

Joinup Generic Installation Procedure

Table Of Content

- 1. Introduction
 - 1.1. Aims
 - 1.2. Reserves
- 2. Architecture
- 3. Front End
 - 3.1. Components
 - 3.2. Prerequisites
 - 3.3. Installation procedure
 - 3.3.1. Files
 - 3.3.2. Database
 - 3.3.3. Cron job
- 4. Source Code Management
 - 4.1. Components
 - 4.2. Prerequisites
 - 4.2.1. Required Apache modules
 - 4.3. Installation procedure
 - 4.3.1. Directories and filesystems
 - 4.3.2. Scripts description
 - 4.3.3. Database setup
 - 4.3.4. Scripts setup
 - 4.3.5. Scripts configuration
 - 4.3.6. Authentication through "pwauth" setup and configuration
 - 4.3.7. "svncron" specific configuration
- 5. Apache configuration file
- 6. Build and Artifacts Repository
 - 6.1. Components
 - 6.2. Prerequisites
 - 6.3. Installation procedure
 - 6.3.1. MySQL DataSource on Tomcat
 - 6.3.2. Installation
 - 6.3.3. First time configuration
 - 6.3.3.1. Server configuration
 - 6.3.3.2. Allow authenticated JoinUp users to log in the User Interface
- 7. Mailing List
 - 7.1. Components
 - 7.2. Prerequisites
 - 7.3. Installation procedure
 - 7.3.1. Obtaining a POP3-enabled mailbox
 - 7.3.2. Configure and enable the mailbox within the Front End
 - 7.3.2.1. Configuration:
 - 7.3.3. Installation of the scripts
 - 7.3.4. Configuration of the scripts
 - 7.3.5. Configuration of the cron jobs
- 8. Indexation and Search
 - 8.1. Components
 - 8.2. Prerequisites
 - 8.3. Installation procedure
 - 8.3.1. Directories and filesystems
 - 8.3.2. Tomcat setup
 - 8.3.3. Link of Solr on the Front End

1. Introduction

1.1. Aims

This document is the entry point for all the technical documentation related to the installation of Joinup. This documentation is generic and has to be adapted to the configuration of the final platform.

1.2. Reserves

This document doesn't include all the tuning of the services such as Apache, Tomcat, Subversion, MySQL...

This document doesn't include the update mechanism of an existing Joinup Platform. For this please refer to the associated delivery note.

2. Architecture

The architecture of Joinup is based on several open source components which deliver different services. We distinguished seven components:

- Front End
- Source Code Management
- Build and Artifacts Repository
- File Sharing
- Mailing List
- Indexation and Search

3. Front End

3.1. Components

The Front End is composed of a Drupal instance based on the Pressflow 6.x core and a database.

3.2. Prerequisites

You need to have an Apache HTTPD instance and a MySQL database with InnoDB engine enabled.



Joinup Front End is not dependant of Apache HTTPD and may be installed on NGINX. It has never been tested on IIS but standard Drupal is compatible with Microsoft technology.

3.3. Installation procedure

3.3.1. Files

The first step is to get the source code and the pressflow environment:

```
# We'll stay generic and mention ${version} for the delivered version
version="V1.1.2"
```

We have to get the full Drupal tree. The development team uses PressFlow (PressFlow is an optimized variant of Drupal) instead of the regular Drupal core:

```
pressflow_version="6.22.105"
wget "https://github.com/pressflow/6/archive/pressflow-${pressflow_version}.tar.gz"
tar xzf pressflow-${pressflow_version}.tar.gz
pressflow_dir="pressflow-${pressflow_version}"
```

We export the sites folder from svn:

```
cd $pressflow_dir
svn export --force https://joinup.ec.europa.eu/svn/joinup/tags/${version}/sites/
```

Then, we apply the patches of the core:

```
cd sites/core_patches/  
./apply_patches.sh  
cd ..  
rm -rf core_patches/
```

