

Zarafa Server Manual

Server side installation and Administrator information



Addresses

Zarafa
Schieweg 2
2627 AN Delft
The Netherlands



Zarafa combines the usability of Outlook with the stability and flexibility of a Linux server.

This document will explain how to install/upgrade, configure and maintain Zarafa on your Linux server. We assume you have knowledge of your Linux distribution, and know a few basic Linux commands and know how to edit files.

Table of contents

1	Zarafa technical overview.....	6
1.1	Technical structure.....	6
1.2	Server.....	7
1.2.1	Configuration.....	7
1.2.2	Logging.....	7
1.3	Client.....	7
1.4	Delivery agent.....	8
1.5	Spooler.....	8
1.5.1	Configuration.....	8
1.6	Database.....	9
1.6.1	Attachments.....	9
1.7	Secure HTTP.....	9
1.8	Caching.....	10
1.9	Protocols and Connections.....	10
1.9.1	SOAP.....	11
1.9.2	HTTP Proxy.....	11
2	Installing Zarafa on a Linux distribution.....	12
2.1	System requirements.....	12
2.1.1	Software requirements.....	12
2.1.2	Hardware recommendations.....	13
2.2	Editions of Zarafa.....	13
2.2.1	Community edition.....	13
2.2.2	Zarafa commercial editions.....	13
2.2.3	User Licenses.....	14
2.2.4	Trial license.....	14
2.3	Installation.....	14
2.3.1	Manual installation.....	15
2.3.2	Notes on Debian and Ubuntu.....	16
2.3.3	Notes on x86_64 architectures.....	16
2.4	Configuration.....	16
2.4.1	Initialising the server.....	16
2.4.2	Setting the server license key.....	17
2.4.3	Configuring the server.....	17
2.4.4	Configuring services logging methods.....	18
2.4.5	Configuring the UNIX plugin.....	18
2.4.6	Configuring the LDAP plugin.....	18
2.5	Starting the services.....	19
2.6	Stopping the services.....	20
2.7	Reloading service configuration.....	20
2.8	Stores.....	20
2.8.1	Updating stores and user information.....	20

2.8.2	Deleting stores.....	21
2.9	Users.....	21
2.9.1	Creating users.....	21
2.9.2	Non-active users.....	21
2.10	Groups.....	22
2.10.1	Creating groups using zarafa-admin.....	22
2.11	Other admin commands.....	22
2.12	Sending email through the server.....	22
2.13	Delivery of email through the server.....	23
2.13.1	Postfix.....	23
2.13.2	Qmail.....	23
2.13.3	Procmail.....	23
2.14	Attachments outside the MySQL database.....	24
2.15	Local Mail Transfer Protocol (LMTP).....	24
2.16	Quota and quota monitoring.....	25
2.16.1	Setup server-wide quota.....	25
2.16.2	Setup quota for a single user.....	26
2.16.3	Monitoring for quota exceedings.....	26
2.16.4	Quota warning templates.....	26
2.17	POP3 and IMAP gateway.....	27
2.18	CALDAV and iCal gateway.....	27
2.19	Configuring webaccess.....	27
2.19.1	Configuring PHP.....	28
2.19.2	Configuring the webserver.....	28
2.20	Optimising Zarafa and MySQL.....	29
2.21	Upgrading.....	29
2.21.1	Finalising the upgrade.....	30
2.22	Troubleshooting.....	31
2.22.1	Server processes.....	31
2.22.2	Webaccess.....	31
2.23	Advanced configurations.....	32
2.23.1	Running as non-root user.....	32
2.23.2	SSL connections and certificates.....	32
2.23.3	Linux Zarafa clients from another server.....	33
2.23.4	Email delivery options.....	34
2.23.5	Junk e-mail.....	35
2.23.6	Temporary deliver failures.....	35
2.23.7	Out of Office message.....	35
2.23.8	'Send as' Permissions.....	35
2.24	Client Auto updater tool.....	36
2.24.1	Server side configuration.....	37
2.24.2	Client side configuration.....	38
2.24.2.1	Zarafa Launch Updater Application.....	38
2.24.2.2	Zarafa Updater Service.....	38
2.25	Soft Deletes.....	39
2.25.1	Configuration.....	39
2.26	Single Instance Attachment Storage.....	40
2.26.1	Single Instance and LMTP.....	40
3	Zarafa CALDAV / iCal gateway configuration.....	40
3.1	Security.....	41

3.2	iCal Gateway Configuration.....	41
3.2.1	SSL/TLS.....	42
3.3	Gateway Starting.....	42
3.4	Calendar access.....	43
4	Zarafa IMAP & POP3 gateway configuration.....	43
4.1	Security.....	44
4.2	Gateway Configuration.....	44
4.2.1	SSL/TLS.....	45
4.3	Gateway Starting.....	46
4.4	Important notes.....	46
5	Configuring LDAP or ADS with Zarafa.....	46
5.1	The Zarafa synchronisation principle.....	47
5.1.1	Add/Remove events.....	47
5.1.2	Group membership.....	48
5.1.3	LDAP server dependency.....	48
5.2	Setting up the LDAP repository.....	48
5.2.1	Setting up Zarafa for LDAP.....	49
5.3	Setting up Zarafa with Microsoft Active Directory.....	49
5.3.1	Setting up Active Directory for SSL access	50
5.3.2	Retrieving the CA certificate from your AD server.....	50
5.3.3	Configuring Zarafa for Microsoft Active Directory Service.....	51
5.3.4	Password checking with MS Active Directory.....	52
5.3.5	Using advanced LDAP attributes with Microsoft Active Directory.....	53
5.3.6	Set up Zarafa Active Directory Extension and schema files.....	53
5.3.6.1	Windows 2000 server.....	53
5.3.6.2	Windows 2003 server.....	54
5.4	Setting up Zarafa with OpenLDAP.....	54
5.4.1	Setting up SSL for OpenLDAP.....	55
5.4.2	Configuring OpenLDAP's slapd to use SSL.....	55
5.4.2.1	Testing your SSL / OpenLDAP connection.....	56
5.4.3	Configuring OpenLDAP to use Zarafa schemas.....	56
5.4.4	Configuring Zarafa for OpenLDAP.....	56
5.4.4.1	Password checking with OpenLDAP.....	57
5.5	Advanced configuration.....	58
5.5.1	Change unique attribute of Zarafa, in a live environment configuration.....	58
5.5.1.1	Example:.....	59
5.5.2	Send as Permissions option.....	60
5.5.3	Addresslists by condition.....	61
5.5.3.1	Setup addresslists.....	61
5.5.3.2	Condition examples.....	61
6	Single SignOn for Zarafa.....	62
6.1	Before attempting Single Sign On (SSO).....	62
6.2	SSO with ADS.....	62
6.2.1	Installing linux software.....	62
6.2.2	ADS: Specific network setup.....	62
6.2.3	Configuring Kerberos library.....	63
6.2.4	Joining the ADS domain.....	63
6.3	SSO with Samba.....	64
6.3.1	Installing linux software.....	64
6.3.2	Joining the domain.....	64

6.4	Up and running.....	64
7	Performance tuning for Zarafa.....	65
7.1	Hardware considerations.....	65
7.1.1	RAM Memory usage.....	65
7.1.2	Multiserver setups.....	65
7.1.3	Hardware considerations.....	65
7.1.4	More RAM is More Speed.....	66
7.1.5	RAID 1/10 is faster than RAID 5.....	66
7.1.6	Multiprocessor is not always faster.....	66
7.1.7	High rotation speed (RPMs) is better for disks.....	66
7.1.8	Hardware RAID.....	66
7.2	Memory Usage setup.....	66
7.2.1	Zarafa's Cell Cache (cache_cell_size).....	67
7.2.2	Zarafa's object cache (cache_object_size).....	67
7.2.3	Zarafa's indexedobject cache (cache_indexedobject_size).....	67
7.2.4	MySQL innodb_buffer_pool_size.....	67
7.2.5	MySQL innodb_log_file_size.....	67
7.2.6	MySQL innodb_log_buffer_size.....	68
7.2.7	MySQL query_cache.....	68
7.2.8	Setup of modules on different servers.....	68
8	Backup.....	68
8.1	Softdelete cache.....	69
8.2	Full database dump.....	69
8.2.1	SQL dump through mysqldump.....	69
8.2.2	Binary data dump via LVM Snapshotting.....	70
8.3	Brick-level backups.....	70
8.3.1	Backup format.....	70
8.3.2	Backup process.....	70
8.3.3	Restore process.....	71
9	Zarafa System Statistics Monitor.....	72
9.1	Installation.....	72
9.2	Internal working of the System Statistics Monitor.....	73
10	Mobile push technology.....	73
10.1	Z-push introduction.....	73
10.2	Security.....	74
10.3	Installation.....	74
11	Multicompany configuration.....	75
11.1	Support plugins.....	75
11.2	Configuring the server.....	75
11.2.1	Enabling Multi company.....	76
11.2.2	Configuring login name.....	76
11.2.3	Configuring store name.....	76
11.2.4	Configuring the LDAP plugin.....	77
11.2.5	Public stores.....	78
11.3	Managing company spaces.....	78
11.4	Managing users and groups.....	79
11.5	Quota levels.....	79
12	Multiserver configuration.....	80
12.1	Introduction.....	80
12.2	Prepare/Setup the LDAP server for multiserver setup.....	81

12.3	Configuring the servers.....	82
12.4	Setup mail delivery.....	83
13	Appendix A; backup container details.....	84
14	Appendix B; Pre-6.2x upgrade strategies.....	85
14.1	Database upgrades from 4.1 or 4.2.....	85
14.2	Upgrades from 5.0 to 5.1x and up.....	86
14.3	Important changes since 4.x and 5.x.....	86

1 Zarafa technical overview

The Zarafa solution fully integrates with default components of a Linux server, like MySQL, PHP, Apache, Openldap and an MTA, like Postfix or Sendmail.

Zarafa support a wide range of clients, like Outlook, webaccess, IMAP/POP3 clients and most mobile devices. In this chapter the different components of Zarafa server will be discussed.

1.1 Technical structure

The system consists of the following parts:

- Zarafa server*

The server process accepts connections for all clients through SOAP, and stores the data in an SQL database.

- Zarafa licensed*

The licensed process checks which features will be available dependent on the license chosen for the Community, Standard, Professional or Enterprise edition.

- Zarafa client*

The Zarafa client provides access to Outlook through an interface known as MAPI. The connections with the server are handled by SOAP.

- Zarafa-dagent and spooler*

The tools which serve the email communication with the outside world. The dagent delivers mail from the Mail Transport Agent (MTA) to a Zarafa user. The spooler sends mail waiting in the outgoingqueue.

- Zarafa-admin*

The commandline administration tool is used to manage users, user information and groups.

- Zarafa-gateway*

Optional service to provide POP3 and IMAP access to Zarafa users.

- Zarafa-monitor*

Monitor service which monitors user stores for quota exceeds.

- Zarafa-ical*

Optional service that provides ical and CALDAV support. CALDAV is recommended due to speed and less data transfer.

- Zarafa-backup*

A brick-level backup tool to create simple backups of stores. This part is only available in Zarafa commercial editions

- Zarafa-restore*

A brick-level restore tool to restore items and stores. This part is only available in Zarafa commercial editions

- Zarafa-ssm*

Optional service to provide monitoring and statistical data to Zarafa support. This part is only available for Professional and Enterprise editions

- Apache*

Serves web pages of the webaccess to the users browser.

- PHP*

The webaccess is written in this programming language.

- PHP-MAPI extension*

Module for PHP to enable use of the MAPI layer. Through this module, MAPI functions are made accessible for PHP developers. This effectively means that MAPI web clients can be written. The

webaccess is such a client.

1.2 Server

The Zarafaserver stores all the data in a MySQL database. It also maintains the connections with the clients like Outlook and Webaccess. Therefore, clients never directly access the database, and need to ask the Zarafa server for the information. The Zarafa server will decide if the client may access the requested data.

Users who want to use Zarafa need to be registered in the system first. Registration is independent of the users present in the Operating System. The MTA (Message Transfer Agent) must be reconfigured to deliver the email for a specific email address to a specific Zarafa user.

The server listen on a TCP port (default port 236) to accept all connections from the different clients. The server also connects to the SQL database to retrieve or store data of the users.

For Outlook client connections, at least one connection will always be open to send notifications from the server to the client. Notifications are used to notify clients of events. A new email is such an event. As soon as the email is in the user's Inbox, the client will be notified to update the contents of the folder. Another example is when an item is added or changed in a public folder. All the connected clients will be notified of the change, therefore making all the clients view the same contents of the same folder.

There is no distinction in local network connections or connections over the Internet.

1.2.1 Configuration

The server can be configured through a configuration file. The file will overwrite default settings. Common changeable settings are the MySQL database credentials, which port to listen for incoming connections and the logging details like the loglevel and logfile location..

1.2.2 Logging

Log messages of the server can be configured. The following options need to be altered in the configuration file:

- log_method

How to log the messages. 'file' sends the messages to a file. On Linux systems, 'syslog' sends the messages to the default maillog through syslog.

- log_file

When the *log_method* is set to 'file', this is the variable that defines the name of file. The server needs write access to the directory and file.

- log_level

Increase the level of messages that will be logged. Level 6 is the highest level.

- log_timestamp

1 or 0; This will enable or disable a timestamp, when using a file as the log method.

Logging of other services than zarafa-server are configured in a same manner as the server.

1.3 Client

A client connects over a network to the server. Since these calls are done with SOAP, they can pass through any webserver or proxy transparently. This makes it possible for the clients to bypass firewalls, since traffic over port 80 is mostly available.

Since the connections can pass through proxies, it is possible to have the Zarafa server as a separate server in the local network, without a direct connection to the internet.

When your webserver is correctly configured for HTTPS connections, it is even possible to use HTTPS encrypted connections to connect to the Zarafa server. This is important when connecting over the Internet. No one will be able to read the traffic between the client and the server.

At the moment, there are two clients which works via the MAPI protocol: a Microsoft Outlook client, which works with Outlook version 2000 and up. The second client is the webaccess. Even without access to your own computer with Outlook, email, calender and other information can be checked. The user can even access public stores and other user's (shared) folders from Webaccess.

1.4 Delivery agent

The delivery agent (zarafa-dagent) is a program which delivers incoming email to Zarafa users. The input of the dagent is a normal RFC2822 internet email. After the dagent has a connection to the server, it converts the email and saves the item in the standard Inbox of the user.

It depends on your setup how the dagent needs to be called. Dagent often will be directly called by the MTA. When the MTA does not deliver the mail itself, it needs to be configured by using programs procmail or maildrop.

In any case, the dagent needs to be aware of the user to deliver the email to. The only mandatory parameter is the username to deliver to. This can be configured in the MTA. For an example using postfix, see chapter 2.13.1

By default, the dagent reads the input from stdin. It is also possible to read/deliver an email directly from a file, by using the -f parameter:

```
zarafa-dagent -f test.eml username
```

1.5 Spooler

The Zarafa-spooler sends email from the global outgoing queue to a SMTP server, which sends the email to the correct address.

When an email message is sent from Outlook or Webaccess, the message is placed in the Outbox folder, and a submit message is sent to the Zarafa server. The server notifies the Zarafa spooler to send the email to the SMTP server. The spooler will now start to convert the message to a normal email message. When the conversion is complete, a connection to the supplied SMTP server is created, and the email is sent to the SMTP server.

The spooler will send the email, and after the mail is sent, will move the mail automatically to the user's Sent Items folder.

If at any time an error was found, the user will be notified with an 'Undeliverable' message. The message will contain an error description on which error was found. Often, the user can retry to send the message.

Note: both external and internal emails will be send via the MTA.

1.5.1 Configuration

The Spooler is configured the same as the server. Options in the spooler configuration file are the name or ip-address of the SMTP server, where to find the Zarafa server, and logging options.

All the options are:

- smtp_server

The name or IP-address of the SMTP server, which will send the email to the destination. This server may also be given as an argument when starting the spooler.

- server_socket

The UNIX socket of the Zarafa server. The spooler will use this socket to create a connection to the server. This value should be the same as set in the server configuration file.

The default value is file:///var/run/zarafa.

- [logging]

The spooler has the same configuration options as the server to configure logging options.

1.6 Database

The Zarafa server uses a SQL database to store all the data. The database contains a number of tables, which will be discussed subsequently. Every Item (email message, appointment, task, note or contact) consists of a set of properties. All properties of an item are stored in the *properties* table. Which properties belong to which item is stored in the *hierarchy* table. This table also contains which folder belongs to a store, which folders are a subfolder, and so on. Attachments of emails and pictures of contacts are stored in the *lob* (large objects) table.

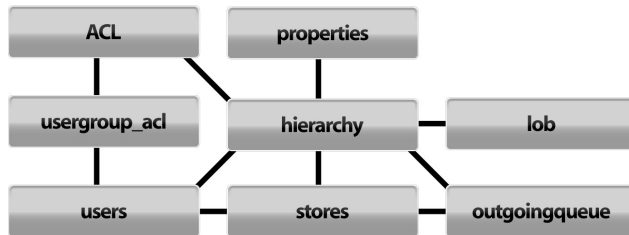


Figure 1: table structure

Users and groups are stored in the *users* table. Relations between users and groups are stored in the *usergroup_acl* table. Access rights are stored in the *acl* table, which has a relation to the *hierarchy* table, so rights can be set on any folder.

When an email is sent, the server will create a link to the message in the *outgoingqueue* table. The spooler receives a notification when the contents of this table is changed, and immediately starts to send the email message. When the spooler is finished, the link will be removed from the *outgoingqueue* and the email of the user will be moved to the *sent-items* folder.

1.6.1 Attachments

Attachments can be placed in the database (default). They will be written in blocks to the *lob* table. This is the default, and can be preferred for small installations.

Attachments can also be stored outside the database, and in a directory structure on the disk. This can decrease strain on the MySQL database, and is preferred for larger installations.

If the attachments are placed outside of the database, a `mysqldump` is not enough anymore for a disaster recovery. The attachments need to be backed up separately.

1.7 Secure HTTP

The Zarafa client has the possibility to create an SSL connection to the server. When you create a profile in Outlook, it is possible to set the connection to use HTTPS. All connections over the network will now be encrypted before sending, making eavesdropping virtually impossible.

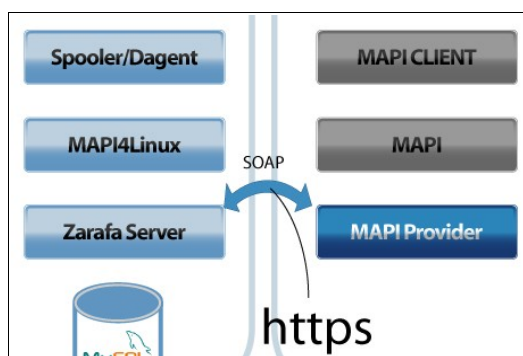


Figure 2: HTTPS connection

The server must be configured to also accept SSL connections. By default this is disabled, because it requires the creation of SSL certificates. When the server certificate is created, SSL connections can be directly accepted from client. As an extra option, Linux Zarafa clients (like the dagent and spooler) can also connect over the SSL connection of the server and authenticate using a private key. If this key is accepted by the server, the client can login without using a password. This results in these tools, like the spooler, monitor, dagent and gateway, to login using an encrypted connection from any other machine to the server. This way, you can securely run all Zarafa parts on different servers, adding some load-balancing.

1.8 Caching

All the data of the Zarafa server is stored in a SQL database. Every time an item is created, like an email, there will be several SQL insert commands. When a client connects, a lot of SQL select commands are executed. Retrieving data from the database happens all the time, and is therefore the most demanding task of the server.

To minimise the read access to the database, Zarafa will cache useful data that is often accessed. The cache is a layer between the database and the Zarafa server. Often accessed data can be returned directly to the client, without asking the database.

After starting the Zarafa server, and the first clients connect, data will be added to the cache. When a client restarts, most of the data is already loaded into memory of the server, and the client will display the correct contents much faster.

More information on how to optimise cache sizes is described in chapter 7.

1.9 Protocols and Connections

All applications which connect to the Zarafa server, use the MAPI standard. This library calls in turn the server if information from the database is needed. Because of this, all MAPI compliant programs should work with Zarafa. Even webaccess uses MAPI through the PHP-MAPI extension.

The Zarafa client is a MAPI provider. It connects to the server and uses the SOAP protocol to communicate with the server. The client is the translator for MAPI calls to server calls, and returns the result of this server call.

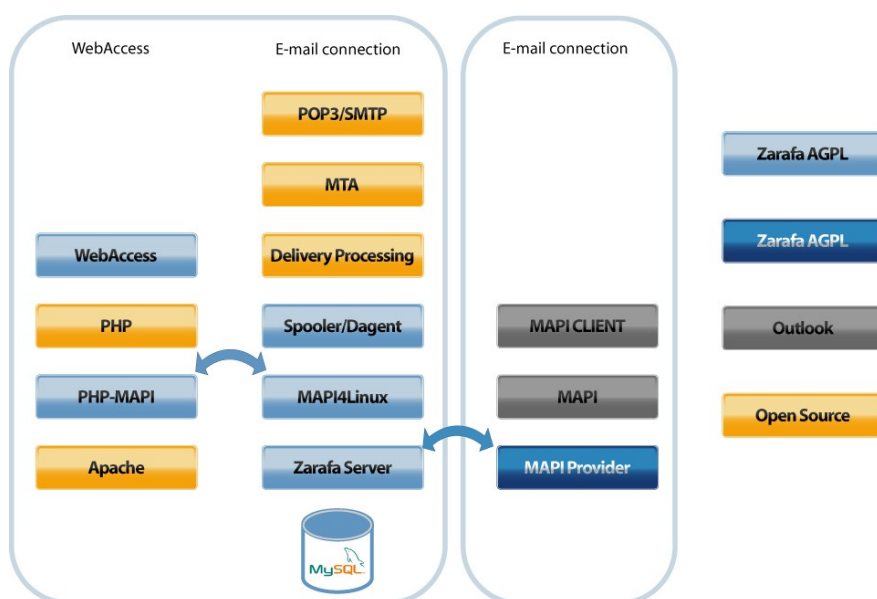


Figure 3: Protocols

1.9.1 SOAP

SOAP is an abbreviation of Simple Object Access Protocol. It is a protocol to exchange data and make Remote Procedure Calls between applications over a network or Internet for that matter.

SOAP is based on XML and HTTP 1.1. Because of these standards it is possible to go through proxies or webservers transparently, making the server available over any connection. Users can make connections over the Internet, without opening or forwarding an extra port in the firewall.

1.9.2 HTTP Proxy

The transmitted data between the client and server is compressed XML, wrapped in a HTTP header. Because of this HTTP header, you can forward the connections using a proxy or webserver.

