the groupware solution. But it determins more or less the extendibility of the directory in the future.

The Kolab server's administration tools must conserve the flexibility of LDAP which is simply needed at this point. At this time Kolab sticks to the domain oriented approach.

6.2.1. Internet Domain Oriented

The directory is layed out after existing internet domains, e. g. bsi.de and bund.de. This approach applies well for example to an Internet Service Provider who provides a lot of hosted domains or a university with a lot of subdomains down to the level of a specific research institute.

```
dn: dc=de top level domain
   dn: o=bsi,dc=de bsi.de
dn: cn=Alfred Mueller,o=bsi,dc=de alfred.mueller@bsi.de
dn: cn=Else Stratmann,o=bsi,dc=de else.stratmann@bsi.de
   dn: o=bund,dc=de bund.de
dn: cn=Post Master,o=bund,dc=de post.master@bund.de
(etc.)
```

6.2.2. Organisation Oriented

The directory layout is done after a organisation's internal structure and applies well to most business applications.

```
dn: o=bsi
```

```
dn: ou=administration,o=bsi
dn: cn=Alfred Mueller,ou=administration,o=bsi
dn: cn=Else Stratmann,ou=administration,o=bsi
dn: ou=postoffice,o=bsi
dn: cn=Post Master,ou=postoffice,o=bsi
(etc.)
```

6.2.3. Location Oriented

A geographical layout may also make sense. A good application scenario may be the administration of a county consisting of all local administrations down to the level of a city.

```
dn: o=bsi,c=de
dn: l=bonn,o=bsi,c=de
dn: cn=Alfred Mueller,o=bsi,l=bonn,c=de
dn: cn=Else Stratmann,o=bsi,l=bonn,c=de
dn: l=berlin,o=bsi,c=de
dn: cn=Hans Schroeder,o=bsi,l=berlin,c=de
(etc.)
```

Chapter 7. Cyrus SASL2 Library

The Cyrus SASL2 project of the Carnegie Mellon University provides a library which a number of other free software projects use for authentication, namely the Cyrus IMAP daemon and the Postfix mailserver.

Starting with version 2 the saslauthd daemon provides an authentication interface to LDAP. Note that two methods exist right now to achieve this:

1. saslauthd against LDAP via PAM (with pam-ldap)
2. saslauthd directly against LDAP

The direct LDAP support is considered experimental as of now. The OpenPKG sasl package already includes the latest source tree but has not enabled the LDAP feature in the RPM spec file. Note that the package was taken from the Current tree of OpenPKG.

The SPEC file was slightly changed to reflect the direct LDAP feature and a patch had to be applied to the configure script of the original source package:

```
# package information
Name:          sasl
Summary:       Simple Authentication and Security Layer
URL:           http://asg.web.cmu.edu/sasl/
Vendor:        Cyrus Project, CMU
Packager:      The OpenPKG Project
Distribution:  OpenPKG [PLUS]
Group:         Cryptography
License:       BSD
Version:       2.1.10
Release:       1.2.0

# package options
%option        with_fsl    no
%option        with_pam    no
%option        with_ldap   yes

# list of sources
%define url    ftp://ftp.andrew.cmu.edu/pub/cyrus-mail/
               cyrus-sasl-%{version}.tar.gz
Source0:       %( [ ! -f %{SOURCE} cyrus-sasl-%{version}.tar.gz ] \
                 && wget -c %url; echo %url )
Source1:       rc.sasl
Source2:       fsl.sasl
Patch0:        sasl-ldap.patch

# build information
```

```

Prefix:      %{l_prefix}
BuildRoot:   %{l_buildroot}
BuildPreReq: OpenPKG, openpkg >= 1.2.0, db >= 4.1.24, openssl, gcc
PreReq:      OpenPKG, openpkg >= 1.2.0
%if "%{with_fsl}" == "yes"
BuildPreReq: fsl
PreReq:      fsl
%endif
%if "%{with_pam}" == "yes"
BuildPreReq: PAM
PreReq:      PAM
%endif
%if "%{with_ldap}" == "yes"
BuildPreReq: openldap
PreReq:      openldap
%endif
AutoReq:     no
AutoReqProv: no

%description
    SASL is the Simple Authentication and Security Layer, a method
    for adding authentication support to connection-based protocols.
    To use SASL, a protocol includes a command for identifying and
    authenticating a user to a server and for optionally negotiating
    protection of subsequent protocol interactions. If its use is
    negotiated, a security layer is inserted between the protocol and
    the connection.

%prep
    %setup -q -n cyrus-sasl-%{version}
    %{l_shtool} subst \
        -e 's;^ *for dbname in ;for dbname in db ;' \
        configure
    %patch0 -p0

%build
    %{l_shtool} subst \
        -e "s;javac;javac-xxx;g" \
        -e "s;javah;javah-xxx;g" \
        -e "s;javadoc;javadoc-xxx;g" \
        configure
    CC="%{l_cc}" \
    CFLAGS="%{l_cflags -O} %{l_cppflags} -static" \
%if "%{with_fsl}" == "yes"
    LDFLAGS="%{l_ldflags} '%{l_prefix}/bin/fsl-config --all --ldflags'" \
    LIBS="-ldb '%{l_prefix}/bin/fsl-config --all --libs'" \
%else
    LDFLAGS="%{l_ldflags}" \
    LIBS="-ldb" \
%endif
    ./configure \
        --prefix=%{l_prefix} \
        --with-pluginindir=%{l_prefix}/lib/sasl \

```

```

--with-saslauthd=${l_prefix}/var/sasl/saslauthd \
--with-dbpath=${l_prefix}/var/sasl/sasldb \
--with-dblib=berkeley \
--with-openssl=${l_prefix} \
--with-bdb-incdir=${l_prefix}/include \
--with-bdb-libdir=${l_prefix}/lib \
%if "%{with_pam}" == "yes"
--with-pam \
%else
--without-pam \
%endif
%if "%{with_ldap}" == "yes"
--with-ldap=${l_prefix} \
%endif
--disable-shared \
--enable-static \
--disable-java \
--disable-sample \
--disable-krb4 \
--disable-gssapi \
--disable-otp \
--without-des \
--without-opie
%{l_make} %{l_mflags}
cd saslauthd
%{l_make} %{l_mflags} testsaslauthd

%install
rm -rf $RPM_BUILD_ROOT
%{l_make} %{l_mflags} install AM_MAKEFLAGS="DESTDIR=$RPM_BUILD_ROOT"
%{l_shtool} install -c -m 755 saslauthd/testsaslauthd \
    $RPM_BUILD_ROOT${l_prefix}/sbin/
%{l_shtool} mkdir -f -p -m 755 \
    $RPM_BUILD_ROOT${l_prefix}/etc/rc.d
%{l_shtool} install -c -m 755 \
    -e 's:@l_prefix@:${l_prefix};g' \
%if "%{with_pam}" == "yes"
    -e 's:@authmech@;pam;g' \
%else
    -e 's:@authmech@;getpwent;g' \
%endif
    %{SOURCE rc.sasl} \
    $RPM_BUILD_ROOT${l_prefix}/etc/rc.d/
%if "%{with_fsl}" == "yes"
%{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT${l_prefix}/etc/fsl
%{l_shtool} install -c -m 644 \
    -e 's:@l_prefix@:${l_prefix};g' \
    %{SOURCE fsl.sasl} \
    $RPM_BUILD_ROOT${l_prefix}/etc/fsl/
%endif
%{l_shtool} mkdir -f -p -m 700 \
    $RPM_BUILD_ROOT${l_prefix}/var/sasl/log
%{l_shtool} mkdir -f -p -m 755 \

```

```

    $RPM_BUILD_ROOT%{l_prefix}/var/sasl/saslauthd
mv $RPM_BUILD_ROOT%{l_prefix}/lib/sasl2 \
    $RPM_BUILD_ROOT%{l_prefix}/lib/sasl
strip $RPM_BUILD_ROOT%{l_prefix}/sbin/* >/dev/null 2>&1 || true
%{l_rpmtree} files -v -ofiles -r$RPM_BUILD_ROOT %{l_files_std} \
%if "%{with_fsl}" == "yes"
    '%config %{l_prefix}/etc/fsl/fsl.sasl' \
    '%not %dir %{l_prefix}/etc/fsl' \
%endif
    '%dir %attr(-,%{l_susr},{l_sgrp}) %{l_prefix}/var/sasl/log'

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

```

Two patches we had to provide for openpkg follow.

```

--- saslauthd/configure.orig Sun Nov 17 15:27:34 2002
+++ saslauthd/configure Sun Nov 17 15:42:13 2002
@@ -4012,7 +4012,7 @@
    echo $ac_n "(cached) $ac_c" 1>&6
    else
        ac_save_LIBS="$LIBS"
-LIBS="-lldap -llber $LIBS"
+LIBS="-lldap -llber -lssl -lcrypto $LIBS"
    cat > conftest.$ac_ext <<EOF
    #line 4018 "configure"
    #include "confdefs.h"
@@ -4044,7 +4044,7 @@
    #define HAVE_LDAP 1
    EOF

-
+
LDAP_LIBS="-lldap -llber"
LDAP_LIBS="-lldap -llber -lssl -lcrypto"
else
    echo "$ac_t" "no" 1>&6
fi

--- lib/Makefile.in Wed Dec 4 10:20:38 2002
+++ lib/Makefile.in Wed Dec 4 10:21:00 2002
@@ -437,7 +437,7 @@

plugin_common.lo: plugin_common.o

```

```
-      ln -s $(top_builddir)/plugins/plugin_common.lo plugin_common.lo
+      ln -sf $(top_builddir)/plugins/plugin_common.lo plugin_common.lo

plugin_common.o:
      ln -s $(top_builddir)/plugins/plugin_common.o plugin_common.o
```

A Source RPM containing above changes is available.

The community has at this point already demanded that further authentication mechanisms may be used with the Kolab server. It seems that the choices offered by the saslauthd satisfy this demand. It requires however that the LDAP database and the additional authentication database must be kept consistent. That is right now out of the focus of the initial Kolab release.

Chapter 8. Postfix Mail Server

The postfix mailservier was taken entirely from OpenPKG. No modifications were done, except that the spec file does a download of the sources in case they are not already locally available.

Chapter 9. Cyrus IMAP Daemon

On basis of the IMAP daemon's source package of OpenPKG a modified spec file was developed:

```
# package information
Name:          imapd
Summary:       Cyrus IMAP Server
URL:           http://asg.web.cmu.edu/cyrus/imapd/
Vendor:        Carnegie Mellon University
Packager:      The OpenPKG Project
Distribution:   OpenPKG [EVAL]
Group:         Mail
License:       BSD
Version:       2.1.11
Release:       20030125

%option        with_fsl no

# list of sources
%define url    ftp://ftp.andrew.cmu.edu/pub/cyrus-mail/
              cyrus-imapd-#{version}.tar.gz
Source0:       %( [ ! -f %{SOURCE cyrus-imapd-#{version}.tar.gz} ] \
                && wget -c %url; echo %url )
Source1:       imapd.conf
Source2:       rc.imapd
Source3:       Makefile
Patch0:        imapd.patch
Patch1:        groupfile.patch
Patch2:        makefile.patch
Patch3:        configure-db4.patch
Patch4:        perl.patch

# build information
Prefix:        %{l_prefix}
BuildRoot:     %{l_buildroot}
BuildPreReq:   OpenPKG, openpkg >= 1.2, sasl, db >= 4.1.24, \
              openssl, make, perl
PreReq:        OpenPKG, openpkg >= 1.2, sasl, openssl, MTA
AutoReq:       no
AutoReqProv:   no
Provides:      IMAP

%description

The Cyrus IMAP server is an IMAP4 and POP3 daemon that differs from
other IMAP server implementations in that it is generally intended to
be run on sealed servers, where normal users are not permitted to log
in. The mailbox database is stored in parts of the filesystem that are
private to the Cyrus IMAP system. All user access to mail is through
the IMAP, POP3, or KPOP protocols.
```

```

%prep
    %setup -q -n cyrus-imapd-%{version}
    %patch0 -p0
    %patch1 -p0
    %patch2 -p0
    %patch3 -p0
    %patch4 -p0

%build
    cflags="-I%{l_prefix}/include"
    ldflags="-L%{l_prefix}/lib"
    case "%{l_target}" in
        *-solaris* ) ldflags="$ldflags -lsocket -lnsl" ;;
    esac
    CC="%{l_cc}" \
    CPPFLAGS="$cflags" \
    CFLAGS="%{l_cflags} -O} $cflags" \
    LDFLAGS="$ldflags" \
%if "%{with_fsl}" == "yes"
    libs="'%{l_prefix}/bin/fsl-config --libs'" \
    ldflags="$ldflags '%{l_prefix}/bin/fsl-config --ldflags'" \
%endif
    LIBS="$libs" \
    LDFLAGS="$ldflags -L%{l_prefix}/lib/sasl" \
    ./configure \
        --prefix=%{l_prefix} \
        --libdir=%{l_prefix}/lib \
        --with-openssl=%{l_prefix} \
        --with-dbdir=%{l_prefix} \
        --with-sasl=%{l_prefix} \
        --with-statedir=%{l_prefix}/var/imapd \
        --with-auth=unix \
        --with-cyrus-prefix=%{l_prefix} \
        --with-cyrus-user=%{l_musr} \
        --with-cyrus-group=%{l_mgrp} \
        --without-ucdsnmp \
        --without-zephyr \
        --with-staticsasl=%{l_prefix} \
        --without-afs \
    --with-perl=%{l_prefix}/bin/perl \
    --with-lock=fcntl

    # redirect the hard coded file paths
    %{l_shtool} subst -e "s;/etc/\.*/.conf; \
        %{l_prefix}/etc/imapd/\.*/.conf;" \
        imap/*.c imap/*.h master/*.c master/*.h
    %{l_shtool} subst -e "s;/etc/imapd.group; \
        %{l_prefix}/etc/imapd/imapd.group;" lib/auth_unix.c
    %{l_make} %{l_mflags}
    %{l_shtool} subst -t -e "s;\(INSTALL.* = \).*\(%{l_prefix}.*\); \
        \1$RPM_BUILD_ROOT\2;g" \
    perl/imap/Makefile perl/sieve/managesieve/Makefile

```

```

%install
  rm -rf $RPM_BUILD_ROOT

  %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/imapd
  %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/var/imapd
  %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/var/spool
  %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}
                                     /var/spool/imap/sieve

  %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d
  for dir in a b c d e f g h i j k l m n o p q r s t u v w x y z
  do
    (
      %{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}
                                     /var/spool/imap/sieve/$dir
    )
  done

  %{l_make} %{l_mflags} install \
    prefix=$RPM_BUILD_ROOT%{l_prefix} \
    exec_prefix=$RPM_BUILD_ROOT%{l_prefix} \
    cyrus_prefix=$RPM_BUILD_ROOT%{l_prefix}

  # offer a sane configuration
  #cp master/conf/small.conf master/conf/cyrus.conf
  #%{l_shtool} subst -e "s;/var/imap/socket;%{l_prefix}
                                     /var/imap/socket;g" \

  # master/conf/cyrus.conf
  %{l_shtool} install -c -m 644 \
    -e 's:@l_prefix@;%{l_prefix};g' \
    -e 's:@l_musr@;%{l_musr};g' \
    %{SOURCE imapd.conf} $RPM_BUILD_ROOT%{l_prefix}
                                     /etc/imapd/imapd.conf

  %{l_shtool} install -c -m 644 \
    -e "s;/var/imap/socket;%{l_prefix}/var/imapd/socket;g" \
    master/conf/small.conf $RPM_BUILD_ROOT%{l_prefix}
                                     /etc/imapd/cyrus.conf

  # install the run command file
  %{l_shtool} install -c -m 755 -e 's:@l_prefix@;%{l_prefix};g' \
    -e 's:@l_musr@;%{l_musr};g' -e 's:@l_mgrp@;%{l_mgrp};g' \
    %{SOURCE rc.imapd} $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/

  # use mkimap to create many directories for us
  cp $RPM_BUILD_ROOT%{l_prefix}/etc/imapd/imapd.conf imapd.conf.hack
  %{l_shtool} subst -e "s;%{l_prefix};$RPM_BUILD_ROOT%{l_prefix};" \
    imapd.conf.hack
  tools/mkimap imapd.conf.hack

  # determine files
  %{l_rpmtree} files -v -ofiles -r$RPM_BUILD_ROOT \
    %{l_files_std} \

