ZCP 7.1 (build 48315)

Zarafa Archiver

The Zarafa Archiver Manual

ZCP 7.1 (build 48315) Zarafa Archiver
The Zarafa Archiver Manual
Edition 7.1

The Zarafa Collaboration Platform (ZCP) combines the usability of Outlook with the stability and flexibility of a Linux server. It features a rich web-interface, the Zarafa WebAccess, and provides brilliant integration options with all sorts of clients including all most popular mobile platforms.

Most components of ZCP are open source, licensed under the *AGPLv3*[1], can therefore be downloaded freely as *ZCP's Community Edition*[2].

Several closed source components exist, most notably:

[4] http://creativecommons.org/licenses/by-sa/3.0/
[1] http://www.gnu.org/licenses/agpl-3.0.html
[2] http://www.zarafa.com/content/community

- the Zarafa Windows Client providing Outlook integration,
- the Zarafa BES Integration providing Blackberry Enterprise Server connectivity,
- the Zarafa ADS Plugin providing Active Directory integration, and
- the Zarafa Backup Tools.

These components, together with several advanced features for large setups and hosters, are only available in combination with a support contract as part of *ZCP's Commercial Editions*[3].

Alternatively there is a wide selection of hosted ZCP offerings available.

This document, the Administrator Manual, describes how to install, upgrade, configure and maintain ZCP on your Linux server. In addition various advanced configurations and integration options are discussed.

[3] http://www.zarafa.com/content/editions

Introduction

Zarafa Archiver provides a way to minimise mailbox sizes by moving old messages to *slower* and presumably *cheaper* storage. Archive storage consists of one or more additional Zarafa servers dedicated to store archived messages.

Zarafa archive 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. Archive mailboxes can be enabled per user and users automatically get access to their archive mailbox from email clients.

Once a message is archived, it may be deleted from the original store. Zarafa Archiver can create a stub to the archived message, allowing a user to view the archived message and open it as if it were a normal message.

Zarafa Archiver uses Zarafa's multi-server technology to seamlessly access archive stores.

Zarafa Archiver is an additional product and is not a standard component of Zarafa Collaboration Platform. Subscriptions of Zarafa Archiver can only be used with Zarafa Professional or Enterprise edition; both editions provide 20 archive users for free.

Conventions

Please read this chapter before installing and deploying Zarafa Archiver, in order to understand its terminology.

- **Primary Server**

  The primary server is the server that contains the stores on which users normally work. Although the term **primary server** suggests that there's only one primary server, multiple primary servers may exist in a multi-server environment. In this document no distinction is made between a single-server or multi-server environment unless explicitly stated.

- **Archive Server**

  An archive server is the server that contains the archives for stores that reside on the primary server. An archive server is a 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.

  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 a primary server and on which a user normally works. Also known as **main store**

- **Archive Store**

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

- **Zarafa-archiver**

  Zarafa-archiver is the application that performs the actual archiving job. Zarafa-archiver is typically started from a daily or weekly cronjob. It can be installed on any Zarafa server that connects to the primary or archive server using SSL authentication. Running zarafa-archiver can be done on any node in a multi-server zarafa environment.

- **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 in the primary store, but show a user that a message was once there. It also acts as an entry point to the archived copy of that message.

- **Single Instances**

  Zarafa Collaboration Platform uses single instance storage whenever possible in order to minimise storage requirements when data is stored more than once. Zarafa Archiver makes use of this technology by remembering which instances it copied to an archive server and referencing that instance whenever possible.

- **Attached/detached stores**

  An archive store is attached, when a data link is made with a primary store. Making such a link is called *attaching* a store, removing such a link *detaching* it. Detached stores are still available for reading.

- **Hooked/unhooked stores**

  A hooked store is available for use as a primary or archive store. Unhooked stores are not available. From the perspective of a mail client or the Archiver they don’t exist. Such unhooked stores are also know as **orphaned stores**. Unhooked stores cannot be read from or written to. The process of making a store available for a user is called *hooking*, the reverse *unhooking*. An unhooked store may be removed.

Installation

3.1. Requirements

To deploy the Zarafa Archiver at least two servers are required: one primary Zarafa server and one Zarafa Archive server. Multi-server technology is used to connect archives to users automatically.