The following lines are an example of how Apache2 can be configured to forward incoming connections on port 80 to the Zarafa server on port 236. When the Apache server also accepts HTTPS connections, the proxied connections can also be encrypted. The proxy and proxy_html modules of Apache need to be loaded.

```
<IfModule mod_proxy.c>
    ProxyPass /zarafa http://127.0.0.1:236/
    ProxyPassReverse /zarafa http://127.0.0.1:236/
</IfModule>
```

This means that URLs that begin with /zarafa will be forwarded to HTTP connections on localhost on port 236, where normally the Zarafa server listens for incoming connections. These lines can be placed globally, or within a VirtualHost declaration.

2 Installing Zarafa on a Linux distribution

2.1 System requirements

2.1.1 Software requirements

The Zarafa packages are built for specific distributions. For each distribution, the default set of distribution packages are used for dependencies. The table below shows the distribution, and which version of the software you must have installed in order to be able to install and run Zarafa.

Distribution	Required software	RPM/DEB package
OpenSuSE 10.0	PHP 5.0 MySQL 4.1 OpenLDAP 2.2.23	php5-5.0.4 mysql-client-4.1.13 openldap2-client-2.2.23
SLES 10 OpenSuSE 10.1	PHP 5.1 MySQL 5.0 OpenLDAP 2.3.19	php5-5.1.2 mysql-client-5.0.18 openldap2-client-2.3.19
RHEL 4	PHP 4.3 MySQL 4.1 OpenLDAP 2.2.13	php-4.3.9 mysql-4.1.7 openldap-2.2.13
RHEL 5	PHP 5.1 MySQL 5.0 OpenLDAP 2.3.27	php-5.1.6 mysql-5.0.22 openldap-2.3.27
Debian 4.0 (Etch)	PHP 5.2 MySQL 5.0 OpenLDAP 2.1.30	libapache-mod-php5 / libapache2-mod-php5 mysql-client-5.0 libldap2
Debian 5.0 (Lenny)	PHP 5.2 MySQL 5.0 OpenLDAP 2.4.11	libapache-mod-php5 / libapache2-mod-php5 mysql-client-5.0 libldap2.4-2
Ubuntu 6.06 LTS	PHP 5.1 MySQL 5.0 OpenLDAP 2.1.30	libapache2-mod-php5 mysql-client-5.0 libldap2
Ubuntu 8.04 LTS	PHP 5.2 MySQL 5.0 OpenLDAP 2.4.7	libapache2-mod-php5 mysql-client-5.0 libldap-2.4-2

Table 1: Software Requirements

The following distributions have support for the AMD64/Intel EM64T architecture:

- RHEL 4 and 5
- SLES 10
- Debian 4.0 (Etch) and 5.0 (Lenny)
- Ubuntu 6.06 LTS
- Ubuntu 8.04 LTS

Please use the x86_64 packages if you installed a 64bit OS on your server. It is recommended to use a 64bits OS for Zarafa. The following distributions have support for the Intel IA64 architecture:

- RHEL 4 and 5
- SLES 10
- Debian 4.0 (Etch) and 5.0 (Lenny)

Other software that must have been previously installed on the target system:

- Apache** or a webserver that supports PHP;
Zarafa is tested with Apache 2.0 and 2.2.
- MySQL** 4.1 or greater server;
Version 4.0 or lower will not work correctly.
Zarafa is tested with MySQL 4.1 and 5.0.
- SMTP** server of choice;
Zarafa is tested with Postfix, Sendmail and Qmail
- Optional LDAP server as userbase;
Zarafa is tested with OpenLDAP, eDirectory and Microsoft Active Directory

2.1.2 Hardware recommendations

Size of all mailboxes	CPU (*)	Memory	Harddisk	Raid level
< 5 Gb	P4 HT - 3 Ghz	512 Mb	SATA, SCSI	RAID1
> 5 - < 10 Gb	Xeon - 3 Ghz	1 Gb	SCSI	RAID1
> 10 - < 20 Gb	Dual Xeon - 3 Ghz	2 Gb	SCSI	RAID1
> 20 - < 50 Gb	Dual Xeon - 3 Ghz	4 Gb	SCSI	RAID1
> 50 Gb - < 100Gb	Dual Xeon - 3,33Ghz MP 64bits	> 4 Gb	SCSI	RAID10
> 100GB - < 250 Gb	Dual Xeon - 3,33Ghz MP 64bits	8 Gb	SCSI	RAID10
> 250 Gb	Dual Xeon - 3,33Ghz MP 64bits	16 Gb	SCSI	RAID10

Table 2: Hardware Recommendations

*: Assuming a non-heavy loaded server with other applications.

2.2 Editions of Zarafa

2.2.1 Community edition

The proven Zarafa groupware solution is now also available as an open source community edition licensed under the [Affero GPLv3](#). This edition includes:

- Ajax based webaccess
- Mobile webaccess
- IMAP / POP3 gateway
- CALDAV / iCal gateway
- Z-Push - ActiveSync compatibility (licensed under GPLv2)

Additionally you can use this edition with the closed source Zarafa Outlook clients up to 3 Outlook users.

Note: To have Outlook support in the community edition, you need to run the zarafa-licensed daemon. A license is not needed however.

2.2.2 Zarafa commercial editions

Standard, Professional and Enterprise editions do need a commercial license.

The non-community editions will be further described in this document. Whenever a feature is not available for the community edition, it will be mentioned.

2.2.3 User Licenses

Zarafa works with per-named-user user licenses. A base license is a license for a fixed number of users, which can be extended by adding extra Client Access Licenses. Client Access Licenses simply extend the existing base license; ie. having a base license for 10 users and a CAL for 10 users, is functionally equivalent to having a 20-user base license.

Licenses are based on named users; i.e. you can add 10 named users in a Zarafa system with 10 licensed users. However, there are also users which do not directly add to this user count; these are so-called 'non-active' users. These are users that exist like any other user, but cannot log in to Zarafa directly. An example of a non-active user is an 'info' user. This is a user in the respect that it can receive email and has all the standard folders, but the user 'info' will never log in. Instead, other users will open the 'info' store as a shared store and retrieve email from there.

Each license automatically allows you to have an extra amount of non-active users. Every license has a minimum of 20. After that, the amount of non-active users is 50% of your 'active' user count.

Examples:

License: 10 users
Active users: 10
Non-Active users: 20

License: 400 users
Active users: 400
Non-Active users: 200

If do not use all your active user accounts, you can use them instead as non-active accounts. For example, in the above scenario, you could also create 500 non-active users, and 100 active users.

Please note that a user is deemed 'active' or 'non-active' when the user back end declares this at the time of creation. You cannot convert active users to non-active users or vice-versa. You must delete the user and re-create the user as a different type.

In LDAP setups you can control the non-active flag of users through the `ldap_nonactive_attribute` configuration directive. When using the DB back end, you can specify the non-active flag with the '-n' option of `zarafa-admin` when creating users. The Unix user plugin uses the unix-shell of the user to determine if the store should be a non-active store.

2.2.4 Trial license

When you use a trial license, you have a period of time to test our software. The software will be fully functional as a normal installation would. You can continue using your current database when you install a valid license.

2.3 Installation

The Zarafa packages come with an `install.sh` script to install Zarafa. You may also manually install Zarafa. The `install.sh` script will automatically execute the actions described under *Manual installation* below. Thus, it will:

- Check package dependencies;
- Install packages;
- Check MySQL database access;
- Ask for configuration options.

You can start the installation script by typing

```
$ sh ./install.sh
```

After running `install.sh`, your server is ready to start. Proceed with creating stores as will be explained by `install.sh`.

You can also start the `install.sh` script with the `-config` parameter. This will skip the installation of the software, and ask the configuration options only.

```
$ sh ./install.sh -config
```

If you have an older version of Zarafa already installed on your server, please read the *Upgrading* section of this document. The `install.sh` script is not usable in that case.

The `install.sh` script configures the server to use the DB user plugin. If you wish to use another user

base, please read the *Configuration* chapter for information on how to configure the server.

2.3.1 Manual installation

Please use the packages for your distribution. See the distribution list in the *Software requirements* section above. For other distributions, choose a distribution that is the most similar.

Note: Do not mix packages of different distributions! Choose one distribution, and use only those packages. If you do not honor this rule, errors will occur!

Use the following commands to install the Zarafa-server on RPM based distributions:

```
$ rpm -Uvh libvmime-[version].[arch].rpm
$ rpm -Uvh libical-[version].[arch].rpm
$ rpm -Uvh zarafa-[version].[arch].rpm
$ rpm -Uvh zarafa-webaccess-[version].noarch.rpm
$ rpm -Uvh zarafa-webaccess-muc-[version].noarch.rpm
$ rpm -Uvh zarafa-licensed-[version].[arch].rpm
```

Replace [version] with the current version shipped, and [arch] with your distribution's default architecture (either i386, i586, x86_64 or ia64).

Use the following commands to install the Zarafa-server on DEB based distributions:

```
$ dpkg -i libvmime0_[version]_[arch].deb
$ dpkg -i libical0_[version]_[arch].deb
$ dpkg -i zarafa_[version]_[arch].deb
$ dpkg -i zarafa-webaccess_[version]_all.deb
$ dpkg -i zarafa-webaccess-muc_[version]_all.deb
$ dpkg -i zarafa-licensed_[version]_[arch].deb
```

Replace [version] with the current version shipped, and [arch] with your distribution's default architecture (either i386, i586, x86_64 or ia64).

Note: In the community edition the package `zarafa-licensed` is not needed, though in order to have Outlook support in the community edition, you need to run the `zarafa-licensed` daemon.

Note: The Multi User Calendar inside the package `'zarafa-webaccess-muc'` is a feature not available in the community edition. A valid license is needed.

The `webaccess` files will be installed in the default webserver directory of your distribution. The list of locations is as follows:

- SuSE: /srv/www/htdocs
- RedHat Enterprise: /var/www/html
- Debian and Ubuntu: /var/www

A directory called `'webaccess'` will be created at the given prefix. If you wish to install the website files in another directory, use:

```
$ rpm -Uvh zarafa-webaccess-[version].noarch.rpm \
--relocate /srv/www/htdocs=/usr/local/htdocs
```

With DEB based distributions it is simplest to just move the `webaccess` directory to a new location. You can place the files anywhere on your server, as long as your webserver is able to read these files.

The Zarafa package will, on every distribution, configure the PHP-extension when PHP is already

installed. The MAPI-extension will be automatically loaded.

2.3.2 Notes on Debian and Ubuntu

To install the correct dependencies for Zarafa, you can use apt-get or similar.

For MySQL, use:

```
$ apt-get install mysql-server-5.0 libmysqlclient15off
```

For Apache with the needed PHP support, use:

```
$ apt-get install apache2-mpm-prefork libapache2-mod-php5
```

If the Zarafa packages fail to install because of dependencies, please use the following command to install these dependencies:

```
$ apt-get -f install
```

If you install Apache with PHP support after you have installed Zarafa, please use the following command to automatically configure PHP:

```
$ dpkg-reconfigure zarafa
```

2.3.3 Notes on x86_64 architectures

The shared libraries which provide the user plugin are installed in `/usr/lib64/zarafa`, instead of the `/usr/lib/zarafa` location. You need to adjust this path in the server `.cfg` configuration file. Set the `plugin_path` to `/usr/lib64/zarafa`, so the server can find the user plugin files.

2.4 Configuration

2.4.1 Initialising the server

The server's storage is kept in a MySQL database. The Zarafa server will create a database and its tables the first time you start the server. Because of this, the `mysql` user you set in the configuration file must have all rights, including the right to create a new database. If you wish, you can revoke the right to create database commands after the database has been created by the server.

If you create a new `mysql` user for zarafa, be sure to give the user enough permissions to connect from localhost to this database. Use this user information in your Zarafa server `.cfg`. (See below *Configuring the server* on the use of server `.cfg`). For example:

```
GRANT ALL PRIVILEGES ON zarafa.* TO  
  'zarafa'@'localhost' IDENTIFIED BY 'secret';
```

If the Zarafa server connects over the network to the MySQL database, you will need to replace `localhost` with the IP-address of the server which Zarafa will run on.

2.4.2 Setting the server license key

Note: In the community edition of Zarafa you don't need a license key.

The server expects a directory containing a file called `base` to read the license key from. The default directory is `/etc/zarafa/license`. To install your license key, use the following command:

```
$ mkdir -p /etc/zarafa/license
$ echo 'license key' > /etc/zarafa/license/base
```

'license key' should of course be replaced with a valid license key.

Note: The license key must be entered in all capitals.

If you also have an extra CAL (Client Access License), you can add this license key to the server as follows:

```
$ echo 'CAL key' > /etc/zarafa/license/cal
```

If you have more than one CAL, please install one CAL per file in the license directory. The filename of the cal is of no importance.

It is not allowed to have subfolders in the license directory. At the moment `Zarafa-licensed` will quit without warning if it finds a subfolder.

2.4.3 Configuring the server

You can create a configuration file to overwrite the default values of the server. After the installation is done, you can find an example in the following directory:

```
/usr/share/zarafa/example.server.cfg
```

The default values and explanations of the options for the configuration file can be found with:

```
$ man zarafa-server.cfg
```

All settings for Zarafa can be accessed through a configuration file. If a line is not present, the default setting will be assumed.

Although the changes you need to make to the configuration file are minimal, you do have to change the `mysql_password` option. Set the correct mysql password here so Zarafa can connect to the database. Also, check the other mysql options to ensure that a connection to the database can be made without error.

At this point you are already able to start Zarafa.

However, the most important setting in Zarafa is the `user_plugin` setting. This setting determines which user backend should be used when listing users and groups. The possibilities are:

- db
- unix
- ldap

The `db` user plugin uses the main MySQL database to store user and group information. You can use the `zarafa-admin` tool to manage your users. This is described below in the *Creating users* section.

The `unix` user plugin uses information from `/etc/passwd`, `/etc/group` and `/etc/shadow`. A range of users and groups will be automatically created in Zarafa. Some information that cannot be stored in these unix files must be set using the `zarafa-admin` tool. This is described in chapter 2.8.1.

The *ldap* uses an LDAP server for users, groups and companies. You can manage your users, groups and companies and all their information in an LDAP or ADS server.

The *db* user plugin will be installed by default and does not require any further configurations. If you choose this as your user plugin, you can continue with the *Starting the services* and *Creating users* sections.

2.4.4 Configuring services logging methods

Normally, all services log to their respective file located in `/var/log/zarafa`. This directory is created when the packages are installed. When this directory is not present, or not writable under the running user, services will not be able to open their log file and will print the log messages to the console.

You can also change the log method in the configuration file to `syslog`. The `syslog` facility will then be used, so the messages will be printed to the default mail log.

2.4.5 Configuring the UNIX plugin

The `unix` plugin is used on a server which has all its user information setup in the `/etc/passwd` file. Group information will be read from `/etc/group`. Passwords are checked against `/etc/shadow`, so the `zarafa-server` process must have read access to this file. Zarafa is normally run as root, so usually that is not a problem.

Since the `unix` files do not contain enough information for Zarafa, there are some properties of a user that will be stored in the database. These properties are the email address, overriding quota settings, and administrator settings. You need to use the `zarafa-admin` tool to update these user properties. All other user properties are done using the normal `unix` tools.

In the configuration file you need to set the `uid` range of users you want to have in the Zarafa server. The same goes for the groups. Users with a special shell, default `/bin/false`, will be created as non-login store. These users cannot login, but the stores can be opened by other users. An administrator user should setup the correct access rights.

An overview of all the `unix` user plugin settings, type:

```
$ man zarafa-unix.cfg
```

2.4.6 Configuring the LDAP plugin

The LDAP plugin is used for coupling an LDAP server with Zarafa. This way, all users, groups and membership information can be retrieved 'live' from an LDAP server. The main features of the LDAP plugin are:

- Coupling with any LDAP-compliant server
- Totally configurable for use with OpenLDAP and Active Directory

The defaults for OpenLDAP and for Active Directory can be found in `/usr/share/zarafa/example-config`. These settings need to be placed in `/etc/zarafa/ldap.cfg` in order for the LDAP plugin to be able to find the settings.

The details for the `ldap` settings can be found with:

```
$ man zarafa-ldap.cfg
```

Please check all options of this configuration. It is very likely you will need to alter settings so that they point to the right DN string and `objectClass` variables.

Note: When using the LDAP plugin, you cannot create or delete either users or groups with the

zarafa-admin tool. You can only create a user from within the LDAP database. You can use any standard LDAP user manager for this task.

2.5 Starting the services

There are 6 services you can run:

- zarafa-server, the server process
- zarafa-spooler, sends outgoing email to an SMTP server
- zarafa-monitor, checks for quota limits
- zarafa-gateway, provides POP3 and IMAP access
- zarafa-ical, provides iCal and CALDAV access for clients that use this type of calendar
- zarafa-licensed, needed when using any closed source zarafa module with zarafa-server
- zarafa-dagent, runs as a service when using local mail transfer protocol (LMTP, see chapter 2.15)

The zarafa-server and zarafa-spooler processes are mandatory to run Zarafa. The zarafa-monitor, zarafa-gateway, and zarafa-ical services are optional. To start a service, type:

```
$ /etc/init.d/zarafa-[servicename] start
```

Replace [servicename] with the service you want to start. To start the zarafa-server, type:

```
$ /etc/init.d/zarafa-server start
```

This script will start the server. The `init.d` scripts can start, stop and restart the services.

If you cannot use the `init.d` script, then you need to start the server manually. You might also want to explicitly tell the zarafa server where the configuration file is. This is done with the `-c` switch:

```
$ /usr/bin/zarafa-server -c /etc/zarafa/server.cfg
```

The zarafa-server will daemonise, so you will almost immediately return to the prompt. Use `-F` to start it in the foreground. The `-F` switch can also be used for programs like daemontools that monitor services.

2.6 Stopping the services

To stop a service, type:

```
$ /etc/init.d/zarafa-[servicename] stop
```

Most services will stop almost immediately. The zarafa-spooler may take up to 10 seconds to stop. The zarafa-server may take up to 60 seconds to stop.

2.7 Reloading service configuration

Some options can be modified and reloaded by the service in a live environment. The options that can be reloaded are described in the manual page of the service configuration file. Example: for the zarafa-server, type the following command to get the configuration manual page:

```
$ man zarafa-server.cfg
```

In the 'reloading' chapter are all the options that can be reloaded for that service. To make a service reload the configuration file, type:

```
$ /etc/init.d/zarafa-[servicename] reload
```

2.8 Stores

Once the server has been correctly started, you can create stores. There are two type of stores: Private and public stores. There can only be one public store. It can be created with the following command:

```
$ /usr/bin/zarafa-admin -s
```

The public store is the folder every user can always open. After installation and configuration of the server you have to add a public store first before you can create a private store. If the `install.sh` script started the server, the public store should be already created, and this command will fail.

If Zarafa is configured for multicompany, a public store will be automatically created per company.

If Zarafa is configured for multiserver, a public store can be set in LDAP as a server attribute (singlecompany) or a company attribute (multicompany). See chapter 12.2 for more information.

2.8.1 Updating stores and user information

Note: Updating users and groups is only supported when using the DB plugin. When using the unix plugin you may only edit a limited set of properties. When using the ldap plugin, all information is kept in sync with the information present in LDAP tree.

The same `zarafa-admin` tool can be used to update the stores and user information. Use the following command to update:

```
$ /usr/bin/zarafa-admin -u <user name> [-U <new user name>] \  
    [-p <new password>] [-e <email>] \  
    [-f <full name>] [-a <0|1>]
```

All the changes are optional. You may, for example, only update the password for an existing user, leaving the other user information the same as it was.

2.8.2 Deleting stores

To delete a user from the server, use the following command:

```
$ /usr/bin/zarafa-admin -d <user name>
```

The user will be deleted from the database. The store of the user will be placed in the public store, in a 'Deleted Stores' folder. This folder is only available to administrative users of Zarafa. Administrators can delete the deleted stores completely by removing the corresponding folder from the 'Deleted stores' folder.

2.9 Users

Users are by default created in the Zarafa database by using the `zarafa-admin` tool. If you have an LDAP server with all your users already setup, you can use this as your user source by using the ldap plugin, set in the server configuration file.

2.9.1 Creating users

Note: Creating and deleting users and groups is only supported when using the DB plugin. When using the UNIX or LDAP plugin, users and groups are kept in sync with the users present in these databases.

To create a private store for a user, use the following command:

```
$ /usr/bin/zarafa-admin -c <user name> -p <password> \  
-e <email> -f <full name> -a <administrator>
```

The fields between <> should be filled in as follows:

- User name: The name of the user. With this name the user will log on to the store.
- Password: The password in plain text. The password will be stored encrypted in the database.
- Email: The email address of the user. Often this is <user name>@<email domain>.
- Full name: The full name of the user. Because the full name will contain space characters, and maybe other non-alphanumeric characters, it is best that you use single or double quotes around the name.
- Administrator: This value should be 0 or 1. When a user is administrator, the user will be allowed to open all Zarafa stores of any user.

All fields except the email address are case sensitive.

The password can also be set using the -P switch. The password is then not given at the command prompt, but asked for by the `zarafa-admin` tool. The password is not echoed on the screen, and needs to be typed twice for verification.

2.9.2 Non-active users

A non-active user cannot login, but email can be delivered, and the store can be opened by users with correct rights.

To create a non-active user, use the following command:

```
$ zarafa-admin -c <user name> -e <email> -f <full name> -n 1
```

In the Unix Plugin, users with a special shell (default `/bin/false`) are non-active users.

Note: In Zarafa version 6 and later, when you mark an account from active to a non-active user, the user (and store) will be deleted and created again.

Note: It is not possible to convert a non active user to an active user

2.10 Groups

The server supports groups. Users can belong to any number of groups. Every user always belongs to the special group *Everyone*. Defining security settings on folders and items are the same for both users and groups.

For example, the group *Everyone* has read access to the Inbox of Peter. At this point, every user may read the email in Peter's Inbox, because all users are a member of the group *Everyone*.

When a new Zarafa user is created, only the free/busy information is open for read access for the group *Everyone* by default.

2.10.1 Creating groups using zarafa-admin

By using the `zarafa-admin` tool, groups can be created and users can be added or removed from groups. In the following example, a user *john* is created, a group *administration* is created, and the user *john* is added to the group *administration*.

```
zarafa-admin -c john -p secret -f "John Doe" -e "j.doe@domain.com"  
zarafa-admin -g administration  
zarafa-admin -b john -i administration
```

Using the options `-l` or `-L`, a list of users or groups can be listed from the server.

When listing users, everyone will always be in the group *Everyone*.

2.11 Other admin commands

The `zarafa-admin` tool can also be used to list users and groups, retrieve user details and set user-specific quota levels. Please refer to the manual page of `zarafa-admin` to see all the commands and options available.

```
$ man zarafa-admin
```

2.12 Sending email through the server

To send email from the server, you have to run the `zarafa-spooler` program:

```
$ /etc/init.d/zarafa-spooler start  
or  
$ /usr/bin/zarafa-spooler -c /etc/zarafa/spooler.cfg
```

The `zarafa-spooler` will daemonise, so you will almost immediately return to the prompt. Use `-F` to start it in the foreground. The spooler will keep running and sending mail when it is needed. You can also specify an SMTP server to send all mail through an MTA other than the one on localhost. For example:

```
$ zarafa-spooler smtp.internet-provider.com
```

To configure the `zarafa-spooler` program, use a configuration file and pass the location of the file to the program using the `-c` switch.