We also need to fill the sites/default/settings.php file with the following values:

```
// Solr-dedicated, environment-specific configuration  
$conf['apachesolr_host'] = 'SOLR_HOST'  
$conf['apachesolr_port'] = 'SOLR_PORT';  
$conf['apachesolr_path'] = '/joinup-solr';  
// The Solr client require a JRE...  
$conf['apachesolr_attachments_java'] = 'PATH_TO_JRE_ON_LOCAL_MACHINE';  
  
// JoinUp-specific configuration for the mailing list. It consists in setting the  
mailing list domain  
$conf['isa_ml_domain'] = 'MAILING_LIST_DOMAIN';  
  
// These variables must be set to zero for the main Drupal website.  
// They will be modified for the sub-site when a new virtual forge is created.  
$conf['isa_vf_access_tid_virtual_forge'] = 0;  
$conf['isa_vf_access_enable_grant'] = 0;
```

3.3.2. Database

You can save the database this way (if it is already existing):

```
mysqldump --skip-extended-insert \--skip-complete-insert \--hex-blob \--add-drop-table  
\--routines \-h DATABASE_HOST \-u DATABASE_USER \-p DATABASE_PASSWORD \ | gzip >  
DATABASE -$(date "+%F.%T").sql.gz
```

You can import a database dump this way:

```
mysql_command="mysql \-h DATABASE_HOST \-u DATABASE_USER \-p DATABASE_PASSWORD"  
${mysql_command}  
mysql> CREATE DATABASE joinup COLLATE utf8_unicode_ci;  
mysql> exit;  
  
# drop all existing tables (useful if the provided script does not include DROP TABLE  
statements)  
echo "show tables" \ | $mysql_command \ | tail \-n \+2 \ | sed 's,^,DROP TABLE ,;s,$,\;,'  
\ | $mysql_command  
  
# import the delivered data  
zcat dbscripts/database_backups/ISA_FULL_BACKUP_${version}.sql.gz \ | $mysql_command
```

The Joinup database also needs some stored procedures for the Nexus-Drupal integration; their setup script can be found in [https://joinup.ec.europa.eu/svn/joinup/tags/\\${version}/isamaven/doc/nexus_drupal_authentication.sql](https://joinup.ec.europa.eu/svn/joinup/tags/${version}/isamaven/doc/nexus_drupal_authentication.sql).

```
wget
https://joinup.ec.europa.eu/svn/joinup/tags/${version}/isamaven/doc/nexus_drupal_authentication.sql
${mysql_command} DATABASE < nexus_drupal_authentication.sql
rm -f nexus_drupal_authentication.sql
```

You can test the stored procedures like this:

```
${mysql_command}
mysql> use DATABASE;
mysql> show procedure status;
```

We recommend (but it is not mandatory) to use a dedicated user for the maven connection

```
mysql> CREATE USER 'maven_user'@'%' IDENTIFIED BY 'Password';
mysql> GRANT EXECUTE ON * . * TO 'maven_user'@'%' IDENTIFIED BY 'Password' WITH
MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0
MAX_USER_CONNECTIONS 0 ;
```

You may also wish to change the admin password:

```
mysql> UPDATE users SET pass = MD5('new password goes here') WHERE uid = 1;
```



Mailbox

This application checks a POP3 mailbox through its cron job in order to get mails incoming from mailing lists that have to be integrated into the database.

All parameters of this mailbox have to be configured through admin/content/mailhandler/edit/1. If you need to disable the POP3 check (e.g. because php-imap is not installed or because the mailbox has not been created yet), you can do it either by using this page or by issuing the following query:

```
mysql> UPDATE mailhandler SET enabled = 0;
```

3.3.3. Cron job

The approach chosen to trigger regularly the cron.php script is to schedule a cron task with a dedicated bash script (but you can use other method if needed).

