



Zarafa



Zarafa is a workgroup sharing solution based on the look-and-feel of Microsoft Outlook, which enables sharing of mail and appointments from Outlook and a web based interface.

The solution is unique because it is available both on Linux and Windows. It is the only Outlook sharing solution world wide storing the data in an open SQL database. Also, the webaccess works in different browsers, including Internet Explorer and Mozilla Firefox. This document focuses on the technical aspects of the solution.

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 a SQL database.
- *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 e-mail 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 administration tool is used to manage users, user information and groups.
- *Zarafa gateway*
Optional service to provide POP3 and IMAP to Zarafa users.
- *Zarafa monitor*
Monitor service which monitors user stores for quota exceeds.
- *Zarafa caldav*
Optional service that currently only provides ical support, but also CalDAV in the near future.
- *Zarafa backup*
A brick-level backup tool to create simple backups of stores.
- *Apache*
Serves web pages of the webaccess to the users browser.
- *PHP*
The webaccess is written in this programming language.
- *PHP MAPI extension*
Plug-in for PHP to enable use of the MAPI layer. Through this plug-in, MAPI functions are made accessible for PHP developers. This effectively means that MAPI web clients can be written. The webaccess is such a client.

This document clarifies the relation between these parts and how they cooperate.

Server

The Zarafa server program stores all the data in a SQL database. It also maintains the connections with the clients like Outlook and the webaccess. Therefore, clients never directly access the database, and need to ask the Zarafa server for the correct 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. This is independent of the users present in the Operating System. The MTA must be reconfigured to deliver the e-mail for a specific e-mail address to a specific Zarafa user. In the near future Zarafa will be able to accept already existing e-mail addresses and users, minimizing the installation effort.

The server opens an TCP port (default port 236) to accept connections from clients. The server also opens multiple connections to the SQL database.

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 e-mail is such an event. As soon as the e-mail is in the users 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.

Configuration

The server can be configured through a configuration file. The file will overwrite default settings. Common changeable settings are which database to use, and how to authenticate to the database server. Other settings are on which port to listen for incoming connections and how to log errors.

Logging

Log messages of the server can be greatly 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. For Windows, '*eventlog*' makes the logmessages go the Windows Eventlog.
- `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 5 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 in the Zarafa spooler is configured in a same manner as the server.

Soft Deletes

Zarafa uses a Soft Delete system.

When a user deletes e-mails, 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.

To purge these deleted items from the database, the lifetime of those items needs to be defined.

Configuration

Soft Deletes always remain in the database, until they are purged. When a 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.

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 your Zarafa server as a separate server in your 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 you connect over the Internet. No one will be able to read your traffic between the client and the server.

At this moment, there are two clients: a Microsoft Outlook client, which works with version 2000 and up. The second client is the webaccess. Even if you don't have access to your own computer with Outlook, you may view your mail, calendar and other information. The user can even access public stores and other peoples folders from the webaccess.

Delivery agent

The delivery agent (zarafa-dagent) is a program that delivers incoming e-mail to Zarafa users. The input of the dagent is a normal Internet e-mail. After the dagent has a connection to the server, it converts the e-mail and saves this in the standard Inbox of the user.

It depends on your set-up how the dagent needs to be called. It 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 e-mail to. The only mandatory parameter is the username to deliver to. This can be configured in the MTA. As an example for postfix, change the *mailbox_command* in the *main.cf* file as follows:

```
mailbox_command = /usr/bin/zarafa-dagent "$USER"
```

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

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

Junk e-mail

The dagent will normally deliver new e-mail to the default Inbox. When you have a spam filter that only marks the e-mail as spam, it is possible to make the Zarafa dagent deliver the spam mail directly in the 'Junk E-mail' folder. This can be done by using the *-j* option:

```
zarafa-dagent -j username
```

The Junk E-mail folder is generated by default in Zarafa. To move spam e-mail in this folder, you can use procmail. The following lines show the procmail configuration.

```
:0
* ^X-Spam-Status: Yes
| zarafa-dagent -j $USER

:0
| zarafa-dagent $USER
```

When a header 'X-Spam-Status: Yes' is found in the e-mail, procmail will start the zarafa-dagent program with the *-j* option. Otherwise the e-mail is normally delivered to the Inbox.

When the Junk E-mail folder was unavailable, the e-mail will again be delivered to the normal Inbox of the user.

Temporary delivery failures

If the dagent fails to delivery the e-mail to the server, it can return different exit codes to notify the SMTP server what has happend in 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 nolonger 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.

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 e-mail. An e-mail will be sent to every incoming message containing the given subject and message. The service can be enabled in the webaccess on the 'settings' page.