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.

3.1.1. Software Requirements

- ZCP 7.0.4+
- Multi-server Zarafa setup with LDAP or ADS
- Valid ZCP Professional or Enterprise subscription
- Zarafa Archiver subscription

3.1.2. Required Packages

The following Zarafa packages are required on a primary node:

- zarafa-client
- zarafa-dagent (for LMTP only)
- zarafa-licensed (with Zarafa Archiver subscription)
- zarafa-multiserver
- zarafa-search or zarafa-indexer
- zarafa-server
- zarafa-spooler
- zarafa-utils

The following Zarafa packages are required on an archive node:

- python-zcp-license
- python-mapi
- zarafa-client
- zarafa-common
- zarafa-libarchiver
- zarafa-libs

- zarafa-licensed (with Zarafa Archiver subscription)
- zarafa-multiserver
- zarafa-server
- zarafa-utils

3.2. Installation

The Zarafa Archiver software can be found on the Zarafa Portal as an additional download. The Zarafa Archiver download will include the following two packages:

- zarafa-archiver containing the Archive controller and configuration files
- zarafa-archiver-extra containing additional utilities for archiving setups

  See *Chapter 8, Zarafa Archiver Extras* for more details.

3.2.1. Default installation location

The Zarafa Archiver packages can be installed on any node in a multi-server setup. However, installing the packages on the archive node is recommended.

3.2.1.1. RPM based distributions

Use the following command to install the **zarafa-archiver** and **zarafa-archiver-extra** package on RPM based distributions:

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

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

3.2.1.2. DEB based distributions

On Debian based distributions use:

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

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

3.2.2. Activate Archiver subscription

To use the Archiver subscription on every node, the Archiver subscription has to be placed in the **/etc/zarafa/license** directory of all your servers. Execute the following commands on every node to use the archive subscription:

```
echo 'Archiver 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.

Important

The Archiver subscription must be placed on **all** server nodes, otherwise de-stubbing will not work.

3.3. Basic Configuration

Zarafa Archiver is configured by default in **/etc/zarafa/archiver.cfg**. If the default configuration file is not found, Archiver will try to work with default configuration settings, which will most probably fail. Archiver will display an error message like "Unable to open admin session on server *https://192.168.1.10:237/zarafa*" and close.

An alternative configuration file may be specified by using command line option **--config**. When such a configuration file is specified and cannot be found, Archiver will emit an error message like: "Unable to open configuration file /tmp/test.cfg" and close.

3.3.1. Connection Parameters

As with all Zarafa components, **zarafa-archiver** needs to know where to connect to and how to authenticate. This is configured using the **server_socket**, **sslkey_file** and **sslkey_pass** settings.

For instance:

```
server_socket   = file:///var/run/zarafa
sslkey_file     = /etc/zarafa/ssl/client.pem
sslkey_pass     = secret
```

For more information about creating and configuring SSL certificates, see *http://doc.zarafa.com/7.1/Administrator_Manual/en-US/html-single/index.html#_creating_ssl_certificates*.

3.3.2. MySQL Settings

zarafa-archiver uses one central MySQL database for managing deduplication of archived attachments. MySQL settings can be configured like this:

```
mysql_host      = localhost
mysql_port      = 3306
mysql_user      = zarafa
mysql_password  = password
mysql_socket    =
mysql_database  = archiver
```

This database should not be the same one that **zarafa-server** is using and needs to be accessible over the network from each server running the **zarafa-archiver** command.

Archive configuration

zarafa-archiver is configured in a configuration file, that can be specified on the command line using command line option **--config** or **-c**. When no configuration file is specified from the command line, Archiver tries to find a default configuration file named **archiver.cfg** located in **/etc/zarafa/**. Please note that when no configuration file can be found, information required to proceed is missing. In such a case Archiver will emit an error message and exit.

Depending on user requirements, Archiver can be configured in many different ways. Five different stages in Archiving can be distinguished, of which any subset can be used. This chapter describes a basic setup. Later more advanced configurations using more complex combinations of stages will be presented.

The five stages of Archiving are:

Figure 4.1. Archiver stages

The ages listed in the rightmost colums are examples only.

Table 4.1. Archiver stages

Stage	Description
No Archiving	No archive store exists and no messages are archived.
Archive copy	A message exist in both main store and archive store.
Stubbed message	A message exists in archive store and a stub referring to the archived message exists in main store.
Fully removed	A message exists only in archive store.
Archive purged	A message has been removed from both main store and archive store. It no longer exists, but may be present in a backup and restored from there.

4.1. Basic Archiver configuration

In a setup all stages as shown in Figure 4.1 can be configures. It should be noted, though, that not all combinations make sense. As a basic setup for Zarafa Archiver, considered best practice, is a **stubbing-only** configuration. In this setup all emails can be accessed from the primary store of the user. Depending on the age of the message the email will be opened on the primary server or a stub will open the item directly in the user's archive mailbox. In this case the user doesn't directly access his or her archive mailbox to view emails.

For a **stubbing only** setup settings like the following must be configured in **/etc/zarafa/archiver.cfg**:

```
archive_enable                  = yes
archive_after                   = 90

stub_enable                     = yes
stub_after                      = 90
stub_unread                     = no

delete_enable                   = no
delete_after                    = 0

purge_enable                    = no
```

```
purge_after                     = 0

cleanup_action                  = store
cleanup_follow_purge_after      = no

enable_auto_attach              = yes
auto_attach_writable        = no
```

Setting **archive_enable = yes** enables archive operation, i.e. copying emails from the primary node to the archive node. Setting **archive_after = 90** indicates that messages older than 90 days are to be archived.

Setting **stub_enable = yes** enables stubbing. Body and attachments messages in the primary store are removed and a reference to the archive store is created. Setting **stub_after = 90** means that messages older than 90 days will be stubbed.

Unread emails are not stubbed by default. Setting **stub_unread = yes** means that even unread messages are stubbed.

Settings **delete_enable** and **purge_enable** must be set to **no**, since a user will not access his archive mailbox directly in this mode.

Setting **cleanup_action = store** means that, in a cleanup run, archived messages of which stubs have been deleted from the archive mailbox will be moved to an archive folder named **Zarafa Archive\Deleted**. See chapter *Section 6.2, "Cleanup"* for an explanation of this cleanup process.

Setting **enable_auto_attach = yes** means that a user automatically gets an archive mailbox when the archive server option is enabled in Active Directory or OpenLDAP. The user will always get read-only permissions, so users can delete messages from the primary mailbox only. This is required to make sure the stubs in the primary mailbox will always point to a existing message.

4.2. Archive-only setup

Another setup is archive-only, in which emails will exist either in the primary mailbox or in an archive mailbox. Stubbing is disabled in this mode.

For an archive-only setup the following settings need to be configured in **/etc/zarafa/archiver.cfg**:

```
archive_enable                  = yes
archive_after                   = 365

stub_enable                     = no

delete_enable                   = yes
delete_after                    = 365

purge_enable                    = yes
purge_after                     = 3650

enable_auto_attach              = yes
auto_attach_writable            = yes
```

To move items to the archive mailbox setting **delete_enable** is set to **yes** and the number of days is configured in **delete_after**. In this example, items older than 1 year are deleted from the primary mailbox and after that can only be accessed from the archive mailbox.

Setting **stub_enable = no** makes sure there will be no stubs. Since there are no stubs, there is no sense in doing cleanup runs, so settings for cleanup need not be given.

To clean up the archive mailbox, items will be completely removed after 10 years.

As stubbing is not used in this mode, the archives can be set writable so users can clean up their archive mailbox themselves.

4.3. Advanced setups

Advanced setups are possible, but may cause some confusion initially, because some settings interfere. An example of an advances setup is the following configuration, that resembles the setup from Figure 4.1:

```
archive_enable                  = yes
archive_after                   = 2

stub_enable                     = yes
stub_after                      = 60
stub_unread                     = no

delete_enable                   = yes
delete_after                    = 365
stub_unread                     = yes

purge_enable                    = yes
purge_after                     = 3650

cleanup_action                  = store
cleanup_follow_purge_after      = yes

enable_auto_attach              = yes
auto_attach_writable            = no
```

In this example mode emails will be archived after two days as an archived copy, while the original remains in main store. After 60 days the copy from main store will be removed and replaced by a stub to the archived copy. After a year, the stub will be removed, while the copy in the archive remains.

To prevent the archive from growing forever, items will be removed after 10 years.

Archive Management

5.1. Automatic archive management

Zarafa-archiver can attach archive stores automatically, based on user attributes stored in LDAP or in Active Directory. When using this way of attaching stores, zarafa-archiver will create archive stores on the archive server and attach the user stores to these archive stores based on information found in LDAP or in Active Directory. When using this method of attaching and detaching, Outlook and Webaccess will load archive stores automatically.

To use this feature, setting **enable_auto_attach** must be set to **yes** in **/etc/zarafa/archiver.cfg**:

```
enable_auto_attach = yes
```

Alternatively **zarafa-archiver** can be run periodically to perform the auto-attach operation:

```
zarafa-archiver --auto-attach
```

5.1.1. Using OpenLDAP

To add an archive store to a user through LDAP, attribute **zarafaUserArchiveStores** needs to be modified. This is a multi-value attribute, which needs to be set for the server name or server names of the servers that contain an archive store for the user.

A typical ldif (LDAP configuration file) would look like this:

```
dn: uid=user,ou=users,dc=example,dc=com
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: top
objectClass: zarafa-user
objectClass: posixAccount
cn: User
gidNumber: 0
homeDirectory: /bin/false
sn: User
uid: user
uidNumber: 1000
givenName: User
mail: user@server.com
userPassword:: e1NTSEF9VzlXV0U3N1NEcW54UkJ3SFJkQUYvVkhrUj
zarafaAccount: 1
zarafaUserServer: userServer
zarafaUserArchiveServers: archiveServer
```

Note

The archive store won't be created or attached until the next run of **zarafa-archiver -A** with **enable_auto_attach = yes** or **zarafa-archiver --auto-attach**.

5.1.2. Using Active Directory

To add an archive store to a user in Active Directory, one has to open the user in the *Active Directory Users and Computers* window and select the **Zarafa Features** tab.

Figure 5.1. ADS Features tab

Next, select the **Archiver** feature and click **Properties**. This will pop up the dialog in which the server names of the servers on which an archive store should exist for the selected user or users.

Figure 5.2. ADS Select archive servers

When a user has an automatic archive and the archiver feature in Active Directory is removed, the automatic archive will be automatically detached the first time an archive run is performed with auto-attach enabled. See also chapter 5.14.

Note

Archive stores won't be created or attached until the next run of **zarafa-archiver -A** with **enable_auto_attach = yes** in its configuration file or by running **zarafa-archiver --auto-attach**.

5.1.3. Listing attached archives

To see which archive stores are attached to a user's primary store execute the following command:

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

This will output a list of attached archives. Each line contains the name of the archive store, the name of the folder that acts as root for the archive and the access rights the owner has for the archive store.

The store name will equal the name that was passed when the archive was attached. The archive folder name will be one of three things: 1. it will be the full name of the user if no name was specified when the archive was attached; 2. if a name was specified, the archive folder will be that name; 3. if the archive was attached in a one-to-one configuration, it will be *Root Folder*.

The access rights will be *Read Only* when the archive was attached without write permissions. If it was attached with write access it will be *Read/write*. If the permissions were manually changed after the archive was attached, the rights field will display the configured role for the rights or all the rights if no matching role exists.

5.1.4. Detaching an archive

To detach an archive mailbox of a user, the archive server has to be removed in the Zarafa feature tab.

Note

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

5.1.5. Listing users that have an archive attached

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

```
zarafa-archiver --list-archiveusers
```

5.1.6. See details of an archive store

The details of an archive store can be obtained in two ways:

- By requesting the details of the user owning the archive:

```
> zarafa-admin --details user1
Username:           user1
Fullname:           User 1
Emailaddress:       user1@cluster.sio2
Active:             yes
Administrator:      no
Address book:       Visible
Auto-accept meeting req:no
Home server:        cnode-1
Last logon:         12/09/2011 03:41:32 PM
Last logoff:        12/09/2011 03:41:32 PM
Mapped properties:
    PR_GIVEN_NAME           User
    PR_SURNAME              One
    PR_EC_ENABLED_FEATURES  pop3
    PR_EC_DISABLED_FEATURES imap
    PR_EC_ARCHIVE_SERVERS   cnode-2
Attached archives:  1
    Root Folder in Archive - User 1 [Read Only]
Quota overrides:    no
Warning level:      unlimited
Soft level:         unlimited
Hard level:         unlimited
Current store size: 14.86 MiB
Groups (1):
    Everyone