```

```

        '%config %{l_prefix}/etc/imapd/imapd.conf' \
        '%config %{l_prefix}/etc/imapd/cyrus.conf'

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

```

In order to avoid problems with the detection of the Berkeley database the following patch (configure-db4.patch) is needed:

```

--- configure.orig 2003-01-26 15:51:04.000000000 +0100
+++ configure 2003-02-01 02:41:18.000000000 +0100
@@ -1977,7 +1977,7 @@

    #Try to detect the name of the DB4/DB3 library
    dbfound="no"
-for dbname in db-4.1 db4.1 db-4.0 db4.0 db-4 db4 db-3.3 \
        db3.3 db-3.2 db3.2 db-3.1 db3.1 db-3 db3 db
+for dbname in db db-4.1 db4.1 db-4.0 db4.0 db-4 db4 db-3.3 \
        db3.3 db-3.2 db3.2 db-3.1 db3.1 db-3 db3
    do
        echo $ac_n "checking for db_create in -l$dbname" "... $ac_c" 1>&6
        echo "configure:1984: checking for db_create in -l$dbname" >&5

```

As cyrus is not completely consistent with respect to avoid the system's /etc/group file we provide the following patch (groupfile.patch). Note that cyrus is not fully consistent in that respect. Users don't need to exist locally on the mailserver and can be taken from LDAP. For groups that doesn't apply. We felt that introducing a separate group file which is not read by the system would help here. The modified cyrus imapd now reads /etc/imapd.group instead of /etc/group. The outcome is that groups of imap users don't exist on the mailserver.

```

--- lib/auth_unix.c Thu Oct 10 16:38:26 2002
+++ lib/auth_unix.c Thu Oct 10 17:48:04 2002
@@ -48,6 +48,7 @@
    #include <stdlib.h>
    #include <pwd.h>
    #include <grp.h>
+   #include <stdio.h>
    #include <ctype.h>
    #include <string.h>

@@ -143,6 +144,26 @@
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

```

```

};

+
+static struct group* fgetgrnam(const char* name)
+{
+    struct group *grp;
+    FILE *groupfile;
+
+    groupfile = fopen("/etc/imapd.group","r");
+    if (!groupfile) groupfile = fopen("/etc/group", "r");
+    if (groupfile) {
+        while ((grp = fgetgrent(groupfile))) {
+            if (strcmp(grp->gr_name, name) == 0) {
+                fclose(groupfile);
+                return grp;
+            }
+        }
+    }
+    if (groupfile) fclose(groupfile);
+    return NULL;
+}
+
+/*
+ * Convert 'identifier' into canonical form.
+ * Returns a pointer to a static buffer containing the canonical form
+@ -185,7 +206,7 @@
+ */
+
+    if (!strncmp(retbuf, "group:", 6)) {
- grp = getgrnam(retbuf+6);
+ grp = fgetgrnam(retbuf+6);
+    if (!grp) return 0;
+    strcpy(retbuf+6, grp->gr_name);
+    return retbuf;
+@ -228,6 +249,7 @@
+    struct passwd *pwd;
+    struct group *grp;
+    char **mem;
+    FILE *groupfile;
+
+    identifier = auth_canonifyid(identifier, 0);
+    if (!identifier) return 0;
+@ -241,20 +263,23 @@
+    newstate->ngroups = 0;
+    newstate->group = (char **) 0;
+
-    setgrent();
-    while ((grp = getgrent())) {
- for (mem = grp->gr_mem; *mem; mem++) {
-     if (!strcmp(*mem, identifier)) break;
- }
-
- if (*mem || (pwd && pwd->pw_gid == grp->gr_gid)) {

```

```

-   newstate->ngroups++;
-   newstate->group = (char **)xrealloc((char *)newstate->group,
- newstate->ngroups * sizeof(char *));
-       newstate->group[newstate->ngroups-1] = xstrdup(grp->gr_name);
-   }
-   }
-   endgrent();
+   groupfile = fopen("/etc/imapd.group", "r");
+   if (!groupfile) groupfile = fopen("/etc/group", "r");
+   if (groupfile) {
+       while ((grp = fgetgrent(groupfile)) {
+           for (mem = grp->gr_mem; *mem; mem++) {
+               if (!strcmp(*mem, identifier)) break;
+           }
+
+           if (*mem || (pwd && pwd->pw_gid == grp->gr_gid)) {
+               newstate->ngroups++;
+               newstate->group = (char **)xrealloc((char *)newstate->group,
+ newstate->ngroups * sizeof(char *));
+               newstate->group[newstate->ngroups-1] = xstrdup(grp->gr_name);
+           }
+       }
+       fclose(groupfile);
+   }
+   return newstate;
}

```

The community process resulted in an additional path to be included as there can be compile errors on some systems due to not found libraries. Therefore the patch `makefile.patch` is needed:

```

--- timsieved/Makefile.in Fri Jan 24 10:41:36 2003
+++ timsieved/Makefile.in Fri Jan 24 10:42:01 2003
@@ -52,7 +52,7 @@
CYRUS_GROUP=@cyrus_group@

DEFS = @DEFS@ @LOCALDEFS@
-CPPFLAGS = -I. -I.. -I../sieve/ -I$(srcdir) -I$(srcdir)/../sieve -I$(srcdir)/../imap \
-I$(srcdir)/../acap -I$(srcdir)/../lib @COM_ERR_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
+CPPFLAGS = -I. -I../et -I.. -I../sieve/ -I$(srcdir) -I$(srcdir)/../sieve -I$(srcdir)/../im
-I$(srcdir)/../acap -I$(srcdir)/../lib @COM_ERR_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
CFLAGS = @CFLAGS@
LDFLAGS = @LDFLAGS@

--- notifyd/Makefile.in Fri Jan 24 10:42:13 2003
+++ notifyd/Makefile.in Fri Jan 24 10:42:26 2003
@@ -52,7 +52,7 @@
CYRUS_GROUP=@cyrus_group@

DEFS = @DEFS@ @LOCALDEFS@

```

```

-CPPFLAGS = -I. -I.. -I../sieve/ -I$(srcdir) -I$(srcdir)/../sieve -I$(srcdir)/../imap \
  -I$(srcdir)/../acap -I$(srcdir)/../lib @COM_ERR_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
+CPPFLAGS = -I. -I../et -I.. -I../sieve/ -I$(srcdir) -I$(srcdir)/../sieve -I$(srcdir)/../im
  -I$(srcdir)/../acap -I$(srcdir)/../lib @COM_ERR_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
CFLAGS = @CFLAGS@
LDFLAGS = @LDFLAGS@

--- imap/Makefile.in Fri Jan 24 10:42:53 2003
+++ imap/Makefile.in Fri Jan 24 10:43:15 2003
@@ -62,7 +62,7 @@
  CYRUS_GROUP=@cyrus_group@

  DEFS = @DEFS@ @LOCALDEFS@
-CPPFLAGS = -I. -I.. -I../sieve -I$(srcdir) -I$(srcdir)/../lib -I$(srcdir)/../acap \
  -I../acap @COM_ERR_CPPFLAGS@ @SIEVE_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
+CPPFLAGS = -I. -I../et -I.. -I../sieve -I$(srcdir) -I$(srcdir)/../lib -I$(srcdir)/../acap
  -I../acap @COM_ERR_CPPFLAGS@ @SIEVE_CPPFLAGS@ @CPPFLAGS@ @SASLFLAGS@
IMAP_LIBS = @IMAP_LIBS@
SIEVE_LIBS = @SIEVE_LIBS@
IMAP_COM_ERR_LIBS = @IMAP_COM_ERR_LIBS@

```

Due to changes in OpenPKG 1.2 perl handling we need the perl.patch:

```

--- perl/Makefile.in.orig 2003-02-02 23:56:22.000000000 +0100
+++ perl/Makefile.in 2003-02-02 23:56:42.000000000 +0100
@@ -52,9 +52,9 @@
  INSTALL = @INSTALL@

  # Note that we use the *dynamic* sasl libraries
-SASL_LIB=@LIB_DYN_SASL@
+SASL_LIB=@LIB_DYN_SASL@ -ldl -ldb
  SASL_INC=@DYNSASLFLAGS@
-OPENSSL_LIB=@OPENSSL_LIB@
+OPENSSL_LIB=@OPENSSL_LIB@ -ldl -ldb
  OPENSSL_INC=@OPENSSL_INC@

  PERL=@PERL@

--- perl/sieve/Makefile.in.orig 2003-02-03 03:54:47.000000000 +0100
+++ perl/sieve/Makefile.in 2003-02-03 03:55:27.000000000 +0100
@@ -63,9 +63,9 @@
  cyrus_prefix = @cyrus_prefix@

  # Note that we use the *dynamic* sasl libraries
-SASL_LIB=@LIB_DYN_SASL@
+SASL_LIB=@LIB_DYN_SASL@ -ldl -ldb
  SASL_INC=@DYNSASLFLAGS@
-OPENSSL_LIB=@OPENSSL_LIB@
+OPENSSL_LIB=@OPENSSL_LIB@ -ldl -ldb
  OPENSSL_INC=@OPENSSL_INC@

```

CC=@CC@

Chapter 10. ProFTP Daemon

The server ProFTPD is built with an altered spec file in order to use additional functionality we need for the Kolab server.

```
# package options
%ifndef with_pam
%define with_pam no
%endif
%ifndef with_fsl
%define with_fsl no
%endif

# package information
Name:          proftpd
Summary:       Professional FTP Daemon
URL:           http://www.proftpd.net/
Vendor:        The ProFTPD Project
Packager:      The OpenPKG Project
Distribution:  OpenPKG [BASE]
Group:         FTP
License:       GPL
Version:       1.2.8rc1
Release:       20030125

# list of sources
%define url ftp://ftp.proftpd.net/distrib/source/
        proftpd-%{version}.tar.bz2
Source0:       %( [ ! -f %{SOURCE proftpd-%{version}.tar.bz2} ] \
        && wget -c %url; echo %url )
Source1:       proftpd.conf
Source2:       proftpd.msg.goaway
Source3:       proftpd.msg.login
Source4:       rc.proftpd
Source5:       Makefile
Patch0:        mod_ldap.patch

# build information
Prefix:        %{l_prefix}
BuildRoot:     %{l_buildroot}
BuildPreReq:   OpenPKG, openpkg >= 1.1.0, make
PreReq:        OpenPKG, openpkg >= 1.1.0
%if "%{with_pam}" == "yes"
BuildPreReq:   PAM
PreReq:        PAM
%endif
%if "%{with_fsl}" == "yes"
BuildPreReq:   fsl
PreReq:        fsl
```

```

%endif

AutoReq:      no
AutoReqProv:  no

%description
  ProFTPD grew out of the desire to have a secure and configurable FTP
  server, and out of a significant admiration of the Apache web server.
  There are currently a very limited number of FTP servers running on unix
  (or unix-like) hosts. The most commonly used server is probably wu-ftp.d.
  While wu-ftp.d provides excellent performance and is generally a good
  product, it lacks numerous features found in newer Win32 FTP servers, and
  has a poor security history. Many people, including the developers who
  work on ProFTPD have spent a great deal of time fixing bugs and hacking
  features into wu-ftp.d. Unfortunately, it quickly became clear that a
  complete redesign was necessary in order to implement the configurability
  and features desired. ProFTPD is not a hack based on any other server,
  it's an independent source tree from the ground up. Click here for a small
  list of some of the sites ProFTPD powers -- many of them handling large
  volumes of traffic on a daily basis.

  Options:
  --define 'with_pam %{with_pam}'

%prep
  %setup -q
  %patch -p0

%build
  # make non-standard "rundir" the same as standard "sysconfdir"
  %{l_shtool} subst \
    -e 's:^(rundir=@localstatedir@)/proftpd;\1;' \
    Make.rules.in

  # configure the ProFTPD source tree
  CC="%{l_cc}" \
  CFLAGS="%{l_cflags -O}" \
  %if "%{with_fsl}" == "yes"
    LDFLAGS="-L %{l_prefix}/lib '%{l_prefix}/bin/fsl-config --ldflags'" \
  %else
    LDFLAGS="-L %{l_prefix}/lib" \
  %endif
  %if "%{with_pam}" == "yes"
    CPPFLAGS="-I'%{l_prefix}/etc/rc --query pam_incdirc'" \
    LDFLAGS="$LDFLAGS -L'%{l_prefix}/etc/rc --query pam_libdir'" \
  %endif
  LDFLAGS="$LDFLAGS -lcrypto"
  ./configure \
    --prefix=%{l_prefix} \
    --sysconfdir=%{l_prefix}/etc/proftpd \
    --localstatedir=%{l_prefix}/var/proftpd \
  --with-libraries=%{l_prefix}/lib \
  --with-includes=%{l_prefix}/include \

```

```

%if "%{with_pam}" == "yes"
    --enable-pam \
%else
--disable-pam \
%endif
    --with-modules=mod_ratio:mod_readme:mod_ldap

%ifndef syslog_facility
    %{l_shtool} subst -e \
        's/syslog([a-zA-Z]*,/syslog(LOG_%{syslog_facility}),/g;' src/log.c
%endif
    # build ProFTPD programs
    %{l_make} %{l_mflags -O}

%install
    rm -rf $RPM_BUILD_ROOT

    # make sure the "install" procedure does not try
    # to perform explicit ownership assignments
    %{l_shtool} subst -v \
        -e 's;-o $(INSTALL_USER) -g $(INSTALL_GROUP);;g' \
        -e 's;chown;true;g' \
        Makefile

    # perform the "install" procedure while redirecting
    # it to the temporarily install area
    %{l_make} %{l_mflags} \
        install-proftpd install-utils install-man \
        prefix=$RPM_BUILD_ROOT%{l_prefix} \
        sysconfdir=$RPM_BUILD_ROOT%{l_prefix}/etc/proftpd \
        localstatedir=$RPM_BUILD_ROOT%{l_prefix}/var/proftpd \
        rundir=$RPM_BUILD_ROOT%{l_prefix}/var/proftpd

    # strip installation
    rm -f $RPM_BUILD_ROOT%{l_prefix}/sbin/in.proftpd

    # install more stuff manually
    %{l_shtool} mkdir -f -p -m 755 \
    $RPM_BUILD_ROOT%{l_prefix}/share/proftpd
    %{l_shtool} install -c -m 644 doc/faq.html \
        $RPM_BUILD_ROOT%{l_prefix}/share/proftpd/faq.html
    %{l_shtool} install -c -m 644 doc/Configuration.html \
        $RPM_BUILD_ROOT%{l_prefix}/share/proftpd/cfg.html

    # extended installation with own stuff
    l_pam="#"
%if "%{with_pam}" == "yes"
    l_pam=""
%endif
    %{l_shtool} install -c -m 644 \
        -e 's;@l_prefix@;%{l_prefix};g' \
        -e 's;@l_nusr@;%{l_nusr};g' \
        -e 's;@l_ngrp@;%{l_ngrp};g' \