The default values of the `zarafa-spooler` configuration can be found in the manual page:

```
$ man zarafa-spooler.cfg
```

2.13 Delivery of email through the server

In order to deliver an email into a user's inbox, `zarafa-dagent` is executed. The `zarafa-dagent` will read an SMTP email message from the standard input or a file. A parameter to the `zarafa-dagent` is needed to select the store to deliver the incoming message to. A few examples are given for different email servers.

2.13.1 Postfix

Modify the `/etc/postfix/main.cf` file, and set the `mailbox_transport` to:

```
mailbox_transport = zarafa:
zarafa_destination_recipient_limit = 1
```

Add the following line to the `/etc/postfix/master.cf` file:

```
zarafa unix - n n - 10 pipe
flags= user=vmail argv=/usr/bin/zarafa-dagent ${user}
```

The `vmail` user should be added as a local user to the Linux system: `adduser vmail`. Change the following option in the `/etc/zarafa/server.cfg` file:

```
local_admin_users = root vmail
```

2.13.2 Qmail

Place in your `$HOME/.qmail` file a line containing:

```
| /usr/bin/zarafa-dagent -q [user name]
```

The `[user name]` tag needs to be replaced with the name of the user to correctly deliver the email. The `-q` switch is to notify the `zarafa-dagent` program to return qmail style error codes. When an email can temporarily not be delivered, it will stay in the qmail queue, which will try to deliver the email again later.

2.13.3 Procmail

Create a file called `.procmailrc` in the users home directory. If you have virtual users, you can create a global `/etc/procmailrc` file. Please read the *email delivery* section in the *Advanced configurations* chapter on more procmail details and setups.

A simple procmailrc file should contain the following lines:

```
:0 w
| /usr/bin/zarafa-dagent [user name]
EXITCODE=$?
```

The `w` after the `:0` tells procmail to wait for the `zarafa-dagent` to finish. You can also type the `c` flag to make a 'carbon copy' of the email. This way, your email will be delivered to Zarafa as well as your normal Mailbox or Maildir. Again, `[user name]` needs to be replaced with the name of the user.

If you have a system wide `/etc/procmailrc` file and unix login names are the same as Zarafa user names, you can replace `[user name]` with `$LOGNAME`.

2.14 Attachments outside the MySQL database

As of version 6.0 is it possible to save the attachments outside the database. This has some advantages, but it also has disadvantages. The default method is to save the attachments inside the database, like the older Zarafa versions, so for a normal upgrade you do not need to change anything.

For first time installations, you must select the storage method before you start the server for the first time. You cannot (easily) switch the attachment storage method.

For upgrades, you need to run a script that copies the attachments from the database to the new file storage. You can find this script in /usr/share/zarafa, called db-convert-attachments-to-files. This script is run as follows:

```
$ db-convert-attachments-to-files <mysqluser> <mysqlpass> <mysqldb>
  <destination path> [delete]
```

It is only possible to convert from database storage to file storage. You cannot switch back. The [delete] switch is optional. If this parameter is given, the attachments are also removed from the database. Keep in mind that for the conversion you will need twice the storage of the attachments on your harddisk. You can check the amount of storage needed with the following mysql command:

```
mysql> show table status;
```

Check the data_length column for the lob table. This contains the number of bytes needed for the attachment storage.

To select this new storage method, change the attachment_storage option in the server.cfg file. When changing this option, the zarafa-server needs to be started with the --ignore-attachment-storage-conflict parameter once.

Advantages of attachments outside the database are:

- MySQL does not save the large data blobs in between the other outlook data in the database. This improves read and write access to the database
- Attachments will not cause cache purges of MySQL

Disadvantages of attachments outside the database are:

- A mysql dump of the database is not enough for a full recovery
- Remote storage of attachments requires a new system, like an NFS or Samba mounted directory

It is very important to note the email backup strategy must be changed when choosing to store the attachments outside the database. There is no change for the zarafa-backup tool. This will still work as usual.

2.15 Local Mail Transfer Protocol (LMTP)

Since version 6.20, Zarafa supports delivery through Local Mail Transfer Protocol (LMTP). The benefit of LMTP is that the SMTP server only needs to pass the email to Zarafa once for every local Zarafa user.

This way, the email only needs to be processed once for many users, which saves resources. LMTP must be used to fully enable the single instance attachment storage feature of zarafa for newly received emails. See chapter 2.26 for more details about single instance attachment storage.

To enable LMTP it is necessary to modify the main.cf file located in etc/postfix/.

```
mailbox_transport = lmtp:localhost:2003
```

or if you use virtual addresses which go to zarafa users:

```
virtual_transport = lmtp:localhost:2003
```

or if using transport maps:

```
transport_maps= lmtp:127.0.0.1:2003
```

Also make sure to have the following line in the list of services in the master .cf file located in etc/postfix/:

```
lmtp      unix  -      -      n      -      -      lmtp
```

To launch the LMTP service as a daemon use the command:

```
$ /etc/init.d/zarafa-dagent start
```

After this command LMTP will be running as a daemon and listening on port 2003. Or, if not wanting to run dagent as a daemon:

```
$ zarafa-dagent -l
```

Note: It can be wise to enable logging of the dagent when running in LMTP mode for monitoring purposes.

2.16 Quota and quota monitoring

Users can collect a lot of email, while disk space can be limited. With this feature quotas can be set server-wide and quotas for users specifically. Zarafa quota system consists of three levels: warn, soft and hard quota. When one of the levels will be reached, the user receives an email with the quota sizes and which quota is reached.

The quota settings can be configured serverwide in the server.cfg or per user via the user plugin.

When a user reaches the warning quota level, the user will receive an email with a warning and quota information. As the user reaches the soft quota limit, the user will not be able to sent new mail until the size of the store is reduced. On the moment that the user reaches the hard quota limit, new mail cannot be delivered to the user anymore, until the size of the store is reduced.

2.16.1 Setup server-wide quota

The server-wide quota can be configured in the configuration file of the server:

```
quota_warn = 100
quota_soft = 150
quota_hard = 200
```

The values are all in Mb. These values will be honoured for all users present in the server. When the values are set to 0, the quota level is 'unlimited', and will never be reached.

2.16.2 Setup quota for a single user

By using the zarafa-admin tool, the user quota can be set for a specific user. Example:

Set the quota of the user John with the settings: Warning level to 80 Mb, soft level to 90 Mb and hard level to 100 Mb.

```
zarafa-admin -u john --qd 0 --qw 80 --qs 90 --qh 100
```

Note: Set user quota with zarafa-admin does not work with LDAP. With LDAP the properties are stored in the LDAP server per user. See: LDAP documentation for more information.

2.16.3 Monitoring for quota exceedings

The zarafa-monitor program checks every hour for users who are exceeding a quota level. The zarafa-monitor program sends an email to a user when the warning or soft quota limit is exceeded. Global quota settings can be set in the server configuration. User specific levels can be set via `zarafa-admin` when using the db or unix plugin, or by editing the LDAP values.

To start the zarafa-monitor, use:

```
$ /etc/init.d/zarafa-monitor start
or
$ zarafa-monitor -c /etc/zarafa/monitor.cfg
```

The zarafa-monitor will daemonise, so you will almost immediately return to the prompt. Use `-F` to start it in the foreground. You can read about the configuration options in the manual page:

```
$ man zarafa-monitor.cfg
```

2.16.4 Quota warning templates

When working with the zarafa-monitor, it is possible to modify the contents of the email which will be sent out when a user or company exceeds its quota. For each quota level a separate quota template can be specified, these can be configured with the following options:

- userquota_warning_template
- userquota_soft_template
- userquota_hard_template
- companyquota_warning_template

By default the templates are stored in `/etc/zarafa/quotamail/`, in each of these templates certain variables are provided which will be substituted for the real value before the email is sent out:

- `_${ZARAFQA_QUOTA_NAME}` - The name of the user or company who exceeded his quota
- `_${ZARAFQA_QUOTA_COMPANY}` - The name of the company to which the user belongs
- `_${ZARAFQA_QUOTA_STORE_SIZE}` - When a user exceeds his quota, this variable contains the total size of the user's store. When a company exceeds its quota this variable contains the total size of all stores, including the public store within the company space.
- `_${ZARAFQA_QUOTA_WARN_SIZE}` - The quota warning limit for the user or company.
- `_${ZARAFQA_QUOTA_SOFT_SIZE}` - The quota soft limit for the user or company.
- `_${ZARAFQA_QUOTA_HARD_SIZE}` - The quota hard limit for the user or company.

Note: Variables containing a size always include the size unit (B,KB,MB,GB) as part of the variable.

2.17 POP3 and IMAP gateway

If you have email client programs other than Outlook and still wish to use Zarafa, you may want to activate the optional gateway program. The zarafa-gateway program provides POP3 and IMAP support.

To start the zarafa-gateway, use:

```
$ /etc/init.d/zarafa-gateway start
or
$ zarafa-gateway -c /etc/zarafa/gateway.cfg
```

The zarafa-gateway will daemonise, Use -F to start it in the foreground. The configuration options are described in the manual page:

```
$ man zarafa-gateway.cfg
```

How to setup SSL support to use encrypted connections with zarafa-gateway is described in chapter 4.

2.18 CALDAV and iCal gateway

Other calendar clients that can **communicate with** the CALDAV or iCal protocol, the iCal gateway can be used to enable these clients to read the calendar from Zarafa.

To start the zarafa-ical, use:

```
$ /etc/init.d/zarafa-ical start
or
$ zarafa-ical -c /etc/zarafa/ical.cfg
```

The zarafa-ical will daemonise, Use -F to start it in the foreground. The configuration options are described in the manual page:

```
$ man zarafa-ical.cfg
```

How to setup SSL support to use encrypted connections with zarafa-ical is described in chapter 3.

2.19 Configuring webaccess

Normally, the zarafa package will configure PHP on the system automatically. In most situations you can skip the next chapter and continue with chapter 2.19.2.

2.19.1 Configuring PHP

PHP is needed in order to use Webaccess. The PHP-extension is installed in the default directory of distribution:

- SUSE: `/usr/lib/php/extensions/`
- RedHat Enterprise Linux: `/usr/lib/php4/` or `/usr/lib/php5/`
- Debian: `/usr/lib/php5/20060613/`
- Ubuntu: `/usr/lib/php5/20060613/`

If you have a different directory for your PHP-extensions, move the `map*.so*` files to this location, eg:

```
$ mv /usr/lib/php/extensions/mapi.so* \  
    /usr/local/lib/php/extensions
```

To find your PHP-extensions location, use the following command:

```
$ php-config --extension-dir
```

After the PHP-extension is in the correct directory, add it to your `php.ini` configuration file. Add the following line to your `php.ini` if it does not already exist:

```
extension = mapi.so
```

Common places for your `php.ini` file are:

```
/etc/php.ini  
/etc/httpd/php.ini  
/etc/php5/apache2/php.ini
```

You can check with the `phpinfo()` function whether the module will be loaded correctly. Search for the 'MAPI' part to check for the module. The `phpinfo` can also be viewed by running `php -i` on the command line if you have the `php cli` installed.

2.19.2 Configuring the webserver

To correctly load the recently added `mapi.so` extension, you need to restart your webserver. The following example shows how to restart Apache2:

```
$ /etc/init.d/apache2 restart
```

or

```
$ /etc/init.d/httpd restart
```

The website files are by default installed in the `webaccess/` directory. Make sure you can see the webclient's login page by browsing to the correct url:

```
http://<ip-address server>/webaccess/
```

If the login page is not shown, you need to configure your webserver to let it access the correct directory. The following example shows a configuration for Apache2:

```
Alias /webaccess /srv/www/htdocs/webaccess/  
<Directory /srv/www/htdocs/webaccess/>  
    AllowOverride None  
    Order allow,deny  
    Allow from all  
</Directory>
```

Make sure you type the correct directory holding the PHP `webaccess` files. Use the following command to let `apache2` reread its configuration:

```
$ /etc/init.d/apache2 reload
```

You should now be able to see the website. If it still does not show up, please see the *Troubleshooting* chapter for more information.

2.20 Optimising Zarafa and MySQL

The default memory settings of both Zarafa and MySQL are quite low, so it runs on any server. To improve speed, increase these memory settings for both servers. The sizes set are dependent on how

many users are active on the server and on how many other memory-intensive programs are run. Make sure the system is not going to swap memory.

This caches actual data requested by clients. Increase this to 25% of your RAM memory:

```
cache_cell_size = 512000000
```

For MySQL, you need to increase the InnoDB memory settings in the my.cnf file:

```
innodb_buffer_pool_size=256M
innodb_additional_mem_pool_size=8M
innodb_log_file_size=100M
innodb_log_buffer_size=32M
```

Please set the `innodb_buffer_pool_size` to 25% of the RAM memory.

The maximum size of the InnoDBs `ibdata-file(s)` can be configured via `innodb_data_file_path`. Please make sure to keep each data-file in an organisable size in terms of fragmentation, backup and disk-size.

For more information about tuning MySQL and Zarafa, see chapter 7.2.

2.21 Upgrading

Before upgrading to a new version of Zarafa, it is advisable to make a backup of the database and the configuration files.

First stop the running services, so you know the database is not in use anymore:

```
$ /etc/init.d/zarafa-spooler stop
$ /etc/init.d/zarafa-server stop
$ /etc/init.d/zarafa-licensed stop
```

And the optional services too, if you started them:

```
$ /etc/init.d/zarafa-dagent stop
$ /etc/init.d/zarafa-gateway stop
$ /etc/init.d/zarafa-ical stop
$ /etc/init.d/zarafa-monitor stop
```

Now you can upgrade the Zarafa packages, just like installing them:

```
$ rpm -Uvh libvmime-[version].[arch].rpm
$ rpm -Uvh libical-[version].[arch].rpm
$ rpm -Uvh zarafa-[version].[arch].rpm
$ rpm -Uvh zarafa-webaccess-[version].noarch.rpm
$ rpm -Uvh zarafa-licensed-[version].[arch].rpm
```

or for Debian based installations:

```
$ dpkg -i libvmime0_[version]_[arch].deb
$ dpkg -i libical0_[version]_[arch].deb
$ dpkg -i zarafa_[version]_[arch].deb
$ dpkg -i zarafa-webaccess_[version]_all.deb
$ dpkg -i zarafa-licensed_[version]_[arch].deb
```

Replace the `[version]` and `[arch]` tags with the correct values for your system.

Note: In the community edition the package `zarafa-licensed` is not needed, though in order to have Outlook support in the community edition, you need to run the `zarafa-licensed` daemon.

After the new packages are installed, you can check the example configuration files found in `/usr/share/zarafa` for new features. This is discussed in chapter 14.3. There are also perl and sql scripts which upgrade the database format for the new version. Some scripts **must** be executed to make Zarafa runnable, while other scripts are recommended for speed increases. The *Upgrading the database* section explains the upgrade path.

2.21.1 Finalising the upgrade

After you have checked the new configuration options, you can start the services again:

```
$ /etc/init.d/zarafa-server start
$ /etc/init.d/zarafa-spooler start
$ /etc/init.d/zarafa-licensed start
```

You can also start the optional services again:

```
$ /etc/init.d/zarafa-dagent start
$ /etc/init.d/zarafa-gateway start
$ /etc/init.d/zarafa-ical start
$ /etc/init.d/zarafa-monitor start
```

Note: In the community edition the package `zarafa-licensed` is not needed, though in order to have Outlook support in the community edition, you need to run the `zarafa-licensed` daemon.

Since the upgrade changed the php extension of Zarafa, you will need to restart the webserver as well:

```
$ /etc/init.d/apache2 restart
or
$ /etc/init.d/httpd restart
```

2.22 Troubleshooting

2.22.1 Server processes

Make sure at least MySQL 4.1 is installed. The server will only run with this version of the database server or a more recent version.

If errors when loading libraries occur or connecting to MySQL fails, the errors are printed in the log. Always check if the service was started correctly.

When an invalid configuration option is present in a configuration file, the service will not start. The wrong options will be printed on the console.

2.22.2 Webaccess

To correctly see the webaccess, you will need the following PHP-extensions:

- gettext
- session
- iconv

Some distributions deliver support for these extension by default through the PHP package. For SUSE

distributions, these modules are provided by separate RPMs, eg:

```
php5-gettext-5.2.8-37.4.x86_64.rpm  
php5-iconv-5.2.8-37.4.x86_64.rpm
```

Versions may differ for newer versions of SUSE.

For Red Hat Enterprise Linux and Debian distributions, these modules are provided by the normal php package which you already installed because of dependencies.

If experiencing problems with sending attachments, make sure the webserver is able to create files under the webaccess/tmp directory.

If you are directly logged off when you login to the webaccess, make sure PHP is configured with:

```
register_globals = off
```

If you are using a distribution in combination with SELinux, you may get an error message while logging in when using the Webaccess. The default message suggests that your password is wrong or the Zarafa server is not running. When SELinux is enabled, it is blocking your connection from the webserver to the Zarafa server. You can solve this by allowing Apache to make network connections:

```
$ setsebool httpd_can_network_connect=1
```

or by disabling SELinux altogether:

```
$ setenforce permissive
```

When you choose to disable SELinux, you will also want to edit `/etc/sysconfig/selinux` to disable it for after reboots too.

Hint: SELinux information can be found here: <http://fedora.redhat.com/docs/selinux-faq/>

2.23 Advanced configurations

While Zarafa can already be used, there are some optional features to check out. Some provide added security and some can be interesting in some circumstances. This chapter describes these features.

2.23.1 Running as non-root user

Normally the Zarafa services are run as root. Since version 5.0 there is the option to change the user the service runs as, and still start the services as root. However, there are several things to do before the services can correctly run as a non-root user.

If the `log_method` is set to `file`, make sure this directory and file is writable by the user or group the service will be running as. When a logrotate happens, by sending the service the HUP signal, a new file is created, which will be owned by the user the service is running under.

The service should still be started as root since it will create a pid file under the system location `/var/run`, and will open the network sockets which most likely have a number under 1024, which may only be opened as root.

The following example shows how to configure the zarafa-server to run as user `zarafa` and group `zarafa`:

```

$ addgroup --system zarafa
$ adduser --system --home /dev/null --no-create-home \
    --ingroup zarafa \
    --disabled-password --gecos 'Zarafa services' \
    --shell /bin/false zarafa
$ mkdir /var/log/zarafa
$ chown zarafa.zarafa /var/log/zarafa

```

Note: The `addgroup` and `adduser` tools may have different syntax on different distributions.

Edit the `run_as_user` and `run_as_group` options in the `server.cfg` file, and set them both to `zarafa`. Make sure the `local_admin_users` option still contains `root` as an administrative user, so that you can still use the `zarafa-admin` tool. Otherwise you will have to use `su` or `sudo` each time you start the `zarafa-admin` tool.

2.23.2 SSL connections and certificates

The Zarafa server is capable of directly accepting encrypted SSL connections. You may already have this feature when you have setup your HTTPS Apache server to proxy these connections to Zarafa as described in the *Configuring Apache for Outlook* section. However, having native SSL connections to the server has some advantages. Zarafa linux clients can login using their SSL certificate, and thus login from any location to the server.

This section will describe how to setup certificates to add native SSL connections to Zarafa.

First, we will create the directory to contain the certificate and setup the permissions, since it contains our private key.

```

$ mkdir /etc/zarafa/ssl
$ chmod 700 /etc/zarafa/ssl

```

If you run Zarafa as another user, as described in the *Running as non-root user* section, do not forget to `chown` the directory as well.

Now we are ready to create a Certificate Authority. This CA will be used to create the server certificate and sign it. We provide a `ssl-certificates.sh` script in the `/usr/share/zarafa` directory, which uses the `openssl` command and the `CA.pl` script from OpenSSL. Depending on your distribution this script can be installed in different directories. The script will try to find it on its own. If it is not found, either OpenSSL is not installed, or the script is in an unknown location, and you can provide the location of the script yourself. Normally, the `ssl-certificates.sh` script can be run without problems.

```

$ cd /etc/zarafa/ssl
$ sh /usr/share/zarafa/ssl-certificates.sh server

```

The parameter `server` is added, so the name of the new certificate will be called `server.pem`. When the CA is not found in the default `./demoCA` directory, it needs to be created. By pressing `enter`, the creation of the new CA is started.

Enter a password (*passphrase*) when asked for. This is the password you will use later to sign certificate requests. Then enter your certificate information. Do not leave the Common Name field blank, otherwise the creation will fail.

Now that we have a CA, we can create self-signed certificates. The `ssl-certificates.sh` script will automatically continue with this step. Enter a password for the request, and enter the certificate details. Some details need to be different from those you typed when creating the CA. At least the field *Organizational Unit Name* needs to be different. The challenge password at the end may be left empty.

This step created a Certificate Request, that needs to be signed by the CA that was created in the first step of the script. Type the password of the CA again when asked for. The details of the certificate will be shown, and asked for acceptance. Accept the certificate.

As the last step, you will be asked if you want the public key of this certificate. Since you created the server certificate you do not need the public key of this certificate.

Now that you have the CA certificate and the server certificate, you can enable SSL in the `server.cfg` file, which is normally disabled. The port 237 is set for SSL connections. You can change this port number if you wish.

```
server_ssl_enabled = yes
server_ssl_port = 237
```

The CA certificate must be set in the `server_ssl_ca_file` setting. The server certificate and password must be set in the `server_ssl_cert_file` and `server_ssl_cert_pass` options.

```
server_ssl_ca_file = /etc/zarafa/ssl/demoCA/cacert.pem
server_ssl_key_file = /etc/zarafa/ssl/server.pem
server_ssl_key_pass = <password>
```

Restart the `zarafa-server` process, and you are now able to directly connect with the SSL port. Create a new Outlook profile, and mark the SSL connection option. Set the port to 237. Your connection to the server is now encrypted.

2.23.3 Linux Zarafa clients from another server

Using the SSL connection with certificates it becomes not only possible to encrypt the connection, but Linux services will also be able to login using their SSL certificate.

Repeat the certificate creation script to create certificates for client programs like the `zarafa-spooler`, `zarafa-monitor`, `zarafa-gateway` and `zarafa-dagent`. You can create one certificate for all these programs, or you can create a certificate for each program separately. These clients can then login on the SSL connections with their certificate as authentication.

```
$ sh /usr/share/zarafa/ssl-certificates.sh client
```

Again, when entering the certificate details, at least make the *Organizational Unit Name* different from the other certificates. Also, do not forget to fill in the *Common Name* field.