joinup-cron.sh

```
#!/bin/sh

# Configuration
stdout_path="/tmp/cron.$$out"
stderr_path="/tmp/cron.$$err"
wget_command="/usr/bin/wget -S -t 1 -O ${stdout_path} -o ${stderr_path}"

# The URL to the cron.php file located in the root folder of the application.
url='http://HOSTNAME/cron.php'
# The target_addr variable is used to send email (you can set several email address
separated by a space)
target_addr="EMAIL_ADDRESS_1 EMAIL_ADDRESS_2"
# The target file is used to trace the last execution of the script - please use
# one file per instance (e.g. applicationname-instancename)
target_file='/tmp/joinup-cron.log'

# Work
echo -e "Last execution of $(whoami)@$(hostname):$(readlink -f $0) on $(date '+%F %T')
\n" > ${target_file}
${wget_command} "${url}"
wget_exit_code=$?

mail_subject="Execution of $(whoami)@$(hostname):$(readlink -f $0)"

# Analyze the wget exit code and output
if [ ${wget_exit_code} -gt 0 ] || ! grep -q '200 OK' "${stderr_path}"; then
(
    echo "$(date '+%F %T'): wget returned ${wget_exit_code} along with the following
output:"
    echo '---'
    cat "${stderr_path}"
    echo '---'
    echo "... when trying to fetch ${url}."
    if [ -s ${stdout_path} ]; then
        echo 'Downloaded content:'
        cat "${stdout_path}"
    else
        echo 'No content was downloaded.'
    fi
) | mail -s "${mail_subject}" ${target_addr}
fi

# Analyze the PHP script output, which is expected to be empty
if [ -s ${stdout_path} ]; then
(
    echo "The PHP script at ${url} output something:"
    cat "${stdout_path}"
) | mail -s "${mail_subject}" ${target_addr}
fi

unlink "${stdout_path}"
unlink "${stderr_path}"

exit 0
```

You have then to enter the script into the crontab.

```
[\\]$ crontab -l
# Drupal's "cron.php" for ISA / JoinUp
\\*/15 * * * * /absolute/path/to/joinup-cron.sh
```

... that will run the cron.php script and send mails to adequate people in case of failure.

4. Source Code Management

4.1. Components

The Source Code Management is composed of a subversion instance, a database and a Apache HTTPD server.

4.2. Prerequisites

You need to have an Apache HTTPD instance, a subversion daemon and a MySQL database.

4.2.1. Required Apache modules

- Ensure the Apache is compiled with the MPM prefork. Indeed, mod_dav_svn does not appear to be thread-safe and thus can not be used along with the MPM worker.

```
$ httpd -l | egrep '(prefork|worker)'
prefork.c
```

- Ensure the following modules are enabled:

```
LoadModule authnz_external_module modules/mod_authnz_external.so
LoadModule dav_module mod_dav.so
LoadModule dav_fs_module modules/mod_dav_fs.so
LoadModule dav_lock_module mod_dav_lock.so
LoadModule dav_svn_module mod_dav_svn.so
```



Note mod_authnz_external is not a standard Apache module and has to be fetched from [its official website](#).

4.3. Installation procedure

4.3.1. Directories and filesystems

Under the Apache DocumentRoot (\${webroot}), create the following directories:

- misc: this directory will host the DAV lock database (called dav_lock.db).
- scripts: this directory will host the "isasvn" scripts. Their setup is described later in this document.
- scripts/tmp: this directory will store temporary and/or lock files for the isasvn scripts. It must be writable by the Apache user.
- subversion: this directory is dedicated to host the Subversion repositories. It must be writable by the Apache user.
- webdav: this directory is dedicated to host the WebDAV directories themselves. It must be writable by the Apache user.

4.3.2. Scripts description

- bin/pwauth is a PHP script that will be called by the Apache daemon to tell whether a given user/password is allowed to access a given Subversion path. Data are passed as environment variables.
- bin/svncommit will be called through a SVN hook for each new SVN commit in order to register it in the Drupal database.
- bin/svncron will be run regularly through Cron to create new SVN repositories.

4.3.3. Database setup

On the front end database, you should create a new user adapted to subversion, having only the usage privileges (select, insert, update, delete).

```
mysql_command="mysql \-h DATABASE_HOST \-u DATABASE_USER \-p DATABASE_PASSWORD"
${mysql_command}
mysql> CREATE USER 'subversion_user'@'%' IDENTIFIED BY 'Password';
mysql> GRANT SELECT, INSERT, UPDATE, DELETE ON * . * TO 'subversion_user'@'%'
IDENTIFIED BY 'Password' WITH MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0;
```