```

```

        -e "s:@l_pam@;${l_pam};g" \
        %{SOURCE proftpd.conf} $RPM_BUILD_ROOT%{l_prefix}/etc/proftpd/
%{l_shtool} install -c -m 644 \
        %{SOURCE proftpd.msg.goaway} \
        %{SOURCE proftpd.msg.login} \
        $RPM_BUILD_ROOT%{l_prefix}/etc/proftpd/
%{l_shtool} install -c -m 644 \
        %{SOURCE proftpd.msg.login} \
        $RPM_BUILD_ROOT%{l_prefix}/share/proftpd/.msg.login
%{l_shtool} install -c -m 644 \
        %{SOURCE proftpd.msg.goaway} \
        $RPM_BUILD_ROOT%{l_prefix}/share/proftpd/.msg.goaway
%{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d
%{l_shtool} install -c -m 755 -e 's:@l_prefix@;${l_prefix};g' \
        %{SOURCE rc.proftpd} $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/

# determine the package ingredients
%{l_rpmtool} files -v -ofiles -r$RPM_BUILD_ROOT \
        %{l_files_std} \
        '%config %{l_prefix}/etc/proftpd/proftpd.conf'

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

%post
%if "%{with_pam}" == "yes"
    # add PAM configuration entry
    if [ $1 -eq 1 ]; then
        $RPM_INSTALL_PREFIX/sbin/pamtool --add --smart --name=proftpd
    fi
%endif

%preun
%if "%{with_pam}" == "yes"
    # remove PAM configuration entry
    if [ $1 -eq 0 ]; then
        $RPM_INSTALL_PREFIX/sbin/pamtool --remove --smart --name=proftpd
    fi
%endif
    if [ $1 -eq 0 ]; then
        $RPM_INSTALL_PREFIX/etc/rc proftpd stop
        rm -f $RPM_INSTALL_PREFIX/var/proftpd/*.log
        rm -f $RPM_INSTALL_PREFIX/var/proftpd/*.pid
    fi

```

In order to have LDAP support linked into proftpd the following patch must be applied:

```
*** contrib/mod_ldap.c.orig 2003-01-26 17:45:20.000000000 +0100
--- contrib/mod_ldap.c 2003-01-26 17:45:30.000000000 +0100
*****
*** 64 ****
! * $Libraries: -lldap -llber$
--- 64 ----
! * $Libraries: -lldap -llber -lssl -lcrypto$
```

Chapter 11. Apache Web Server

The Apache source package from the Current OpenPKG branch was modified to include Webdav and LDAP support (SPEC file).

```
# the additionally used Apache modules (can be enabled without thinking)
%{!?with_mod_ssl:           %define with_mod_ssl           no}
%{!?with_mod_perl:         %define with_mod_perl         no}
%{!?with_mod_php:          %define with_mod_php          no}
%{!?with_mod_php3:         %define with_mod_php3         no}
%{!?with_mod_dav:          %define with_mod_dav          no}
%{!?with_mod_layout:       %define with_mod_layout       no}
%{!?with_mod_macro:        %define with_mod_macro        no}

# the additionally used Apache modules (you have to know what you do)
%{!?with_mod_auth_pam:     %define with_mod_auth_pam     no}
%{!?with_mod_gzip:         %define with_mod_gzip         no}
%{!?with_mod_zmod:         %define with_mod_zmod         no}
%{!?with_mod_fastcgi:     %define with_mod_fastcgi     no}
%{!?with_mod_throttle:    %define with_mod_throttle    no}
%{!?with_mod_access_referer: %define with_mod_access_referer no}
%{!?with_mod_roaming:     %define with_mod_roaming     no}
%{!?with_mod_relocate:    %define with_mod_relocate    no}

# more optional PHP4 specific settings
# (requires "with_mod_php" set to "yes" above!)
%{!?with_mod_php_calendar: %define with_mod_php_calendar no}
%{!?with_mod_php_mysql:    %define with_mod_php_mysql    no}
%{!?with_mod_php_gd:       %define with_mod_php_gd       no}
%{!?with_mod_php_db:       %define with_mod_php_db       no}
%{!?with_mod_php_debug:    %define with_mod_php_debug    no}
%{!?with_mod_php_pdflib:   %define with_mod_php_pdflib   no}
%{!?with_mod_php_zlib:     %define with_mod_php_zlib     no}
%{!?with_mod_php_bzip2:    %define with_mod_php_bzip2    no}
%{!?with_mod_php_openssl:  %define with_mod_php_openssl  no}
%{!?with_mod_php_openldap: %define with_mod_php_openldap no}
%{!?with_mod_php_mm:       %define with_mod_php_mm       no}
%{!?with_mod_php_pcre:     %define with_mod_php_pcre     no}
%{!?with_mod_php_ftp:      %define with_mod_php_ftp      no}
%{!?with_mod_php_java:     %define with_mod_php_java     no}
%{!?with_mod_php_oci8:     %define with_mod_php_oci8     no}
%{!?with_mod_php_freetype: %define with_mod_php_freetype no}
%{!?with_mod_php_gettext:  %define with_mod_php_gettext  no}
%{!?with_mod_php_imap:     %define with_mod_php_imap     no}
%{!?with_mod_php_xml:      %define with_mod_php_xml      no}
%{!?with_mod_php_bc:       %define with_mod_php_bc       no}
%{!?with_mod_php_transsid: %define with_mod_php_transsid no}
%{!?with_mod_php_cyrus:    %define with_mod_php_cyrus    no}
```

```

# more optional PHP3 specific settings
# (requires "with_mod_php3" set to "yes" above!)
%{!?with_mod_php3_ftp:      %define with_mod_php3_ftp      no}
%{!?with_mod_php3_gd:      %define with_mod_php3_gd      no}
%{!?with_mod_php3_jpeg:    %define with_mod_php3_jpeg    no}
%{!?with_mod_php3_mysql:   %define with_mod_php3_mysql   no}
%{!?with_mod_php3_openssl: %define with_mod_php3_openssl no}
%{!?with_mod_php3_zlib:    %define with_mod_php3_zlib    no}

# fixing implicit inter-module dependencies and correlations
%if "%{with_mod_php}" == "yes"
%if "%{with_mod_php3}" == "yes"
%{error: with_mod_php conflicts with with_mod_php3}
# FIXME: error macro does not terminate execution
exit 1
%endif
%if "%{with_mod_ssl}" == "yes"
%define with_mod_php_openssl yes
%define with_mod_php_mm      yes
%endif
%if "%{with_mod_php_freetype}" == "yes"
%define with_mod_php_gd      yes
%endif
%if "%{with_mod_php_mysql}" == "yes" || "%{with_mod_php_pdflib}" \
    == "yes" || "%{with_mod_php_gd}" == "yes"
%define with_mod_php_zlib    yes
%endif
%endif
%if "%{with_mod_php3}" == "yes"
%if "%{with_mod_ssl}" == "yes"
%define with_mod_php3_openssl yes
%endif
%if "%{with_mod_php3_mysql}" == "yes"
%define with_mod_php3_zlib    yes
%endif
%endif

# package component versions
%define V_apache          1.3.27
%define V_mod_ssl         2.8.12-1.3.27
%define V_mod_perl        1.27
%define V_mod_php         4.3.0
%define V_mod_php3        3.0.18
%define V_mod_dav         1.0.3-1.3.6
%define V_mod_layout      3.2
%define V_mod_macro       1.1.2
%define V_mod_auth_pam    1.0a
%define V_mod_gzip        1.3.19.1a
%define V_mod_zmod        2_3
%define V_mod_fastcgi     2.4.0
%define V_mod_throttle    312
%define V_mod_access_referer 1.0.2
%define V_mod_roaming     1.0.2

```

```

#define      V_mod_relocate      1.0

#   package information
Name:       apache
Summary:    Apache HTTP Server
URL:        http://httpd.apache.org/
Vendor:     Apache Software Foundation
Packager:   The OpenPKG Project
Distribution: OpenPKG [BASE]
Group:      Web
License:    ASF
Version:    %{V_apache}
Release:    20030125

#   list of sources
#define url   http://www.apache.org/dist/httpd/apache_%{V_apache}.tar.gz
Source0:     %( [ ! -f %{SOURCE apache_%{V_apache}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://www.modssl.org/source/mod_ssl-%{V_mod_ssl}.tar.gz
Source1:     %( [ ! -f %{SOURCE mod_ssl-%{V_mod_ssl}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://perl.apache.org/dist/mod_perl-%{V_mod_perl}.tar.gz
Source2:     %( [ ! -f %{SOURCE mod_perl-%{V_mod_perl}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://www.php.net/distributions/php-%{V_mod_php}.tar.gz
Source3:     %( [ ! -f %{SOURCE php-%{V_mod_php}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://www.webdav.org/mod_dav/mod_dav-%{V_mod_dav}.tar.gz
Source4:     %( [ ! -f %{SOURCE mod_dav-%{V_mod_dav}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://software.tangent.org/download/
mod_layout-%{V_mod_layout}.tar.gz
Source5:     %( [ ! -f %{SOURCE mod_layout-%{V_mod_layout}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://www.cri.ensmp.fr/~coelho/mod_macro/
mod_macro-%{V_mod_macro}.tar.gz
Source6:     %( [ ! -f %{SOURCE mod_macro-%{V_mod_macro}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://pam.sourceforge.net/mod_auth_pam/
dist/mod_auth_pam.tar.gz
Source7:     %( [ ! -f %{SOURCE mod_auth_pam.tar.gz} ] \
&& wget -c %url; echo %url )
# define url   http://www.remotecomunications.com/apache/mod_gzip/
src/%{V_mod_gzip}/mod_gzip.c
# Source8:     %( [ ! -f %{SOURCE mod_gzip.c} ] \
&& wget -c %url; echo %url )
Source8:     mod_gzip.c
#define url   http://download.at.kde.org/opsys/linux/OpenPKG/
sources/DST/apache/src.apapi.FIN%{V_mod_zmod}.tar.gz
Source9:     %( [ ! -f %{SOURCE src.apapi.FIN%{V_mod_zmod}.tar.gz} ] \
&& wget -c %url; echo %url )
#define url   http://www.fastcgi.com/dist/
mod_fastcgi-%{V_mod_fastcgi}.tar.gz

```

```

Source10:      %( [ ! -f %{SOURCE mod_fastcgi-%{V_mod_fastcgi}.tar.gz} ] \
&& wget -c %url; echo %url )
%define url    http://www.snert.com/Software/mod_throttle/
mod_throttle%{V_mod_throttle}.tgz
Source11:      %( [ ! -f %{SOURCE mod_throttle%{V_mod_throttle}.tgz} ] \
&& wget -c %url; echo %url )
%define url    http://download.sourceforge.net/accessreferer/
mod_access_referer-%{V_mod_access_referer}.tar.gz
Source12:      %( [ ! -f %{SOURCE mod_access_referer-
%{V_mod_access_referer}.tar.gz} ] && wget -c %url; echo %url )
%define url    http://www.klomp.org/mod_roaming/
mod_roaming-%{V_mod_roaming}.tar.gz
Source13:      %( [ ! -f %{SOURCE mod_roaming-%{V_mod_roaming}.tar.gz} ] \
&& wget -c %url; echo %url )
%define url    http://software.tangent.org/download/
mod_relocate-%{V_mod_relocate}.tar.gz
Source14:      %( [ ! -f %{SOURCE mod_relocate-%{V_mod_relocate}.tar.gz} ] \
&& wget -c %url; echo %url )
%define url    http://www.php.net/distributions/php-%{V_mod_php3}.tar.gz
Source15:      %( [ ! -f %{SOURCE php-%{V_mod_php3}.tar.gz} ] \
&& wget -c %url; echo %url )
%define url    http://www.muquit.com/muquit/software/
mod_auth_ldap/mod_auth_ldap.tar.gz
Source16:      %( [ ! -f %{SOURCE mod_auth_ldap.tar.gz} ] \
&& wget -c %url; echo %url )
Source20:      apache.conf
Source21:      apache.base
Source22:      apache.vhost
Source23:      rc.apache
Source24:      Makefile
Patch0:        mod-ldap.patch

# build information
Prefix:        %{l_prefix}
BuildRoot:     %{l_buildroot}
BuildPreReq:   OpenPKG, openpkg >= 1.1.0, gdbm
PreReq:        OpenPKG, openpkg >= 1.1.0
%if "%{with_mod_ssl}" == "yes"
BuildPreReq:   openssl, mm
%endif
%if "%{with_mod_auth_ldap}" == "yes"
BuildPreReq:   openldap
%endif
%if "%{with_mod_perl}" == "yes"
BuildPreReq:   perl
PreReq:        perl
%endif
%if "%{with_mod_php}" == "yes"
BuildPreReq:   make, bison, flex
%if "%{with_mod_php_mysql}" == "yes"
BuildPreReq:   mysql
%endif
%endif

```

```

%if "%{with_mod_php_gd}" == "yes"
BuildPreReq: GD, jpeg, png
%endif
%if "%{with_mod_php_db}" == "yes"
BuildPreReq: db
%endif
%if "%{with_mod_php_pdflib}" == "yes"
BuildPreReq: pdflib, jpeg, png
%endif
%if "%{with_mod_php_zlib}" == "yes"
BuildPreReq: zlib
%endif
%if "%{with_mod_php_bzip2}" == "yes"
BuildPreReq: bzip2
%endif
%if "%{with_mod_php_openssl}" == "yes"
BuildPreReq: openssl
%endif
%if "%{with_mod_php_openldap}" == "yes"
BuildPreReq: openldap, openssl
%endif
%if "%{with_mod_php_mm}" == "yes"
BuildPreReq: mm
%endif
%if "%{with_mod_php_pcre}" == "yes"
BuildPreReq: pcre
%endif
%if "%{with_mod_php_java}" == "yes"
BuildPreReq: j2se
%endif
%if "%{with_mod_php_freetype}" == "yes"
BuildPreReq: freetype
%endif
%if "%{with_mod_php_gettext}" == "yes"
BuildPreReq: gettext, libiconv
%endif
%if "%{with_mod_php_imap}" == "yes"
BuildPreReq: c-client
%endif
%if "%{with_mod_php_xml}" == "yes"
BuildPreReq: expat
%endif
%endif

%if "%{with_mod_php3}" == "yes"
BuildPreReq: make, bison, flex
%if "%{with_mod_php3_gd}" == "yes"
BuildPreReq: GD
%endif
%if "%{with_mod_php3_jpeg}" == "yes"
BuildPreReq: jpeg
%endif
%if "%{with_mod_php3_mysql}" == "yes"

```

```

BuildPreReq:  mysql
%endif
%if "%{with_mod_php3_openssl}" == "yes"
BuildPreReq:  openssl
%endif
%if "%{with_mod_php3_zlib}" == "yes"
BuildPreReq:  zlib
%endif
%endif

%if "%{with_mod_auth_pam}" == "yes"
BuildPreReq:  PAM
PreReq:       PAM
%endif
AutoReq:      no
AutoReqProv:  no

%description
    The Apache Project is a collaborative software development effort
    aimed at creating a robust, commercial-grade, featureful, and
    freely-available source code implementation of an HTTP (Web) server.
    The project is jointly managed by a group of volunteers located
    around the world, using the Internet and the Web to communicate,
    plan, and develop the server and its related documentation. These
    volunteers are known as the Apache Group. In addition, hundreds
    of users have contributed ideas, code, and documentation to the
    project.

Options (additional modules I):
--define 'with_mod_dav'           %{with_mod_dav}' \
--define 'with_mod_layout'        %{with_mod_layout}' \
--define 'with_mod_macro'         %{with_mod_macro}' \
--define 'with_mod_perl'          %{with_mod_perl}' \
--define 'with_mod_php'           %{with_mod_php}' \
--define 'with_mod_php3'          %{with_mod_php3}' \
--define 'with_mod_ssl'           %{with_mod_ssl}' \

Options (additional modules II):
--define 'with_mod_access_referer' %{with_mod_access_referer}' \
--define 'with_mod_auth_pam'       %{with_mod_auth_pam}' \
--define 'with_mod_fastcgi'        %{with_mod_fastcgi}' \
--define 'with_mod_gzip'           %{with_mod_gzip}' \
--define 'with_mod_relocate'       %{with_mod_relocate}' \
--define 'with_mod_roaming'        %{with_mod_roaming}' \
--define 'with_mod_throttle'       %{with_mod_throttle}' \
--define 'with_mod_zmod'           %{with_mod_zmod}' \

Options (additional extensions for mod_php):
--define 'with_mod_php_bzip2'      %{with_mod_php_bzip2}' \
--define 'with_mod_php_bc'         %{with_mod_php_bc}' \
--define 'with_mod_php_calendar'   %{with_mod_php_calendar}' \
--define 'with_mod_php_db'         %{with_mod_php_db}' \
--define 'with_mod_php_debug'      %{with_mod_php_debug}' \