When asked for the creation of the public key, enter `y` and press enter. You now have a new certificate called `client.pem` and a public key called `client-public.pem`. As an example, the configuration options needed to edit on the `spooler.cfg` file are as follows:

```
server_socket = https://name-or-ip-address:237/zarafa
sslkey_file = /etc/zarafa/ssl/client.pem
sslkey_pass = ssl-client-password
```

Enter the correct name or IP-address in the `server_socket` option. If you use another port number for the SSL connections on the server, enter the right port number as well. Replace the password with the password you entered to create the certificate.

Copy the `client-public.pem` file to the server location:

```
$ mkdir /etc/zarafa/sslkeys
$ mv client-public.pem /etc/zarafa/sslkeys
```

Now the client knows the private key, and the server knows the public key. The client can login with this key to the server from anywhere on the network or internet.

Note: Be careful with the `client.pem` file. Anybody who has this private key can login to the Zarafa

server and will be the internal SYSTEM user, who can do anything without restriction.

2.23.4 Email delivery options

The zarafa-dagent has a lot of options when it comes to controlling the delivery location. However, in the configuration of your SMTP server, it is likely to have only one type of delivery command, without being able to set any options for special email accounts or when specific headers are present in the email.

This is where tools like procmail or maildrop step in. This document uses procmail as an example. You can also find a lot of examples and documentation on the internet about procmail. You may also find the procmail manual pages a good thing read:

```
$ man procmail
$ man procmailrc
$ man procmailex
```

The procmail manual is about procmail itself. The procmailrc manual is about the config file layout and options, and the procmailex is about regular expressions which you can use in the rules to match headers.

For each zarafa-dagent option we will give you an example with a procmail rule of how to use the option.

When your Zarafa users are present on the system in /etc/passwd, and have a home directory, you can create a file called .procmailrc in the users homedirectory. If you have virtual users, you can create a global file called procmailrc in the /etc directory.

A normal procmailrc file contains atleast the following rule:

```
:0 w
| /usr/bin/zarafa-dagent $USER
EXITCODE=$?
```

The line starting with :0 tells procmail a new rule is starting. It also has the w flag, which tells procmail that when this rule is executed, it should wait for the delivery to finish before continuing. This is because we need the exit result of the zarafa-dagent to pass. This is what happens in the last line.

The rule has no condition, so it will always be executed. The \$USER variable will be replaced with the unix name of the user. When the zarafa-dagent is done, its exit code will be placed in the EXITCODE variable of procmail. This value will be returned to the SMTP server when procmail is done.

2.23.5 Junk e-mail

A filter for spam mail can be easily created, using the -j option of the zarafa-dagent. When this option is set the email will be delivered in the Junk mail folder instead of the Inbox. Let us set up a rule to find a spam header:

```
:0 w
* ^X-Spam-Status: Yes
| /usr/bin/zarafa-dagent -j $USER
EXITCODE=$?
```

When the spamfilter adds the header X-Spam-Status: Yes . . . procmail will match this rule, and start the zarafa-dagent using the -j option.

The zarafa-dagent can also deliver mail into a specific folder or into a folder of the public store. For example, this can be used when email is sent to a specific email address.

```

:0 w
* ^TO_(info)
| /usr/bin/zarafa-dagent --public 'Public folders\info mail' -C info
EXITCODE=$?

```

When an email is sent to the info address, it will be placed in the public store. The `Public folders` string is language dependent, so replace it if your language is other than English. The `-C` flag tells the `zarafa-dagent` it should create the folder if it could not find it.

2.23.6 Temporary deliver failures

If the `dagent` fails to delivery the email to the server, it can return different exit codes to notify the SMTP server what happened during the delivery process. One exit code tells the SMTP server that delivery has failed, and will always fail, no matter how many times is retried. Examples of such a failure is when a user does not exist in `zarafa` or the user is over quota, and may no longer receive new email. The SMTP server will act on this error by bouncing the error to the sender of the mail, notifying the user that the mail cannot be delivered at the given destination.

When the `dagent` fails because the `zarafa` server cannot be contacted, it will return a special error code, telling the SMTP server to retry to deliver the mail in a while. The SMTP server will keep the mail in it's queue, instead of immediately bouncing the mail back to the sender. The mail will be retried after sufficient time has passed.

The `dagent` has a special option to use `qmail` style error codes, in case you use `qmail` with the default `.qmail` delivery files.

2.23.7 Out of Office message

`Zarafa` contains simple support for sending out of office messages. A user can enable this auto-reply service for the incoming email. An email will be sent to every incoming message containing the given subject and message. The service can be enabled in `Webaccess` on the 'settings' page.

2.23.8 'Send as' Permissions

Since `Zarafa 6.20` two kinds of delegation are available:

- Send on Behalf permissions:** if a user grants the appropriate permission to another user, the latter can send items 'on behalf of' the other user. In this case an email or meeting request will be sent with the following `From:` field: `<delegate> on behalf of <user>`.
- Send As permissions:** if the system administrator gives the rights to user B to 'send as' user A, the receiver of an email will not see that user B sent an email. The receiver will only see user A in the `From:` field

Before version 6.20, a user could use the send on behalf of permissions. This meant letting a user send an email *on behalf of* another user from inside the inbox of the other user. You could always see who sent the email. For example: Pete enters the inbox of 'info' and sends an email as the non-active user 'info', you would see: "pete@example.com on behalf of info@example.com".

Since 6.20 it is possible to send emails as other users without the *on behalf of* part. Due to security reasons the new 'send as' permission is only configurable by the administrator on the server side. This setting can always be overruled by the user itself and the old 'on behalf of' permission can still be set by the user. See the client manual how to set the user based 'on behalf of' delegation and/or overruling of the admin based 'send as' delegation.

Send As permissions

The use of delegation via `zarafa-admin` is only applicable with the DB or UNIX plugin. For setting up LDAP or Active Directory, see [chapter 5](#)

- Adding a delegate: `zarafa-admin -u <delegate> --add-sendas <user>`

Add user to the list of the delegate you are updating as a 'send as' user. The delegate can now send mails as the updated users' name, unless the updated user set the delegate as a user based delegate. This option is only valid with the -u update action.

•Deleting a delegate: `zarafa-admin -u <delegate> --del-sendas <user>`

Remove user from the list of the delegate you are updating as a 'send as' user. This option is only valid with the -u update action.

•View list of users of the delegate: `zarafa-admin --list-sendas <delegate>`

List all users who are in the list of the delegate.

Note: All previous settings concerning delegation have to be reconfigured when upgrading to 6.2x or later. Unfortunately a reset of these settings is needed in order to have this additional feature available.

2.24 Client Auto updater tool

The zarafa-server allows clients to update themselves to the latest version of the client update available with the zarafa-server.

Note: The autoupdater will be only available in the Professional and Enterprise edition.

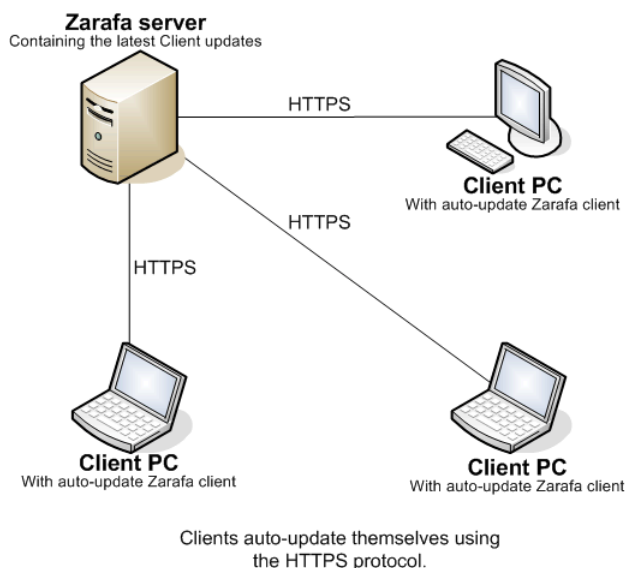


Figure 4: Auto-update structure.

Restrictions:

- The auto-update mechanism does not support the ability to downgrade the client to a certain version, it will update the zarafa client to the highest version of the client update available.
- The autoupdater is not available for windows 2000 environments or earlier at this moment.
- The Autoupdate mechanism doesn't support the ability to downgrade the client.

2.24.1 Server side configuration

The autoupdater can be enabled by setting the following setting to 'yes' in the server .cfg of the zarafa-server:

```
client_update_enabled = yes
```

When a zarafa-server is upgraded, it will copy the latest updated client installer to the path which is specified in the server configuration file server .cfg. As shown below.

```
client_update_path = /var/lib/zarafa/client
```

The updates at the client update folder have a naming convention. Zarafa-server will work only with those updates that adhere to this file naming convention. The updates need to have the following file name format:

```
zarafaclient-<major version>.<minor version>.<update number>-<build number>.msi
```

For example zarafaclient-6.20.2-12345.msi is a valid name of an update.

Based on this naming convention the autoupdate mechanism finds out if an update of the client is available. If a suitable version is available for a client, zarafa-server will send the update to the client machine to update itself with the latest client version.

Clients communicate with the server via the HTTP protocol. The zarafa-server acts as an HTTP server on port 236. Clients send a request to download a virtual file, this file is the most current version of the client available on the server.

The client communicates with the server using an encrypted message format. This prevents misuse of the zarafa-updater mechanism by a person with any malicious intent.

Important note: If the default profile is set to use encryption via port 237, the root CA certificate needs to be installed on the desktop used.

2.24.2 Client side configuration

The zarafa client auto-update mechanism consists of an application to start the auto-update process known as zarafa launch updater application (ZarafaLaunchUpdater.exe) and a windows service known as zarafa updater service (ZarafaUpdaterService.exe).

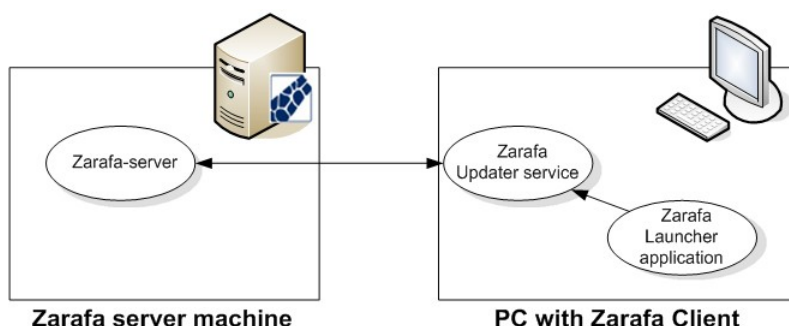


Figure 5: Autoupdate structure.

2.24.2.1 Zarafa Launch Updater Application

The zarafa launch updater application will be launched at windows startup. The command to run the application is placed in the registry here:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run.

This application will find out client's current version from the following registry key.

HKEY_LOCAL_MACHINE\Software\Zarafa\Client\Version

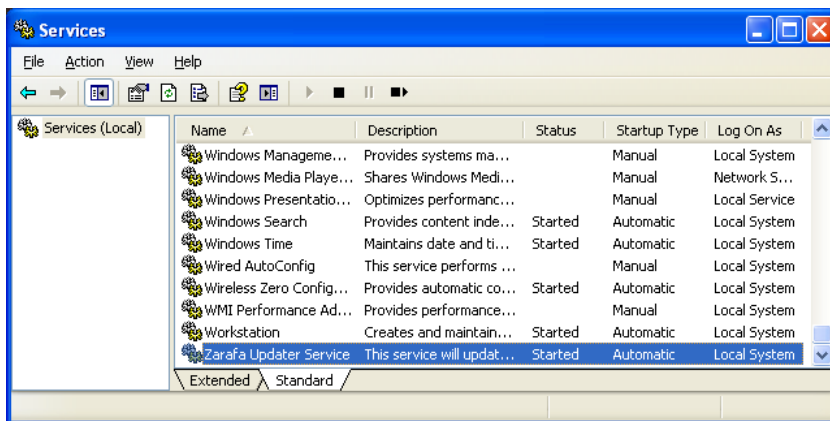
This is a new registry key introduced for updater mechanism, it will contain the version of zarafa client installed on the machine.

The zarafa launch updater application will then find out zarafa server details by reading default outlook profile from the registry. It will then send the current version of the client and the zarafa server details to zarafa updater service on a pipe and then it will terminate.

The zarafa launch updater application communicates with the zarafa updater service using an encrypted message format. This prevents misuse of zarafa updater mechanism by a person with any malicious intent.

2.24.2.2 Zarafa Updater Service

The zarafa updater service runs as a local system account. Therefore, it has all the needed privileges to install the zarafa client on the desktop.



Screenshot 1: Services.

The zarafa updater service will wait on a pipe for zarafa launch updater application to send it the current version of the client and the zarafa-server to connect to. If there is a suitable update, the service downloads it at the following location- c:\windows\temp\zarafaclient.msi. The zarafa updater service launches this update for installation in a silent mode.

Although, the entire update process is silent, logs can be generated for troubleshooting. To generate logs the Updater service startup parameter needs to have -v option, similarly for the launcher the registry key needs to have the variable '-v'. The Updater service log will be written in the "All users\Application data\" directory and the Launch updater log will be written in the "<user>\Application data\" directory.

Note: The client will successfully look for updates only if the default outlook profile is configured to work with a zarafa server, and if updates are available at the zarafa-server which is configured in default outlook profile.

2.25 Soft Deletes

If a user deletes emails, calendar items or complete folders, there are by default moved to the 'Deleted Items' folder.

When the items are removed from the 'Deleted Items', the items still will not be removed from the Database. Rather, they are marked as deleted, so the user does not see the items.

This makes restoring of items quick and easy from Outlook. To restore deleted items, choose Extra in

the Outlook menu, and click on 'Restore deleted items'. Items are grouped by the folder they were deleted from. Most items will appear from the 'Deleted Items' folder, since they were really deleted from that location.

Even when a user uses the <Shift>+<delete> action on items, directly removing items from the folder without placing them in the 'Deleted Items' folder, they are not directly removed from the database, but actually only marked as deleted, so the user will not see the items any more.

2.25.1 Configuration

Soft Deletes always remain in the database, until they are purged. When an item will be purged is dependent of a configuration value. This option defines how long a deleted item will remain in the database:

```
softdelete_lifetime = 30
```

In this example, the value is set to 30. This means that deleted items will be purged from the database 30 days after they were deleted. When this option is set to 0, the items will never be removed from the database.

The purge can also be done ad hoc via zarafa-admin:

```
$ zarafa-admin --purge-softdelete <days>
```

2.26 Single Instance Attachment Storage

Zarafa 6.30 implements single instance attachment store to avoid redundant storage of attachments on the server's database and increase efficiency. When using single attachment store, if a message is sent to multiple recipients within the same server, only one copy of every attachment is saved in the database. This mechanism, thus, minimizes the disk space requirements and remarkably enhances delivery efficiency of messages with attachments sent to large distribution lists.

Let's assume the following situation:

user A belongs to a Zarafa server; he sends a message with 10 MB of attachments to 30 users that reside on the same server. In a normal situation 30 copies of the files would be saved on the database, leading to an inefficient usage of the storage space (310 MB of data). With single instance attachment store, only one copy of each attachment is saved on the database (only 10 MB of data in this example) and all the 30 users can access the attachment through a reference pointer.

2.26.1 Single Instance and LMTP

In order to make maximum usage of Single instance, use it in combination with LMTP and make use of virtual addresses in Postfix.

With the abovementioned setup, externally received email with an attachment sent to multiple internal users will be processed efficiently by saving the attachment only once.

The usage of 'virtual_transport' in Postfix will deliver only one email with a list of the internal users to the daemon instead of one email per internal user. Without virtual transport option, Single Instance can not know that the attachment is similar in the email item(s).

Note for multi-company environment:

Single instance attachments are accessible between companies as well (even when the companies cannot view each other), the handling of single storage will be transparent. Thus, considering the example above, if user A sends the message to 30 users of company1 and 50 users of company2, provided that the companies reside on the same server, only one copy of the attachments is saved.

Note for multi server:

Single attachments will be handled per server, when sending an email with attachment to multiple Zarafa users spread over multiple servers, each server will get its own Single instance attachment.

3 Zarafa CALDAV / iCal gateway configuration

The Zarafa iCal gateway enables users to view their Zarafa calendars using clients like Sunbird or Evolution.

The Zarafa iCal gateway acts as layer between calendar clients which use CALDAV (or iCal) with Zarafa. It resides between the clients and MAPI4Linux.

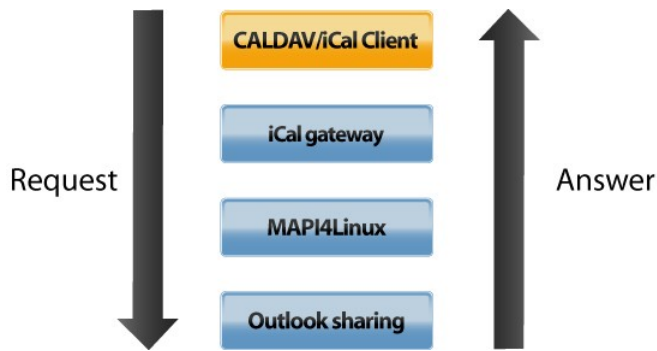


Figure 6: CALDAV / iCal communication

CALDAV and iCal push and retrieve complete calendars. Sunbird and other clients supports retrieval and pushes, though Evolution does only support retrieving complete calendars.

The gateway can be configured using a configuration file the same way as the Zarafa server. This allows the change of settings described in chapter 3.2; 'iCal Gateway Configuration'.

3.1 Security

The Zarafa iCal gateway supports both plain and secure connections using SSL/TLS. This secure connection governs authentication and encryption. Authentication allows the clients to verify the server by its certificate. The encryption makes sure that nobody can read the data and passwords while they are being transported.

For high security, enable secure CALDAV / iCal and disable plain CALDAV / iCal in the configuration.

The gateway does not support secure password authentication (SPA) because the gateway can not retrieve the password from the Zarafa server. SSL/TLS should be used instead of SPA. This makes all data transfer between the gateway and the client encrypted.

3.2 iCal Gateway Configuration

The gateway is configured the same way as the server. Options in the gateway configuration file are the SSL certificates and the Zarafa server to connect to. Other settings are on which ports to listen for incoming connections and how to log errors. If the ports for incoming connections are used by other email server software, disable that software or use other ports.

All the options are:

•**server_bind**

IP address to bind to. 0.0.0.0 for any address.

Default value: 0.0.0.0

•**ical_enable**

Enable plain service with value yes

Default value: yes

•**ical_port**

The plain service will listen on this port for incoming connections.

Default Value: 8080

•**icals_enable**

Enable secure service with value yes

Default value: no

•**icals_port**

The secure service will listen on this port for incoming connections.

Default value: 8443

•**server_socket**

The http address of the Zarafa server.

Default value: <http://localhost:236/zarafa>

•**ssl_private_key_file**

The file that contains the private key used for encrypting the ssl connections. The absolute path to the file should be used.

Default value: /etc/zarafa/privkey.pem

•**ssl_certificate_file**

The file that contains the certificate for the server. The absolute path to the file should be used.

Default value: /etc/zarafa/cert.pem

•**ssl_verify_client**

Enable client certificate verification with value yes

Default value: no

•**ssl_verify_file / ssl_verify_path**

The file or path to the files to verify the clients certificate with. The absolute path should be used for both options.

No default value.

•**[logging]**

The gateway has the same configuration options as the server to configure logging options.

3.2.1 SSL/TLS

The gateway supports SSL/TLS using openssl.

The gateway needs a key for the encryption and a certificate for the authentication. The private key and the certification file can be set in the gateway settings file with `ssl_private_key_file` and `ssl_certificate_file`.

The client can check the servers certificate for validity.

The server can also authenticate the users certificate by verifying the clients certificate using his verification file(s). This can be set with `ssl_verify_client`, `ssl_verify_file`, `ssl_verify_path`.

Certificates can be self-signed or signed by a trusted certifying agency. To generate a RSA key of 2048

bytes:

```
openssl genrsa -out /etc/zarafa/privkey.pem 2048
```

Creating a self-signed test certificate for 3 years:

```
openssl req -new -x509 -key /etc/zarafa/privkey.pem -out /etc/zarafa/cert.pem  
-days 1095
```

3.3 Gateway Starting

To start the Zarafa iCal gateway manually, use:

```
/usr/bin/zarafa-ical -c /etc/ical.cfg
```

The Zarafa will daemonise automatically. Use the -F flag to start in the foreground.

It can also be started and stopped using init scripts:

```
/etc/init.d/zarafa-ical start  
/etc/init.d/zarafa-ical restart  
/etc/init.d/zarafa-ical stop
```

3.4 Calendar access

To access calendar folders on Zarafa the following options are available:

URL	Calendar
http://server:8080/ical/	user's own default calendar via ical (not recommended)
http://server:8080/caldav/	user's own default calendar
http://server:8080/caldav/<other-user>	Other-user's calendar
http://server:8080/caldav/<user>/<calendar>	user's self created calendar in user's (own) store
http://server:8080/caldav/<user>/<calendar>/<subcal>	user's self created subcalendar in a self created calendar
http://server:8080/caldav/public/<calendar>/	Calendar folder in the public folder.
URL For MAC OSX ical client	
http://server:8080/caldav/	User's calendar list
http://server:8080/caldav/<other-user>	Other-users calendar list
Http://server:8080/caldav/public	Public folders list

Table 3: CALDAV and iCal URLs

Note: The <other user> or <user>/<calendar> is only reachable if the correct permissions are available.

Additional information regarding client side setup is described in the Zarafa User manual.

4 Zarafa IMAP & POP3 gateway configuration

The Zarafa IMAP & POP3 gateway enables to view mail stored on the Zarafa server with an IMAP or POP3 client. This enables to view the mail with other applications and operating systems. For example Mozilla Thunderbird on linux or a mobile device with Microsoft Pocket Outlook.

The Zarafa IMAP & POP3 gateway connects to the Zarafa server for IMAP and POP3 clients that connect. Requests from the client are interpreted by the gateway. The Zarafa IMAP & POP3 gateway requests necessary data like emails and folders from the Zarafa server through MAPI4Linux and answers the clients request.

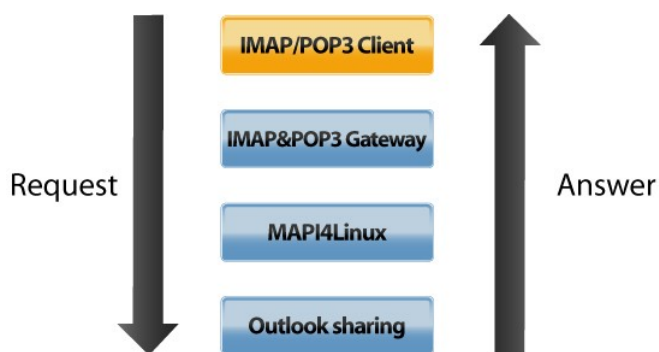


Figure 7: IMAP / POP3 communication

POP3 can only retrieve the mail in the Inbox from the server.