Archive details on node 'cnode-2':
Current store size: 114.68 MiB
```

All attached archive details are appended at the end.

- By explicitly requesting the archive details on a specific node:

```
> zarafa-admin --details user1 --type archive --node cnode-2
Current store size: 114.68 MiB
```

Important

The node on which the archive is stored must be specified in order to find the archive.

These methods only apply to genuine archive stores. Details of regular stores that are manually attached as archives can be obtained by obtaining the details of the user owning that store:

```
> zarafa-admin --details archive
Username:           archive
```

```
Fullname:           Archive Store
Emailaddress:       archive@cluster.sio2
Active:             no
Administrator:      no
Address book:       Hidden
Auto-accept meeting req:no
Home server:        cnode-2
Last logon:         12/09/2011 03:41:32 PM
Last logoff:        12/09/2011 03:41:32 PM
Mapped properties:
    PR_GIVEN_NAME           Archive
    PR_SURNAME              Archive
    PR_EC_ENABLED_FEATURES  pop3
    PR_EC_DISABLED_FEATURES imap
Quota overrides:    no
Warning level:      unlimited
Soft level:         unlimited
Hard level:         unlimited
Current store size: 114.68 MiB
Groups (1):
    Everyone
```

5.1.7. Unhooking an archive store

An archive store can be unhooked the same way as a regular store, but with the addition of the **`type`** and **node** arguments:

```
> zarafa-admin --unhook-store user1 --type archive --node cnode-2
Store unhooked.
```

5.1.8. Hooking an archive store

An archive store can be hooked the same way as a regular store, but with the addition of the **`type`** and **node** arguments:

```
> zarafa-admin --list-orphans --node cnode-2
Stores without users:
    Store guid                          Guessed username    Last login      Store size  Store
 type

 ----------------------------------------------------------------------------------------------------
    F1A6BFCD67604B0FB733F746F1D00A91    user1               <unknown>       0
 archive
> zarafa-admin --hook-store F1A6BFCD67604B0FB733F746F1D00A91 -u user1 --type archive --node
 cnode-2
Store hooked.
```

5.1.9. Removing an archive store

An archive store can be removed the same way as a regular store, but with the addition of the **`type`** and **node** arguments:

```
> zarafa-admin --unhook-store user1 --type archive --node cnode-2
Store unhooked.
> zarafa-admin --list-orphans --node cnode-2
Stores without users:
    Store guid                          Guessed username    Last login      Store size  Store
 type

 ----------------------------------------------------------------------------------------------------
```

```
    F1A6BFCD67604B0FB733F746F1D00A91    user1               <unknown>       0
 archive
> zarafa-admin --remove-store F1A6BFCD67604B0FB733F746F1D00A91 --node cnode-2
Store removed.
```

Running the Archiver

6.1. Performing archiving task

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

6.1.1. From the command line

The following command performs an archive run for all users:

```
zarafa-archiver -A
```

Passing option **`--local-only`** to **`zarafa-archiver`** tells it to archive only the primary stores that live on the server to which zarafa-archiver is connected. Unless configured otherwise in the configuration file, this is the server on which zarafa-archiver is run:

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

To run **`zarafa-archiver`** for a specific user use the -u option:

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

It's recommended to perform an Archiver run every night. This can be done by adding the following line to **`/etc/crontab`** :