```

```

--define 'with_mod_php_ftp'           %{with_mod_php_ftp}' \
--define 'with_mod_php_freetype'     %{with_mod_php_freetype}' \
--define 'with_mod_php_gd'           %{with_mod_php_gd}' \
--define 'with_mod_php_gettext'      %{with_mod_php_gettext}' \
--define 'with_mod_php_imap'         %{with_mod_php_imap}' \
--define 'with_mod_php_java'         %{with_mod_php_java}' \
--define 'with_mod_php_mm'           %{with_mod_php_mm}' \
--define 'with_mod_php_mysql'        %{with_mod_php_mysql}' \
--define 'with_mod_php_oci8'         %{with_mod_php_oci8}' \
--define 'with_mod_php_openldap'     %{with_mod_php_openldap}' \
--define 'with_mod_php_openssl'      %{with_mod_php_openssl}' \
--define 'with_mod_php_pcre'         %{with_mod_php_pcre}' \
--define 'with_mod_php_pdflib'       %{with_mod_php_pdflib}' \
--define 'with_mod_php_transsid'     %{with_mod_php_transsid}' \
--define 'with_mod_php_xml'          %{with_mod_php_xml}' \
--define 'with_mod_php_zlib'         %{with_mod_php_zlib}' \

Options (additional extensions for mod_php3):
--define 'with_mod_php3_ftp'         %{with_mod_php3_ftp}' \
--define 'with_mod_php3_gd'          %{with_mod_php3_gd}' \
--define 'with_mod_php3_jpeg'        %{with_mod_php3_jpeg}' \
--define 'with_mod_php3_mysql'       %{with_mod_php3_mysql}' \
--define 'with_mod_php3_openssl'     %{with_mod_php3_openssl}' \
--define 'with_mod_php3_zlib'        %{with_mod_php3_zlib}' \

%prep
#   unpack Apache distribution
%setup0 -q -c
#   unpack optional extension modules
%if "%{with_mod_ssl}" == "yes"
    %setup1 -q -T -D -a 1
%endif
%if "%{with_mod_perl}" == "yes"
    %setup2 -q -T -D -a 2
%endif
%if "%{with_mod_php}" == "yes"
    %setup3 -q -T -D -a 3
%endif
%if "%{with_mod_dav}" == "yes"
    %setup4 -q -T -D -a 4
%endif
%if "%{with_mod_layout}" == "yes"
    %setup5 -q -T -D -a 5
%endif
%if "%{with_mod_macro}" == "yes"
    %setup6 -q -T -D -a 6
%endif
%if "%{with_mod_auth_pam}" == "yes"
    %setup7 -q -T -D -a 7
%endif
%if "%{with_mod_zmod}" == "yes"
    %setup9 -q -T -D -a 9
%endif

```

```

%if "%{with_mod_fastcgi}" == "yes"
    %setup10 -q -T -D -a 10
%endif
%if "%{with_mod_throttle}" == "yes"
    %setup11 -q -T -D -a 11
%endif
%if "%{with_mod_access_referer}" == "yes"
    %setup12 -q -T -D -a 12
%endif
%if "%{with_mod_roaming}" == "yes"
    %setup13 -q -T -D -a 13
%endif
%if "%{with_mod_relocate}" == "yes"
    %setup14 -q -T -D -a 14
%endif
%if "%{with_mod_php3}" == "yes"
    %setup15 -q -T -D -a 15
%endif
%if "%{with_mod_auth_ldap}" == "yes"
    %setup16 -q -T -D -a 16
    ( cd modauthldap
      %patch0 -p0
    )
%endif

%build
#   prepare environment
rm -rf $RPM_BUILD_ROOT
%{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}

#   optionally prepare mod_ssl
%if "%{with_mod_ssl}" == "yes"
    ( cd mod_ssl-%{V_mod_ssl}
      ./configure \
        --with-apache=../apache_%{V_apache} \
        --with-ssl=%{l_prefix} \
        --with-mm=%{l_prefix} \
        --expert --force
    )
%endif

#   optionally pre-configure Apache for mod_php, mod_php3 and mod_dav
%if "%{with_mod_php}" == "yes" || "%{with_mod_php3}" == "yes" \
|| "%{with_mod_dav}" == "yes"
    ( cd apache_%{V_apache}
      CC="%{l_cc}" \
      CFLAGS="%{l_cflags -O}" \
      ./configure \
%if "%{with_mod_ssl}" == "yes"
        --enable-rule=EAPI \
%endif
      --target=apache \

```

```

--with-layout=GNU \
--prefix=${l_prefix} \
--sbindir=${l_prefix}/sbin \
--sysconfdir=${l_prefix}/etc/apache \
--libexecdir=${l_prefix}/lib/apache \
--datadir=${l_prefix}/share/apache \
--localstatedir=${l_prefix}/var/apache
)
%endif

# optionally prepare mod_perl
%if "%{with_mod_perl}" == "yes"
( cd mod_perl-%{V_mod_perl}
eval `${l_prefix}/bin/perl -V:archname`
eval `${l_prefix}/bin/perl -V:version`
%{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT${l_prefix}/bin
perl=$RPM_BUILD_ROOT${l_prefix}/bin/perl
echo "#!/bin/sh" >$perl
echo "exec ${l_prefix}/bin/perl \\" >>$perl
echo " -I$RPM_BUILD_ROOT${l_prefix}/lib/perl5/${version} \\" \
>>$perl
echo " -I$RPM_BUILD_ROOT${l_prefix}/lib/perl5/
${version}/${archname} \\" >>$perl
echo " -I$RPM_BUILD_ROOT${l_prefix}/lib/perl5/
site_perl \\" >>$perl
echo " -I$RPM_BUILD_ROOT${l_prefix}/lib/perl5/
site_perl/${version} \\" >>$perl
echo " -I$RPM_BUILD_ROOT${l_prefix}/lib/perl5/
site_perl/${version}/${archname} \\" >>$perl
echo " \\"$@" >>$perl
chmod a+x $perl
$perl Makefile.PL \
PREFIX=$RPM_BUILD_ROOT${l_prefix} \
APACHE_SRC=../apache_%{V_apache}/src \
DO_HTTPD=1 \
USE_APACI=1 \
PREP_HTTPD=1 \
EVERYTHING=1 \
PERL_TIE_TABLES=1 \
PERL_DIRECTIVE_HANDLERS=1
%{l_make} %{l_mflags}
%{l_make} %{l_mflags} install
mkdir $RPM_BUILD_ROOT${l_prefix}/perl5
mv $RPM_BUILD_ROOT${l_prefix}/lib/* $RPM_BUILD_ROOT${l_prefix}/perl5/
mv $RPM_BUILD_ROOT${l_prefix}/perl5 $RPM_BUILD_ROOT${l_prefix}/lib/
%{l_shtool} subst -e "s;^\((PERL = \\\).*;\1 $perl;" \
../apache_%{V_apache}/src/modules/perl/mod_perl.config
)
%endif

# optionally prepare mod_php
%if "%{with_mod_php}" == "yes"
( cd php-%{V_mod_php}

```

```

CC="%{l_cc}"; export CC
CFLAGS="%{l_cflags -O} -I%{l_prefix}/include"; export CFLAGS
CPPFLAGS="%{l_cflags -O} -I%{l_prefix}/include"; export CPPFLAGS
LDFLAGS="%{l_cflags -O} -L%{l_prefix}/lib"; export LDFLAGS
LIBS=""; export LIBS
%if "%{with_mod_ssl}" == "yes"
    CFLAGS="$CFLAGS -DEAPI"
%endif
%if "%{with_mod_php_gd}" == "yes"
    LIBS="$LIBS -lpng -lz"
%endif
%if "%{with_mod_php_gettext}" == "yes"
    LIBS="$LIBS -liconv" \
%endif
%if "%{with_mod_php_openssl}" == "yes"
    LIBS="$LIBS -lsasl2"
%endif
    ./configure \
        --prefix=%{l_prefix} \
        --with-apache=../apache_%{V_apache} \
        --with-config-file-path=%{l_prefix}/etc/apache \
%if "%{with_mod_php_calendar}" == "yes"
        --enable-calendar \
%endif
%if "%{with_mod_php_mysql}" == "yes"
        --with-mysql=%{l_prefix} \
%endif
%if "%{with_mod_php_gd}" == "yes"
        --with-gd=%{l_prefix} \
        --with-jpeg-dir=%{l_prefix} \
        --with-png-dir=%{l_prefix} \
%endif
%if "%{with_mod_php_freetype}" == "yes"
        --enable-gd-imgstrttf \
        --enable-gd-native-ttf \
        --with-freetype-dir=%{l_prefix} \
%endif
%if "%{with_mod_php_db}" == "yes"
        --with-db3=%{l_prefix} \
%endif
%if "%{with_mod_php_debug}" == "yes"
        --with-debug=yes \
%else
        --with-debug=no \
%endif
%if "%{with_mod_php_zlib}" == "yes"
        --with-zlib=%{l_prefix} \
%endif
%if "%{with_mod_php_bzip2}" == "yes"
        --with-bz2=%{l_prefix} \
%endif
%if "%{with_mod_php_pdf}" == "yes"
        --with-pdf=%{l_prefix} \

```

```

        --with-jpeg-dir=${l_prefix} \
        --with-png-dir=${l_prefix} \
    %endif
    %if "%{with_mod_php_openssl}" == "yes" \
        || "%{with_mod_php_openldap}" == "yes"
        --with-openssl=${l_prefix} \
    %endif
    %if "%{with_mod_php_openldap}" == "yes"
        --with-ldap=${l_prefix} \
    %endif
    %if "%{with_mod_php_cyrus}" == "yes"
        --with-cyrus=${l_prefix} \
    %else
        --without-cyrus \
    %endif
    %if "%{with_mod_php_mm}" == "yes"
        --with-mm=${l_prefix} \
        --enable-session \
    %endif
    %if "%{with_mod_php_pcre}" == "yes"
        --with-pcre=${l_prefix} \
    %endif
    %if "%{with_mod_php_ftp}" == "yes"
        --enable-ftp \
    %endif
    %if "%{with_mod_php_java}" == "yes"
        --with-java=${l_prefix}/libexec/j2se \
    %endif
    %if "%{with_mod_php_oci8}" == "yes"
        --with-oci8 \
    %endif
    %if "%{with_mod_php_gettext}" == "yes"
        --with-gettext=${l_prefix} \
    %endif
    %if "%{with_mod_php_imap}" == "yes"
        --with-imap=${l_prefix} \
    %endif
    %if "%{with_mod_php_xml}" == "yes"
        --with-xml=${l_prefix} \
    %endif
    %if "%{with_mod_php_bc}" == "yes"
        --enable-bcmath \
    %endif
    %if "%{with_mod_php_transsid}" == "yes"
        --enable-trans-sid \
    %endif
        --without-pear \
        --disable-shared \
        --enable-inline-optimization \
        --enable-track-vars
    %{l_make} %{l_mflags}
    %{l_shtool} subst \
        -e "s;^\((EXTENSION_DIR = \\)\)\(%{l_prefix}\)\);";

```

```

\\$RPM_BUILD_ROOT\2;" \
    -e "s;^\((PEAR_INSTALLDIR = \\)\)\(%{l_prefix}\);
\\$RPM_BUILD_ROOT\2;" \
    config_vars.mk
    %{l_make} %{l_mflags} install \
        prefix=$RPM_BUILD_ROOT%{l_prefix}
    )
%endif

# optionally prepare mod_php3
%if "%{with_mod_php3}" == "yes"
    ( cd php-%{V_mod_php3}
      CC="%{l_cc}" \
%if "%{with_mod_ssl}" == "yes"
        CFLAGS="%{l_cflags -O} -I%{l_prefix}/include -DEAPI" \
%else
        CFLAGS="%{l_cflags -O} -I%{l_prefix}/include" \
%endif
      CPPFLAGS="%{l_cflags -O} -I%{l_prefix}/include" \
      LDFLAGS="%{l_cflags -O} -L%{l_prefix}/lib" \
      ./configure \
        --prefix=%{l_prefix} \
        --with-apache=../apache_%{V_apache} \
        --with-config-file-path=%{l_prefix}/etc/apache \
%if "%{with_mod_php3_ftp}" == "yes"
        --with-ftp \
%endif
%if "%{with_mod_php3_mysql}" == "yes"
        --with-mysql=%{l_prefix} \
%endif
%if "%{with_mod_php3_zlib}" == "yes"
        --with-zlib=%{l_prefix} \
%endif
%if "%{with_mod_php3_jpeg}" == "yes"
        --with-jpeg=${prefix} \
%endif
%if "%{with_mod_php3_gd}" == "yes"
        --with-gd=${prefix} \
%endif
%if "%{with_mod_php3_openssl}" == "yes"
        --with-openssl=%{l_prefix} \
%endif
        --disable-shared \
        --enable-track-vars

# FIXME:
# --enable-safe-mode
# --with-exec-dir[=DIR]
# --enable-magic-quotes
# --enable-memory-limit
# --enable-sysvsem
# --enable-sysvshm
    %{l_make} %{l_mflags}
    %{l_make} %{l_mflags} install \

```

```

        prefix=$RPM_BUILD_ROOT%{l_prefix}
    )
%endif

    # optionally prepare mod_dav
%if "%{with_mod_dav}" == "yes"
    ( cd mod_dav-%{V_mod_dav}
      CC="%{l_cc}" \
%if "%{with_mod_ssl}" == "yes"
      CFLAGS="%{l_cflags -O}" \
%else
      CFLAGS="%{l_cflags -O} -DEAPI" \
%endif
%endif
    LDFLAGS="%{l_cflags -O}" \
    ./configure \
        --with-apache=../apache_%{V_apache}
    %{l_make} %{l_mflags}
    %{l_make} %{l_mflags} install
    )
%endif

    # optionally prepare mod_layout
%if "%{with_mod_layout}" == "yes"
    ( cd mod_layout-%{V_mod_layout}
      mkdir ../apache_%{V_apache}/src/modules/layout
      cp * ../apache_%{V_apache}/src/modules/layout/ \
2>/dev/null || true
      chmod -R u+w ../apache_%{V_apache}/src/modules/layout
    )
%endif

    # optionally prepare mod_macro
%if "%{with_mod_macro}" == "yes"
    ( cd mod_macro-%{V_mod_macro}
      cp mod_macro.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

    # optionally prepare mod_auth_pam
%if "%{with_mod_auth_pam}" == "yes"
    ( cd mod_auth_pam-%{V_mod_auth_pam}
      cp mod_auth_pam.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

    # optionally prepare mod_gzip
%if "%{with_mod_gzip}" == "yes"
    cp %{SOURCE mod_gzip.c} apache_%{V_apache}/src/modules/extra/
%endif

    # optionally prepare mod_zmod
%if "%{with_mod_zmod}" == "yes"
    ( cd src