In order to run, the service is attached to a database which is **mandatory** local to the subversion service.

```
mysql_svn_command="mysql \-h DATABASE_SVN_HOST \-u DATABASE_SVN_USER \-p
DATABASE_SVN_PASSWORD"
${mysql_svn_command}
mysql> CREATE DATABASE joinup_subversion COLLATE utf8_unicode_ci;
mysql> CREATE USER 'joinup_subversion'@'localhost' IDENTIFIED BY 'Password';
mysql> GRANT ALL ON joinup_subversion.* TO 'joinup_subversion'@'localhost' IDENTIFIED
BY 'Password' WITH MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0;
```

4.3.4. Scripts setup

The subversion service comes with a bunch of scripts available on the Subversion repository of the Joinup project [https://joinup.ec.europa.eu/svn/joinup/tags/\\${version}/isasvn](https://joinup.ec.europa.eu/svn/joinup/tags/${version}/isasvn) (to be adapted for later versions)

Export the content of this directory into \${webroot}/scripts/isasvn.

4.3.5. Scripts configuration

Most SVN-related scripts read the conf/isasvn-conf.inc.php file:

```

<?php
/*
  Common configuration for Joinup/SVN scripts.
*/
// The local server is a MySQL database located on the current machine
// It does not contain anything but a cache table used to check whether a given
// user/password couple is valid or not.
define('LOCAL_SERVER',      'localhost'); // connection through Unix socket
define('LOCAL_DB_USERNAME', 'USER_LOCAL_DATABASE'); // we created previously the
account 'joinup_subversion'
define('LOCAL_DB_PASSWORD', 'PASSWORD_LOCAL_DATABASE');
define('LOCAL_DB_NAME',     'joinup_subversion');

// The ISA server is the MySQL database of the ISA Drupal application.
define('ISA_SERVER',        'DATABASE_HOST');
define('ISA_DB_USERNAME',   'DATABASE_SVN_USER'); //We created previously the user
'subversion_user'
define('ISA_DB_PASSWORD',   'DATABASE_SVN_PASSWORD');
define('ISA_DB_NAME',       'joinup');

// This directory will host lock files
define('LOCK_DIR', '/absolute/path/to/document/root/scripts/tmp'); // Replace by the
absolute path to document root

// Name of the table storing repositories operations
// Note this value must include the Drupal table prefix, if any
define('REPOSITORIES_MANAGEMENT_TABLE', 'repositories_management');
// Name of the table storing commit operations
// Note this value must include the Drupal table prefix, if any
define('COMMIT_MANAGEMENT_TABLE', 'commit_management');

// Parent directory for all Subversion repositories
// Note the directory must exist and be writable.
define('SVN_ROOT_DIR', '/absolute/path/to/subversion');
// Skeleton directory when creating a Subversion repository
// Do not define this constant if you do not want repositories to be initialized this
way
define('SVN_SKELETON', dirname(__FILE__) . '/svn-skeleton');
define('SVN_SKELETON_COMMIT_MESSAGE', 'Repository initialization');
define('SVN_SKELETON_COMMIT_AUTHOR', 'Joinup Subversion service');

// This prefix will be stripped from the received URI when authenticating a user.
// the service is given through the AuthExternalContext Apache directive
$service = ($_ENV['CONTEXT'] == 'subversion') ? 'svn' : 'webdav';
//Allows to use different permissions
define('AUTH_CONTEXT', $service);

if (preg_match("/^(\\/(?:(?:isa|joinup)-$service)/", getenv('URI'), $matches)) {
  $prefix = $matches[1];
  define('SVN_URI_PREFIX', $prefix);
}

```


4.3.6. Authentication through "pwauth" setup and configuration

For the pwauth script to work, create the structures contained in doc/auth_cache.sql.

 As soon as you have an existing SVN repository (e.g. "deploytest"), you can test pwauth this way:

```
$debug = TRUE;
$debug_file = '/tmp/debug.txt';
if ($debug) {
  if (file_exists($debug_file)) {
    @chmod($debug_file, 0600);
  }
  $fh = fopen($debug_file, 'a');
  fwrite($fh, sprintf('[%s] authenticating %s/%s for %s (%s)' . "\n", date('d-m-Y
H:i:s'), getenv('USER'), getenv('PASS'), getenv('URI'), $prefix));
  fwrite($fh, print_r($_ENV, TRUE));
  fclose($fh);
}
```

This should output either 0 (access granted) or 1 (access denied).