```
0 1 * * * root [ -x /usr/bin/zarafa-archiver ] && /usr/bin/zarafa-archiver -A
```

6.2. Cleanup

To keep the archive synchonised with primary stores after items are deleted from the primary store, start the archiver in cleanup mode. This is a separate operation from archiving. Since cleanup may be a long process, it is recommended to run it less often than archive operations.

When **`zarafa-archiver`** performs a cleanup run in which it encounters messages that exist in an archive and for which in main store a stub used to present which has now been deleted, it will remove the message from the archive. This is useful to avoid the situation that archived messages exist which have properties set that indicate that they are stubbed, while no corresponding stub exists in main(store.)

What happens to such archived messages depends on configuration:

1. cleanup_action is set to **`delete`**. In this case messages for which no stubs exist will be removed from the archive.

2. cleanup_action is set to **`store`**. In this case messages for which no stubs exist will be moved to a folder named **`Zarafa Archive\Deleted`**.

Note

Archiver can remove *soft deleted* items from the main store (or stores) for which a cleanup run is performed. This avoids the situation in which *soft deleted* stubs may be restored for which the archived versions no longer exist. To enable this feature, configuration item **`purge_soft_deleted`** must be set to **`yes`**. See chapter 10.1. *Softdelete restore* in the *The Administrator Manual* for more information on *soft deleted* items.

6.2.1. From the command-line

The following command performs a cleanup for all users:

```
zarafa-archiver -C
```

Passing option **`--local-only`** to **`zarafa-archiver`** causes it to cleanup only archives of users who have a store on the server to which **`zarafa-archiver`** is connected. This is the server on which **`zarafa-archiver`** is executed unless otherwise configured in the configuration file.

```
zarafa-archiver -C --local-only
```

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

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

It's recommended to perform the cleanup run once a week. This can be done by adding the following line to **`/etc/crontab`**.

```
0 3 * * 0 root [ -x /usr/bin/zarafa-archiver ] && /usr/bin/zarafa-archiver -C
```

This will clean up all archives every sunday at 3:00 am.

Client features

To allow a user to browse and search in his archive or in the archives of shared stores, the archives are automatically added in Outlook and in WebAccess.

Figure 7.1. Outlook Hierarchy with Archives

In Webaccess that looks like this:

Figure 7.2. Webaccess Hierarchy with Archives

Important

Please note that it's currently (7.1.9 with zarafa-search 2.0) not possible to search in the body of stubbed messages from the users primary store. Instead the search has to be performed from within the archive store.

Zarafa Archiver Extras

The Zarafa Archiver Extras package contains additional tools to enhance basic Zarafa Archiver functionality.

8.1. Installation

Zarafa Archiver Extras can be found in the zarafa-archiver-extra package.

8.2. The tools

8.2.1. Zarafa Archiver ACL Sync

8.2.1.1. Description

`za-aclsync` synchronises archive ACL settings with those of the primary store.

When a user has set permissions for other users or groups on his or her store or folders, those other users will need at least read-permissions on this persons archive as well, so that they can read stubbed messages or access the archive directly. These permissions cannot be set by the owner of the archive when the archive was attached without write privileges. Even when another user has write permissions, it's a nuisance to set all the permissions twice, or possibly more often when multiple archives are attached.

Note that no user will ever get more rights on a store or folder than the owner of the archive. When the archive was attached without write permissions, no user will get write permissions on the archive stores. For every archived folder in an archived store **`za-aclsync`** will first determine the rights of the owner of the archive. After that it will get all the entries from the ACL of the current folder except those of the owner. Each right will be masked with the rights of the owner before being added to the ACL of the archive folder.

8.2.1.2. Usage

```
za-aclsync [options] [users]

options:
-h serverpath           : Host to connect to.
-s sslkey_file          : SSL key file for authentication.
-p sslkey_pass          : Password for the SSL key file.
```

`users` is a space separated list of users for which to synchronize the ACL settings. If no user is specified all users will be processed.

8.2.2. Zarafa Archiver ACL Set

8.2.2.1. Description

`za-aclset` sets or updates permissions on an archive store for the owner of that store.

When an archive is attached to a store, the owner of the store gets read-only or read/write permissions on the archive, depending on the configuration file or command line options used at the time of

attaching. Also when an archive is attached to a store of a non-active user it's impossible to set permissions for that non-active user.

In the first case it may be desirable to change the permissions at a later time because of company policy changes or because the original setting was wrong. In the second case, when the user is converted to an active user, permissions need to be reset because the owner won't have any permissions on his archive.

8.2.2.2. Usage

```
za-aclset [OPTIONS] [users...]