IMAP on the other hand displays all folders that can contain mail, such as Drafts and Deleted Items. All subfolders are shown in same way as Microsoft Office Outlook and Zarafa webaccess.

The Zarafa IMAP & POP3 gateway can be configured with a configurations file in the same way as the Zarafa server. This allows changing what ports to listen on and logging settings. Also security settings can be set in this file as explained later in this Chapter.

4.1 Security

Both IMAP and POP3 communication can be done with a secure connection using SSL/TLS. This is referred to as IMAPS and POP3S. This secure connection covers authentication and encryption. Authentication allows the clients to verify the server by its certificate. The encryption makes sure that nobody can read the email and passwords while they are being transported. It also checks if the data is altered.

For high security enable secure IMAPS / POP3s and disable plain IMAP / POP3 in the configuration.

The gateway does not support secure password authentication (SPA) because the gateway can not retrieve the password from the Zarafa server. SSL/TLS should be used instead of SPA. This makes all data transfer between the gateway and the client encrypted.

4.2 Gateway Configuration

The gateway is configured the same way as the server. Options in the gateway configuration file are the SSL certificates and the Zarafa server to connect to. Other settings are on which ports to listen for incoming connections and how to log errors. The ports for incoming connections can be used by other email server software, disable that software or use other ports.

All the options are:

- server_bind

IP address to bind to. 0.0.0.0 for any address.

Default value: 0.0.0.0

- imap_enable
Enable IMAP service with value yes
Default value: yes
- imap_port
The IMAP service will listen on this port for incoming connections.
Default Value: 143
- imaps_enable
Enable secure IMAP service with value yes
Default value: no
- imaps_port
The secure IMAP service will listen on this port for incoming connections.
Default value: 993
- pop3_enable
Enable POP3 service with value yes
Default value: yes
- pop3_port
The POP3 service will listen on this port for incoming connections.
Default value: 110
- pop3s_enable
Enable secure POP3 service with value yes
Default value: no
- pop3s_port
The secure POP3 service will listen on this port for incoming connections.
Default value: 995
- imap_only_mailfolders
Enable only mailfolders to be shown with value yes
Default value: yes
- server_socket
The http address of the Zarafa server.
Default value: <http://localhost:236/zarafa>
- ssl_private_key_file
The file that contains the private key used for encrypting the ssl connections. The absolute path to the file should be used.
Default value: /etc/zarafa/privkey.pem
- ssl_certificate_file
The file that contains the certificate for the server. The absolute path to the file should be used.
Default value: /etc/zarafa/cert.pem
- ssl_verify_client
Enable client certificate verification with value yes
Default value: no
- ssl_verify_file / ssl_verify_path
The file or path to the files to verify the clients certificate with. The absolute path should be used for both options.
No default value.
- [logging]
The gateway has the same configuration options as the server to configure logging options.

4.2.1 SSL/TLS

The gateway supports SSL/TLS using openssl.

The gateway needs a key for the encryption and a certificate for the authentication. The private key and the certification file can be set in the gateway settings file with `ssl_private_key_file` and `ssl_certificate_file`.

The client can check the servers certificate for validity.

The server can also authenticate the users certificate by verifying the clients certificate using his verification file(s). This can be set with `ssl_verify_client`, `ssl_verify_file`, `ssl_verify_path`.

Certificates can be self-signed or signed by a trusted certifying agency.

To generate a RSA key of 2048 bytes:

```
openssl genrsa -out /etc/zarafa/privkey.pem 2048
```

Creating a self-signed test certificate for 3 years:

```
openssl req -new -x509 -key /etc/zarafa/privkey.pem -out /etc/zarafa/cert.pem  
-days 1095
```

4.3 Gateway Starting

To start the Zarafa IMAP & POP3 gateway manually use:

```
/usr/bin/zarafa-gateway -c /etc/zarafa/gateway.cfg
```

The Zarafa IMAP & POP3 gateway will not daemonise. Use `&` to start it in the background. It can also be started and stopped using init scripts:

```
/etc/init.d/zarafa-gateway start  
/etc/init.d/zarafa-gateway restart  
/etc/init.d/zarafa-gateway stop
```

4.4 Important notes

The addressbooks/contacts from the native client are used. External tools may be able to copy the contacts from Microsoft Office Outlook to the client.

Setting the Out of Office message is not possible with an IMAP or POP3 client.

Rules set in Microsoft Office Outlook do not work using the Zarafa IMAP & POP3 gateway. Some clients can set rules. These rules are not related to the Microsoft Office Outlook rules.

Deleting a mail using IMAP will mark the mail for deletion. This is not shown in Microsoft Office Outlook and Zarafa webaccess. The mail will be deleted when the client expunges the folder. Some clients allow to expunge folders manually and some have settings when to expunge a folder. Other clients expunge the folder automatically when a mail is deleted.

Moving mail to a different folder with IMAP is done by copying the mail to the new folder and mark the originating mail for deletion. As long as the the original mail is not expunged from its folder, the mail will be shown in both folders as stated above.

5 Configuring LDAP or ADS with Zarafa

Starting from version 5.00 of Zarafa, the server backend features a system whereby the administrator of a server can specify an LDAP-based server to retrieve user, group and company information. This means that user management can be simplified for installations and standard LDAP administration tools can be used for user management. Also, using an LDAP server makes it possible to integrate Zarafa into an existing environment.

Various LDAP server systems are supported, and Zarafa will communicate with any standard LDAP protocol version 3 or later server. This means that Zarafa works in combination with industry-standard solutions as Microsoft Active Directory, OpenLDAP and eDirectory.

This document describes loosely how Zarafa uses the LDAP server as a source for user, group and company information, and how administrators can set up their new or existing LDAP servers to work with Zarafa. In most cases, your particular setup will require other options and settings than those described in this document. It is therefore assumed that the reader has a good understanding of how LDAP trees work, and how they are configured in their network.

For more information, please refer to the example configurations and manual pages available on all systems on which Zarafa is installed.

5.1 The Zarafa synchronisation principle

In any Zarafa server, there is a database holding the actual data needed while running Zarafa. Apart from the actual folder and item data, the database also holds information on data access rights, user settings, and user meta-data set for users and groups. A lot of this data refers to a specific user ID. For example, an ACL (Access Control List) for the 'inbox' for user A will be stored in the database as a record in the ACL table. This record holds the actual access rights for the objects, and the user ID to whom the access control entry has been assigned.

The user ID stated above is therefore a reference to a user ID within the Zarafa database. This ID is stored in the 'users' table, along with a reference to the ID of the user in the external user database (in this case, an LDAP server). For example, user A may have user ID 5 in the Zarafa system, and may refer to the item (`dn=cn=user, dc=domain, dc=com`) on the LDAP server.

Keeping a list of users in this way also solves the problem of creating the store for a user; There is no way to trigger a store creation event on the Zarafa server whenever a user is added in the LDAP server. The 'users' table provides a convenient way to track which users are new to the system and therefore require a new store. The same goes for deleting users, as the user store needs to be removed when the user is deleted.

So, the 'users' table in Zarafa is almost exclusively a mapping between the user ID which is used internally in Zarafa, and an external reference to a user in the LDAP database. Naturally, when any new users are added or users are removed from the LDAP server, this table must be kept in-sync with the changes.

There are many ways of keeping the 'users' table synchronised with the LDAP server, but Zarafa has chosen for a 'just-in-time' approach. This means that any time a user is requested from the system, it is first checked in the LDAP server for existence, and then it is checked in the 'users' table for existence. If the user does not exist locally on the Zarafa server, then the user is created on-the-fly, before returning the information to the caller.

This means that for users and administrators, the synchronisation seems to be real-time; never will there be a delay between adding or removing users from the LDAP server and the users showing up in Zarafa.

Because all Zarafa components use the same MAPI interface to connect to the server backend, a situation can't arise with any of the Zarafa tools where the user database is out-of-sync. For example,

delivering an email to a user that was just created will never fail due to the user not existing in the Zarafa users table.

The drawback to this is that any user and any process can trigger the addition and deletion of users; the timing of store creation and deletion is therefore not easy to predict. However, for the vast majority of installations, this is not a problem.

5.1.1 Add/Remove events

The mechanism above creates a situation in which there are six events that can be signaled:

- 1.user creation
- 2.group creation
- 3.company creation
- 4.user deletion
- 5.group deletion
- 6.company deletion

These six events can be coupled to a script (which will be described later) so that system administrators can perform specific actions on their servers with these events. By default, Zarafa will only perform the absolute necessary actions during these events; ie store creation and removal. Any other events can be scripted by the system administrator. This means that by default, no actions are performed during group creation and group deletion.

5.1.2 Group membership

Zarafa synchronises users, groups and companies so that it can assign user ID's to them, but the group membership for users is never stored on the Zarafa server. This means that group membership changes are real-time also, and the Zarafa server will query group membership for a user or a user list for a group directly from the LDAP server. How the mapping between group members and users is done will be discussed later.

5.1.3 LDAP server dependency

Due to the fact that the Zarafa 'users' database doesn't actually hold the user or group information, but only a reference to the LDAP server, the Zarafa server cannot function without a running and accessible LDAP server. If the LDAP server goes down while Zarafa is running, Zarafa tools will not be able to perform any actions, as almost all server-side actions require some kind of interaction with the LDAP server. For example, just opening an email requires a query to the LDAP server for the groups that the current user has been assigned to. Only after fetching this information, can Zarafa determine whether the current user has the access rights to open the message.

When using OpenLDAP as an LDAP source, you can use LDAP replication to guarantee that an LDAP server is available at all times by running an OpenLDAP server on the same machine as Zarafa. This will make sure that the local LDAP server will always be reachable, and Zarafa will always keep running as normal.

5.2 Setting up the LDAP repository

While in principle almost any LDAP repository can be used with Zarafa, this document describes how Zarafa requests the data from the server, and how that data is used within the Zarafa server and tools.

The following information is required from the LDAP server:

- User details (name, email address, etc)
- Group details (name of group)

- Company details
- User/Group relationships (group membership)
- Company members (users and group membership)
- Company relationships (cross-company view and administrator permissions)

The objects that are classified as users, groups or companies and the attributes that contain the data can be configured within the Zarafa configuration files, so Zarafa can almost always meet your LDAP schema needs. However, here are some pointers to keeping you LDAP repository clean and easy-to-manage:

- Always use some sort of graphical user interface for user and group management. There are many LDAP configuration tools. (For example, phpldapadmin for OpenLDAP as a web based interface)
- If you have users that will be using Zarafa, while other users will not, try to group these users into separate 'folders'. You can use an OU record or any other dc-type object to create these folders.
- If you are running Microsoft Active Directory, make sure that your real users are in a separate LDAP folder so that Zarafa doesn't need to import the standard users like 'Administrator' and 'Guest' into the database. It is also possible to filter the users using an LDAP search query, but these search queries can become unsatisfactorily large when using ADS.

As a general rule, always use the LDAPS (SSL) protocol while contacting the LDAP server. When SSL is not used, information will be transmitted clear-text over the wire. This opens possibilities to sniffing user (and administrator!) passwords from the network wire. Zarafa supports connecting through ldaps via SSL and a certificate specified in /etc/ldap/ldap.conf which is compatible with both Microsoft Active Directory as OpenLDAP servers. Zarafa does **not** support STARTTLS-type encryption.

5.2.1 Setting up Zarafa for LDAP

All that is needed for Zarafa to use the LDAP backend are two configuration directives in the server.cfg configuration file:

```
user_plugin = ldap
user_plugin_config = ldap.cfg
```

All ldap-specific settings are stored in ldap.cfg. These will be discussed separately for MS Active Directory and OpenLDAP, apart from the following generic settings:

The host to connect to, can be an IP address or DNS name

```
ldap_host = localhost
```

The port to connect to, normally 389 for LDAP and 636 for LDAPS

```
ldap_port = 389
```

or

```
ldaps_port = 636
```

The protocol to use, currently only 'ldap' and 'ldaps' are supported

```
ldap_protocol = ldaps
```

The bind DN to connect to the server. This can be any user with read-only access to the attributes and objects that are accessed for Zarafa.

```
ldap_bind_user = cn=admin,dc=zarafa,dc=com
```

The bind password to connect to the server. This is the password for the ldap_bind_user.

```
ldap_bind_passwd = ldapadmin
```

5.3 Setting up Zarafa with Microsoft Active Directory

Using Zarafa with a Microsoft Active Directory makes it possible for administrators to use the standard LDAP administration for users, groups and companies and therefore provide a zero-maintenance Zarafa server running on Linux.

Communication with Microsoft Active Directory can be done via standard LDAP access either through port 389 (ldap) or port 636 (ldaps).

5.3.1 Setting up Active Directory for SSL access

Make sure that the Certificate Authority is installed on the DC running your Active Directory. If it is not installed, you can install it as follows:

1. Click **Start -> Control Panel -> Add or Remove Programs**
2. Click **Add/Remove Windows Components** and select **Certificate Services**
3. Follow the procedure provided to install the **Certificate Services CA**.

After installation, you must reboot your Active Directory server to make sure the Active Directory server is accepting TLS/SSL connections.

Now, you must configure your Linux server to connect to the SSL port of the Active Directory. This must be done in the system-wide configuration file `/etc/ldap/ldap.conf` with the configuration option `TLS_CACERT`. This must be configured to point to a CA (Certificate Authority) that can authorize the server certificate on the AD Server.

This can be done either by using the AD server itself as a Certificate Authority (CA) or by using an online CA server. The latter is not recommended due to the time it takes to request the certificate on the internet. If you want to use an online CA, you will need a line like

```
TLS_CACERTDIR /etc/ssl/certs
```

This assumes your CA certificates are installed in `/etc/ssl/certs`. Please refer to your Linux distribution's documentation on where to find the CA certificates or how to install them. (Tip: on debian, you must install the 'ca-certificates' package, while SuSE and RedHat install the certificates together with the 'openssl' package, sometimes in `/usr/share/ssl/certs`).

5.3.2 Retrieving the CA certificate from your AD server

You can also retrieve the CA certificate from your local AD server so that all communication is local during LDAP accesses. To retrieve the certificate from the MS Windows Server:

1. Click **Start -> Control Panel -> Administrative Tools -> Certificate Authority** to open the CA Microsoft Management Console (MMC) GUI.
2. Highlight the CA machine and right-click to select **Properties** for the CA.
3. From General menu, click **View Certificate**.
4. Select the **Details** view, and click the **Copy to File** button on the lower-right corner of the window.

5. Use the Certificate Export Wizard to save the CA certificate in a file. (Use ASCII mode)

The certificate will be saved as a .CER file, but you can simply rename the file to a .PEM file. The filename of the .PEM file is not important.

You can now copy the certificate to your Linux server, for example into /etc/ssl/certs/AD.pem

To use this certificate, please specify

```
TLS_CACERT /etc/ssl/certs/AD.pem
```

in your /etc/ldap/ldap.conf file, or use (also in /etc/ldap/ldap.conf)

```
TLS_CACERTDIR /etc/ssl/certs
```

to accept any CA in the /etc/ssl/certs directory. If you use TLS_CACERTDIR, you must also create the hash link in /etc/ssl/certs: In debian, this is accomplished by running 'update-ca-certificates'. In other Linux distributions, you must create the link manually with

```
ln -s /etc/ssl/certs/AD.pem `openssl x509 -noout -hash -in /etc/ssl/certs/AD.pem`
```

You can check whether the SSL connection is working and see what is happening by issuing the command:

```
openssl s_client -connect <ip>:636 -CApath /etc/ssl/certs
```

To test whether the SSL connection is working correctly with LDAP, try the following command:

```
ldapsearch -x -H ldaps://ads.domain.com -b <BASEDN> -D <binddn> -w <password>
```

If ldapsearch fails, while the s_client test returns with 'Verify return code 0 (ok)', please make sure that the URL you are connecting with after the -H option contains the exact same hostname as is specified behind CN= in the output of s_client (at the very beginning of the output from s_client).

5.3.3 Configuring Zarafa for Microsoft Active Directory Service

If ldapsearch returns data, you are now ready to configure Zarafa to connect to your Active Directory Server with SSL. The configuration directives that need to be set up in ldap.conf are:

The LDAP search base (base DN) that the search for user objects should start at. This should be the 'root' of your users folder which contains your users.

```
ldap_user_search_base = cn=users,dc=zarafa,dc=com
```

The search filter that should be applied. All objects matching this search will be treated as user objects (This should all be on one line in the ldap.cfg file).

```
ldap_user_search_filter =  
    (&(objectClass=person)  
    (objectCategory=CN=Person,  
    CN=Schema,CN=Configuration,DC=zarafa,DC=com))
```

The attribute in the object that should uniquely identify this user. For Active Directory, you can use the objectSid attribute, which is a globally unique identifier which will never change over the lifetime of the AD. Because this is a binary attribute, you also need to specify this in the configuration file.

```
ldap_user_unique_attribute = objectSid
ldap_user_unique_attribute_type = binary
```

The same configuration directives are for groups:

```
ldap_group_search_base = cn=groups,dc=zarafa,dc=com
ldap_group_search_filter = (objectClass=group)
ldap_group_unique_attribute = objectSid
ldap_group_unique_attribute_type = binary
```

The same configuration directives are for companies:

```
ldap_company_search_base = dc=zarafa,dc=com
ldap_company_search_filter = (objectClass=organizationalUnit)
ldap_company_unique_attribute = objectSid
ldap_company_unique_attribute_type = binary
```

Note: When working in a multi-company environment, the `ldap_user_search_base` and `ldap_group_search_base` will not be used in favour of `ldap_company_search_base`. All users and groups which are located hierarchily below a company are considered member of that company.

For the various user, group and company attributes, you can specify which LDAP attribute is used to retrieve the information:

```
ldap_fullname_attribute = displayName
ldap_loginname_attribute = sAMAccountName
ldap_emailaddress_attribute = mail
ldap_groupname_attribute = cn
ldap_companyname_attribute = cn
```

For the membership relationships between groups and users, each group object in the AD has an attribute 'member'. This can be configured in Zarafa with

```
ldap_groupmembers_attribute = member
```

The member attribute contains the DN of zero or more users in that same LDAP database. This must be specified in the configuration file also:

```
ldap_groupmembers_attribute_type = dn
```

5.3.4 Password checking with MS Active Directory

Password checking with Zarafa against an MS Active Directory works by trying to connect to the LDAP server with the user DN and the given password. This means that password checking also makes use of the LDAP protocol. This can be achieved by using the 'bind' password authentication method in Zarafa:

```
ldap_authentication_method = bind
```

With the previous configuration options in place, you should be able to start Zarafa with `'/etc/init.d/zarafa-server start'` and then be able to get a user list and group list with

```
zarafa-admin -l
and
zarafa-admin -L
```

when multi-company support is enabled, the list of companies can be requested with:

```
zarafa-admin -list-companies
```

If no user list is shown, or no groups are shown, please check your zarafa log file (usually /var/log/zarafa/zarafa.log) and re-check your configuration. The first time you run the -l or -L options, it may take a long time to get the user list, depending on how many users must be created due to the query to the server.

5.3.5 Using advanced LDAP attributes with Microsoft Active Directory

Some user information that Zarafa can use is not available in the LDAP database of an Active Directory server by default. These are:

- User/Company Quota settings
- Multiple email address support (aliases)
- Company rights settings

It is possible to add support for this information to the AD, which requires installing the Zarafa Active Directory extension. This extension registers new attributes for these features, and provides a new tab in the AD user management console.

Note: Zarafa itself does not make any use of the multiple email aliases, but they can be used by your MTA on the Linux host, e.g. postfix).

5.3.6 Set up Zarafa Active Directory Extension and schema files

With the Zarafa Active Directory extension, the extra attributes required can be set and viewed through the standard active directory 'users and computers' administration console. The extra attributes are: Quota, Zarafa administrator and email aliases.

To use the Zarafa Active Directory Extension, download and install it with the setup program on your Active Directory.

Note: The Active Directory Extension need be installed on the Active Directory server that's the schema master, because the installer will update the schema files. It's currently not possible to install only the graphical interfaces without updating the schemas.

Note: Installing the Zarafa Active Directory Extension will cause the Active Directory server to update its schema files. This action is *not reversible*, although the administration interface *can* be deinstalled.

5.3.6.1 Windows 2000 server

When you run the installation on a Windows 2000 server, the setup requires write access to update the Active Directory Schema. To get the write access you must enable the registry key "Schema Update Allowed".

To edit the registry key, perform the follow steps:

1. Click **Start**, click **Run**, and then in the **Open** box, type: 'regedit' Then press ENTER.

2. Locate and click the following registry key:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Parameters

3. On the **Edit** menu, click **New**, and then click **DWORD Value**.
4. Enter the value data when the following registry value is displayed:
 Value Name: Schema Update Allowed
 Data Type: REG_DWORD
 Base: Binary
 Value Data: *Type 1 to enable this feature, or 0 (zero) to disable it.*
5. Quit Registry Editor.

Now you can run the Zarafa Active Directory extension setup.

For more information take a look at: <http://support.microsoft.com/kb/285172>

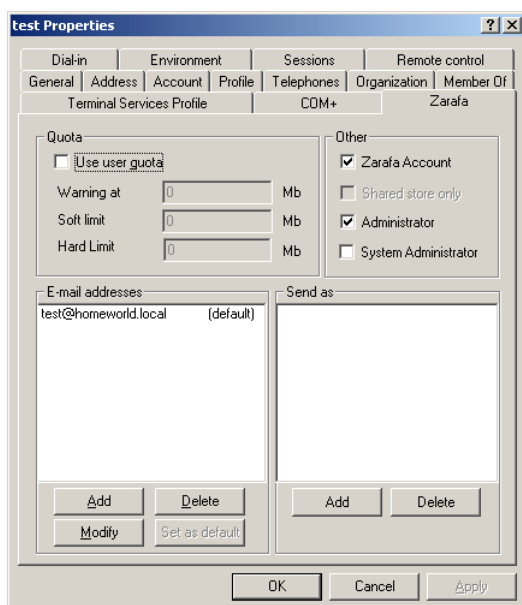
Note: Don't forget to switch the registry key back.