```

```

        mkdir ../apache_%{V_apache}/src/modules/zmod
        %{l_shtool} subst -e 's;"compat.h";"ap_compat.h";' \
modules/zmod/mod_zmod.c
        cp modules/zmod/* ../apache_%{V_apache}/src/modules/zmod/
    )
%endif

# optionally prepare mod_fastcgi
%if "%{with_mod_fastcgi}" == "yes"
    ( cd mod_fastcgi-%{V_mod_fastcgi}
      mkdir ../apache_%{V_apache}/src/modules/fastcgi
      cp -rp * ../apache_%{V_apache}/src/modules/fastcgi/
    )
%endif

# optionally prepare mod_throttle
%if "%{with_mod_throttle}" == "yes"
    ( cd mod_throttle-*
      cp mod_throttle.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

# optionally prepare mod_access_referer
%if "%{with_mod_access_referer}" == "yes"
    ( cd mod_access_referer-%{V_mod_access_referer}
      cp mod_access_referer.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

# optionally prepare mod_roaming
%if "%{with_mod_roaming}" == "yes"
    ( cd mod_roaming-%{V_mod_roaming}
      cp mod_roaming.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

# optionally prepare mod_relocate
%if "%{with_mod_relocate}" == "yes"
    ( cd mod_relocate-%{V_mod_relocate}
      cp mod_relocate.c ../apache_%{V_apache}/src/modules/extra/
    )
%endif

# mod_auth_ldap
%if "%{with_mod_auth_ldap}" == "yes"
    %{l_shtool} subst -e "s;\@PREFIX\@;%{l_prefix};g" modauthldap/*
    cp -r modauthldap apache_%{V_apache}/src/modules/ldap
%endif

# configure Apache
( cd apache_%{V_apache}
  cflags="%{l_cflags -O}"
  ldflags="-L%{l_prefix}/lib"

```

```

    libs=""
%if "%{with_mod_auth_pam}" == "yes"
    pam_incdir=`%{l_prefix}/etc/rc --query pam_incdir`
    if [ ".$pam_incdir" != "./usr/include" -a ".$pam_incdir" \
    != "./include" ]; then
        cflags="$cflags -I$pam_incdir"
    fi
    pam_libdir=`%{l_prefix}/etc/rc --query pam_libdir`
    if [ ".$pam_libdir" != "./usr/lib" -a ".$pam_libdir" \
    != "./lib" ]; then
        ldflags="$ldflags -L$pam_libdir"
    fi
    libs="$libs -lpam"
%endif
    CC="%{l_cc}" \
    CFLAGS="$cflags" \
    LDFLAGS="$ldflags" \
    LIBS="$libs" \
%if "%{with_mod_ssl}" == "yes"
    EAPI_MM="%{l_prefix}" \
    SSL_BASE="%{l_prefix}" \
%endif
%if "%{with_mod_auth_ldap}" == "yes"
    LIBS="$LIBS -lldap -llber" \
%endif
    ./configure \
        --target=apache \
        --with-layout=GNU \
        --prefix=%{l_prefix} \
        --sbindir=%{l_prefix}/sbin \
        --sysconfdir=%{l_prefix}/etc/apache \
        --libexecdir=%{l_prefix}/lib/apache \
        --datadir=%{l_prefix}/share/apache \
        --localstatedir=%{l_prefix}/var/apache \
%if "%{with_suexec}" == "yes"
        --enable-suexec \
        --suexec-caller=%{l_nusr} \
        --suexec-userdir=.www \
%endif
        --enable-module=most \
        --with-perl=%{l_prefix}/bin/perl \
%if "%{with_mod_ssl}" == "yes"
        --enable-rule=EAPI \
        --enable-module=ssl \
%endif
%if "%{with_mod_perl}" == "yes"
        --activate-module=src/modules/perl/libperl.a \
%endif
%if "%{with_mod_php}" == "yes"
        --activate-module=src/modules/php4/libphp4.a \
%endif
%if "%{with_mod_php3}" == "yes"
        --activate-module=src/modules/php3/libphp3.a \

```

```

%endif
%if "%{with_mod_dav}" == "yes"
    --activate-module=src/modules/dav/libdav.a \
%endif
%if "%{with_mod_layout}" == "yes"
    --activate-module=src/modules/layout/liblayout.a \
%endif
%if "%{with_mod_macro}" == "yes"
    --activate-module=src/modules/extra/mod_macro.o \
%endif
%if "%{with_mod_auth_pam}" == "yes"
    --activate-module=src/modules/extra/mod_auth_pam.o \
%endif
%if "%{with_mod_gzip}" == "yes"
    --activate-module=src/modules/extra/mod_gzip.o \
%endif
%if "%{with_mod_zmod}" == "yes"
    --activate-module=src/modules/zmod/libzmod.a \
%endif
%if "%{with_mod_fastcgi}" == "yes"
    --activate-module=src/modules/fastcgi/libfastcgi.a \
%endif
%if "%{with_mod_throttle}" == "yes"
    --activate-module=src/modules/extra/mod_throttle.o \
%endif
%if "%{with_mod_access_referer}" == "yes"
    --activate-module=src/modules/extra/mod_access_referer.o \
%endif
%if "%{with_mod_roaming}" == "yes"
    --activate-module=src/modules/extra/mod_roaming.o \
%endif
%if "%{with_mod_relocate}" == "yes"
    --activate-module=src/modules/extra/mod_relocate.o \
%endif
%if "%{with_mod_auth_ldap}" == "yes"
    --activate-module=src/modules/ldap/mod_auth_ldap.o \
%endif
    --enable-module=so \
    --disable-module=auth_dbm
    %{l_make} %{l_mflags -O} build-quiet
)

%install
#   install Apache
( cd apache_%{V_apache}
#   perform standard Apache installation procedure
%{l_make} %{l_mflags} install root=$RPM_BUILD_ROOT
#   post-adjustments to installation tree
mv $RPM_BUILD_ROOT%{l_prefix}/share/apache/icons/small/* \
    $RPM_BUILD_ROOT%{l_prefix}/share/apache/icons/
rm -rf $RPM_BUILD_ROOT%{l_prefix}/share/apache/icons/small
rm -f $RPM_BUILD_ROOT%{l_prefix}/share/apache/icons/README*
rm -f $RPM_BUILD_ROOT%{l_prefix}/etc/apache/*.default

```

```

rm -f $RPM_BUILD_ROOT%{l_prefix}/etc/apache/srm.conf
rm -f $RPM_BUILD_ROOT%{l_prefix}/etc/apache/access.conf
mv $RPM_BUILD_ROOT%{l_prefix}/share/apache/htdocs/index.html.en \
    $RPM_BUILD_ROOT%{l_prefix}/share/apache/htdocs/index.html
rm -f $RPM_BUILD_ROOT%{l_prefix}/share/apache/htdocs/index.html.*
chmod a+rx $RPM_BUILD_ROOT%{l_prefix}/share/apache/cgi-bin/*
mv $RPM_BUILD_ROOT%{l_prefix}/share/apache/cgi-bin \
    $RPM_BUILD_ROOT%{l_prefix}/cgi/
rm -rf $RPM_BUILD_ROOT%{l_prefix}/cgi/test-cgi
)

# optionally cleanup for mod_perl
%if "%{with_mod_perl}" == "yes"
    rm -f $RPM_BUILD_ROOT%{l_prefix}/bin/perl
%endif

# create default configuration
l_hostname=%{l_shtool} echo -e %h`
l_domainname=%{l_shtool} echo -e %d | cut -c2-`
%{l_shtool} install -c -m 644 \
    -e 's:@l_prefix@;%{l_prefix};g' \
    -e "s:@l_hostname@;$l_hostname;g" \
    -e "s:@l_domainname@;$l_domainname;g" \
    -e 's:@l_nusr@;%{l_nusr};g' \
    -e 's:@l_ngrp@;%{l_ngrp};g' \
    %{SOURCE apache.base} \
    %{SOURCE apache.conf} \
    %{SOURCE apache.vhost} \
    $RPM_BUILD_ROOT%{l_prefix}/etc/apache/
mv $RPM_BUILD_ROOT%{l_prefix}/etc/apache/magic \
    $RPM_BUILD_ROOT%{l_prefix}/etc/apache/mime.magic
    find $RPM_BUILD_ROOT%{l_prefix} -name perllocal.pod \
    -print | xargs rm -f

# create run-command script
%{l_shtool} mkdir -f -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d
%{l_shtool} install -c -m 755 -e 's:@l_prefix@;%{l_prefix};g' \
    -e 's:@l_musr@;%{l_musr};g' -e 's:@l_mgrp@;%{l_mgrp};g' \
    %{SOURCE rc.apache} $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/

# strip installation binaries
strip $RPM_BUILD_ROOT%{l_prefix}/bin/* 2> /dev/null || true
strip $RPM_BUILD_ROOT%{l_prefix}/sbin/* 2> /dev/null || true

# determine installation tree files
%{l_rpmtree} files -v -ofiles -r$RPM_BUILD_ROOT \
    %{l_files_std} \
%if "%{with_suexec}" == "yes"
    '%attr(4755,root,%{l_mgrp}) %{l_prefix}/sbin/suexec' \
%endif

    '%config %{l_prefix}/etc/apache/*' \
%if "%{with_mod_ssl}" == "yes"
    '%config %{l_prefix}/etc/apache/ssl.crl/*.crl' \

```

```

        '%config %{l_prefix}/etc/apache/ssl.crt/*.crt' \
        '%config %{l_prefix}/etc/apache/ssl.csr/*.csr' \
        '%config %{l_prefix}/etc/apache/ssl.key/*.key' \
        '%config %{l_prefix}/etc/apache/ssl.prm/*.prm' \
%endif
        '%config %attr(444,%{l_musr},{l_mgrp}) %{l_prefix}
/etc/apache/apache.base'

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

%post
%if "%{with_mod_auth_pam}" == "yes"
    # add PAM configuration entry
    if [ $1 -eq 1 ]; then
        $RPM_INSTALL_PREFIX/sbin/pamtool --add --smart --name=apache
    fi
%endif

%preun
%if "%{with_mod_auth_pam}" == "yes"
    # remove PAM configuration entry
    if [ $1 -eq 0 ]; then
        $RPM_INSTALL_PREFIX/sbin/pamtool --remove --smart --name=apache
    fi
%endif

```

To reflect the OpenPKG dependencies a patch was created for the Apache configure script.

```

--- srclib/apr-util/configure    2002-09-23 21:41:29.000000000 +0200
+++ srclib/apr-util/configure    2002-09-23 21:41:51.000000000 +0200
@@ -4255,7 +4255,7 @@

     if test ${apu_has_ldap} != "define"; then
         ldaplib="ldap"
-        extralib="-llber"
+        extralib="-llber -ldl -lresolv -lcrypt -lsasl2 -lssl -lcrypto -ldb"
         unset ac_cv_lib_${ldaplib}_ldap_init
         as_ac_Lib=`echo "ac_cv_lib_${ldaplib}_ldap_init" | $as_tr_sh`
         echo "$as_me:$LINENO: checking for ldap_init in -l${ldaplib}" >&5

```

Inconsistencies with respect to the linking of LDAP libraries made it necessary to introduce the mod-ldap.patch:

```

--- mod_auth_ldap.module Sun Oct 13 03:51:05 2002
+++ mod_auth_ldap.module Sun Oct 13 03:52:10 2002
@@ -8,14 +8,14 @@
     # if you installed LDAP headers in an unusual place,
     # modify the variable below to specify the ldap libraries, example:
     #     LDAP_INCLUDES="-I/usr/local/foo/include"
-   LDAP_INCLUDES=""
+   LDAP_INCLUDES='-I@PREFIX@/include'

##### LDAP Libraries #####
# if you installed LDAP stuff in an unusual place,
# modify the variable below to specify the ldap libraries, example:
#     LDAP_LIB="-L/usr/foo/lib -lldap -llber"
-   LDAP_LIBS=""
+   LDAP_LIBS='-L@PREFIX@/lib'

     error_occurred=0
--- Makefile Sun Oct 13 01:30:17 2002
+++ Makefile Sun Oct 13 01:47:12 2002
@@ -24,16 +24,16 @@
     CPP=gcc -E
     TARGET=httpd
     OPTIM=
-SSL_BASE=/usr/local/ssl
+SSL_BASE=@PREFIX@
     SSL_BINDIR=$(SSL_BASE)/bin
     SSL_INCDIR=$(SSL_BASE)/include
     SSL_LIBDIR=$(SSL_BASE)/lib
-SSL_PROGRAM=/usr/local/ssl/bin/openssl
+SSL_PROGRAM=@PREFIX@/bin/openssl
     SSL_VERSION=-DMOD_SSL_VERSION=\"2.7.1\"
     SSL_CFLAGS= -DSSL_COMPAT -DSSL_USE_SDBM -I$(SSL_INCDIR)
     SSL_VENDOR_OBJS=
     SSL_VENDOR_OBJS_PIC=
-CFLAGS1= -DLINUX=2 -DMOD_SSL=207101 -I/usr/local/open-ldap/include \
    -DUSE_HSREGEX -DEAPI -DUSE_EXPAT -I$(SRCDIR)/lib/expat-lite \
    -DNO_DL_NEEDED
+CFLAGS1= -DLINUX=2 -DMOD_SSL=207101 -I@PREFIX@/include -DUSE_HSREGEX \
    -DEAPI -DUSE_EXPAT -I$(SRCDIR)/lib/expat-lite -DNO_DL_NEEDED
INCLUDES1=
LIBS_SHLIB=
LDFLAGS1= -L$(SSL_LIBDIR)
@@ -41,7 +41,7 @@
     REGLIB=regex/libregex.a
     EXPATLIB=lib/expat-lite/libexpat.a
     RANLIB=ranlib
-LIBS1= /usr/local/open-ldap/lib/libldap.a \
    /usr/local/open-ldap/lib/liblber.a -lm -lcrypt -lssl -lcrypto
+LIBS1= @PREFIX@/lib/libsas12.a @PREFIX@/lib/libldap.a \
    @PREFIX@/lib/liblber.a -lm -lcrypt -lssl -lcrypto
##

```

```
## (End of automatically generated section)  
##
```

Chapter 12. Monit Daemon

As the Kolab server requires different services like MTA, webserver etc. it is necessary to control and monitor them. This is done via the daemon Monit. Unfortunately this program is not available from OpenPKG. So we made an OpenPKG specfile to have monit available within the Kolab environment.

```
%ifndef      with_fsl
%define      with_fsl      no
%endif

#  package information
Name:        monit
Summary:     Process monitor and restart utility
URL:         http://www.tildeslash.com/monit/
Vendor:      Contributors to the monit codebase
Packager:    The OpenPKG Project
Distribution: OpenPKG
Group:       Utilities/Console
License:     GPL
Version:     3.1
Release:     20030125

#  list of sources
%define url  http://www.tildeslash.com/monit/dist/
%{name}-%{version}.tar.gz
Source0:     %( [ ! -f %{SOURCE monit-%{version}.tar.gz} ] \
&& wget -c %url; echo %url )
Source1:     rc.monit
Source2:     Makefile
Patch0:      nopermcheck.patch
Patch1:      kolab.patch

#  build information
Prefix:      %{l_prefix}
BuildRoot:   %{l_buildroot}
BuildPreReq: OpenPKG, openpkg >= 1.1.0, gcc
PreReq:      OpenPKG, openpkg >= 1.1.0
%if "%{with_fsl}" == "yes"
BuildPreReq: fsl
PreReq:      fsl
%endif
AutoReq:     no
AutoReqProv: no

%description
monit is an utility for monitoring daemons or similar programs running on
a Unix system. It will start specified programs if they are not running
and restart programs not responding.
```

```

%prep
%setup -q
%patch -p0
%patch1 -p0

%build
    CC="%{l_cc}" \
    CFLAGS="%{l_cflags -O}" \
%if "%{with_fsl}" == "yes"
    LIBS="`%{l_prefix}/bin/fsl-config --libs`" \
    LDFLAGS="`%{l_prefix}/bin/fsl-config --ldflags`" \
%endif
    ./configure \
--prefix=%{l_prefix} \
--with-ssl-dir=%{l_prefix} \
--sysconfdir=%{l_prefix}/etc/monit \
--sharedstatedir=%{l_prefix}/var
    %{l_shtool} subst -e "s;check_rcfile\(controlfile\);FALSE;" p.y
    %{l_shtool} subst -e "s;/var/run;%{l_prefix}/var/monit;g" config.h
    %{l_shtool} subst -e "s;@@@l_prefix@@@;%{l_prefix};g" files.c
    %{l_shtool} subst -e "s;@@@l_prefix@@@;%{l_prefix};g" monitor.h
    %{l_make} %{l_mflags}

%install
    rm -rf $RPM_BUILD_ROOT
    %{l_make} %{l_mflags} install DESTDIR=$RPM_BUILD_ROOT
    strip $RPM_BUILD_ROOT%{l_prefix}/bin/* >/dev/null 2>&1 || true
    mkdir -p $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/
    mkdir -p $RPM_BUILD_ROOT%{l_prefix}/etc/monit
    mkdir -p $RPM_BUILD_ROOT%{l_prefix}/var/monit
    %{l_shtool} install -c -e "s;l_prefix@;%{l_prefix};g" \
-m 755 %{SOURCE rc.monit} \
$RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/rc.monit
    %{l_shtool} install -c -m 700 monitrc \
$RPM_BUILD_ROOT%{l_prefix}/etc/monit/monit.conf
    %{l_rpmtree} files -v -ofiles -r$RPM_BUILD_ROOT %{l_files_std}

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