To configure your server to send these auto-reply messages, install a program or script called `zarafa-autorespond`. The location of the program can be overwritten by using the `-a` option of the `dagent`:

```
zarafa-dagent -a [autoresponder] username
```

[autoresponder] should be replaced by the location and program name of your auto-reply program. When a user enabled the auto-reply service, the `dagent` will execute the program with the following options:

```
</path/to/autoresponder> <from> <to> <subject> <username>  
<messagefile>
```

These arguments can be used in the program. The message file is a simple Internet e-mail message, containing To, From and Subject headers. One special header line is added called 'X-Zarafa-Vacation'. When this header is present, the Zarafa `dagent` will not send an reply, avoiding a loop between two users.

An example of a possible `autorespond` script is added in the Zarafa installation.

Spooler

The Zarafa spooler sends e-mail from the global outgoingqueue to a SMTP server, which sends the e-mail to the correct address.

When an e-mail message is sent from Outlook or the webaccess, the message is placed in the Outbox folder. The client asks the server to send this message. If the server responds with an 'OK' signal, the client moves the message directly to the sent-items folder. At this point, the server notifies the spooler process that an e-mail is waiting to be processed. The spooler will now start to convert the message to a normal internet e-mail message. When the conversion is complete, a connection to the supplied SMTP server is created, and the e-mail is send to the SMTP server.

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.

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 e-mail 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.

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.

To create users in the default way, the `zarafa-admin` tool is used as follows:

```
zarafa-admin -c username -p password -e username@domain.com \  
-f "Full username" -a yes -n no
```

The username is the name which the user uses to logon with the password. You can also use the `-P` switch to let the tool ask for a password, so it is not seen on the screen. The email address is used on outgoing emails. Note the quotes when setting the fullname as it needs to be one parameter. In this example, the user is created as `admin`, and not as `inactive`.

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 e-mail in Peter's Inbox, because all users are a member of the group *Everyone*.

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

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*.

Quota and quota monitoring

Users can collect a lot of email, while your disk space is limited. With this feature you can set quotas server-wide and specific quotas for users. 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.

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.

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 can never be reached.

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.

Quota monitoring

The zarafa-monitor keeps an eye on the sizes of the stores. When a user is over his quota, the monitor will send the user a warning email explaining that it's over quota and will need to remove information from its store.

Global Addressbook

The Global Addressbook of outlook returns a list of users and groups in the server. Using this addressbook, you can easely send emails to your co-workers. The addressbook works automatically when you create you Outlook profile. When you upgrade from a previous version, you will need to create a new outlook profile.

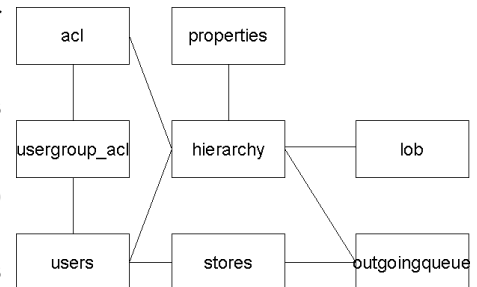
Database

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 (e-mail 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 e-mails and pictures of contacts are stored in the *lob* (large objects) table.

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 placed on any folder.

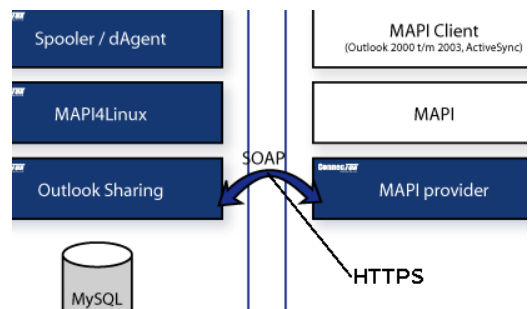
When an e-mail is sent, the server will create a link to the message in the *outgoingqueue* table. The spooler receives a notification when the contents of this table is changed, and immediately starts to send the e-mail message. When the spooler is finished, the link will be removed from the *outgoingqueue*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



Secure HTTP

The Zarafa client has the possibility to create an SSL connection to the server, using HTTPS. 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.

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, linux Zarafa clients can also connect to the SSL connection of the server, and authenticate using their certificate. 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.



Outlook connections can use SSL too, but the profile must still be created for a user with it's password to correctly login.

Caching

All the data of the Zarafa server is stored in a SQL database. Every time an item is created, like an e-mail, 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 minimize 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.