5.3.6.2 Windows 2003 server

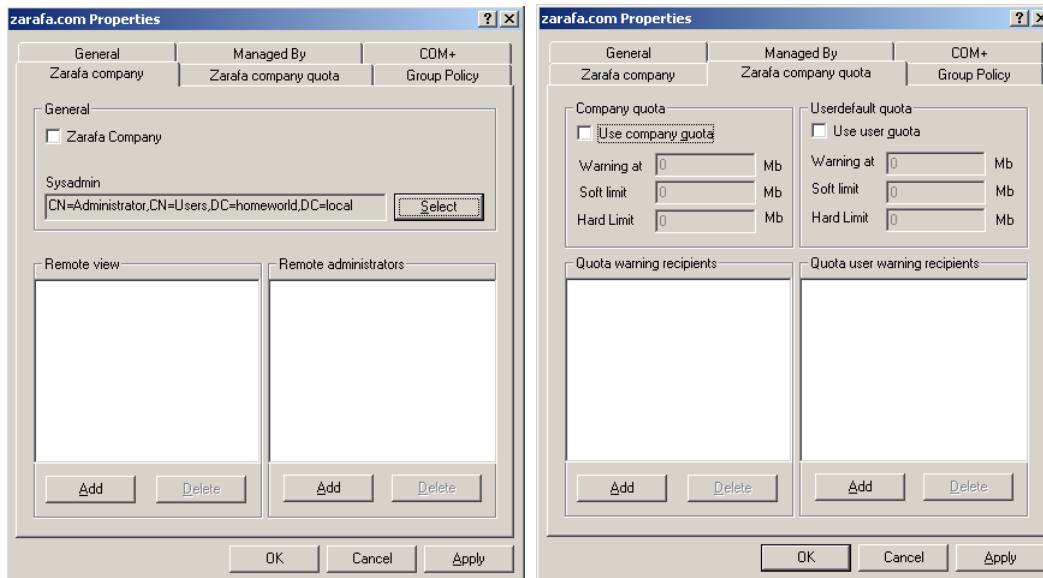
For windows server 2003 you can step through the setup to click the next button.

If the Zarafa Active Directory extension is installed, it is possible to edit the Zarafa attributes. For edit an user go to 'users and computers', select an user and get the properties. Now you see a tab with the name Zarafa.

To configure a company within the Active Directory environment, tabs have been installed on the properties page of organizational Units. The first tab is for the company settings including inter-company rights, and the second tab is for company quota settings.



Screenshot 2: Zarafa user tab



Screenshot 3: Zarafa company tab and Zarafa company quota tab

5.4 Setting up Zarafa with OpenLDAP

In many network configurations, OpenLDAP is used as the main LDAP server, and Zarafa can be used to connect to OpenLDAP servers, either via port 389 or 636 (SSL). For best speed and reliability, it is always best to install an OpenLDAP server on the same physical host as the Zarafa server which replicates with your main LDAP server. This makes sure that should the main server go down, Zarafa can keep functioning on the local host.

In the follow paragraphs the configuration will be explain. Before you begin, check the location of the the configuration files.

The OpenLDAP server is mostly installed in the default directory of your distribution:

- SUSE: /etc/openldap
- Red Hat Enterprise Linux: /etc/openldap
- Debian & Ubuntu: /etc/ldap

5.4.1 Setting up SSL for OpenLDAP

Setting up SSL for OpenLDAP requires you to create an SSL certificate, install the certificate, and set up slapd (the OpenLDAP server process) to accept connections on the SSL port. If you already have an SSL certificate (for example, for Apache), then you can use that certificate for slapd also.

First, create a private key with:

```
openssl genrsa -out private.pem 2048
```

This will create the file 'private.pem' with a 2048-bit private RSA key. You can then create a self-signed SSL certificate with:

```
openssl req -new -x509 -key private.pem -out cert.pem -days 1095
```

Make sure that the private.pem file stays secret to anybody except the server process serving the SSL (ie slapd), while the cert.pem file is freely available to clients wanting to access the server. Normally, you would place private.pem in /etc/ldap/private.pem and cert.pem in /etc/ssl/certs/<servername>.pem.

You must also create the hash link in /etc/ssl/certs: In debian, this is accomplished by running 'update-ca-certificates'. In other Linux distributions, you must create the link manually with:

```
ln -s /etc/ssl/certs/server.pem `openssl x509 -noout -hash -in /etc/ssl/certs/server.pem`
```

5.4.2 Configuring OpenLDAP's slapd to use SSL

To configure slapd to use SSL, you need to add the following configuration directives to /etc/ldap/slapd.conf:

```
TLSCipherSuite HIGH:MEDIUM:+SSLv2
TLSCACertificateFile /etc/ssl/certs/server.pem
TLSCertificateFile /etc/ssl/certs/server.pem
TLSCertificateKeyFile /etc/ldap/private.pem

TLSVerifyClient never
```

You must then restart slapd with

```
/etc/init.d/slapd restart
```

5.4.2.1 Testing your SSL / OpenLDAP connection

You can check whether the SSL connection is working and see what is happening by issuing the command:

```
openssl s_client -connect <ip>:636 -CApath /etc/ssl/certs
```

To test whether the SSL connection is working correctly with LDAP, try the following command:

```
ldapsearch -x -H ldaps://openldap.domain.com -b <BASEDN> -D <binddn> -w <password>
```

5.4.3 Configuring OpenLDAP to use Zarafa schemas

To configure slapd to use Zarafa LDAP schemas, you need to add the following configuration directives to /etc/ldap/slapd.conf:

```
include /etc/ldap/schema/zarafa.schema
```

Copy the schema file to the ldap directory:

```
cp /usr/share/zarafa/zarafa.schema /etc/ldap/schema/zarafa.schema
```

5.4.4 Configuring Zarafa for OpenLDAP

The configuration directives that need to be set up for Zarafa in /etc/zarafa/ldap.conf are:

The LDAP search base (base DN) that the search for user objects should start at. This should be the

'root' of your users folder which contains your users.

```
ldap_user_search_base = cn=users,dc=zarafa,dc=com
```

The search filter that should be applied. All objects matching this search will be treated as user objects (This should all be on one line in the `ldap.cfg` file).

```
ldap_user_search_filter = (objectClass=posixAccount)
```

The attribute in the object that should uniquely identify this user. This is normally 'uid' or 'uidNumber'. If you use 'uid', which normally contains a username, you will not be able to rename the user. Using `uidNumber` makes sure the user is identified even if the username is changed.

```
ldap_user_unique_attribute = uidNumber  
ldap_user_unique_attribute_type = text
```

The same configuration directives are for groups:

```
ldap_group_search_base = cn=groups,dc=zarafa,dc=com  
ldap_group_search_filter = (objectClass=posixGroup)  
ldap_group_unique_attribute = gidNumber  
ldap_group_unique_attribute_type = text
```

The same configuration directives are for companies:

```
ldap_company_search_base = dc=zarafa,dc=com  
ldap_company_search_filter = (objectClass=organizationalUnit)  
ldap_company_unique_attribute = cn  
ldap_company_unique_attribute_type = text
```

Note: When working in a multi-company environment, the `ldap_user_search_base` and `ldap_group_search_base` will not be used in favour of `ldap_company_search_base`. All users and groups which are located hierarchily below a company are considered member of that company.

For the various user, group and company attributes, you can specify which LDAP attribute is used to retrieve the information:

```
ldap_fullname_attribute = cn  
ldap_loginname_attribute = uid  
ldap_emailaddress_attribute = mail  
ldap_groupname_attribute = cn  
ldap_companyname_attribute = cn
```

For the membership relationships between groups and users, each group object in `posixGroup` has an attribute 'memberUid'. This can be configured in Zarafa with:

```
ldap_groupmembers_attribute = memberUid
```

The member attribute contains the uid of zero or more users in that same LDAP database. This must be specified in the configuration file also:

```
ldap_groupmembers_attribute_type = name  
ldap_groupmembers_relation_attribute = uid
```

From Zarafa 6.20.8, additional information from Zarafa users can be stored in LDAP, so it is available in the global address book. This information is specified in the following attributes:

```
ldap_user_telephone_attribute =  
ldap_user_department_attribute =  
ldap_user_location_attribute =  
ldap_user_fax_attribute =
```

5.4.4.1 Password checking with OpenLDAP

The easiest way of enabling password checking is to use the 'bind' authentication method. This means that a password will be checked by attempting to bind to the server using the DN of the user, and the passed password. In default installations of OpenLDAP, this is enabled, so this is the easiest way of checking the passwords:

```
ldap_authentication_method = bind
```

With the previous configuration options in place, you should be able to start zarafa with:

```
/etc/init.d/zarafa-server start
```

And then be able to get a user list and group list with:

```
zarafa-admin -l  
and  
zarafa-admin -L
```

when multi-company support is enabled, the list of companies can be requested with:

```
zarafa-admin -list-companies
```

If no user list is shown, or no groups are shown, please check your zarafa log file (usually /var/log/zarafa/zarafa.log) and re-check your configuration. The first time you run the -l or -L options, it may take a long time to get the user list, depending on how many users must be created due to the query to the server.

5.5 Advanced configuration

5.5.1 Change unique attribute of Zarafa, in a live environment configuration

When you want to change the Zarafa unique relationship between Zarafa and LDAP in a live environment you can do this by following these steps:

1. First shutdown Zarafa server
2. Make a backup from the Zarafa database, for your own security
3. Update the unique attribute in the Zarafa "ldap.cfg"
4. Now you must update the Zarafa database

Login with mysql on the command line or another interface
To enter the Zarafa database, type:

```
mysql> USE zarafa;
```

To show the existing users with the old unique attribute, type:

```
mysql> SELECT id, object_type, externid FROM users;
```

You see the table structure:

- “id” field, internal Zarafa number
- “object_type” field is the objecttype. 1=active user, 2=group, 4=company, 5=nonactive user.
- “externid” is the relation between Zarafa and LDAP.

When the attribute “ldap_user_unique_attribute” is changed and you want to keep the current user information, it is necessary to update all items with “object_type” 1 and 5. The “externid” must be updated with the new user relation id.

When the attribute “ldap_group_unique_attribute” is changed and you want to keep the current group information, it is necessary to update all items with “object_type” 2. The “externid” must be updated with the new group relation id.

To update the “externid” for a user or group you can use:

```
mysql> UPDATE users SET externid=<newvalue> WHERE id=<id>;
```

When the unique id is a binary field you must insert the data binary like:

```
mysql> UPDATE users SET externid=0x1234567890 WHERE id=<id>;
```

1. When all items are updated, you can start the Zarafa server
2. Check the user list: `$ zarafa-admin -l`
3. Check the group list: `$ zarafa-admin -L`
4. Take a look of the Zarafa server log. When a user is deleted you see the message: “Auto-deleting user from external source” or “Auto-deleting group from external source”.

When you see the aforementioned message, something goes wrong.

5.5.1.1 Example:

When you want to change the ldap_user_unique_attribute attribute from “cn” to “uidNumber”

The information in your ldap is for example:

```
dn: cn=name,ou=People,dc=Zarafa,dc=com
cn=name
uidNumber=1000
```

1. The first step is to shutdown server.
2. Open the ldap.cfg and change the ldap_user_unique_attribute attribute from “cn” to “uidNumber”
3. Now the Zarafa database must be updated. Go to the Zarafa users table.

```
$ mysql -p "databasename"
```

The databasename is most of the time “Zarafa”

4. Show the users information:

```
mysql> SELECT id, object_type, externid FROM users;
```

id	object_type	externid
1	1	NULL
2	2	NULL
3	1	Jan
4	1	Piet

Table 4: User information

5. Change the externid from the value "jan" to the value "1000" and "Piet" into "1001"

```
mysql> UPDATE users SET externid="1000" WHERE id=3;
mysql> UPDATE users SET externid="1001" WHERE id=4;
```

6. Start the Zarafa server

7. Check the user list with : `$ zarafa-admin -l`

5.5.2 Send as Permissions option

After inserting the zarafa schema's and setting up your user structure it is possible to use specific settings intended for Zarafa.

One important setting is the Send as Permissions. This setting will be explained for LDAP, though also applicable for ADS.

To read more about Permissions, please read the Zarafa server manual if you are interested in the admin side and the Zarafa client manual for the user side.

1. Find the user who will send as another user
2. Add the *ObjectClass* 'zarafa-user'
3. Add new attribute 'ZarafaSendAsPrivilege'
4. Find the uid number of the user to be 'impersonated'.
5. Add the uidnumber as a value in the attribute ZarafaSendAsPrivilege.
6. Edit ldap.cfg and edit if needed, the values:

```
ldap_user_sendas_attribute = zarafaSendAsPrivilege
ldap_user_sendas_attribute_type = text
ldap_user_sendas_relation_attribute =
```

If `ldap_user_sendas_relation_attribute` is empty LDAP will use the attribute that is available in `ldap_user_unique_attribute` by default

objectClass required

*

(add value)

sn required

*

(add value)

uidNumber required

*

User Name alias, required, rdn

*

(rename)

userPassword

[Check password...](#)

(add value)

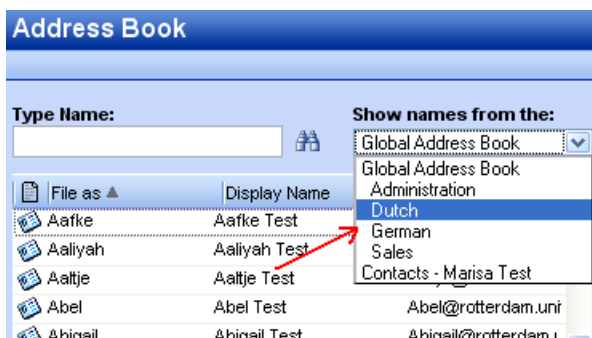
zarafaSendAsPrivilege

(add value)

Screenshot 4: SendAsPrivilege option.

5.5.3 Addresslists by condition

Addresslists are groups of users that agree with any custom condition, so users do not need to be assigned to groups. These groups are accessible for users in the global address book



Screenshot 5: Addresslists in the Address book

5.5.3.1 Setup addresslists

Create an Organisation Unit for the addresslists in the LDAP tree. The cn attribute for the addresslist objects will act as the addresslistname. The zarafaFilter attribute will contain the condition query.

Name	Value	Type
objectClass	posixGroup	Attribute
objectClass	zarafa-addresslist	Attribute
zarafaFilter	(mail=*@zarafa.nl)	Attribute
gidNumber	10003	Attribute
cn	Dutch	Attribute

Screenshot 6: addresslists in LDAP

Change or add in `ldap.cfg` the following configuration settings for the addresslist objects and change the value from `ldap_addresslist_search_base` to the LDAP location.

```
ldap_addresslist_search_base = ou=Addresslists,dc=domain
ldap_addresslist_search_filter = (objectClass=posixGroup)
ldap_addresslist_unique_attribute = gidNumber
ldap_addresslist_unique_attribute_type = text
ldap_addresslist_filter_attribute = zarafaFilter
ldap_addresslist_name_attribute = cn
```

After restarting the `zarafa-server`, the addresslists should be visible in the global addressbook.

5.5.3.2 Condition examples

For example, the global addressbook contains Dutch and German users. It is possible to view these users per country by creating two addresslists in the LDAP tree. All German users have the domain `zarafa.de` in the mail address, and all the Dutch have `zarafa.nl`.

In this situation, the condition `mail=*@zarafa.de` is used for the addresslist German, and `mail=*@zarafa.nl` for the addresslist Dutch.

Any combination with LDAP attributes are applicable. This following example selects everyone that is a Zarafa administrator **and** has the character `p` in the `cn` value.

```
(&(cn=*p*)(zarafaAdmin=1))
```

This example selects all users with mailaddress `piet@example.com` **or** `klaas@example.com`.

```
(|(mail=piet@example.com)(mail=klaas@example.com))
```

6 Single SignOn for Zarafa

This chapter will define steps on how to configure your pristine linux installation to be able to authenticate users using NTLM to an ADS domain server or a Samba server as Primary Domain Controller (PDC).

6.1 Before attempting Single Sign On (SSO)

This document assumes you already have a Windows ADS domain server up and running or a Samba server as PDC.

In the case of ADS, you also need to have knowledge on how you can add names to your local DNS service. This document only provides some linux software and commands you will need to be able to use the single signon through NTLM.

The first step is to enable SSO in Zarafa; change the parameter `enable_sso_ntlmauth` in `server.cfg` to `enable_sso_ntlmauth = yes`

In chapter 6.2 SSO with ADS is discussed, in chapter 6.3 SSO with Samba.

6.2 SSO with ADS.

6.2.1 Installing linux software

You will need to following software installed:

- winbind
- kinit

Depending on your linux distribution, this comes through various package names. On debian type:

```
apt-get install krb5-user winbind
```

krb5-user will also install the kerberos library configuration files in /etc. The command 'winbind' will also install samba-common which provides some samba commands like net you will need to use.

On RedHat Enterprise you need to install krb5-workstation and samba-common packages for this.

6.2.2 ADS: Specific network setup

You need to have the following things straight in your network:

- Every server must have a DNS name, so you can use DNS requests to lookup their ip-address
- All servers must be in sync. Time cannot lag for a few minutes.

This document has the following names as example:

FQDN of your Windows ADS server: ADSSERVER.ADSDOMAIN.LOCAL

Therefore, your windows server is named: ADSSERVER, the realm is ADSDOMAIN.LOCAL, and the Workgroup name is ADSDOMAIN.

Workstations can therefore either join your domain using the ADSDOMAIN or ADSDOMAIN.LOCAL name.

FQDN of your linux server: LINUXSERVER.LOCAL

The name of your linux server doesn't matter much, as long as it's available in your DNS server.

6.2.3 Configuring Kerberos library

First we are going to configure the kerberos library. The configuration file is /etc/krb5.conf

Under the [libdefaults] section, set:

```
default_realm = ADSDOMAIN.LOCAL
```

Under the [realms] section, add the windows realm:

```
[realms]
ADSDOMAIN.LOCAL = {
    kdc = 192.168.0.1
    admin_server = 192.168.0.1
    password_server = 192.168.0.1
```

```
        default_domain = ADSDOMAIN.LOCAL
    }
```

Here, 192.168.0.1 is the IP-address of your Windows ADS domain server.

Now that the kerberos library is configured. You can login using kinit on your linux server:

```
$ kinit Administrator
Password of Administrator@ADSDOMAIN.LOCAL:
```

Type your administrator password there, and you should receive a kerberos ticket from your ADS server. You are now known as Administrator on your linux server for kerberos actions.

6.2.4 Joining the ADS domain

First we'll configure samba. Edit the /etc/samba/smb.conf file, and add/set the following options:

```
[global]
realm = ADSDOMAIN.LOCAL
use kerberos keytab = true
security = ads
```

With this ticket we can join the Windows domain, without typing the password again:

```
$ net ads join -S ADSDOMAIN -U Administrator
```

This command may also be different for different versions of Samba. When this command asks you for a password, something goes wrong and you need to stop it with Ctrl-C.

When all goes well, you will see the line:

```
Joined 'LINUXSERVER' to realm 'ADSDOMAIN.LOCAL'
```

or some other success message.

Now we need to restart the winbind daemon, because it keeps too many items cached:

```
$ /etc/init.d/winbind restart
```

And that's it. To see if authentication to the actually works, try the following command:

```
$ ntlm_auth --username=john
```

Where john is a user on your ADS server.

The program will asks for a password. After entering the password, it should say:

```
NT_STATUS_OK: Success (0x0)
```

If this step does not work, try restarting winbind, check your DNS names, check with `strace` what `ntlm_auth` tries to do, check with `tcpdump` if there is actual traffic on your network from `ntlm_auth` to the domain server, and other lowlevel debugging tools. That's what we did too.

6.3 SSO with Samba.

6.3.1 Installing linux software

You will need to following software installed on your zarafa server:

- winbind

Depending on your linux distribution, this comes through various package names. On debian type:

```
$ apt-get install winbind
```

6.3.2 Joining the domain

Now we can join the domain by typing the following command:

```
$ net rpc join
```

Finish by typing the Administrator password. If succesfull the prompt should give:

```
Joined domain <DOMAIN>
```

6.4 Up and running

Now that SSO seems to work on your linuxserver, this will automatically be used by zarafa-server. Login on a Windows workstation on the domain. Create a new Outlook profile for the user you just logged on as, but leave the password field empty. Outlook should create the profile without the password.

If you check your zarafa server logfile, you should see messages about the user being authenticated by ntlm_auth.

7 Performance tuning for Zarafa

When installing a Linux server with Zarafa, it is imperative that you correctly configure MySQL to achieve maximum performance on your server; almost all performance bottlenecks are within the database access itself, so getting your SQL queries to run as quickly as possible is very important.

For large installations, it is also a good idea to tune Zarafa's cache parameters as well; These are normally set quite low to make sure that Zarafa can run on relatively low-end servers, but in anything but the smallest installations, these defaults needs to be upped. Any installation with 50 or more users should definitely tune the cache parameters for maximum performance.

This document assumes the primary role of your server is to run Zarafa. Always make sure that you take other factors into account – for example an anti-spam system or a webserver running a site other than the Zarafa WebAccess.

7.1 Hardware considerations

There are also various different hardware setups to consider when setting up a server for Zarafa. We will discuss the various types of hardware that affect performance.

7.1.1 RAM Memory usage

Tuning RAM usage is one of the best ways of increasing performance on your server; as RAM is generally cheap, using the large amount of RAM in your server properly can boost performance by orders of magnitude.

On the other hand, setting RAM usage too high may cause the server to swap out parts of the memory which need to be swapped back in later, causing a large slowdown in all parts of the server. It is therefore important to set the RAM usage of various components to a high enough setting to use the RAM that you have, and at the same time not to set the RAM usage too high.

To make use of your RAM as best as possible, Zarafa is designed to use only a fixed amount of physical RAM; the memory usage does increase per user that connects, but only by a small amount – the largest part of the memory usage is due to cache settings in your configuration file. This makes it very easy to control the exact amount of memory that will be used in a live situation, and you can be pretty sure that the actual amount of RAM used will never go far beyond your settings.

So, in general, the optimum RAM usage is as high as you can go, without making the system needing to swap out important parts of your memory.

It is very difficult to give a fixed value for what the optimal memory usage distribution is for a given server, as data access patterns vary wildly from server to server. We will describe some rule-of-thumb parameters here and make the RAM usage patterns as clear as possible here.

7.1.2 Multiserver setups

By splitting the load over different servers, you can achieve better performance because you are basically load-balancing the server chain over multiple Zarafa servers. There are many possibilities in which you can use multiple servers to distribute the load over multiple servers.

7.1.3 Hardware considerations

In servers running Zarafa, the main performance bottleneck will be the route between the data on your harddrive, and the time it takes to get to the client. This means that generally, I/O performance is more important than CPU performance. Using this as a basis, the following pointers may help you in selecting the correct hardware for your system:

7.1.4 More RAM is More Speed

More RAM means better caching and therefore better speed.

Zarafa is specifically designed to make use of the large amounts of RAM that is available in modern servers. On the other hand, please remember that in normal Linux server the maximum amount of usable RAM in a 32-bit server is 3Gb unless PAE (physical address extension) is supported in the kernel, CPU and mainboard. If more than 3Gb is needed without some sort of limitation, use a 64 bit system, a 64 bit Linux OS, and a 64 bit Zarafa package.