 For security reasons, this debug shall be turned off as soon as the authentication is working and the /tmp/debug.txt file shall be deleted.

4.3.7. "svncron" specific configuration

The `$(webroot)/scripts/isasvn/conf/post-commit.tmpl` file is copied in each newly created repository (in its "hooks" subdirectory) by svncron. The content of the `$(webroot)/scripts/isasvn/conf/svn-skeleton` directory is imported as an initial commit for each newly created repository.

1. Edit `conf/post-commit.tmpl` and fix the paths it contains (php + svncommit itself).
2. Require the svncron script to be scheduled (don't forget to remove the variable and replace it by the real value):

```
[\\]$ crontab -l
# Subversion scripts
*/1 * * * * WEBROOT/scripts/isasvn/bin/svncron > /dev/null 2>&1
```

5. Apache configuration file

Integrate the following directives in Apache configuration file:

```
# External authentication configuration
<IfModule mod_authnz_external.c>
  # This script allows authenticating WebDAV and Subversion users against a Drupal
  database
  AddExternalAuth pwauth WEBROOT/scripts/isasvn/bin/pwauth
  # Data are passed to the script as environment variables
  SetExternalAuthMethod pwauth environment
  <Directory "WEBROOT/scripts">
    Order Deny,Allow
    Deny From All
  </Directory>
</IfModule>

# WebDAV configuration
<IfModule mod_dav_fs.c>
  DavLockDB WEBROOT/misc/dav_lock.db
  # Prevents direct access to data stored in the DocumentRoot
  <Directory "WEBROOT/misc">
    Order Deny,Allow
    Deny From All
```

```
</Directory>

# WebDAV root directory
Alias /webdav "WEBROOT/webdav"
<Directory "WEBROOT/webdav">
    DAV on
    Options Indexes FollowSymLinks
    <IfModule mod_authnz_external.c>
        AuthType Basic
        AuthName "WebDAV service"
        AuthBasicProvider external
        AuthExternal pwauth
        AuthExternalContext webdav
        Require valid-user
    </IfModule>
</Directory>
</IfModule>

# Subversion configuration
<IfModule mod_dav_svn.c>
    # Subversion root directory
    <Location /svn>
        DAV svn
        SVNParentPath WEBROOT/subversion
        Options Indexes FollowSymLinks
        <LimitExcept GET OPTIONS PROPFIND REPORT>
            <IfModule mod_authnz_external.c>
                AuthType Basic
                AuthName "Subversion service"
                AuthBasicProvider external
                AuthExternal pwauth
                AuthExternalContext subversion
                Require valid-user
            </IfModule>
        </LimitExcept>
    </Location>

# Prevents direct access to data stored in the DocumentRoot
<Directory "WEBROOT/subversion">
    Order Deny,Allow
    Deny From All
```

```
</Directory>
</IfModule>
```



In the Subversion configuration: it is not possible to fully merge the <Location> directives in a single <LocationMatch> because the SVNParentPath directive has a tricky behaviour when it comes to extract the SVN path from the received request URI. See <http://svn.haxx.se/dev/archive-2005-09/0470.shtml> for details. However, the <LimitExcept> and Options could be moved to a common <LocationMatch>.

6. Build and Artifacts Repository

6.1. Components

The Build and Artifacts Repository is composed of a maven repository built on Sonatype Nexus instance.

6.2. Prerequisites

In order to install the nexus, you need to have the following components:

- Tomcat 6 up and running
- MySQL 5.x
- Joinup Stored procedure (installed with the Joinup Database)

6.3. Installation procedure

6.3.1. MySQL DataSource on Tomcat

Copy the MySQL JDBC driver to the \$CATALINA_HOME/lib folder, you can get this Java connector from the page [MySQL connector](#).

From the Archive got, use the mysql-connector-java-x.x.x-bin.jar which should be put in the \$CATALINA_HOME/lib folder.