options:
-h | --host         Host to connect to. Default: file:///var/run/zarafa
                    Three formats are allowed for this option:

                      UNIX socket : file://<path to the UNIX socket>
                             HTTP : http://<host or IP>:<port>/zarafa
                      Secure HTTP : https://<host or IP>:<port>/zarafa

-s | --sslkey-file  SSL key file for authentication.
-p | --sslkey-pass  Password for the SSL key file.
-w |                Grant write permissions on the archive.
--writable <y|n>    Enable or disable write permissions.
--help              Show this help message.
```

users is a space separated list of users for which to synchronize the ACL settings. If no user is specified all users will be processed.

Za-aclset should be executed on a regular basis from a cronjob to synchronise the rights between the primary store and the archive store .

8.2.3. Zarafa Archiver Restore

8.2.3.1. Description

za-restore is a utility that restores archived messages from an archive. Use **za-restore** to destub all stubbed messages and restore all messages that were deleted after archiving. Do not drag and drop messages back from an archive to a main store in Outlook or Webaccess. Using **za-restore** ensures that restored messages are sanitised, so that they can be re-archived properly later.

8.2.3.2. Usage

```
Usage: za-restore [OPTIONS] user
OPTIONS:
  -h | --host           : Host to connect to. Default: file:///var/run/zarafa
  -s | --sslkey_file    : SSL key file for authentication.
  -p | --sslkey_pass    : Password for the SSL key file.
  -l | --log-file       : Specify log file.
  --detach              : Detach the selected or all archive stores before
                          starting the restore procedure. This avoids the
                          Archiver from rearchiving restored messages.
  --unhook              : Unhook the selected or all archive stores once
                          the restore process has completed. This implies
                          --detach and only works on archive stores.
  --remove              : Remove the selected or all archive stores once
                          the restore process has completed. This implies
```

```
                          --unhook and only works on archive stores.
  --select-source       : Select the source archive(s) by providing a comma
                          separated list of archive indexes. The indexes specify
                          which archives to restore from. The --detach, --unhook
                          and --remove options only apply to the selected
                          archives.
                          The archive indexes can be obtained by listing the
                          attached archives for a user: zarafa-archiver -u
                          <user> -l.
  -v | --verbose        : Increase console loglevel. Can be specified multiple
                          times.
  -q | --quiet          : Decrease console loglevel. Can be specified multiple
                          times.
  -N | --dry-run        : Don't actually modify anything.
  --help                : Show this help message.
```

8.2.3.3. Example

The following example will completely restore the store of john_doe and detaches and unhooks all archive stores while logging to **`/tmp/john_doe_restore.log`**