7.1.5 RAID 1/10 is faster than RAID 5

In general, a RAID1 or RAID10 array is faster at database accesses than RAID5. If you have the choice, always go for RAID10 if this is an option

7.1.6 Multiprocessor is not always faster

A multi-processor system may boast good CPU performance, but this isn't that important in a Zarafa setup. Also, running multiple CPU's may cause the kernel to do more context switching – indeed,

turning off Hyperthreading on some processors has shown to increase performance in some cases. In practice the differences are minor though, and if there is a trade-off between CPU power and I/O, always go for the I/O

7.1.7 High rotation speed (RPMs) is better for disks

High-end SCSI disks regularly have high rotation speeds of 10K or even 15K RPMs. The rotation speed of the disks affects seek times on the disk. Although the Zarafa database format is optimized to have data available on the disk in a serial fashion, and most reads are done fairly localised on the disk, seek time is still a large speed factor for I/O. The higher the rotation speed, the lower the seek time.

7.1.8 Hardware RAID

Hardware RAID controllers often have large amounts of Cache RAM. This can also increase performance and data throughput of the I/O subsystem. If you are using a hardware RAID controller however, always make sure that you are either not using write-back cache, or you have a functioning UPS and shutdown process for the server, as write-cached data will be lost when the power fails. This is not only bad for the data you were writing at that moment, the write could actually corrupt the on-disk innodb data.

7.2 Memory Usage setup

There are basically 4 large parts of your server setup that use server memory:

- Zarafa's cell cache (caches individual cell data within a table view)
- MySQL's buffer size (caches reads and writes from the `ibdata` file) - MySQL's query cache (caches exactly repeated SQL queries)

In a server purely running Zarafa, you want to setup these caches to use around 80% of the RAM in your server. The other 20% should be free for system processes, other processes (like your MTA) and the webserver.

For a general rule-of-thumb, you should use to following RAM distribution:

Zarafa caches:

- `cache_cell_size`: around 25% of total RAM size
- `cache_object_size`: about 100kb per user
- `cache_indexedobject_size`: about 512kb per user

MySQL caches:

- `innodb_buffer_pool_size`: around 25% of total RAM size
- `mysql_query_cache`: 8Mb
- `innodb_log_file_size`: 25% of the `innodb_buffer_pool_size`
- `innodb_log_buffer_size`: 32M

This will cause around 50%-60% of your RAM to be tied up in caches for MySQL and Zarafa. The actual memory usage of the MySQL and Zarafa will then be slightly more than this, giving a total of around 80% of your RAM size.

Please refer to the MySQL documentation for the setting of the `innodb_log_file_size` and related settings, as you will also want to set these somewhat higher than the defaults to increase write performance. They don't affect read performance.

The 4 settings will now shortly be discussed to illustrate the need of each of these cache settings.

7.2.1 Zarafa's Cell Cache (`cache_cell_size`)

Data that is actually shown to the user in table views, passes through the 'cell cache'. This means that any view of a table in Outlook will only retrieve the information from the database of the cells that are not already in the cache. The cache lifetime is as long as the entire server lifetime, so opening an inbox twice in succession should result in 0 disk accesses for the second access. It is a good idea to set the cell cache as high as you can manage, usually about the same size as the MySQL buffer size.

7.2.2 Zarafa's object cache (`cache_object_size`)

The Zarafa object cache is used to cache the hierarchy table. Each object that is accessed will be placed in this cache, making it faster to retrieve the information again without accessing the database. The more items users have in their folders, the more important this cache becomes. Since the information is quite small, this cache does not need to be large. About 1Mb for 10 users is even an overestimation.

7.2.3 Zarafa's indexedobject cache (`cache_indexedobject_size`)

To open a specific item, the program needs to send the server a unique key, called an `entryid`, to the server to request that item. This cache is a 2 way index of the MAPI key to a database key and the other way around. The translation of the keys are quite important. This cache is filled per folder, so large folders will push out otherwise important information. Normal usage is about 0.5 Mb per user.

7.2.4 MySQL `innodb_buffer_pool_size`

The MySQL buffer is used to cache reads and writes to the `ibdata` file. In a dedicated MySQL machine, this would be anywhere between 50% to 80% of the physical RAM size in your machine. When you are running MySQL on the same machine as Zarafa, it is recommended to be around 25% of physical RAM size (so that Zarafa's Cell Cache can also be set to this value)

7.2.5 MySQL `innodb_log_file_size`

The `innodb_log_file_size` is the size of the transaction log. By default there are two logfiles. The preferred value size for the `innodb_log_file_size` is 25% of the `innodb_buffer_pool_size`.

7.2.6 MySQL `innodb_log_buffer_size`

The size of the `innodb_log_buffer_size` that InnoDB uses to write to the log files on disk. A large log buffer allows large transactions to run without a need to write the log to disk before the transactions commit. If you have big transactions, making the log buffer larger will save disk I/O. This value should be 25% of the `innodb_log_file_size`.

7.2.7 MySQL `query_cache`

The MySQL query cache is normally disabled. Enabling the `query_cache` can cause a small performance increase, but increasing it to more than a few MB of memory isn't any use, as most recurring SQL queries are rather small.

7.2.8 Setup of modules on different servers

There are several parts of the Zarafa server that can be hosted on different servers. In fact, almost each part of the server can be run on a different system. However, in practice, it often won't help you to improve performance to split all modules of the server into different servers. The main parts that should be considered are:

- Server1: MySQL server

- Server2: Zarafa server
- Server3: MTA + AntiSpam/AntiVirus
- Server4: WebServer

If these 4 parts were to be hosted on 4 servers, each server would communicate with the others to work as a single system. This setup can be made quite easily simply by configuring the various parts of the system to communicate with another server.

For the MySQL server, this only has to be accessed by the zarafa-server process on Server2. This can very easily be done by setting the correct login and host configuration in Zarafa's server.cfg.

The Zarafa Server will itself be contacted by Outlook Clients, Server3 (MTA), and Server4 (WebServer). This can be done because the zarafa-server process is listening on port 236 on Server2, and the other servers can connect with it.

Server3 will accept email on port 25 or fetch email via some email protocol like POP3. After passing the email through anti-spam and anti-virus, the email will be passed to the zarafa-dagent process. The zarafa-dagent process can be configured to connect with an SSL certificate with Server2. This SSL certificate is required because the zarafa-dagent needs to be authenticated because it is connecting from a different server over port 236. When this is configured in both Server3 and Server2, the email can be delivered directly to Server2 by Server3.

Server4 is the webaccess server, running Apache, and accepting connections on port 80 (or 443 for SSL). The Zarafa webaccess can be configured (in config.php) to connect over port 236 (or port 237 for SSL) to Server2 for the actual data. Once this has been configured, this server is ready to serve users. No additional configuration is required.

8 Backup

Currently, Zarafa provides three ways of restoring items:

- Through the softdelete cache
- Via a full database dump
- Using the brick-level backup system

8.1 Softdelete cache

The softdelete cache can be used by users from Outlook with the 'Restore deleted items' dialog from the 'Tools' menu to restore deleted items. This will cover most accidental deletions.

Items that are deleted by the user (by emptying the deleted items folder or with a hard delete like shift-delete in Outlook), are simply placed in the deleted items cache. This means that the item will not actually be removed from the database until the retention time of the item has expired. This expiration time in can be specified in the server.cfg configuration time and it set to 30 days by default.

Note that the 'restore deleted items' dialog works on the currently selected folder.

In the following overview, you will see which possibilities can be performed by whom, and when it's most likely used.

Restore request	% of time spent	Backup solution	Performer
Items < 30 days old	60 %	Softdelete system	User and Administrator

Items >= 30 days old	20 %	Bricklevel	Administrator
Items from a specific sender	10 %	Bricklevel	Administrator
Items over a specific time period	8 %	Bricklevel	Administrator
Disaster recovery	2 %	MySQL Dump	Administrator

Table 5: recovery options

As you can see, the most common restore request can be performed by the user itself. This is because the softdelete system is accessible through Outlook.

When older messages are requested to be restored, the Administrator will need to consult the backups. It is not possible to restore a single item with a MySQL dump, so this is the point where the zarafa-backup tool steps in.

The bricklevel backups from the zarafa-backup tool contain not enough information for a disaster recovery. You will need a complete dump of the MySQL database to be able to perform this type of recovery.

8.2 Full database dump

All the data that is stored by Zarafa is stored within a MySQL database. This means that for a disaster recovery, all that is needed is a full backup/restore of the database in question. This can be done in many ways, but we will explain two ways of doing a good backup here. Also, there are some ways *not* to do a backup

8.2.1 SQL dump through mysqldump

You can save the contents of an entire Zarafa database to file by using the 'mysqldump' command. There are, however, some options that are important in this case: you should always specify the '--single-transaction' option to mysqldump. When you do this, it will cause mysqldump to write a single snapshot of the database to disk. This will make sure that any writes done in the database during the backup will **not** be backed up. In effect, the dump that is made is a 'snapshot' of the database at the moment that the dump started.

When using mysqldump, it is very important not to do any table locking. This means that the '--opt' option and '--lock-tables' should never be used while dumping a Zarafa database. The reason is that these options will 'lock' the tables while they are being dumped to disk, causing any accesses to the database to 'freeze' while the backup runs. This is firstly unnecessary and secondly may cause emails that are arriving during backup to bounce (depending on your MTA settings).

A simple

```
mysqldump --single-transaction -p <database> > <dumpfile>
```

will start a good dump of the database.

8.2.2 Binary data dump via LVM Snapshotting

This technique uses the 'LVM Snapshot' feature to effectively 'freeze' a binary view of the database file, while the database keeps running. This 'frozen' view is then simply binary copied to a remote server. This works because innodb makes sure that a single snapshot of a database will always be coherent (ie. It will be able to recover the database when you start up mysql on this dataset.)

As setting up LVM and configuring LVM for snapshots is a complex process, we refer the user to the LVM documentation and tools on how to set up an LVM volume for the MySQL data, and how to create and delete snapshot partitions.

8.3 Brick-level backups

A new tool since the 5.10 version of Zarafa is a brick-level backup tool. This means that you create a backup of the stores to separate files. The second time you perform this backup for a store, only the added items are added to the backup.

Please note that this kind of backup is not meant for disaster recovery. Only items are written in the backup. No information about the users, or specific information the user create, like rules, are not backed up.

8.3.1 Backup format

The backup tool creates 2 files for each store: a data file and an index file.

The index file contains information about folders, the hierarchy and messages. The fields are colon separated. There are 3 types of entries in the index file, which are R, C and M. The R stands for Root, and is always the first and the only R entry in the index. It contains a key which folders use as their parent key to denote that they are directly connected to the root container of the store.

The C stands for Container, which can be any type of folder. It has 2 keys, one parent and one to identify the container itself. It also has a unique restore key. This key can be used to select the folder for the restore tool. It has an indicator of how many items there are in the folder, a last modification unix timestamp, and a type of the folder (eg. IPF.Note for a mail folder, IPF.Appointment for a calendar). The last part of a C entry is the name of the folder, which may contain a colon itself, so therefore it is the last part in the entry. A detailed list of the fields for a Container can be found in the appendix.

The M in the index stands for Message, which can be any type of message or item. It has a parent key, which matches a folder key. Then it has a restore key, which you can use to restore this specific message. A unix timestamp follows which is the last modification time of the message. If a user changed the message, this timestamp will be updated. The index entry continues with the type of message (mail, calendar, meeting request, etc). The entry contains an offset where the item starts in the data file, and lastly contains the subject of the item. Since this subject may contain colons, it is at the end of the entry. A detailed list of the fields for a Message can be found in the appendix.

The data file is a binary dump of all the message properties, recipients and attachments. Folders are only set in the index file, thus only the name is backed up, since that is enough to recreate the folder.

8.3.2 Backup process

When you create a first backup of a store, the backup tool will perform the following actions:

- create a list of all the folders and their contents of the store
- for all items found, write them to disk

Because it first creates a list of everything in the store, newly created items during the backup will not be seen and thus will not be backed up. Moved items will still be in the backup, but in the original location they were found in. Hard deleted items during the backup will not be backed up because they cannot be opened anymore.

When you start the backup again, it will find the previous backup and automatically start an incremental backup, and will perform the following actions:

- read the index file and create a tree of the previous backup
- create a list of all the folders and their contents of the store
- per container, find the items which are already backed up and did not change and remove those from the list
- remove the old index file
- backup the items left in the list, and append them to the data file

There are a few things to notice about this behavior of the backup tool. When the lists of the previous index and the current contents of the store are compared, it does this per matching container. This means that if the user moved items from one folder to another, they will not be found, thus will be backed up again because they will be marked as 'new' in the other folder they we're moved to.

If a message was changed by the user since the last backup, the item will have a new 'last modification date', and will be backed up again in it's totality since the backup would become unbearably slow if it would need to check all the properties of a message to see which property changed and which not. Overwriting the old message is also problematic because the new message may be bigger than the old, and it will not fit on the old space of the message.

Then when the actual backup process starts, it will first remove the old index. The index file will then be rebuild while the backup processes each message found in the list. The data file will be appended with the new data, keeping the old information which was still available and did not need to be stored again.

8.3.3 Restore process

In order to restore items from the zarafa-backup tool, use the zarafa-restore tool. To restore items or complete folders, find the corresponding restore key in the user.index.zbk file. This index file isn't humanly readable with a text editor. Instead, use the readable-index.pl perl script, which can be found in /usr/share/zarafa/zarafa-backup-helpers/. To identify items, use the container name field or the subject to find the items needed to be restored.

When the items are found, place the restore keys in a separated file, or give them as parameters to the zarafa-restore tool. If the restore key of a container is entered, the complete container with all its items will be restored on one level. If the subcontainers of the selected container need to be restored, add the -r parameter to the command. The following example restores the inbox with subcontainers from userA. The restore key AF000000 is found in the userA.index.zbk file and needs to be defined at the end of the command.

```
zarafa-restore -u userA -r -c userA.index.zbk AF000000
```

The -c parameter as a reference for the index file is not necessary when using an index file from the same user. For example, if using zarafa-restore -u userA, the zarafa-restore tool will automatically use the userA.index.zbk file when index.zbk is in the same directory as where the command is executed.

In the next example a file (keys.txt) containing multiple restore keys from multiple items and folders from user userA is used. Every restore key in the file needs to be separated with a new line.

```
zarafa-restore -u userA -r -i keys.txt
```

9 Zarafa System Statistics Monitor

9.1 Installation

First the software needs to be downloaded to a Linux server. That can be realised with the following command:

```
$ wget http://download.zarafa.com/zarafa/extra/zarafa-ssm-6.20-extra.tar.gz
```

If `wget` is not available, that software package needs to be installed, since the statistics monitor will use `wget` as well. After downloading the file, the statistics monitor can be installed with the following command:

```
$ tar xzpvf zarafa-ssm-6.20-extra.tar.gz -C /
```

The following additional packages need to be installed on your Linux server for the statistics monitor to correctly work:

- `sysstat`
- `wget`
- `mktemp`
- `coreutils`
- `zarafa`

To check whether the required software is installed, the following command is used:

For RPM based distributions (RHEL, SLES):

```
$ rpm -q sysstat wget mktemp coreutils zarafa
```

For DEB based distributions (Debian, Ubuntu):

```
$ dpkg -l sysstat wget mktemp coreutils zarafa
```

The `sysstat` software will monitor your system every 10 minutes, and log what has happened in the last 10 minutes. This includes, but not limited to, memory, disk, cpu and network usage. To enable this software, you will need to edit the file `/etc/default/sysstat`. Edit the `ENABLED` line to correctly enable activity logging of your server. After editing this file, start logging with the following command:

```
$ /etc/init.d/sysstat start
```

Now `zarafa-ssm` need to be enabled. For this, edit the file `/etc/default/zarafa-ssm`. In this file edit the `ENABLED` line too, set the value to 1. Now every sunday night, the statistical information of the server will be uploaded to a server of Zarafa. This data will be used to preventively find and problems or bottlenecks that might arise on your zarafa server.

9.2 Internal working of the System Statistics Monitor

The `zarafa-ssm` program will gather information about the server, along with the usage information that the `sysstat` program gathers. Currently the following information will be sent to a server of Zarafa:

- cpu type
- disk size and formatting information
- memory sizes
- hostname
- kernel version
- distribution version
- zarafa version

Also some information about the Zarafa installation is gathered. This information includes:

- Zarafa statistics, including cache usage and SQL usage.
- User information, including store sizes
- Company information (if running in hosted mode)

All this information is sent over a secure SSL connection.

In a later version, Zarafa will provide a webbased monitoring tool for this information. This feature will be available in the Professional and Enterprise editions.

10 Mobile push technology

This chapter describes how to configure the Z-Push software to synchronise PDA's and Smartphones with a server based solution.

Z-Push is available as an opensource project on Sourceforge - <http://z-push.sourceforge.net>

In this manual only the server part of Z-push is discussed, please see the User manual for configuring the Mobile clients.

10.1 Z-push introduction

The Z-Push software allows users with PDA's and Smartphones to synchronise their email, contacts, calendar items and tasks directly from a compatible server over UMTS, GPRS, WiFi or GSM data connections. The following devices are native supported by Z-Push:

- Windows Mobile 5 and 6
- Nokia E-series
- Sony Ericsson P990, W950 and M600
- All other ActiveSync compatible devices
- Apple Iphone

The devices can be synchronised because the Z-Push module emulates an MS Exchange server on the server side, allowing users to synchronise without installing specialised synchronisation software on their devices.

10.2 Security

The SSL feature of the PDA can only be used when SSL is configured on the server and the server has an acceptable certificate. This means that either an official SSL certificate from a commercial certificate authority is needed, or the certificate needs to be installed on the PDA. Installing SSL certificates is beyond the scope of this document, though many HOWTOs can be found on the internet.

10.3 Installation

Download the latest Z-Push software on the following website: <http://z-push.sourceforge.net>

To Install Z-Push, simply untar the Z-Push tar to the webroot with:

```
tar zxvf z-push-<version>.tar.gz -C /var/www
```

The -c option is the destination where the files need to be installed. In the following table the default webroot directories of where some distributions lets the Apache webserver search for files.

Distribution	Default webroot
SuSE	/srv/www/htdocs
RedHat Enterprise Linux	/var/www/html
Debian and Ubuntu	/var/www

Table 6: webroot directories

Make sure that the 'state' directory is writeable for the webserver process, so either change the owner of the 'state' directory to the UID of your apache process, or make the directory world writeable:

```
chmod 777 /var/www/z-push/state
```

Relaxing the permissions a bit is possible, and correct the user and/or group of the directory, so only Apache can write in the directory:

```
chmod 755 /var/www/z-push/state  
chown www-data:www-data /var/www/z-push/state
```

The user and group name of Apache will differ per Linux distribution. Below a table is shown with an overview of the correct username and groupname for Apache:

Distribution	Apache username	Groupname
SuSE	wwwrun	www
Red Hat Enterprise Linux	apache	apache
Debian and Ubuntu	www-data	www-data

Table 7: User and groupnames per distribution

Now, Apache must be configured to redirect the URL 'Microsoft-Server-ActiveSync' to the index.php file in the z-push directory. This can be done by adding the line to the httpd.conf file

```
Alias /Microsoft-Server-ActiveSync /var/www/z-push/index.php
```

Make sure that the line is added to the correct part of the Apache configuration, taking care of virtual hosts and other Apache configurations.

Note: You CAN NOT simply rename the Z-Push directory to Microsoft-Server-ActiveSync. This will cause Apache to send redirects to the PDA, which will definitely break your PDA synchronisation.

Lastly, make sure that PHP has the following settings:

```
php_flag magic_quotes_gpc off  
php_flag register_globals off  
php_flag magic_quotes_runtime off  
php_flag short_open_tag on
```

Set this in the `httpd.conf`, in `php.ini` or in an `.htaccess` file in the root of Z-Push. If not setup correctly, the PDA will not be able to login correctly via Z-Push.

After doing this, the PDA should be able to synchronise.

11 Multicompany configuration

This section will provide information regarding the multi-company functionality which was introduced in Zarafa 6.10.

Note: This is not the same as multi-domain, which should be handled in your SMTP server. The domain `bicycle.com` and `car.com` may actually be the same company. This feature is only needed to host different companies on the same server and the same database, but users from the different companies may not be able to "see" each other.

11.1 Support plugins

Multi-company support can only be enabled when using the DB or LDAP plugin, at this time it is not possible to use the Unix plugin. When using the DB plugin, the `zarafa-admin` tool can be used to manage companies, while with the LDAP plugin all information will come directly from LDAP or Active Directory.

11.2 Configuring the server

The following configuration options in `server.cfg` are provided for multi-company support.

- `enable_hosted_zarafa`

Enable multi-company environment. When set to true it is possible to create companies within the Zarafa instance and assign all users and groups to particular companies. When set to false, the normal single-company environment is created.

- `createcompany_script`

Location of the `createcompany` script which will be executed when a new company has been created.

- `deletecompany_script`

Location of the `deletecompany` script which will be executed when a company has been deleted.

- `loginname_format`

See section 'Configuring login name' for more details about this configuration option.

- `storename_format`

See section 'Configuring store name' for more details about this configuration option.

11.2.1 Enabling Multi company

To enable multi company support in Zarafa change the following configuration option in `server.cfg`:

```
enable_hosted_zarafa = yes
```

11.2.2 Configuring login name

The loginname of a user must be unique in order to correctly allow the login attempt. When enabling multi-company support in Zarafa, having a unique loginname can become difficult as the number of companies increases. It is easier when the loginname contains the companyname as well in order to only force loginnames to be unique within a single company.

The way the companyname is 'attached' to the username to create the loginname can be configured with the 'loginname_format' configuration option in server.cfg. This configuration option can contain the following variables:

- %u – The username
- %c – The companyname to which the user belongs

As separation character between the username and companyname a character should be chosen that does not appear inside the username or companyname itself. Valid characters for example are '@' and '\

Some example loginname format for a user named John Doe who is member of company Zarafa:

- "%u" → john
- "%u@%c" → john@zarafa
- "%c\%u" → \zarafa\john

Although having a loginname that contains a %c is mandatory for the DB plugin, it is optional for the LDAP plugin. Managing unique loginnames is easier in LDAP because it is possible to use the email address as the loginname attribute. See the LDAP configuration file for more information about the loginname attribute.

Note that all interactions with the zarafa-admin tool which requires the username, the formatted loginname is expected (thus including the companyname if the 'loginname_format' includes that variable).

11.2.3 Configuring store name

When relations between multiple companies are allowed, it is possible that users share their store with users from other companies. To easily differentiate stores from different companies, the store name can be formatted to contain the company name to which the user/store belongs.

In server.cfg the configuration option 'storename_format' is provided for exactly this purpose. In the format different variables are provided which can be used to different kinds of information.