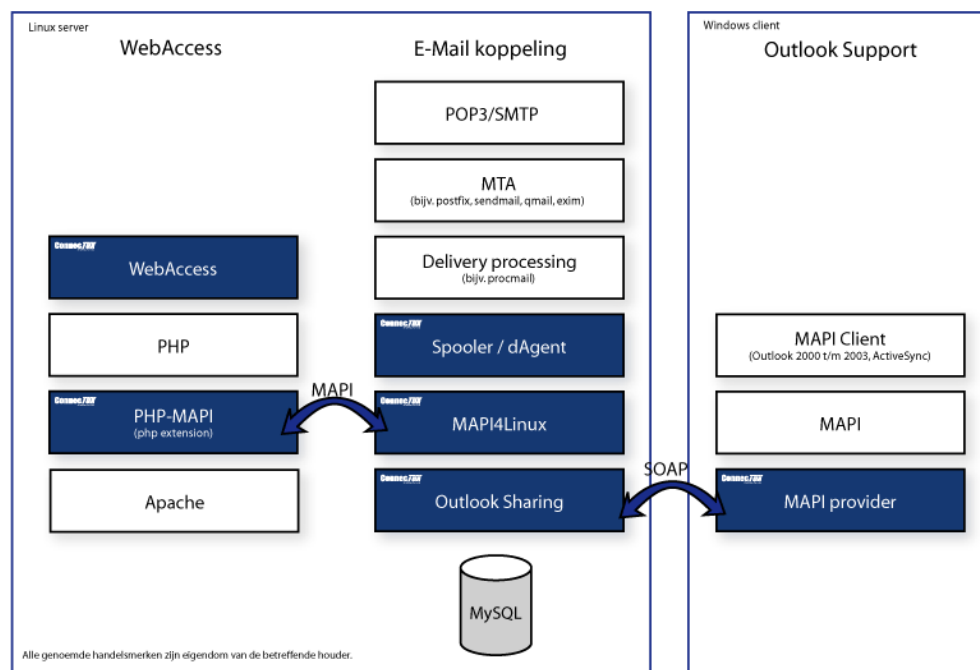
Configuration

Since internal memory of the server is limited, the size of the cache should be limited as well. The total size of the cache can be configured in the configuration file of the server.

```
cache_sortkey_size = 16777216
cache_cell_size = 16777216
```

The value is set in bytes. An optimal value is around 1 Mb per user.

Connections and protocols



All applications which connect to the Zarafa server, use the MAPI layer. This Application Programming Interface (API) is implemented in the MAPI provider. 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 the 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.

SOAP

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

SOAP is based on XML en 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.

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.

Tuning Zarafa Performance

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.

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.

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.

Multi-server setups

By splitting the load over different servers, you can achieve better performance because you are basically load-balancing the server chain over multiple servers. Although Zarafa itself does currently not support failover clustering of the main server process, there are many possibilities in which you can use multiple servers to distribute the load over multiple servers.

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:

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. If you want more than 3Gb, you must use a 64 bit system, a 64 bit Linux OS, and a 64 bit Zarafa package.

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

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

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.

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.

Memory Usage

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

- Zarafa's sortkey cache (caches the sort keys for tables)
- 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:

- `sortkey_cache`: 1MB per concurrent user
- `cell_cache`: around 25% of total RAM size
- `innodb_buffer_size`: around 25% of total RAM size
- `mysql_query_cache`: a few MB

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.

Zarafa's SortKey cache (`cache_sortkey_size`)

The Zarafa sortkey cache is used to cache the sorting keys in tables. This means that when you sort a table 'by date', the dates of the items in that view can be retrieved from a cache, instead of from the SQL engine. This makes loading table much quicker, as a normal table load requires the reading of all 'sorted' columns, and for the other columns, only the rows that are in the current view. As the sortkey cache doesn't use vast amounts of memory, a setting of around 1MB per logged-on user should be fine.

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.

MySQL innodb_buffer_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)

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.

Note:

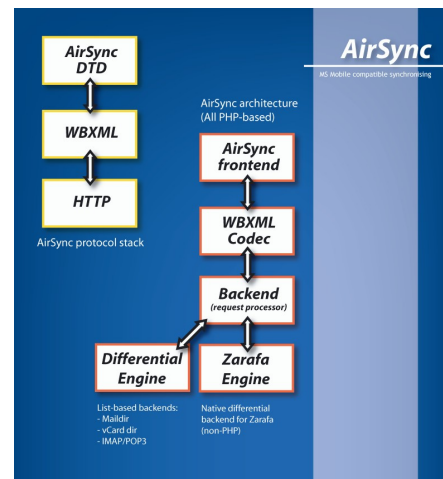
For larger databases we advise to set the innodb_log_file_size to ¼ of the innodb_buffer_size.

Mobile devices

In april 2007 Zarafa will release the opensource project "PHP-Airsync". The PHP-AirSync is an implementation of Microsoft's ActiveSync protocol which is used 'over-the-air' for Microsoft Windows Mobile 2002, 2003 and 5.0, named 'AirSync'. PHP-AirSync enables any PHP-based groupware application to become fully sync'able with any ActiveSync-compliant device.

By using PHP-Airsync you don't need to install any additional plugins on your Windows Mobile device, like Funambol or another SyncML client.

