ZCP 7.0 (build 27578)

Zarafa Collaboration Platform

Zarafa Archiver Manual

ZCP 7.0 (build 27578) Zarafa Collaboration Platform
Zarafa Archiver Manual
Edition 1.0

Copyright © 2011 Zarafa BV.

The text of and illustrations in this document are licensed by Zarafa BV under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at *the creativecommons.org website*[1]. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

All trademarks are the property of their respective owners.

Disclaimer: Although all documentation is written and compiled with care, Zarafa is not responsible for direct actions or consequences derived from using this documentation, including unclear instructions or missing information not contained in these documents.

The Zarafa Archiver provides an integrated archiving solution for Zarafa installations.

[1] http://creativecommons.org/licenses/by-sa/3.0/

Introduction

The Zarafa Archiver product provides a way to minimize mailbox sizes by moving older messages to *slower* and thus *cheaper* storage. The slow storage consists of one or more additional Zarafa servers who’s sole task is to store archived messages.

The archive Zarafa servers have exactly the same storage architecture as a normal Zarafa server, all mapi properties are stored in a MySQL database and all attachments are stored compressed on disk.

Every user that should get an archive will be coupled to an archive mailbox, which is located on the archive server.

Once a message is archived, it can be deleted from the original store. Optionally a stub to the archive can be created that allows a user to see the archived message and open it as if it were a normal message.

An alternate way of opening archived messages is to open the archive as a shared store in Outlook or the WebAccess.

The Zarafa Archiver uses the Zarafa’s multi-server technology to access archive stores in a seamless way. Nonetheless, the Zarafa Archiver can be used in a single-server setup with limited functionality.

Zarafa Archiver is an additional product and is not a default component of the Zarafa Collaboration Platform. Subscriptions of Zarafa Archiver can only be used with the Zarafa Professional or Enterprise edition.

Chapter 2.

Conventions

Before starting to install and deploy the Zarafa Archiver it's strongly advised to read this chapter to understand the different terminology.

- **Primary Server**

  The primary server is the server with the best performance and best IO subsystem, that contains the stores on which users normally work.

  > **Note**
  >
  > Although the term **Primary Server** suggests that there's only one primary server, multiple primary servers can exist in a multi-server environment. In this document no distinction will be made between a single-server or multi-server environment unless explicitly stated.

- **Archive Server**

  The archive server is the server, with normally a slower IO subsystem, that contains the archives for the stores that reside on the primary server.

  > **Note**
  >
  > An archive server is just another zarafa-server with the sole purpose of providing storage for one or more archive stores. In a multi-server environment this server will be just another node in the cluster.

  > **Note**
  >
  > Unlike primary servers, there's no need for a multi-server environment to have a multi-archive server setup.

- **Primary Store**

  The primary store is the store that resides on the primary server and on which a user normally works.

- **Archive Store**

  The archive store is the store that resides on the archive server and which is used for storing the archived messages from the primary store.

- **zarafa-archiver**

  The archiver is the application to manage the archiving. Basically it can be used to attach primary stores to archive stores and execute archive runs. It can be installed on any Zarafa server to connect to the primary or archive server using SSL authentication. It can also be used on a single server, using `zarafa-server`'s unix socket.

- **Stubbed Message**

  A stubbed message is a message in the primary store that acts as a placeholder for the archived message. These messages occupy virtually no space, but allow a user to see that a message was once there. On top of that it acts as an entry point to the archived version of that message.

- **Archive Configuration**

  An archive store can be configured in two ways, one-to-one and one-to-many. This is not a system wide configuration and can be setup for each archive independently.

  This allows for hybrid systems where N users with small to medium stores can be placed on M archive stores (where M is significantly smaller than N) and users with big to huge stores can be placed on dedicated archive stores.

- **One-to-One Configuration**

  In a one-to-one configuration one archive store is attached to one primary store.

  The advantage of this configuration is that it’s faster as the archive store itself is kept smaller.

  The disadvantage is that for each user an additional non-active user needs to be created (since there’s a one-to-one mapping between stores and users in Zarafa).

- **One-to-Many Configuration**

  In a one-to-many configuration one archive store is attached to multiple primary stores. For each attached primary store a folder is created in the archive store that will act as the root of the archive for that particular primary store.

  The advantage of this configuration is that less additional non-active users are required.

  The disadvantage is that the archive will become slower if the total amount of archive data in it grows.

Archiving

The archiving operation is split into four distinct stages:

1. **Copying** - In this stage all messages that are eligible for archiving are copied from the primary store to the archive store. When a copy of a message is made, a link to this copy is placed in the original message.

2. **Stubbing** - In this stage all messages that are eligible for stubbing are stubbed. A stub is defined as the original message with the body and attachments removed. A message is eligible for stubbing when it reaches the specified minimum age AND archived copies are present. So a message is never stubbed if it's not yet copied to the archive store.