- %u – The username
- %f – The fullname of the user
- %c – The companyname to which the user belongs

Some examples for a user named 'John Doe' who is member of the company 'Zarafa':

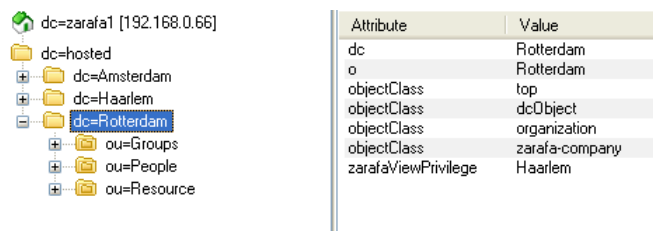
- "%u" → john
- "%f" → John Doe
- "%f (%c)" → John Doe (Zarafa)

11.2.4 Configuring the LDAP plugin

When using the DB plugin no additional configuration is required. For the LDAP plugin there are several configuration options that might require changes.

When using the LDAP plugin with multicompany, first organize the LDAP tree. Best practice is to have a

directory with domain components (dc) in the LDAP tree acting as the companies. Every company has organization units (ou), containing the users, groups and resources



Screenshot 7: Ldap tree MC environment

The next step is to change the `ldap.cfg` config file in `/etc/zarafa/`. An example config file for LDAP or AD can be found in `/usr/share/zarafa/example-config/`. Explanation about the general LDAP attributes for Zarafa can be found in chapter 5. The multicompartment related attributes are shown below.

```
ldap_company_unique_attribute
ldap_company_search_base
ldap_companyname_attribute
ldap_company_scope
ldap_company_view_attribute
ldap_company_view_attribute_type
ldap_company_view_relation_attribute
ldap_company_admin_attribute
ldap_company_admin_attribute_type
ldap_company_admin_relation_attribute
ldap_company_system_admin_attribute
ldap_company_system_admin_relation_attribute
```

The required attributes to point the Zarafa server to the companies with their users on the LDAP server are shown below. These values are for the LDAP tree in screenshot 7.

```
ldap_company_unique_attribute = dc
ldap_company_search_base = dc=hosted,dc=zarafa1
ldap_user_search_base = dc=hosted,dc=zarafa1
```

Test the settings by using `zarafa-admin -list-companies` and `zarafa-admin -l`

For creating policies between the companies in the LDAP plugin, use the multicompartment related attributes in the LDAP tree. To read company policies and settings, the `zarafa-admin` tool can still be used.

See the `zarafa-ldap.cfg` manpage for more detailed information about the multicompartment related LDAP attributes.

```
man zarafa-ldap.cfg
```

11.2.5 Public stores

Once the server has been correctly started, you can create stores. There are two type of stores: Private and public stores. There can only be one public store per company space.

When creating a company, the public store will be created simultaneously. If for some reason the public store for the specific company is not created, the public store can be created manually by issuing the following command:

```
$ /usr/bin/zarafa-admin -s -I <company>
```

Replace `<company>` with the name of the company for which the public store should be created. When

the -I option is not used, the public store will be created for a single-company environment (And will not be accessible when multi-company Zarafa is enabled).

The public store is the store every user can always open. After installation and configuration of the server, you have to add a public store first.

11.3 Managing company spaces

Note: Management of company spaces through zarafa-admin is only required when using the DB plugin. When the LDAP plugin is used, all administration can be done through the LDAP or Active Directory server.

To create a company space use the following command:

```
$ /usr/bin/zarafa-admin --create-company <companyname>
```

To delete a company space use the following command:

```
$ /usr/bin/zarafa-admin --delete-company <companyname>
```

To change a company space use the following command:

```
$ /usr/bin/zarafa-admin --set-company <companyname>
```

This command can be combined with the option -qw for setting the quota warning level for the specified company space.

To control the view privileges for company spaces the following commands can be used:

```
$ /usr/bin/zarafa-admin --add-view <viewer> -I <companyname>  
$ /usr/bin/zarafa-admin --del-view <viewer> -I <companyname>  
$ /usr/bin/zarafa-admin --list-view -I <companyname>
```

The <viewer> is the companyname which receives or loses permission to view company <companyname>.

```
$ /usr/bin/zarafa-admin --add-admin <admin> -I <companyname>  
$ /usr/bin/zarafa-admin --del-admin <admin> -I <companyname>  
$ /usr/bin/zarafa-admin --list-view -I <companyname>
```

The <admin> is the loginname of the user who receives or loses admin privileges over the company <companyname>.

11.4 Managing users and groups

When using the DB plugin users and groups should be created using the zarafa-admin tool. For details about using the zarafa-admin tool see 'man zarafa-admin'. The user- or groupname that should be given to the zarafa-admin tool depend on the 'loginname_format' configuration option.

For example, when 'loginname_format' is set to '%u@%c' creating a user for company 'zarafa' would be:

```
$ /usr/bin/zarafa-admin -c john@zarafa ...other options...
```

And creating a new group for company 'zarafa' would be:

```
$ /usr/bin/zarafa-admin -g group@zarafa ...other options...
```

11.5 Quota levels

When using a multi-company installation there are 2 types of quota, namely the quota for the company and the quota for the user. The quota for the company is checked over the total store size of all users within that company plus the public store.

At this time only the warning quota can be configured for a company, this means you cannot set the soft or hard quota to limit the companies email capabilities.

Just like the user quota, there are multiple levels for the company quota, and there is even a new level for the user quota. A summary of the possible quota levels which can be set in a multi-company environment:

1. Company quota:

a) *Global company quota*: Configured in `/etc/zarafa/server.cfg` and affects all companies within the system.

b) *Specific company quota*: The quota level for a company configured through the plugin (LDAP or zarafa-admin tool).

2. User quota:

a) *Global user quota*: This is configured in `/etc/zarafa/server.cfg` and affects all users from all companies

b) *Company user quota*: This is the default quota level for all users within the company, and is configured through the plugin at the company level

c) *Specific user quota*: This is the quota level for a specific user, and is configured through the plugin.

As mentioned above the "Global company quota" and "Global user quota" can be configured in the `/etc/zarafa/server.cfg` file, in there the options "quota_warn", "quota_soft" and "quota_hard" for the user quota, and the options "companyquota_warn" for the company quota.

To configure the "Specific company quota" the zarafa-admin tool can be used when using the DB plugin. The following command will set the various quota levels over the company:

```
$ zarafa-admin --update-company <company> --qo y --qw <warningquota>
```

To configure the "Specific user quota" the zarafa-admin tool can be used when using the DB plugin. The following command will set the various quota levels over the user:

```
$ zarafa-admin -u <user> --qo y --qh <hardquota> --qs <softquota> --qw <warningquota>
```

To configure the "Company user quota" the zarafa-admin tool can be used when using the DB plugin by using the '--update-company' argument. The following command will set the various user default quota levels over the company:

```
$ zarafa-admin --update-company <company> --udqo y --udqh <hardquota> --udqs <softquota> --udqw <warningquota>
```

When using the LDAP plugin, the attributes which control the quota levels can be configured in `/etc/zarafa/ldap.cfg`.

12 Multiserver configuration

This chapter will provide information regarding the multiserver functionality which was introduced in Zarafa 6.30

Note: In order to use this feature a valid Zarafa Enterprise license key is necessary and a running zarafa-licensed is required.

12.1 Introduction

The Zarafa multiserver feature enables you to distribute Zarafa over multiple servers. In this situation the Zarafa-user-stores are divided over several servers, but is still acting as one system. The users, groups and companies are managed in one LDAP or Active Directory server.

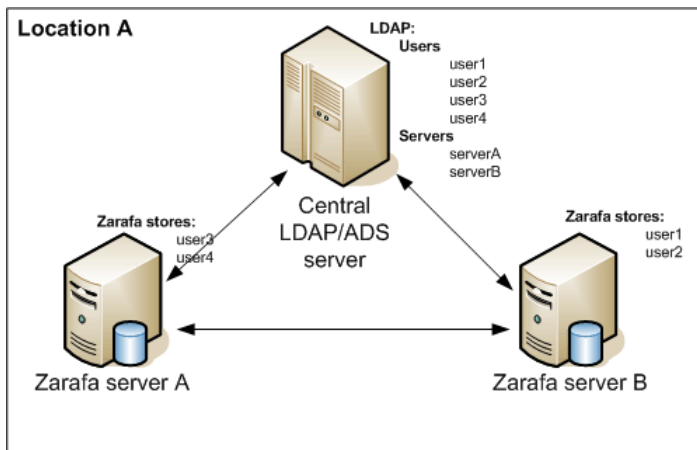


Figure 8: Multiserver environment in one location

This way, the server load can be divided over multiple servers and the system can provide more active users. Another advantage is creating more efficient user-server connections if you separate the mailboxes per geographical location as in figure 2.

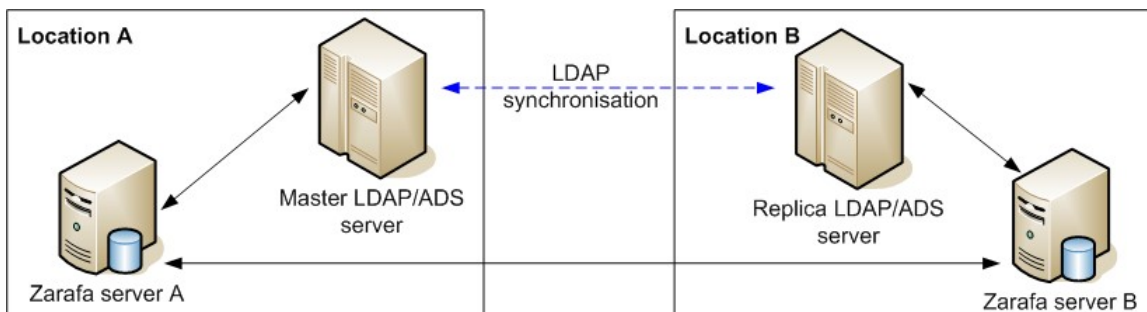


Figure 9: Multiserver environment on two locations

Also, for establishing a connection, the Zarafa users are not bound to the store location. Each user can use all servers to establish a connection.

The multiserver user data will be stored on the Zarafa server where the user is located. In this first multiserver version it is not possible to migrate automatically a user from one server to another server. This feature will be available in the 6.40 version.

12.2 Prepare/Setup the LDAP server for multiserver setup

The Zarafa multiserver version can only be used with an LDAP or ADS server at this time, so the first step for implementing a multiserver environment is preparing the LDAP server(s). The following steps are applicable for an existing Zarafa situation using the LDAP plugin.

1. Setup the ldap server using chapter 5 in this manual.

2. Add in the LDAP structure a folder for the Zarafa servers, containing the server attributes which can be found in the Zarafa schema. The attributes CN, ipHostNumber, ObjectClass and zarafaHttpPort are required. The CN attribute is the primary key for the server information and represents the servername which is defined per server in `/etc/zarafa/server.cfg`. The server name has to be unique, and must be able to translated using DNS or hosts.

3. In a non-hosted multiserver situation, only one server is allowed to host the public store. Therefore add the attribute `zarafaContainsPublic` to the server that host the public store and add the value 1 as in screenshot 8

4. In a multicompany situation, all companies in ldap needs to be updated with the `zarafaCompanyServer` attribute. Use the servername for the value.

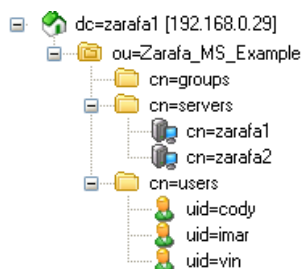
Name	Value
cn	ZdsMaster
objectClass	device
objectClass	ipHost
objectClass	zarafa-server
objectClass	top
zarafaContainsPublic	1
zarafaFilePath	/var/run/zarafa
zarafaHttpPort	236
zarafaSslPort	237
ipHostNumber	192.168.0.63

Screenshot 8: Ldap server attributes

5. The LDAP attributes for the Multiserver feature needs also to be verified in the `/etc/zarafa/ldap.cfg` file. An `ldap.cfg` example file with the Multiserver attributes included can be found in `/usr/share/zarafa/example-config/`. Probably you need to change the value from attribute `ldap_server_search_base` to your own LDAP location in file `/etc/zarafa/ldap.cfg`. Here are the ldap attributes for the Multiserver feature:

```
ldap_server_search_base = dc=servers,dc=zarafa1
ldap_user_server_attribute = zarafaUserServer
ldap_server_address_attribute = ipHostNumber
ldap_server_http_port_attribute = zarafaHttpPort
ldap_server_ssl_port_attribute = zarafaSslPort
ldap_server_file_path_attribute = zarafaFilePath
ldap_server_scope = sub
ldap_server_search_filter = (objectClass=ipHost)
ldap_server_unique_attribute = cn
```

6. Every Zarafa user on the LDAP server needs to be assigned to a Zarafa server. For this, add the attribute 'ZarafaUserServer' to every Zarafa user, using your LDAP admin client. Use the Zarafa server CN as value for the 'ZarafaUserServer' attribute.



12.3 Configuring the servers

The following configuration options in `server.cfg` are provided for Multiserver support.

•`enable_distributed_zarafa`

Enable Multiserver environment. When set to `true` it is possible to spread users and companies over multiple servers. When set to `false`, the single-server environment is created.

•`server_name`

The unique server name used to identify each server in the 'cluster'. This hostname should be correctly configured in your DNS.

To enable multiserver support in Zarafa change the following configuration options in `server.cfg`:

```
user_plugin = ldapms
enable_distributed_zarafa = yes
server_name = <servername>
server_ssl_enabled = yes
```

Note: For multiserver you need a clean database to start with. It is not possible to change an existing DB (single company or multicompany) to a multiserver environment.

Note: Each server needs to have a unique name. Make sure the `server_name` property in `server.cfg` matches the attribute "ZarafaUserServer" in LDAP

Note: If you are using an old `server.cfg` version, you can copy a new version containing also the Multiserver configuration options from `/usr/share/zarafa/example-config/`

12.4 Setup mail delivery

To order to deliver mail in a Multiserver environment, the delivery agent needs to open stores without having user credentials. Normally this is done through Unix-socket, but in a Multiserver environment this has to be done through an SSL authentication.

In order to do this, a public/private key pair needs to be created. Each server running the `dagent` will need its' own private key, and its' public key needs to be copied to the other servers' `sslkeys` directory. Please note that all keys need to be created with the same Client Authority (CA).

Create the SSL authentication using these steps:

1.First, create the directory which will contain the certificate and setup permissions on this folder.

```
mkdir /etc/zarafa/ssl
chmod 700 /etc/zarafa/ssl
```

2.If you run Zarafa as another user, as described in chapter 2.23.1. Do not forget to `chown` the directory as well.

3.Create a Certificate Authority. This CA will be used to create the server certificate and sign it. Use the `ssl-certificates.sh` script in the `/usr/share/zarafa` directory, which uses the `openssl` command and the `CA.pl` script. An easy way is to first copy the `ssl-certificates.sh` script to your SSL location and then execute it.

```
cp /usr/share/zarafa/ssl-certificates.sh /etc/zarafa/ssl/
cd /etc/zarafa/ssl/
sh ssl-certificates.sh server
```

4.The parameter `server` is added, so the name of the new certificate will be called `server.pem`. When the CA is not found in the default CA directory, it needs to be created. By pressing enter, the creation of the new CA is started.

5.Enter a password (passphrase) when asked for. This is the password you will use later to sign certificate requests. Then enter your certificate information. Give extra attention to the Common Name. This has to be the valid hostname.

6.Now the CA is created, it is possible to create self-signed certificates. The `SSL-certificates.sh` script will automatically continue with this step. Enter a password for the request, and enter the certificate details. Some details need to be different from those you typed when creating the CA. At least the field `Organizational Unit Name` needs to be different. The challenge password at the end may be left empty. This step created a Certificate Request, that needs to be signed by the CA that was created in the first step of the script. Type the password of the CA again when asked for. The details of the certificate will be shown, and asked for acceptance. Accept the certificate.

7.The last step in this script, you will be asked if you want the public key of this certificate. Since you created the server certificate, you do not need the public key of this certificate.

8.After completing the `ssl-certificates.sh` script, there are generated two files in the `/etc/zarafa/ssl` directory: `server.pem` and `server-public.pem`. Also you need the `cacert.pem` file, which can be found in the default SSL directory (`/etc/CA` in RHEL distributions). It is recommended to copy this file to the `/etc/zarafa/ssl` directory. These file locations need also to be defined in the `/etc/zarafa/server.cfg` file, using the following configuration options:

```
server_ssl_ca_file = /etc/zarafa/ssl/demoCA/cacert.pem
server_ssl_key_file = /etc/zarafa/ssl/server.pem
server_ssl_key_pass = <ssl-password>
```

9.To use another Zarafa server as central MTA, change the `server_socket` configuration value in the `/etc/zarafa/dagent.cfg` file. Use the just created SSL connection for an secure connection in order to use the SSL authentication. Change the to server socket to:

```
server_socket = https://<server-IP>:237/zarafa
```

And add the following lines to the file:

```
sslkey_file = /etc/zarafa/ssl/server.pem
sslkey_pass = <ssl-client-password>
```

10.Restart the `zarafa-server` process, and you are now able to directly connect with the SSL port.

Note: Make sure in `/etc/postfix/master.cf` you add the `dagent.cfg` argument to the postfix pipe command.

13 Appendix A; Backup index details

Every line in the indexfile represents one item. An item can represent either a container (C), or a message (M). There are 2 extra lines at the beginning of the index. It starts with V:2. This indicates the version of the index file, currently version 2.

The second line is always the root container, indicated by the letter R. The format for this line is:

```
R <self <length> <entryid>> <username>
```

where <self> is the identifier for the root container. This number can be present in the parent field for C and M entries. The self data is first represented by the data length, and then the data itself. The last field in this line is the username the backup was made for.

The format for a container is as follows:

```
C <parent <length <entryid>> <self <length> <entryid>> <rc <length> <hex>> <#msg>
<lmt> <ipc-type> <dn>
```

The format for a message is as follows:

```
M <parent <length <entryid>> <rc <lengths> <bin>> <lmt> <ipc-type> <filepos>
<subject>
```

Container details in the index file, sorted on field number:

1.C

Indicates the following information denotes a container.

2.parent key string length

The length in characters of the following *parent key*.

3.parent key

The key of the parent.

4.own key string length

The length in characters of the following *key*.

5.own key

The key of this container. Other containers and items may have this in the *parent key* field.

6.restore key length

The length in characters of the following *restore key*.

7.restore key

Use this key with the restore tool to restore this folder and its contents.

8.number of items in this container

An indicator of how many items there are in this folder.

9.last modification timestamp

The date of when this container was last modified, (eg its name changed).

10.container type

The MAPI type of the container.

11.container name

The name of the container.

The message details in the index file, sorted on field number:

1.M

Indicates the following information denotes a message.

2.parent key string length

The length in characters of the following *parent key*.

3.parent key

The key of the parent container

4.restore key length

The length in characters of the following *restore key*.

5.restore key

Use this key with the restore tool to restore this message.

6.last modification timestamp

The date of when this message was last modified, (eg the date of a calendar item because it was moved).

7.message type

The MAPI type of the message (eg mail, calendar item, contact, etc.)

8.data offset

The offset into the data file where the message begins.

9.subject

This is the subject of the message.

14 Appendix B; Pre-6.2x upgrade strategies

14.1 Database upgrades from 4.1 or 4.2

Before you can start Zarafa again, you must upgrade the database. There are several scripts required, depending on which version you are upgrading from. Upgrade scripts are only needed when you are upgrading from a 5.0x version or older. The scripts are as follows:

`db-convert-4.1-to-4.2`

This perl script upgrades your database from 4.1 to the 4.20 format. These are changes that regard how users are stored in the database. This script is required, and should be run as follows:

```
$ perl /usr/share/zarafa/db-convert-4.1-to-4.2 \  
  <dbuser> <dbpass> <dbname>
```

Replace <dbuser> with the username you use to connect to the database with. Replace <dbpass> with the password of the database user. If there is no password, enter 2 ' ' single quotes here. Replace <dbname> with the name of the Zarafa database. This will result in something like:

```
$ perl /usr/share/zarafa/db-convert-4.1-to-4.2 root '' zarafa
```

`db-convert-4.20-to-4.21`

This perl script upgrades your database from 4.20 to the 4.21 format. It will replace an indexing key to improve database speed. This script is highly recommended, and should be run as explained for the `db-convert-4.1-to-4.2` script.

Depending on the size of your database and the speed of your system, this script might take a while, but it will probably complete within 10 to 30 minutes.

`db-convert-4.20-to-innodb.sql`

This SQL script converts your converted 4.20 database to InnoDB format. Installations that started at version 4.0 created MyISAM tables. However, the current SQL database layout is optimized for the InnoDB format. Therefore, converting your MyISAM database to InnoDB will result in a huge speed increase. Also, the InnoDB format is less error prone and has less overall table locking. It is highly recommended to convert your database to InnoDB. On the MySQL prompt, import the script:

```
mysql> source /usr/share/zarafa/db-convert-4.20-to-innodb.sql
```

Depending on the size of your database and the speed of your system, this script will take a long to very long time. Reserve up to 8 hours of time for this conversion to complete for a database with several gigabytes of data. If you optimize the MySQL memory settings before you start this script, it will run much faster.

```
db-convert-4.2x-to-5.00
```

This perl script upgrades your database from 4.2x to the 5.0 format. This script calculates and adds a store column to the properties table. This makes the table sorted on the disk, increasing data throughput. Execute this script as described for the `db-convert-4.1-to-4.2` script.

Depending on the size of your database and the speed of your system, this script might take a while, but it will probably complete within 10 to 30 minutes on a fast machine.

Note: It is advisable to start this script with `screen`, so this script can continue in the background.

14.2 Upgrades from 5.0 to 5.1x and up

The Zarafa 5.10 server can upgrade the database itself. It can do this from the database version which is needed in 5.0. If you upgrade from 4.x installations to 5.10 or higher, you first need to upgrade the database yourself with the scripts described above to the 5.0 format. You can then start the 5.10 server which will finalize the upgrade from 5.0 to 5.10 itself.

Later versions of Zarafa can always upgrade from a 5.0 database format or newer.

14.3 Important changes since 4.x and 5.x

A configuration option in the `server.cfg` has been changed since 4.20. The option `server_name` has been renamed to `server_bind`. A configuration file with typing errors in the option names or non-existing options will render a service inoperable, and it will not start. All the errors found in the configuration file will be printed.

For the 5.0 version some unused options have been removed from the server configuration. SQLite support was removed, so the option `internal_path` was also removed. If this option is in your `server.cfg` file, please remove this line before starting the `zarafa-server` process.

Options not set in a configuration file will keep their default value. Default values can be found in the example configuration file found in `/usr/share/zarafa`. Alternatively you can read the specific manual page for the service:

```
$ man zarafa-[service].cfg
```

The Zarafa services did not daemonise in versions before 5.0. However, versions 5.0 and newer do daemonise, and run in the background. To revert this behavior, use the `-F` switch of a service to keep it running in the foreground.

Other configuration changes are found in the gateway. The defaults for the `ssl_private_file_key` and `ssl_certificate_file` have been changed. The default directory is now

/etc/zarafa/gateway/, to distinguish it from the service ssl files.