```
> za-restore --unhook -s /etc/zarafa/ssl/archiver.pem -p password \
  -l /tmp/john_doe_restore.log john_doe
```

Note that no host is specified, causing za-restore to connect to **`file:///var/run/zarafa`**. The sslkey_file and sslkey_pass are specified in order to connect to the other nodes in the cluster.

Archiver command line

The following table describes command line arguments and options that apply to Zarafa Archiver. Please note that if no command line arguments are supplied the same help information is displayed as that from **`--help`**.

Table 9.1. Command line

Option	Alternative	Type	Meaning and usage
-u <name>		string	Select the archive of user <name>.
-l	--list	n/a	List archives for a specified user.
-L	--list-archiveusers	n/a	List users that have an archive attached.
-A	--archive	n/a	Perform archive operation. If no user is specified all user stores will be archived.
-C	--cleanup	n/a	Perform a cleanup of the archive stores attached to the user specified with **`-u`**. If no user is specified, all archives are cleaned up.
	--local-only	n/a	Archive or cleanup only those users that have their store on the server on which the archiver is invoked.
-a	--attach-to <archive store>	n/a	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.
-d	--detach-from <archive store>	n/a	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.
-D	--detach <archive no>	n/a	Detach the archive specified by <archive no>. This number can be found by running **`zarafa-archiver -l`**
	--auto-attach	n/a	When no user is specified with **`-u`**, all users will have their archives attached or detached based on the LDAP/ADS settings. When a user is specified only that user's store will be processed. This option can be combined with **`-A/--archive`** to force an auto-attach run regardless of the enable_auto_attach configuration option.
-f	--archive-folder <name>	string	Specify an alternate name for the subfolder that acts as the root of the archive.
-s	--archive-server <path>	string	Specify the server on which the archive resides.
-N	--no-folder	n/a	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			Grant write permissions on the archive. This will override the **`auto_attach_writable`** config option.
	--writable	yes/no	Enable or disable write permissions. This will override the **`auto_attach_writable`** config option.

Option	Alternative	Type	Meaning and usage
-c	--config	string	Use alternate config file.
	--help	n/a	Show a help message.

Archiver configuration settings

The following table describes configuration settings that apply to Zarafa Archiver. When settings marked **yes** in column *Required* aren't specified Archiver prints an error message and quits execution.

Table 10.1. Archiver settings

Setting	Type	Default	Re	Meaning and usage
server_socket	string		Yes	Location of server unix socket.
archive_enable	yes/no	yes	No	Setting this to **no** disables archiving.
archive_after	integer	30	No	Archive messages older than this number of days.
stub_enable	yes/no	no	No	Enables or disables stubbing.
stub_unread	yes/no	no	No	Enables or disables stubbing of unread messages.
stub_after	integer	0	No	Stub messages older than this number of days. When **archive_after** has a larger value than this setting, **archive_after** is used instead.
delete_enable	yes/no	no	No	Delete archived messages from the main store. Only archived messages can be deleted. Note that if this option is set to **yes**, no stubbing is performed.
delete_unread	yes/no	no	No	Enables or disables deleting unread archived messages from the main store.
delete_after	integer	0	No	Delete messages older than this number of days. When **archive_after** has a larger value than this setting, **archive_after** is used instead.
purge_enable	yes/no	no	No	Enables or disables purging of messages in the archive server(s).
purge_after	integer	2555 (7 years)	No	Purge messages older than this number of days. When **archive_after** has a larger value than this setting, **archive_after** is used instead.
cleanup_action	string	store	No	Specify what action should be taken on archive messages whose primary message has been deleted. Possible options are: **store**: the archived message in a special folder named **Zarafa Archive\Deleted**; **delete**, delete the archived messages.

Setting	Type	Default	Re	Meaning and usage
cleanup_follow_purge_after	yes/no	no	No	Specify that items that are cleaned up from the archive are at least the age that is specified in setting **purge_after**. This is to avoid messages being deleted from the archive when they were deleted from the primary store by a delete operation. Note that setting this to **yes** will cause a cleanup run to seemingly do nothing. A rule of thumb is to set this to the same value as **delete_enable**. So when **delete_enable** is set to **yes**, make sure **cleanup_follow_purge_after** is also set to **yes**.
enable_auto_attach	yes/no	no	No	Specify whether on each archive run an auto-attach run is performed in order to match the attached archives to the requested state as specified in LDAP or ADS.
auto_attach_writable	yes/no	yes	No	Specify whether an auto attached archive will be granted write permissions for the user the arcive is attached to.
purge_soft_deleted	yes/no	no	No	Specify whether archiver must purge soft-deleted items before a clean-up run
log_method	string	file	No	Logging method, either **syslog** of **file**.
log_file	string	-	No	File name of log file when **log_method = file**, use "-" for stderr
log_level	integer	2	No	Log level (0 = no logging, 5 = full logging)
log_timestamp	integer	1	No	Log timestamp - prefix each log line with timestamp in file logging mode.
sslkey_file	string		Yes	Log in to Zarafa server using this SSL Key.
sslkey_pass	string		Yes	Password of SSL Key.
pid_file	string	/var/run/zarafa-archiver.pid	No	Name of control pid file.
mysql_host	string	localhost	No	MySQL hostname to connect to for database access.
mysql_port	string	3306	No	MySQL port to connect with (usually 3306).

Setting	Type	Default	Re	Meaning and usage
mysql_user	string	root	No	Name of the user with which to connect to MySQL.
mysql_password	string		No	The password for **mysql_user** (leave empty for no password).
mysql_socket	string		No	Override the default MySQL socket to access mysql locally. Works only if the **mysql_host** value is empty or **localhost**.
mysql_database	string	zarafa-archiver	No	Database to connect to.