```

We need to alter the configuration in order to be able to adapt it with the bootstrapping procedure. See this kolab.patch:

```

--- monitor.h.orig 2003-01-26 16:14:56.000000000 +0100
+++ monitor.h 2003-01-27 11:34:25.000000000 +0100
@@ -39,7 +39,7 @@
#include "ssl.h"

```

```

#define VERSION                PACKAGE_VERSION
-#define MONITRC                "monitrc"
+#define MONITRC                "@@l_prefix@@/etc/monit/monit.conf"
#define TIMEFORMAT              "%Z %b %e %T"
#define STRERROR                strerror(errno)
#define STRLEN                  256
--- files.c.orig 2003-01-26 16:15:14.000000000 +0100
+++ files.c 2003-01-26 16:17:57.000000000 +0100
@@ -159,7 +159,8 @@
    if(exist_file(MONITRC)) {

        memset(rcfile, 0, STRLEN);
-       snprintf(rcfile, STRLEN, "%s/%s", Run.Env.cwd, MONITRC);
+       // snprintf(rcfile, STRLEN, "%s/%s", Run.Env.cwd, MONITRC);
+       snprintf(rcfile, STRLEN, "@@l_prefix@@/etc/monit/monit.conf");

        return (rcfile);

@@ -171,8 +172,8 @@
    if(exist_file(rcfile))
        return (rcfile);

- log("%s: Cannot find the control file at ~/.%s, ./%s or at /etc/%s\n",
-     prog, MONITRC, MONITRC, MONITRC);
+ log("%s: Cannot find the control file at ~/.%s, ./%s or at \
@@l_prefix@@/etc/monit/monit.conf\n",
+     prog, MONITRC, MONITRC);

    exit(1);

```

As there was a bug within monit we corrected it right away with the nopermcheck.patch:

```

--- p.y.orig 2003-01-26 16:12:46.000000000 +0100
+++ p.y 2003-01-26 16:13:04.000000000 +0100
@@ -644,11 +644,11 @@

    processlist= tail= current= NULL;

- if(! check_rcfile(controlfile)) {
+/* if(! check_rcfile(controlfile)) {

    return (FALSE);

- }
+ */

    if ((yyin = fopen(controlfile,"r")) == (FILE *)NULL) {

```

To make monit aware of the changed config file location (/kolab/etc/monit/monit.conf) this patch is supplied:

```

--- monitor.h.orig      2003-01-26 16:14:56.000000000 +0100
+++ monitor.h          2003-01-27 11:34:25.000000000 +0100
@@ -39,7 +39,7 @@
 #include "ssl.h"

 #define VERSION          PACKAGE_VERSION
-#define MONITRC          "monitrc"
+#define MONITRC          "@@l_prefix@@/etc/monit/monit.conf"
 #define TIMEFORMAT      "%Z %b %e %T"
 #define STRERROR        strerror(errno)
 #define STRLEN          256
--- files.c.orig       2003-01-26 16:15:14.000000000 +0100
+++ files.c            2003-01-26 16:17:57.000000000 +0100
@@ -159,7 +159,8 @@
     if(exist_file(MONITRC)) {

         memset(rcfile, 0, STRLEN);
-        snprintf(rcfile, STRLEN, "%s/%s", Run.Env.cwd, MONITRC);
+        // snprintf(rcfile, STRLEN, "%s/%s", Run.Env.cwd, MONITRC);
+        snprintf(rcfile, STRLEN, "@@l_prefix@@/etc/monit/monit.conf");

         return (rcfile);

@@ -171,8 +172,8 @@
     if(exist_file(rcfile))
         return (rcfile);

-    log("%s: Cannot find the control file at ~/.%s, ./%s or at /etc/%s\n",
-        prog, MONITRC, MONITRC, MONITRC);
+    log("%s: Cannot find the control file at ~/.%s, ./%s or at @@l_prefix@@/etc/monit/moni
+        prog, MONITRC, MONITRC);

     exit(1);

```

An OpenPKG startup script was developed:

```

#!@l_prefix@/lib/openpkg/bash @l_prefix@/etc/rc
#
##
## rc.monit -- Run-Commands for monit
##

```

```

# description: monit is a simple daemon process to restart processes if \
#             they die. It can also check tcp and udp ports to make sure \
#             that they are responding.

%config
    monit_enable="yes"

%start -p 200 -u root
    if opServiceEnabled monit; then
        @l_prefix@/bin/monit -c @l_prefix@/etc/monit/monit.conf -l syslog -d 60
    fi

%stop -p 200 -u root
    if opServiceEnabled monit; then
        if [ -f @l_prefix@/var/monit/monit.pid ]; then
            kill -TERM `cat @l_prefix@/var/monit/monit.pid`
        fi
    fi

%restart -u root
    if opServiceEnabled monit; then
        if [ -f @l_prefix@/var/monit/monit.pid ]; then
            kill -TERM `cat @l_prefix@/var/monit/monit.pid`
            sleep 2
        fi
        @l_prefix@/bin/monit -c @l_prefix@/etc/monit/monit.conf -l syslog -d 60
    fi

%reload -u root
    if opServiceEnabled monit; then
        if [ -f @l_prefix@/var/monit/monit.pid ]; then
            kill -HUP `cat @l_prefix@/var/monit/monit.pid`
        fi
    fi

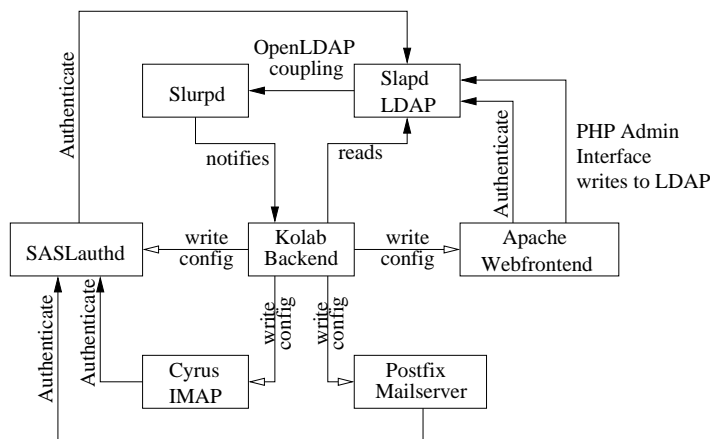
```

Chapter 13. Getting Things Together

We describe here how all services are configured in order to get a working Kolab server. The major efforts in creating the Kolab server were integrating the various open source projects and programming of the PHP4 administrative web interface.

13.1. Kolab Runtime Configuration

All configuration data is stored inside LDAP. The connection between LDAP and the various configuration files of the Kolab services is done via the Kolab configuration backend. An overview follows.



The slapd serves LDAP requests and is tightly coupled with the slurpd daemon. Slurpd is meant to keep a slave LDAP server consistent with the master LDAP server. The Kolab configuration backend acts as slave LDAP server in that regard.

When the Kolab configuration backend receives LDAP updates, it creates configuration files for all services. Therefore template files get filled up with the LDAP data and are placed into the according servers config file location. For the Cyrus imapd further actions need to be taken (adding/removing/modifying users, shared folders, or acl's).

The exact files that the Kolab backend creates are:

1. /kolab/var/kolab/www/admin/include/session_vars.php
2. /kolab/etc/postfix/main.cf
3. /kolab/etc/postfix/master.cf
4. /kolab/etc/postfix/transport

5. /kolab/etc/postfix/canonical
6. /kolab/etc/saslauthd.conf
7. /kolab/etc/imapd/imapd.conf
8. /kolab/etc/imapd/cyrus.conf
9. /kolab/etc/imapd/imapd.group
10. /kolab/etc/apache/apache.conf
11. /kolab/etc/apache/legacy.conf
12. /kolab/etc/apache/php.ini
13. /kolab/etc/proftpd/proftpd.conf
14. /kolab/etc/openldap/slapd.conf
15. /kolab/etc/monit/monit.conf

The following logic is applied here

- configuration setting can be found beneath the LDAP root DN in the k=kolab subtree
- parameters have speaking names with underscores
- configuration template files have placeholders named after the scheme "@@@" < parameter name > "@@@"
- the kolab backend reads all kolab parameters and applies a search/replace operation while copying the modified templates over the existing configuration files of the various services

In result, adding a parameter to the kolab configuration system consists of the following steps:

- Add support for the parameter to the web administration interface
- Add a placeholder to the according configuration template

It is assured that the respective daemons learn about their new configuration and reload it with minimal service interruption.

The FTP daemon plays a special role here. It is only activated as required, to support FTP freebusy uploads for legacy clients.

The apache servers configuration gets the legacy.conf included as required, to support the HTTP freebusy download function for legacy clients.

Technically, the Kolab configuration backend is a perl script which listens in an endless loop for LDAP updates and keeps the LDAP directory in sync with the rest of the Kolab configuration. The script is called "kolab". A startup script is provided. The usual precautions (pid file) have been taken to ensure a daemon-like operation. This script is the reason for the dependency to various perl CPAN modules, which we have mentioned earlier.

13.2. Kolab Administrative Web Interface

The Web interface is entirely written in PHP4. The Apache webserver with mod_php is used to serve request to the web interface.

The web interface supports SSL encryption and authentication against LDAP. It is important to understand that all critical operations on LDAP are done only with the credentials the user has supplied. The web interface does not act with administrative permissions on the LDAP server unless the user has provided the according credentials. All access permission control is done entirely by the LDAP server.

The web interface does not work on local files on the Kolab server. It only communicates with the LDAP server. The Kolab configuration backend learns asynchronously about LDAP changes and takes appropriate actions independently.

The actual user interface is described in a separate document.

13.3. Kolab Installation Process

13.3.1. Services installation

The dependency list of kolab looks like this:

1. openldap
2. sasl
3. imapd
4. apache
5. proftpd
6. monit
7. and finally: kolab

13.3.2. Kolab Bootstrapping

If the Kolab RPM is installed the bootstrapping procedure will be started right away. The user does not need to invoke the bootstrapping script manually. We present the script here just for documentation purposes.

```
#!/kolab/bin/perl

# (c) 2002 Tassilo Erlewein <tassilo.erlewein@erfrakon.de>
# (c) 2002 Martin Konold <martin.konold@erfrakon.de>
# (c) 2002 Achim Frank <achim.frank@erfrakon.de>

# kolab_bootstrap Version 0.91
# create unix configuration files from data source (files or LDAP)
# and templates

use strict;
use vars qw($opt_b);

use URI;
use IO::File;
use IO::Select;
use Net::LDAP;
use Net::LDAP::Entry;
use File::Copy;
use Getopt::Std;
use Sys::Hostname;

my $kolab_prefix = "/kolab";
my $kolab_config = $kolab_prefix."/etc/kolab/kolab.conf";

print "\nKOLAB BOOTSTRAP\n\n";

my $fd = IO::File->new($kolab_config, "r")
  || die "could not open $kolab_config";
my %kolab_config;
foreach (<$fd>) {
  if (/^(.*) : (.*)/) {
    $kolab_config{$1} = $2;
  }
}
undef $fd;
my $bind_dn = $kolab_config{'bind_dn'} || die \
"could not read bind_dn from $kolab_config";
my $bind_pw = $kolab_config{'bind_pw'} || die \
"could not read bind_pw from $kolab_config";
my $ldap_uri = $kolab_config{'ldap_uri'} || die \
```

```

"could not read ldap_uri from $kolab_config";
my $base_dn = $kolab_config{'base_dn'} || \
die "could not read base_dn from $kolab_config";
my $php_dn = $kolab_config{'php_dn'} || \
die "could not read php_dn from $kolab_config";
my $php_pw = $kolab_config{'php_pw'} || \
die "could not read php_pw from $kolab_config";

if (!$bind_dn || !$bind_pw || !$ldap_uri || !$base_dn) {
    print "Please check $kolab_config/kolab.conf \
(seems to be incomplete)\n";
    die "and run kolab_bootstrap afterwards, manually";
}
my $fqdn = `hostname -f`;
chomp($fqdn);
(my $dummy, my $domain) = split(/\./, $fqdn, 2);
if (!$domain) { $domain = $fqdn; }

if ($base_dn =~ /\@\@\@\/ || $bind_dn =~ /\@\@\@\/ || \
$bind_pw =~ /\@\@\@\/) {
    print "Generating default configuration:\n";
    if ($base_dn =~ /\@\@\@\/) {
        $base_dn = "";
        foreach my $dc ((split(/\./,$fqdn))) { $base_dn .= "dc=$dc,"; }
        chop $base_dn;
        print " base_dn : $base_dn\n";
    }
    if ($bind_dn =~ /\@\@\@\/) {
        $bind_dn =~ s/\@\@\@kolab_basedn\@\@\@/$base_dn/g;
        print " bind_dn : $bind_dn\n";
    }
    if ($bind_pw =~ /\@\@\@\/) {
        $bind_pw = `/kolab/bin/openssl passwd kolab`;
        chomp $bind_pw;
        print " bind_pw : $bind_pw\n";
    }
    if ($php_dn =~ /\@\@\@\/) {
        $php_dn =~ s/\@\@\@kolab_basedn\@\@\@/$base_dn/g;
    }
    if ($php_pw =~ /\@\@\@\/) {
        $php_pw = `/kolab/bin/openssl passwd nobody`;
        chomp $php_pw;
    }
}

$fd = IO::File->new($kolab_config, "w+") || die \
"could not open $kolab_config";
print $fd "base_dn : $base_dn\n";
print $fd "bind_dn : $bind_dn\n";
print $fd "bind_pw : $bind_pw\n";
print $fd "ldap_uri : $ldap_uri\n";
print $fd "php_dn : $php_dn\n";
print $fd "php_pw : $php_pw\n";
undef $fd;