Add the following xml element to the tomcat context.xml file

```
<Resource
name="jdbc/drupal"
auth="Container"
type="javax.sql.DataSource"
maxActive="100"
maxIdle="30"
maxWait="10000"
validationQuery="SELECT 1"
testOnBorrow="true"
username="maven_user"
password="Password"
driverClassName="com.mysql.jdbc.Driver"
url="jdbc:mysql://DATABASE_HOST:DATABASE_PORT/DATABASE" />
<!-- Replace DATABASE_HOST / DATABASE_PORT / DATABASE by the value of your drupal
database -->
<!-- maven_user is the user created previously which only have the right to execute
the stored procedure -->
```

and fill in the following attributes

- username
- password
- url

Finally, restart Tomcat

6.3.2. Installation

1. Download the web application from the Joinup Nexus <https://joinup.ec.europa.eu/nexus/content/repositories/releases/eu/europa/ec/joinup/joinup/nexus-webapp/1.9.2-joinup-1.0.0/nexus-webapp-1.9.2-joinup-1.0.0.war>
2. Rename the file nexus.war
3. Deploy the war using the tomcat manager

6.3.3. First time configuration

Navigate to the Nexus console <https://web/path/to/nexus/index.html>

1. Log in using the default admin credentials: admin/admin123
2. Change admin password (Security > Change password)

6.3.3.1. Server configuration

In the Administration > Server part

1. Configure the SMTP settings
2. In the Security settings, select "Drupal realm" and move it from the available realms to selected realms
3. In the Application server settings, set the final URL of your application (i.e. if you have a mapping, enter the mapped URL)
4. In the Default HTTP proxy settings, set the proxy information if necessary (including the authentication)

6.3.3.2. Allow authenticated JoinUp users to log in the User Interface

As admin, ensure the "Authenticated user Drupal" role exists in Security > Roles; if not, click Add "External Role Mapping" and fill the first form this way:

- Realm: Drupal User Manager
 - Role: Authenticated user Drupal
- Click "Create Mapping".

The second form is automatically created with the following values:

- Role Id: drupal_authenticated
- Name: Authenticated user Drupal
- Description: External mapping for Authenticated user Drupal (Drupal)

Click "Add" and check:

- Nexus Anonymous Role
- Repo: All Repositories (Read)
- UI: Base UI Privileges
- Artifact Upload

Click "OK" then "Save".

The mapping between Joinup and Nexus is now active. To provision the maven repository you have to make some specific request for each project. It is detailed in the Maven Repository Management document.

7. Mailing List

7.1. Components

The mailing list component is aimed to offer a service list for the user to have online discussion through emails. There is an interconnection with the Front End in two manners:

- Each project generates a mailing list and might have more mailing list per project
- Each mailing list generates a log in the "mailing list" part of the forum on the Front End.

7.2. Prerequisites

You need to have a fully installed and configured Mailman (+Web interface), MTA (exim, postfix or other), antivirus if necessary. The machine must also have Apache HTTPD, PHP 5.3.x and an access to the Front End database.

This document is only explaining how to interconnect the mailman with the Front End.

7.3. Installation procedure

7.3.1. Obtaining a POP3-enabled mailbox

The Front End will read the data coming from a POP3 mailbox, you can configure it locally to your MTA.

7.3.2. Configure and enable the mailbox within the Front End

7.3.2.1. Configuration:

The following code needs to be executed on the Front End database

```
{mysql_command}
mysql> INSERT INTO `mailhandler` (`mid`, `mail`, `domain`, `port`, `name`, `pass`,
`security`, `replies`, `fromheader`, `commands`, `sigseparator`, `enabled`, `folder`,
`imap`, `mime`, `mailto`, `delete_after_read`, `extraimap`, `format`,
`authentication`)
VALUES (1, 'EMAIL_ADDRESS', 'POP3_SERVER', POP3_PORT, 'POP3_USERNAME',
'POP3_PASSWORD', 0, 0, 'From', '', '', 1, 'INBOX', 0, 'TEXT/HTML,TEXT/PLAIN',
'EMAIL_ADDRESS', 1, '', 2, 'isa_handler');
mysql> exit;
```

(!)You need to replace the values of EMAIL_ADDRESS, POP3_SERVER, POP3_PORT, POP3_USERNAME and POP3_PASSWORD by your configuration values

You can check your configuration using the URL: <http://url/to/your/front/end/admin/content/mailhandler/edit/1>