3. **Deleting** - In this stage all messages that are eligible for deletion are deleted from the primary store. A message is eligible for deletion when it reaches the specified minimum age AND archived copies are present. So a message is never deleted from the primary store if it's not yet copied to the archive store.

4. **Purging** - In this stage messages that reached a specific age will be deleted completely from the archive store. In this case messages will not be available on both the primary and archive server.

Installing

4.1. Requirements

To deploy the Zarafa Archiver at least two servers are required: one primary server and one archive server. When an Zarafa Archiver subscription is available these servers can be configured in a multi-server setup. A multi-server setup is required to use the stubbing feature. For more information about creating a multi-server setup, please read the Administrator Manual.

Note

A multi-server setup requires a central LDAP or Active Directory. It's not possible to use multi-server with the **`DB`** or **`unix`** user plugin.

Alternatively one primary server and an archive server can be setup in a individual setup. In this configuration there will be no support for archive stubs, so when a user wants to open the archived messages, the user has login seperately to the archive server.

4.1.1. Software Requirements

- ZCP 6.40.8+ or ZCP 7.00RC1
- Multi-server Zarafa setup
- Valid ZCP Professional or Enterprise subscription
- Archiver subscription

4.2. Installation

4.2.1. Default installation location

The zarafa-archiver package can be installed on any node in the multi-server setup. However, the recommended node to install the package on is the node that acts as the archive server.

This makes sure that the **`zarafa-archiver`** always has a fast local connection to the archive server.

4.2.2. Enhanced installation location

In an environment where multiple archive servers exist and performance is crucial, an alternative approach can be taken. In such an environment the zarafa-archiver package can be installed on all primary servers and configured to only archive stores that exist on the server on which the **`zarafa-archiver`** is running. This can be done by passing the **`--local-only`** argument to **`zarafa-archiver`** when starting an archive run.

4.2.2.1. RPM based distributions

Use the following command to install the **`zarafa-archiver`** packages on RPM based distributions:

```
rpm -Uvh zarafa-archiver_<version>_<platform>.rpm
```

Replace **`<version>`** with the correct version and **`<platform>`** with the required target platform (**`i386`**, **`i586`**, **`ia64`**, **`x86_64`**).

4.2.2.2. DEB based distributions

On Debian based distributions use:

```
dpkg -i zarafa-archiver_<version>_<platform>.deb
```

Replace **`<version>`** with the correct version and **`<platform>`** with the required target platform (**`i386`**, **`ia64`**, **`x86_64`**).

4.2.3. Activate archiver subscription

To use the archiver subscription on every node archiver subscription has to be placed in the **`/etc/zarafa/license`** directory. Execute the following commands on every node to use the archive subscription:

```
echo 'Archiver license code' > /etc/zarafa/license/archiverbase
/etc/init.d/zarafa-licensed restart
```

Additional Archiver CALs can be added in the **`/etc/zarafa/license`** directory, like normal ZCP CALs.

Configuration

The Zarafa-archiver package will install the actual archive controller binary and the archive configuration file in **`/etc/zarafa/archive.cfg`**. *Table 5.1, “Configuration Options”* lists all available configuration options for the Zarafa Archiver.

Table 5.1. Configuration Options

Option	Default value	Description
server_socket	*file:///var/run/zarafa*	Unix socket to find the connection to the Zarafa Server.
sslkey_file	*not set*	Use this file as key to logon to the server. This is only used when server_socket is set to an HTTPS transport. See the zarafa-server(1) manual page on how to setup SSL keys.
sslkey_pass	*not set*	The password of the SSL key file that is set in sslkey_file.
archive_enable	*yes*	Specifies if the archive step will be performed. The archive step is basically the copy to the archive.
archive_after	*30*	Specifies the age in days of a message before it will be archived.
stub_enable	*no*	Specifies if the stub step will be performed. Stubbing is the process of removing the data from a message and only leaving a reference to one or more archived versions of that message. When such a message is opened it will be de-stubbed on the fly, leaving the user with a normal message to work with.
stub_unread	*no*	Specifies if unread messages will be stubbed.
stub_after	*0*	Specifies the age in days of a message before it will be stubbed.
delete_enable	*no*	Specifies if the delete step will be performed. The delete step deletes messages from the users store if they’re old enough. This way even more space is freed.
delete_unread	*no*	Specifies if unread messages will be deleted.
delete_after	*0*	Specifies the age in days of a message before it will be deleted.
purge_enable	*no*	Specifies if message will be purged from the archive after a certain amount of time.
purge_after	*0*	Specifies the age in days of a message before it will be purged. **Note that the age is measured from the time it was originally delivered in the primay store.**
track_history	*no*	Specifies the behaviour of the archiver when re-archiving messages that were changed after they were archived. When track_history is set to *no*, the archived version will be updated to contain the most actual information. When track_history is set to *yes*, a new copy of the message will be created in the archive, and the previous version is linked to it. This way the change history can be tracked.
log_method	*file*	The method which should be used for logging. See *Table 5.2, “Log Methods”* for valid values.
log_file	-	When logging to a file, specify the filename in this parameter. Use - (minus sign) for stderr output.
log_timestamp	*1*	Specify whether to prefix each log line with a timestamp in *file* logging mode.