```

```

    print "done modifying $kolab_config\n\n";
    print "IMPORTANT NOTE:\n";
    print "use login=manager and passwd=$bind_pw when you
log into the webinterface!\n\n";
}

# remove all application specific fsl config files as these are yet to be done
# having invalid files there hinders applications from starting up properly
# we delay this until there's a better understanding of the fsl stuff

unlink("$kolab_prefix/etc/fsl/fsl.postfix");
unlink("$kolab_prefix/kolab/etc/fsl/fsl.sasl");
unlink("$kolab_prefix/etc/fsl/fsl.apache");
unlink("$kolab_prefix/etc/fsl/fsl.slapped");
unlink("$kolab_prefix/kolab/etc/fsl/fsl.imapd");

my $confname = "$kolab_prefix/lib/sasl/smtpd.conf";
copy("$kolab_prefix/etc/kolab/smtpd.conf.template", $confname) || die \
"could not write to $confname";

getopts('b');

if ($opt_b) {
    print "prepare LDAP database...\n";
    if ($ldap_uri =~ /127\.0\.0\.1/ || $ldap_uri =~ /localhost/) {
        print "kill running slapd (if any)\n";
        system("killall -INT slapd >/dev/null 2>&1");
        sleep 1;
        system("killall -INT slapd >/dev/null 2>&1");
        sleep 1;
        system("killall -9 slapd >/dev/null 2>&1");
        sleep 1;
        system("killall -9 slapd >/dev/null 2>&1");
        sleep 1;
        my $tmpl = IO::File->
new("$kolab_prefix/etc/kolab/slapd.conf.template", "r")
    || die \
"could not read $kolab_prefix/etc/kolab/slapd.conf.template";
        my $slpd = IO::File->
new("$kolab_prefix/etc/openldap/slapd.conf", "w+")
    || die "could not write to $kolab_prefix/etc/openldap/slapd.conf";
        foreach (<$tmpl>) {
            s/\@\@\@base_dn\@\@\@/$base_dn/g;
            s/\@\@\@bind_dn\@\@\@/$bind_dn/g;
            s/\@\@\@bind_pw\@\@\@/$bind_pw/g;
            print $slpd $_;
        }
        undef $slpd;
        undef $tmpl;
        # now we must startup slapd
        print "temporarily start slapd under local port 7777\n";
        system("$kolab_prefix/libexec/slapd -u kolab
-h ldap://127.0.0.1:7777/ -f $kolab_prefix/etc/openldap/slapd.conf");
    }
}

```

```

    $ldap_uri = "ldap://127.0.0.1:7777/";
    sleep 3;
}

my $ldapuri = URI->new($ldap_uri) || warn \
"error: could not parse given uri";
my $ldap = Net::LDAP->new($ldapuri->host, port=> \
$ldapuri->port) || warn "could not connect ldap server";
if ($ldap) {
    $ldap->bind($bind_dn, password=> $bind_pw) || \
warn "could not bind to ldap";
    my $mesg = $ldap->search(base=> "$base_dn", scope=> \
'exact', filter=> "(objectclass=*)");
    if ($mesg && $mesg->count != 1) {
        print "no $base_dn object found, creating one\n";
        $mesg = $ldap->add ($base_dn, attr=> ['dc'=> \
$domain, 'objectclass'=> ['top', 'domain'] ]);
    }
    $mesg && $mesg->code && warn "failed to write basedn entry : ", \
$mesg->error;
    $mesg = $ldap->search(base=> "k=kolab,$base_dn", \
scope=> 'exact', filter=> "(objectclass=*)");
    if ($mesg && $mesg->count != 1) {
        print "no kolab config object in ldap, generating
a reasonable default\n";
    } else {
        print "modifying existing kolab config object\n";
    }

    # create kolab config object
    my $ldapobject = Net::LDAP::Entry->new;
    my $mynetworkinterfaces = "127.0.0.1/8";
    my @net=`ifconfig|grep -v 127.0.0|grep \"inet addr\"| \
sed 's/:/ /g'|awk \'{print \$3 \"/\" 24}\'';
    chomp @net;
    foreach my $nets (@net){
        $mynetworkinterfaces .= ", ".$nets;
    }
    print "mynetworkinterfaces: ".$mynetworkinterfaces."\n";

    $ldapobject->replace(
        'fqhostname' => $fqdn,
        'postfix-mydomain' => $domain,
        #'postfix-relaydomains' => "",
        'postfix-mydestination' => "\$mydomain",
        'postfix-mynetworks' => $mynetworkinterfaces,
        #'postfix-relayhost' => "",
        #'postfix-transport' => "",
        'cyrus-autocreatequota' => 100000,
        'cyrus-admins' => "manager",
        'cyrus-imap' => "TRUE",
        'cyrus-pop3' => "TRUE",
        'cyrus-imaps' => "TRUE",

```

```

        'cyrus-pop3s' => "TRUE",
        'cyrus-sieve' => "TRUE",
        'apache-http' => "FALSE",
        'proftpd-ftp' => "FALSE",
        #'proftpd-defaultquota' => 100000,
        #'proftpd-userPassword' => "freebusy",
'uid' => "freebusy",
        'userPassword' => "freebusy",
        'objectclass' => ['top', 'kolab' ] );
$ldapobject->dn("k=kolab,$base_dn");
$msg = $ldapobject->update($ldap);
$msg && $msg->code && warn "failed to write entry: ", \
$msg->error;
undef $ldapobject;

# create internal user topnode
$ldapobject = Net::LDAP::Entry->new;
$ldapobject->replace('cn' => 'internal', \
'objectclass' => ['top', 'namedObject']);
$ldapobject->dn("cn=internal,$base_dn");
$msg = $ldapobject->update($ldap);
$msg && $msg->code && warn "failed to write entry: ", \
$msg->error;
undef $ldapobject;

# create external user topnode
$ldapobject = Net::LDAP::Entry->new;
$ldapobject->replace('cn' => 'external', \
'objectclass' => ['top', 'namedObject']);
$ldapobject->dn("cn=external,$base_dn");
$msg = $ldapobject->update($ldap);
$msg && $msg->code && warn "failed to write entry: ", \
$msg->error;
undef $ldapobject;

# create admin group
$ldapobject = Net::LDAP::Entry->new;
$ldapobject->replace('cn' => 'admin',
'objectclass' => ['top', 'groupOfNames'],
        'member' => "cn=manager,$base_dn");
$ldapobject->dn("cn=admin,$base_dn");
$msg = $ldapobject->update($ldap);
$msg && $msg->code && warn "failed to write entry: ",
$msg->error;
undef $ldapobject;

# create manager user
$ldapobject = Net::LDAP::Entry->new;
$ldapobject->replace('cn' => 'manager', 'sn' => 'n/a',
'uid' => 'manager', 'userPassword' => $bind_pw,
'objectclass' => ['top', 'inetOrgPerson']);
$ldapobject->dn($bind_dn);
$msg = $ldapobject->update($ldap);

```

```

    $mesg && $mesg->code && warn "failed to write entry: ",
$mesg->error;
    undef $ldapobject;

    # create php read-only user
    $ldapobject = Net::LDAP::Entry->new;
    $ldapobject->replace('cn' => 'nobody', 'sn' => 'n/a n/a',
'uid' => 'nobody', 'userPassword' => $php_pw,
'objectclass' => ['top','inetOrgPerson']);
    $ldapobject->dn("cn=nobody,$base_dn");
    $mesg = $ldapobject->update($ldap);
    $mesg && $mesg->code && warn "failed to write entry: ",
$mesg->error;
    undef $ldapobject;

    # create maintainer group
    $ldapobject = Net::LDAP::Entry->new;
    $ldapobject->replace('cn' => 'maintainer',
'objectclass' => ['top','groupOfNames']);
    $ldapobject->dn("cn=maintainer,$base_dn");
    $mesg = $ldapobject->update($ldap);
    $mesg && $mesg->code && warn "failed to write entry: ",
$mesg->error;
    undef $ldapobject;

    $ldap->unbind;
}
print "LDAP setup finished\n\n";

print "Create initial config files for postfix, apache,
proftpd, cyrus imap, saslauthd\n";
print " running $kolab_prefix/etc/kolab/kolab -v -o -l$ldap_uri\n";
system("$kolab_prefix/etc/kolab/kolab -v -o -l$ldap_uri");

if ($ldap_uri =~ /127\.0\.0\.1/ || $ldap_uri =~ /localhost/) {
    print "\nkill temporary slapd\n\n";
    system("killall -INT slapd >/dev/null 2>&1");
    system("killall -INT slapd >/dev/null 2>&1");
    system("killall -9 slapd >/dev/null 2>&1");
    system("killall -9 slapd >/dev/null 2>&1");
}
exit;
}

```

13.3.3. Kolab configuration

Here we provide the appropriate RPM spec for Kolab file. The source RPM of kolab also holds all the necessary files needed to operate to services provided by the server. This includes the HTML pages of

the web frontend. For space reasons we can not contain all the files within this document but all the files are publicly available via anonymous CVS as described earlier.

By installing the Kolab RPM all the necessary configurations of the services provided by the kolab server will be made automatically by bootstrapping scripts. This includes the creation of OpenSSL certificates used by the services apache, postfix and openldap/sasl. Certificate creation is done using an extra script. This is documented in the openssl section of this document.

```
##
## kolab.spec -- OpenPKG RPM Specification
## Copyright (c) 2002 Erfrakon
## Copyright (c) 2002 Martin Konold <martin.konold@erfrakon.de>
## Copyright (c) 2002 Tassilo Erlewein <tassilo.erlewein@erfrakon.de>

%define          V_kolab    1.0beta2

#  package information
Name:            kolab
Summary:         Kolab Groupware Server
URL:             http://ftp.kde.org/pub/kde/unstable/server/
kolab/kolab-current
Vendor:          Erfrakon http://www.erfrakon.de
Packager:        The Kroupware Project
Distribution:    Kolab
Group:           core
License:         GPL
Version:         %{V_kolab}
Release:         20030209

#  list of sources
Source0:         ftp://ftp.kde.org/pub/kde/unstable/server/kolab/
                 kolab-current/kolab-%{version}.tar.gz
Source1:         Makefile
Source2:         rc.kolab

#  build information
Prefix:          %{l_prefix}
BuildRoot:       %{l_buildroot}
BuildPreReq:     OpenPKG, openpkg >= 1.1.0
PreReq:          OpenPKG, openpkg >= 1.1.0, openldap, postfix, imapd,
sasl, apache, proftpd, perl-ldap, monit
AutoReq:         no
AutoReqProv:     no

%description
    Kolab is the KDE Groupware Server that provides full groupware features
    to either KDE kolab clients or Microsoft Outlook[tm] clients with the
    Bynari Insight Connector http://www.bynari.net. In addition it is
    a robust and fleixle general imap mail server with LDAP addressbook
    and nice web gui.
```

```

%prep
    %setup -q -c

%build
    #echo "Finished"

%install
    rm -rf $RPM_BUILD_ROOT

    %{l_shtool} mkdir -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/kolab
    %{l_shtool} mkdir -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d
    %{l_shtool} mkdir -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/var/kolab/log
    %{l_shtool} mkdir -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/var/kolab/
www/cgi-bin
    %{l_shtool} mkdir -p -m 777 $RPM_BUILD_ROOT%{l_prefix}/var/kolab/
www/freebusy
    %{l_shtool} mkdir -p -m 755 $RPM_BUILD_ROOT%{l_prefix}/var/kolab/
www/icons
    %{l_shtool} mkdir -p -m 777 $RPM_BUILD_ROOT%{l_prefix}/var/kolab/
www/locks

    %{l_shtool} install -c -m 755 -e "s:@l_prefix@;%{l_prefix};g" \
    %{SOURCE rc.kolab} \
    $RPM_BUILD_ROOT%{l_prefix}/etc/rc.d/
    # gets overwritten later
    %{l_shtool} install -c -m 600 CAcert.pem \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 600 cert.pem \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 600 key.pem \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 744 -e \
    's:@@kolab_prefix@@;%{l_prefix};g' \
    kolab_sslcert.sh $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 smtpd.conf.template \
    canonical.template transport.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 -e \
    's:@@kolab_prefix@@;%{l_prefix};g' cyrus.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 \
    -e 's:@@kolab_prefix@@;%{l_prefix};g' \
    -e 's:@l_nusr@;%{l_nusr};g' \
    -e 's:@l_ngrp@;%{l_ngrp};g' httpd.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 -e \
    's:@@kolab_prefix@@;%{l_prefix};g' legacy.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 -e \
    's:@@kolab_prefix@@;%{l_prefix};g' imapd.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 imapd.group.template \

```

```

$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 744 -e \
's;@@@kolab_prefix@@@;%{l_prefix};g' kolab \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 kolab.conf \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 kolab.schema \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 744 -e \
's;@@@kolab_prefix@@@;%{l_prefix};g' \
-e "s;@@@l_musr@@@;%{l_musr};g" \
    -e "s;@@@l_rgrp@@@;%{l_rgrp};g" kolab_bootstrap \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 \
    -e 's;@@@kolab_prefix@@@;%{l_prefix};g' \
    -e "s;@@@l_musr@@@;%{l_musr};g" \
    -e "s;@@@l_rgrp@@@;%{l_rgrp};g" \
    -e "s;@@@l_nusr@@@;%{l_nusr};g" main.cf.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 master.cf.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 -e \
's;@@@kolab_prefix@@@;%{l_prefix};g' monit.conf.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 php.ini.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 \
    -e 's;@@@kolab_prefix@@@;%{l_prefix};g' \
-e 's;@@@l_nusr@@@;%{l_nusr};g' -e \
's;@@@l_nuid@@@;%{l_nuid};g' \
-e 's;@@@l_ngrp@@@;%{l_ngrp};g' -e \
's;@@@l_ngid@@@;%{l_ngid};g' proftpd.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 saslauthd.conf.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 session_vars.php.template \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 644 -e \
's;@@@kolab_prefix@@@;%{l_prefix};g' slapd.conf.template \
    $RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    %{l_shtool} install -c -m 744 -e \
's;@@@kolab_prefix@@@;%{l_prefix};g' workaround.sh \
$RPM_BUILD_ROOT%{l_prefix}/etc/kolab/
    cp -r admin/ $RPM_BUILD_ROOT%{l_prefix}/var/kolab/www/

    %{l_rpmtree} files -v -ofiles -r$RPM_BUILD_ROOT %{l_files_std} \
    '%config %{l_prefix}/etc/kolab/*.pem' \
    '%config %{l_prefix}/etc/kolab/*.schema' \
    '%config %{l_prefix}/etc/kolab/kolab.conf'

%post
    %{l_prefix}/etc/kolab/kolab_bootstrap -b
    %{l_prefix}/etc/kolab/kolab_sslcert.sh 2>/dev/null

```

```

echo
echo "Kolab should now be setup - you should be able to start"
echo "the server with '%{l_prefix}/etc/rc.d/rc.monit start'"
echo "- please read the documentation!"

%files -f files

%clean
    rm -rf $RPM_BUILD_ROOT

```

13.4. LDAP Directory Schema

We found it necessary to slightly modify the pre-defined OpenLDAP schemes. Fields were added to some object types, conserving the already existing fields.

The resulting kolab scheme is part of the kolab RPM file, which is way too long to present it here. Most parts of it come with openldap and simply contain standardized object types.