PHP-Airsync for Zarafa has full support for push-mail and synchronisation of calendars and contacts.



Appendix A. E-mail rules

System requirements

Software that must have been installed on the server system:

- Zarafa 4.20 or later

Software that must be installed on the client system:

- Outlook XP or later

Outlook 2000 can not be used at the moment. It is unknown if Outlook 2000 will be supported in the future.

Rules

When you want to organize your mail, rules are a very handy tool. Rules can perform actions on a newly delivered e-mail like moving the mail to another folder or even another store. With Zarafa, these actions may be performed by the server. This way, even when you're not logged in with Outlook, the rules will still be applied on the e-mails.

However, there are some restrictions and shortcomings when creating rules. In general, the following rules can be created which will be performed by the server:

- copy or move mail to another folder
- delete mail

All other actions are not supported by the server, and will not be executed. Amongst these actions are:

- permanently delete mail
- mark email in any way, e.g. mark as read, mark with urgency, add to category
- reply or forward mail
- any client action, e.g. print, start application, play a sound

Rules with client actions can still be used, but the user has to press the 'Run rules now' button in Outlook.

Outlook XP/2002 notes

Rules created with Outlook XP/2002 should work once created.

Do not edit rules created with Outlook 2003. This will break all the created rules beyond repair.

Outlook 2003 notes

Rules created with Outlook 2003 do not work right after creation. Outlook adds an option to the rule named 'On this machine only'. Edit the rule and remove this option. Press Apply and the rule should now work correctly.

Rules created with Outlook XP/2002 will be reformatted to Outlook 2003 format. This upgrade path is one way only. Do not use Outlook XP/2002 here on after.

Appendix B. ZarafaLiveSync

ZarafaLiveSync is a generic MAPI synchronisation utility that is able to synchronise two MAPI stores. ZarafaLiveSync has the following main features:

- Generic MAPI sync'er, so it works with any MAPI-compliant store
- Synchronises incrementally (ie only changes are uploaded)
- Can Synchronise 'Live', so changes are uploaded on-the-fly

How ZarafaLiveSync works

After setting up which profiles to synchronise, ZarafaLiveSync starts a synchronisation cycle on the default store of both profiles, which works as follows:

1. A *dumpfile* is opened in common files\application data\This dumpfile contains the state of both stores when the sync'er was last run. This dumpfile contains entryid's, modification times and other data so that the syncer can detect any

changes.

2. All the items, messages and folders, are listed in both default stores, along with the same properties that were already in the dump.
This is a standard set of properties that is different for the various types of MAPI objects.
3. A compare is done between store1 and the dump, and a list of changes is generated.
4. A compare is done between store2 and the dump, and a list of changes is generated.
5. Conflicts that can be automatically resolved are removed (ie object deleted in both stores).
6. The differences are collated into one list, and conflicts are resolved.
When an object has been changed in both stores is an example of a conflict. This basically means choosing which objects override which. Currently, both the commandline syncer and the GUI simply give the first profile precedence.
7. The changes are duplicated in the opposite stores.
Thus, an object added in store1 is added in store 2. This is done in the following order:
 - Add folders
 - Modify or rename folders
 - Delete folders

 - Add messages
 - Modify messages
 - Delete messages

The ordering within this list is by EntryID. This results in a rather random ordering of objects for the user.
8. The dumpfile with updated message id's and properties is saved.

Store 1 and Store 2 now contain the same data.

Running ZarafaLiveSync for the first time

When you run ZarafaLiveSync for the first time on two profiles, the dumpfile will be missing. When this happens, ZarafaLiveSync couples the default folders in your store (inbox, outbox, sent items, etc) by ID (not by name) so that you can sync different-language stores without problems.

After this coupling is done, a standard sync is started (see above) at which time all messages and folders are detected as being 'new', and through the standard system the complete stores are copied to each other.

If you wish to reset a sync, simply delete the .dat file from the application data directory, after which a complete (initial) sync will be done when ZarafaLiveSync is next started.

ZarafaLiveSync 'Live' option

When the 'Live' option is activated in ZarafaLiveSync, an extra feed of changes is also started; A notification callback for all changes is requested on each store, and when a change is received from the MAPI system, this change is automatically propagated to the other store. Also, the dump file is updated to reflect these changes. This makes it possible to keep two stores (that support notification) to be kept in sync through-out a session.

Appendix C. 64bit versions

For the following distributions are AMD64/Intel EM64T packages available:

-
- Fedora Core 5
 - Redhat Enterprise 4 / Redhat Enterprise 5
 - Suse Linux Enterprise 10 / OpenSuse 10.1
 - Debian Etch
 - Ubuntu LTS 6.06

These packages run on native x86_64 architectures. There is currently no support for Intel Itanium ia64 architectures.