Option	Default value	Description
log_level	*2*	The level of output for logging in the range from 0 to 5. 0 means no logging, 5 means full logging.

Note

Disabling archive_enable effectively causes the Zarafa Archiver to do nothing at all.

Important

A message will never be stubbed or deleted if it’s not yet archived.

Table 5.2. Log Methods

Setting	Description
file	Log to a file. The filename will be specified in log_file.
syslog	Use the Linux system log. All messages will be written to the mail facility. See also syslog.conf(5).

Usage

The Zarafa Archiver has two distinct functions.

1. Archive Management (See *Section 6.2, “Archive Management”*)

2. Archiving (See *Section 6.3, “Archiving”*)

Both functions are performed with the tool **`zarafa-archiver`** followed by one or more arguments.

6.1. Command-line options

```
Usage:
zarafa-archiver [options]

Options:
  -u <name>                        : Select user
  -l|--list                        : List archives for the specified user
  -L|--list-archiveusers           : List users that have an archived attached
  -A|--archive                     : Perform archive operation
                                     If no user is specified all user stores will
                                     be archived.
     --local-only                  : Archive only those users that have their store
                                     on the server on which the archiver is
                                     invoked.
     --attach-to <archive store>   : Attach an archive to the specified user.
                                     By default a subfolder will be created with
                                     the same name as the specified user. This
                                     folder will be the root of the archive.
     --detach-from <archive store> : Detach an archive from the specified user.
                                     If a user has multiple archives in the same
                                     archive store, the folder needs to be
                                     specified with --archive-folder.
     --archive-folder <name>       : Specify an alternate name for the subfolder
                                     that acts as the root of the archive.
     --archive-server <path>       : Specify the server on which the archive should
                                     be found.
     --no-folder                   : Don't use a subfolder that acts as the root
                                     of the archive. This implies that only one
                                     archive can be made in the specified archive
                                     store.
  -w|--writable                    : Grant write permissions on the archive.
  -c|--config                      : Use alternate config file.
                                     Default: archiver.cfg
     --help                        : Show this help message.
```

6.2. Archive Management

`zarafa-archiver` allows one to attach archives to, detach archives from and list archives of users. A user can have more than one archive attached to allow redundant archives. In those cases these archives are usually on a different server, but that’s not mandatory.

6.2.1. Creating an archive store

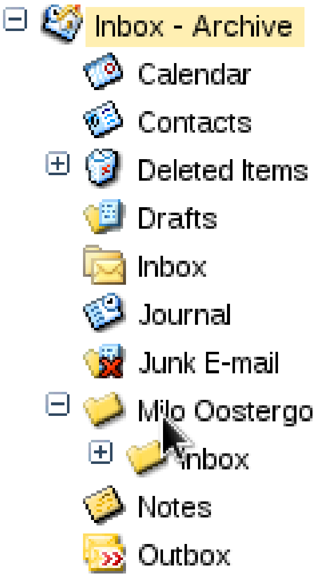
Creating an archive store is no different from creating a non-active store. So for each archive store a new non-active user must be created in the LDAP/ADS backend. To keep the addressbook clean it’s recommended (but not required) to hide the user by setting the **`zarafaHidden`** attribute to 1.

6.2.2. Attaching an archive

After an archive mailbox is created, a user can be attached to the archive mailbox. The simplest way to attach an archive mailbox to a user is as follows:

```
zarafa-archiver -u <user name> --attach-to <archive store>
```

This causes a folder with the full name of **`user name`** to be created in the archive store. This folder is attached to the primary store for **`user name`** and used as the root of the archive, see screenshot.



Inbox - Archive
Calendar
Contacts
Deleted Items
Drafts
Inbox
Journal
Junk E-mail
Milo Oostergo
Inbox
Notes
Outbox

Figure 6.1. Archive store folderlist

A user will have by default readonly permissions on the archive store. To change the default permissions on the archive store, use the **`-w`** option when attaching the store.

```
zarafa-archiver -u <user name> --attach-to <archive store> -w
```

Note

When multiple users are connected to a single archive store. A user will **`only`** have access to his/her personal archive and not to the other archives.