13.4.1. Person

```

objectclass ( 2.5.6.6 NAME 'person' SUP top STRUCTURAL
    MUST ( sn $ cn )
    MAY ( userPassword $ telephoneNumber $ seeAlso $ description $ mail
$displayName $ uid $givenname ) )

```

13.4.2. OrganisationalPerson

```

objectclass ( 2.5.6.7 NAME 'organizationalPerson' SUP person STRUCTURAL
    MAY ( title $ x121Address $ registeredAddress $ destinationIndicator $
    preferredDeliveryMethod $ telexNumber $
teletexTerminalIdentifier $
    telephoneNumber $ internationaliSDNNumber $
    facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
    postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l $ c ) )

```

13.5. LDAP Server Configuration

The configuration template used by the Kolab bootstrapping procedure is given here. The file is expected to live in `/kolab/etc/kolab/slapd.conf.template`.

```
include          /kolab/etc/kolab/kolab.schema
pidfile          /kolab/var/openldap/slapd.pid
argsfile         /kolab/var/openldap/slapd.args
repllogfile      /kolab/var/openldap/repllog
schemacheck      on
lastmod          on
TLSCertificateFile  /kolab/etc/kolab/cert.pem
TLSCertificateKeyFile /kolab/etc/kolab/key.pem
require          none
allow            bind_v2
loglevel         0
database         ldbm
suffix           "@@base_dn@@"
directory        /kolab/var/openldap/openldap-data
rootdn           "@@bind_dn@@"
rootpw           "@@bind_pw@@"
replica host=127.0.0.1:9999
               binddn="cn=replicator"
               bindmethod=simple credentials=secret
index objectClass eq
index uid         eq
index mail        eq
index alias       eq

access to attr=userPassword
      by group="cn=admin,@@base_dn@@" write
      by group="cn=maintainer,@@base_dn@@" write
      by self write
      by anonymous auth
      by * none
      stop

access to dn="(.*,)?cn=internal,@@base_dn@@"
      by group="cn=admin,@@base_dn@@" write
      by group="cn=maintainer,@@base_dn@@" write
      by self write
      by dn="cn=nobody,@@base_dn@@" read
      by anonymous auth stop

access to dn="k=kolab,@@base_dn@@"
      by group="cn=admin,@@base_dn@@" write
      by * read stop

access to *
      by self write
```

```

by group="cn=admin,@@@base_dn@@@" write
by group="cn=maintainer,@@@base_dn@@@" write
by * read stop

```

13.6. SASL

The ldap-enhanced saslauthd daemon is configured by the file `/kolab/etc/kolab/saslauthd.conf.template`.

```

ldap_servers: @@@ldap_uri@@@
ldap_version: 3
ldap_search_base: @@@base_dn@@@
ldap_filter: (|(uid=%u)(mail=%u)(alias=%u))

```

Note that in the configuration file the LDAP search filter is determined by the behaviour of the authenticating daemons. Postfix authenticates using the fully qualified E-Mail address. The Cyrus IMAP daemon and the Apache webserver authenticate forwarding the uid taken from the user's application.

13.7. Postfix Configuration

The configuration template used by the Kolab bootstrapping procedure is given here. The file is expected to live in `/kolab/etc/kolab/main.cf.template`.

```

command_directory = /kolab/sbin
daemon_directory = /kolab/libexec/postfix
queue_directory = /kolab/var/postfix
mail_owner= kolab
setgid_group= kolab-r
default_privs= kolab-n
myhostname = @@@fqhostname@@@
mydomain = @@@postfix-mydomain@@@
myorigin = $myhostname
masquerade_domains = $mydomain
masquerade_exceptions = root
mynetworks = @@@postfix-mynetworks@@@
mydestination = @@@postfix-mydestination@@@
relay_domains =
canonical_maps = hash:/kolab/etc/postfix/canonical
virtual_maps = hash:/kolab/etc/postfix/virtual
relocated_maps = hash:/kolab/etc/postfix/relocated

```

```

transport_maps = hash:/kolab/etc/postfix/transport
alias_maps = hash:/kolab/etc/postfix/aliases
alias_database = hash:/kolab/etc/postfix/aliases
local_recipient_maps = $alias_maps
recipient_delimiter = +
mailbox_transport = lmtp:unix:/kolab/var/kolab/lmtp
smtpd_use_tls = yes
smtpd_tls_auth_only = no
smtpd_starttls_timeout = 300s
smtpd_timeout = 300s
smtpd_tls_CAfile = /kolab/etc/kolab/CAcert.pem
smtpd_tls_cert_file = /kolab/etc/kolab/cert.pem
smtpd_tls_key_file = /kolab/etc/kolab/key.pem
smtpd_tls_received_header = no
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
smtpd_recipient_restrictions = permit_mynetworks,
permit_sasl_authenticated,check_relay_domains
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain = $myhostname
smtpd_sasl_security_options = noanonymous

```

Within `master.cf` one has to create sockets for connection types of SMTP-TLS (Port 465/tcp) and to be compatible on Port 587/tcp for type connection. So the postfix master process listens on the tcp ports 25, 465 and 587 for incoming connections. Note: although with the use of ESMTP commands the allocation of other ports than 25/tcp can be omitted some clients (especially MS Outlook) need port 465/tcp in order to setup a TLS connection. Standard ESMTP uses port 25/tcp also for TLS. On connection after an EHLO the server states STARTTLS. The client can now switch to encryption on also issuing STARTTLS.

`/kolab/etc/kolab/master.cf.template` reads:

25	inet	n	-	n	-	-	smtpd
pickup	fifo	n	-	n	60	1	pickup
cleanup	unix	n	-	n	-	0	cleanup
qmgr	fifo	n	-	n	300	1	qmgr
rewrite	unix	-	-	n	-	-	trivial-rewrite
bounce	unix	-	-	n	-	0	bounce
defer	unix	-	-	n	-	0	bounce
flush	unix	n	-	n	1000?	0	flush
smtp	unix	-	-	n	-	-	smtp
showq	unix	n	-	n	-	-	showq
error	unix	-	-	n	-	-	error
local	unix	-	n	n	-	-	local
virtual	unix	-	n	n	-	-	virtual
lmtp	unix	-	-	n	-	-	lmtp

13.8. Cyrus IMAP Daemon

The configuration template used by the Kolab bootstrapping procedure is given here. The file is expected to live in `/kolab/etc/kolab/imapd.conf.template`. There also exists an empty groups file into which the bootstrapping procedure inserts groups used by the web administrative interface. At this development stage of Kolab there will be inserted the groups `admin` and `maintainer`.

```
configdirectory:      /kolab/var/imapd
partition-default:   /kolab/var/spool/imap
admins:              @@@cyrus-admins@@@
sasl_pwcheck_method: saslauthd
sasl_mech_list:      plain
sendmail:            /kolab/sbin/sendmail
allowanonymouslogin: no
allowplaintext:      yes
servername:          @@@fqhostname@@@
autocreatequota:    @@@cyrus-autocreatequota@@@
reject8bit:          no
munge8bit:           no
quotawarn:           90
timeout:             30
sievedir:            /kolab/var/spool/imap/sieve
lmtpsocket:          /kolab/var/kolab/lmtp
tls_cert_file:       /kolab/etc/kolab/cert.pem
tls_key_file:        /kolab/etc/kolab/key.pem
```

13.9. Apache Webserver

As the configuration file of the apache webserver is quite big and we only change a few lines we only present the relevant changes needed for Kolab setup. The file is expected to live in `/kolab/etc/kolab/httpd.conf.template`.

13.9.1. Global configuration

```
ServerRoot "/kolab"

User kolab-n
Group kolab-n

Listen 80
```

Listen 443

```

SSLVerifyClient      none
SSLCACertificateFile /kolab/etc/kolab/CAcert.pem
SSLSessionCache     dbm:/kolab/var/apache/log/ssl_scache
SSLSessionCacheTimeout 300
SSLMutex            file:/kolab/var/apache/log/ssl_mutex
SSLRandomSeed       startup builtin
SSLRandomSeed       connect builtin

```

<VirtualHost _default_:443>

```

SSLEngine           on
SSLCipherSuite      ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:
+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile  /kolab/etc/kolab/cert.pem
SSLCertificateKeyFile /kolab/etc/kolab/key.pem

```

<Files ~ "\.(cgi|shtml|phtml|php4|php3?)\$" >

```

    SSLOptions +StdEnvVars
</Files>

```

<Directory /kolab/var/kolab/www/cgi-bin>

```

    SSLOptions +StdEnvVars
</Directory>

```

</VirtualHost>

UseCanonicalName Off

DocumentRoot "/kolab/var/kolab/www"

<Directory />

```

    Options FollowSymLinks
    AllowOverride None
</Directory>

```

<Location />

```

    ErrorDocument 403 https://@@@fqdn@@@/admin/index.php
</Location>

```

HostnameLookups On

ErrorLog /kolab/var/apache/log/apache-error.log

LogLevel warn

```

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\"
\"%{User-Agent}i\" combined

```

```

LogFormat "%h %l %u %t \"%r\" %>s %b" common

```

```

LogFormat "%{Referer}i -> %U" referer

```

```

LogFormat "%{User-agent}i" agent

```

CustomLog /kolab/var/apache/log/apache-access.log common

ServerTokens Full

ServerSignature On

Alias /icons/ "/kolab/var/kolab/www/icons/"

```

<Directory "/kolab/var/kolab/www/icons">
  Options Indexes MultiViews
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>

ScriptAlias /cgi-bin/ "/kolab/var/kolab/www/cgi-bin/"

<Directory "/kolab/var/kolab/www/cgi-bin">
  AllowOverride None
  Options None
  Order allow,deny
  Allow from all
</Directory>

```

13.9.2. Providing free/busy

To access the free/busy list of all participating users in legacy mode it is necessary to configure the webserver so that it provides the free/busy lists via http(s). The default mode to upload and retrieve free/busy lists is via webdav over https.

The webdav protocol is supported by Microsoft since Windows 98 provided at least Internet Explorer 5.x and Outlook 2000 are installed. We verified correct operation on Windows 98, IE 6.0 and Outlook 2002.

Authentication for both uploads and downloads is done via AuthType Basic and the LDAP database. In order to protect the credentials from abuse it is a requirement to use TLS.

```

DavLockDB /kolab/var/kolab/www/locks/DAVlock
@@@legacy-mode@@@
<Directory "/kolab/var/kolab/www/freebusy">
  Dav On
  AllowOverride None
  Options None
  Order allow,deny
  <Limit GET PUT LOCK UNLOCK PROPFIND HEAD OPTIONS>
    Allow from all
    Require valid-user
  </Limit>
  AuthType Basic
  AuthName "Kolab Freebusy (webdav)"

  LDAP_Server @@@ldap_ip@@@
  LDAP_Port @@@ldap_port@@@

```

```

Base_DN "###base_dn###"
Bind_DN "###bind_dn###"
Bind_Pass "###bind_pw###"
UID_Attr uid
DavMinTimeout 600
<Directory>

```

13.9.3. Web administration

The configuration interface of the Kolab server is provided by HTML pages which are accessible via HTTPS. At the first login you need to supply the username manager and the password which is created by the Kolab bootstrapping procedure. You can lookup the password which is written into the file `/kolab/etc/openldap/slapd.conf`. The password you need for the user manager provided within the line at `rootpw`.

```

<Directory "/kolab/var/kolab/www/admin">
  AllowOverride None
  Options None
  Order allow,deny
  Allow from all
  AuthName "Kolab Admin Area"
  AuthType Basic
  LDAP_Server ###ldap_ip###
  LDAP_Port ###ldap_port###
  Bind_DN "###bind_dn###"
  Bind_Pass "###bind_pw###"
  Base_DN "###base_dn###"
  UID_Attr uid
  require valid-user
  SSLRequireSSL
</Directory>

```

13.10. ProFTPD (Legacy Support)

As the configuration file of the apache webserver is quite big and we only change a few lines we only present the relevant changes needed for Kolab setup. The file is expected to live in `/kolab/etc/kolab/proftpd.conf.template`.

```

ServerType          standalone
DefaultServer      on

```

```

Port                21
PersistentPasswd    off
LDAPServer          @ldap_ip@@@
LDAPDNInfo          " " "
LDAPDoAuth          on "@@base_dn@" "(uid=freebusy)"
ScoreBoardFile      /kolab/var/proftpd/score
LDAPDefaultUID      1002
LDAPForceDefaultUID on
LDAPDefaultGID      1002
LDAPForceDefaultGID on
LDAPHomedirOnDemand on
LDAPHomedirOnDemandPrefix /tmp
MaxInstances        40
User                kolab-n
Group               kolab-n
Umask               022
UseReverseDNS       off
MultilineRFC2228    on
ShowSymlinks        on
AllowOverwrite      on
RequireValidShell   no
LsDefaultOptions    "-l"
TimeoutLogin        60
TimeoutNoTransfer   60
TimeoutStalled      60
TimeoutIdle         60
LogFormat            default "%h %l %u %t \"%r\" %s %b"
LogFormat            auth    "%v [%P] %h %t \"%r\" %s"
LogFormat            write   "%h %l %u %t \"%r\" %s %b"
SystemLog            /kolab/var/proftpd/proftpd.log
DefaultRoot          /kolab/var/kolab/www/freebusy

<Global>
  IdentLookups       off
  DeferWelcome        off
  WTmpLog             off
</Global>
<Directory /*>
  AllowOverwrite      on
  <Limit STOR>
    AllowAll
  </Limit>
  <Limit WRITE READ DIRS>
    IgnoreHidden     on
    DenyAll
  </Limit>
</Directory>

```

Chapter 14. Possible Extensions

14.1. Virus Checking

By using the AMaViS (<http://www.amavis.org/>) interface it is possible to attach a mail virus scanner. AMaViS does not have the capability of filtering mails - it needs an additionally installed virus scanner.

Please note that currently no powerful virus scanner exists as free software.

14.2. Spam Filtering

Spam mails are quite common and annoying for users. There are two main possibilities how to handle this spam mails.

1. The MTA - postfix - is capable of handling header and body checks. Look here (<http://www.hispalinux.es/~data/postfix/>) for a detailed HOWTO with respect to the regular expression filtering capabilities of the MTA Postfix. It is possible to define multiple rules that match body or header area of incoming emails.
2. Additional programmes like SpamAssassin (<http://spamassassin.org/>) which provide a means of tagging incoming mails. The tags are created as additional headers in the mail. Apart from header and body regular expressions SpamAssassin can also use databases which provide well known spam sources. Within the client filters can be defined according to the individual preference of every user on how much spam rating he likes to accept.

14.3. Webmail Interface

It is possible to use a webinterface to access emails from a remote site. This is primarily intended for users who are off site and can not use a mail user agent. Note that not the full groupware functionality can be provided at this point in time by using a webmail interface. This can change in the future.

14.4. LDAP over TLS

The directory service OpenLDAP2 which Kolab uses can handle authentication and encryption. Accessing LDAP via TLS of course makes sense as the authentication also transfers password hashes that could be intercepted - and be exploited in replay attacks - if transferred in the clear. The service SASL which authenticates the services Postfix and Cyrus can access LDAP via TLS. As we have to address also the legacy clients we can not restrict direct access to LDAP to TLS.

14.5. LDAP Migration Tool

Many sites already have site wide user data available in some sort of directory service. If it is possible to export this user data into so called University of Michigan LDAP 3.3 / OpenLDAP ldif format it can be imported into Kolab's OpenLDAP.

Especially Microsoft Exchange 5.5 and also Microsoft Exchange 2000 user data can be exported after authentication into a ldif file which in turn can be inserted into the Kolab OpenLDAP service.

All directory services use slightly different schema's. The challenge of migration hereby is to adapt the exported LDAP data into Kolab's schema. This can be done by using scripts that are adapted to the originating directory service.

Chapter 15. Appendix

15.1. Handling Kolab's LDAP database on the commandline

As already described the actual layout of the LDAP directory is quite flexible. One can expect that it may be necessary to modify it some time in the future and the "migration" issue arises. This is a bit tricky but surely can be done. But be aware that the described command `slapcat` only reads the LDAP database locally on the server and does not connect via the network.

```
/kolab/etc/rc.d/openldap stop
slapcat -b suffix -l <ldap ldif file>
suffix may be "dc=my,dc=domain,dc=org"
mv /kolab/var/openldap/openldap-data/* <backup directory>
(process the ldif, adding new fields, etc.)
slapadd -b suffix </dev/null
"<" is a redirection here
slapadd -l <edited ldap ldif file>
/kolab/etc/rc.d/openldap start
```

The LDAP Interchange Format (LDIF) consists of ASCII text and may be processed with the usual tools (Sed, Awk, Perl, etc.). A syntax check is done when importing it back into the LDAP database. Unfortunately `slapadd` reports errors on the basis of whole records, seldomly an exact line number and a more descriptive help text is printed. But the LDIF format itself is reasonably simple. Also other tool programs may be used to help out here.