7.3.3. Installation of the scripts

This step consists in installing the php scripts which will automatically create the mailing list associated to the projects.

```
mlscripts_path=/path/to/mailing/list/scripts
mkdir ${mlscripts_path}
cd ${mlscripts_path}
svn export --force https://joinup.ec.europa.eu/svn/joinup/tags/${version}/isaml/
```

7.3.4. Configuration of the scripts

The scripts are configured with PHP code, edit the file located in the folder \${mlscripts_path}/conf and set the values:

- ISA_SERVER: the database host
- ISA_DB_USERNAME: the database username
- ISA_DB_PASSWORD: the database password
- ISA_DN_NAME: the name of the database

Depending on your mailman configuration, you might need to modify the configuration related to the variable ML_*. You need to set the path the associated tools.

7.3.5. Configuration of the cron jobs

The scripts are run by the cron every x minutes (by default, we set it to 2).

```
[\\]$ vim /etc/cron.d/isaml
MAILTO=supervision.email.adapted.to.your.environment@your.domain
*/2 * * * * list /path/to/mailing/list/scripts/bin/mlcron
```

8. Indexation and Search

8.1. Components

The indexation system is composed of two elements. Solr will index the content coming from the Front-End and Tika will look through the documents attached to the content of the Front End.

8.2. Prerequisites

In order to run the indexation and search service, you need to have:

- Tomcat 6 up and running
- The Front End must have a JRE 6 for the indexation within the attachments

8.3. Installation procedure

8.3.1. Directories and filesystems

```
# Create the destination folder
solr_path="path/to/solr"
mkdir ${solr_path}
cd ${solr_path}
# Download Solr
wget http://apache.crihan.fr/dist/lucene/solr/3.1.0/apache-solr-3.1.0.tgz
# Extract the instance example and create the "solr" path
tar xzv --strip-components=3 -f apache-solr-3.1.0.tgz apache-solr-3.1.0/example/solr
# and the Solr war
tar xzv -O -f apache-solr-3.1.0.tgz apache-solr-3.1.0/dist/apache-solr-3.1.0.war >
apache-solr.war

# Download the "apachesolr" Drupal module
wget http://ftp.drupal.org/files/projects/apachesolr-6.x-1.5.tar.gz
# extract solrconfig.xml and schema.xml and insert them into the Solr home conf
directory
mv conf/solrconfig.xml conf/solrconfig.xml.old
tar xzv -O -f apachesolr-6.x-1.5.tar.gz apachesolr/solrconfig.xml >
conf/solrconfig.xml
mv conf/schema.xml conf/schema.xml.old
tar xzv -O -f apachesolr-6.x-1.5.tar.gz apachesolr/schema.xml > conf/schema.xml

# We'll need a data directory to store indexes generated by Solr and a lib directory
if we want to use plugins in the future.
mkdir -p {lib,data}
```

Ensure at least the "data" directory is writable by the tomcat web user.

8.3.2. Tomcat setup

We now need to deploy the Solr application above Tomcat. We will add a new context to the Catalina configuration: \$CATALINA_HOME/conf/Catalina/localhost/solr.xml. It should contain:

```
<?xml version="1.0" encoding="utf-8"?>
<Context docBase="/path/to/solr/apache-solr.war" debug="1" crossContext="true">
  <Environment name="solr/home" type="java.lang.String" value="/path/to/solr/"
  override="true"/>
</Context>
```

Restart Tomcat.

Your solr application is now available at the URL http://TOMCAT_SOLR_HOST:TOMCAT_SOLR_PORT/solr.

8.3.3. Link of Solr on the Front End

Tika is already available in the source available for the front end (folder sites/all/modules/contributed/modified/apachesolr_attachments/tika/). In order to activate it, you have to set up the following elements in the settings.php configuration file of the Front End application:

```
// Solr-dedicated, environment-specific configuration
$config['apachesolr_host'] = 'TOMCAT_SOLR_HOST'; //Host which handle the solr
configuration
$config['apachesolr_port'] = 'TOMCAT_SOLR_PORT';
$config['apachesolr_path'] = '/solr';
// Tika requires a JRE
$config['apachesolr_attachments_java'] = '/path/to/jre/6/bin/java';
```