If desired, the automatic folder name can be overridden by providing an alternate name:

```
zarafa-archiver -u <user name> --attach-to <archive store> \
        --archive-name <folder name>
```

This causes a folder with the name **`folder name`** to be created in the archive store. This folder is attached to the primary store for **`user name`** and used as the root of the archive.

In a one-to-one archive configuration, it's usually not desired to create a folder in the archive store at all. In this case the creation of a folder can be inhibited:

```
zarafa-archiver -u <user name> --attach-to <archive store> --no-folder
```

This causes the root of the archive store to be attached to the primary store for **user name**.

In a single-server configuration, **zarafa-archiver** will not be able to connect to the correct archive server based on the archive store name. In this case the full path to the archive server must be specified:

```
zarafa-archiver -u <user name> --attach-to <archive store> \
        --archive-server <full server path>
```

The full server path is in the form **http[s]://<address>:<port>/zarafa**.

6.2.3. Detaching an archive

Normally an archive can be detached with the following command:

```
zarafa-archiver -u <user name> --detach-from <archive store>
```

If an entry for the archive store is found in the list of attached archives in the primary store, that entry will be removed.

In the rare occasion where a user has multiple attached archives from the same archive store, **zarafa-archiver** will not be able to determine which one the detach from. In that case the folder name also needs to be specified:

```
zarafa-archiver -u <user name> --detach-from <archive store> \
        --archive-name <folder name>
```

Note

When detaching an archive that already contained archived and stubbed messages, the stubbed messages can still be opened.

6.2.4. Listing attached archives

To see which archives are attached to a users primary store execute the following command:

```
zarafa-archiver -u <user name> --list
```

6.2.5. Listing users that have an archive attached

To see which users have an archive attached execute the following command:

```
zarafa-archive --list-archiveusers
```

6.3. Archiving

Archiving can be done for all users, all users on one primary server or on a per-user basis.

6.3.1. From the command-line

The following command performs one archive run for all users:

```
zarafa-archiver -A
```

Passing the **`--local-only`** option to **`zarafa-archiver`** causes it to only archive the primary stores that live on the server to which **`zarafa-archiver`** is connected. This is the server on which **`zarafa-archive`** is executed unless otherwise configured in the config file.

```
zarafa-archiver -A --local-only
```

It's also possible to explicitly specify which users primary store to archive:

```
zarafa-archive -u <user name> -A
```

6.3.2. Automated Archiving

Since archiving can create quite some load on the primary server, it is prefered to run the archiver once per day when there are as few people as possible working on it.

The easiest way to accomplish this is to run the **`zarafa-archiver`** from a cron job:

```
# m h  dom mon dow   command
  0 3  *   *   *     /usr/bin/zarafa-archiver -A
```

6.3.3. Usage Example

In this example the user *John* on the primary server will be attached to the archive store on the archiver server, next we will execute an archive run for user john. The **`zarafa-archiver`** client is installed on the primary server.

First we will attach user *John* to store *arch* on the archive server.

```
zarafa-archiver -u john --attach-to arch --archive-server https://archive-server:237/zarafa
```

Check if user john is attached to the arch store.

```
zarafa-archiver -u john --list
        0: Store: 'Inbox - arch', Folder: 'John'
```

Start archiving

```
zarafa-archiver -u john -A
```

The archived messages from user john are now archived in folder **John** in the archive store **arch** on the archive server.

6.4. Understanding logfile

When performing an archive run the actions will be logged in the default **`/var/log/zarafa/archive.log`** or to **`syslog`**. With log_level set to 6 in the **`archiver.cfg`** the stages are fully logged.

```
Archiving store for user john
Archiving messages
```

```
Processing batch of 50 messages
Opening folder
 (000000008BD0E8AA08784B5D86A489D1D791200E0100000003000000 5DC81BC1816D4F3C8ABA9DAFBDAB5A7100000000)
Opening message
 (000000008BD0E8AA08784B5D86A489D1D791200E0100000005000000FD92252E06854BBFBE1959316126197C00000000)
Message not yet archived to archive
 (0000000038A1BB1005E5101AA1BB08002B2A56C200007A617261666136636C6965677914C23617698B4570736575646F3A2F2F
Archive folder not found in cache
Archive folder: Inbox
Archive folder entryid:
 00000000B2055D0C98D7449B861ADE623FDA38290100000003000000 3B05A14D01F945B6B0904AD80E82D41600000000
Executing stub operation.
Opening message
 (000000008BD0E8AA08784B5D86A489D1D791200E0100000005000000FD92252E06854BBFBE1959316126197C00000000)
Stub operation executed.
```

The log show the archive process for the user john. It will start archiving the messages that are not yet archive mailbox. The messages are archive per folder in batches of 50 items. After the items are archived, the stub operation is started. This operation will open the message remove the body and attachments, when this is successfully done the stub will be placed.