Security Access Control Configuration

The Admin UI can be configured to use security. When configured, whenever a user tries to access the UI via a browser, they will be redirected to the Login page.
After the session is authenticated successfully, access to all pages can continue as normal.

Since version 4.0, Aspire has role-based security access which provides an option to disable some users from executing sensitive actions on Aspire.

On this page

> **Important**
>
> This is important since Aspire contains credentials with Read permissions to most parts of a document repository, which may contain documents that should remain restricted even from Aspire Administrators.

The role system on Aspire consists of two different roles:

- Administrator - with unrestricted "view" permissions but limited "execution" and "modification" permissions
- Developer - with unrestricted access to everything

The following table illustrates what an Administrator and a Developer can and can not do.

Task	Administrator	Developer
View content sources / services	•	•
Execute crawls	•	•
View errors / statistics	•	•
Modify a schedule	•	•
Add new content source / service		•
Export content source / service	•	•
Duplicate content source / service		•
Import content source / service		•
Modify content source / service configuration	•	•
Modify Workflow		•
Application / bundle control from debug console page		•
Groovy script execution from debug console page		•
XSLT Transformation test execution from debug console page		•

> ⚠ By default, security is disabled.

Configuration

Access the **settings.xml** file in your distribution under the **config** folder.

1. Add the following element (if not present):

```
<authentication>
  <type>None</type>
</authentication>
```

2. Change the authentication type based on your needs. Currently, there are three options:

1. None
   - No security is used. Access to administration pages is unrestricted.
2. ConfigFile
   - Access to administration pages is restricted. There are two default users: "*admin*" and "*developer*".

- These users are able to access administration pages after successfully entering a password that is configured in the top level application properties (inside the properties section) of the settings file.

3. LDAP
   - Access to administration pages is restricted. Aspire is configured to connect to an LDAP or Active Directory (AD) server. Users can access administration pages after being successfully validated by the LDAP or AD server.
   - Access to the administration pages can be further restricted to only users that are members of a specific LDAP or AD group, which can be configured to any role (Administrator or Developer).

ConfigFile Authentication - Set Up a Password

When using *ConfigFile* authentication, you must set up a password that the *Administrator or Developer* user will be validated against.

The password is set in the system properties in the settings file in the *adminPassword & developerPassword* properties.

An example is shown below:

```
<!-- System properties -->
<properties>
  <property name="adminPassword">encrypted:4E23628AFEEFA56C507DD86985C12E69</property>
  <property name="developerPassword">encrypted:4E23628AFEEFA56C507DD86985C12GF8</property>
</properties>
```


Security Recommendation

For security reasons, it is a must that you encrypt these passwords when setting the properties. See Password Encryption for details on how to encrypt the passwords.

Important note

If only the "adminPassword" is configured, the "admin" user will function as "Developer" because of backwards compatibility with previous versions of the settings.xml.

LDAP Authentication - Configuration

When *LDAP* authentication is selected, you must extend the *authentication* section to provide the LDAP server to authenticate with. By default, *group based access* is disabled, meaning that any user who is able to authenticate with the given LDAP server will be able to access the Aspire administration pages as a "*Developer*".

To *enable group based access*, you must further extend your configuration, adding a distinguished name (dn) for each of the groups that controls access, an LDAP query that will allow Aspire to establish the distinguished name (dn) of the user attempting log on from the user name they provided, the name of the attribute that holds the user to group membership information and an indicator as to whether that information is held in the user object or in the group object.

User specifc access is also available by extending the configuration, you can add one or more specific users.

You can configure any number of users and/or groups for the "*Administrator*" or "*Developer*" roles.

LDAP Configuration Parameters

The following parameters may be used when configuring *LDAP* authentication:

Parameter	Type	Default	Description
/authentication /ldap/serverUrl	String*		The host and port of the LDAP server to validate against
/authentication /ldap/protocol	String		
/authentication /ldap /contextFactory	String	com.sun.jndi. ldap. LdapCtxFactory	The context factory class
/authentication /ldap /authentication	String*	anonymous	LDAP authentication to use (anonymous/simple/DIGEST-MD5)
/authentication /ldap/readTimeout	Integer	5 minutes	The timeout when making reads. Numbers without suffixes are considered as milliseconds. Suffixes of ms (milliseconds), s (seconds), m (minutes), h (hours), or d (days) may be used
	Integer	1 minute	The timeout when connecting. Numbers without suffixes are considered as milliseconds. Suffixes of ms (milliseconds), s (seconds), m (minutes), h (hours), or d (days) may be used

/authentication /ldap /connectTimeout			
/authentication /ldap /connectionPool	Boolean	True	Specifies if a connection pool is used
/authentication /ldap/referralType	String	Ignore	How LDAP referrals should be handled (ignore/follow)
/authentication /ldap/searchBase	String	dc=search, dc=local	The search base for LDAP queries for the user's dn
/authentication /ldap /adminGroupDN	String		Holds the dn of one group that may access Aspire with "Administrator" role. NOTE: currently, the users must be direct members of this group (not a member of a group that is a member of this group). Multiple adminGroupDN can be used if necessary.
/authentication /ldap /adminUserDN	String		Holds the dn of one user that may access Aspire with "Administrator" role. Multiple adminUserDN can be used if necessary.
/authentication /ldap/devGroupDN	String		Holds the dn of one group that may access Aspire with "Developer" role. NOTE: currently, the users must be direct members of this group (not a member of a group that is a member of this group). Multiple devGroupDN can be used if necessary.
/authentication /ldap/devUserDN	String		Holds the dn of one user that may access Aspire with Developer role. Multiple devUserDN can be used if necessary.
/authentication /ldap /groupsHoldMembers	Boolean	False	Indicates that LDAP group objects hold the membership information. By default, user objects are expected to hold group membership
/authentication /ldap /userDNQuery	String*		The LDAP query to use to find the dn for the user that logged in. The information provided by the user as they logged in is available for substitution. See below for details.
/authentication /ldap/memberAttr	String	memberOf	The attribute of the LDAP object (user or group) that holds information about group membership
/authentication /ldap /defaultDomain	String		A domain to be added to usernames when logging in from the Administration UI if no domain is given

*** Mandatory**

> **Permissions Settings**
>
> If any user falls in to both "Developer" and "Administrator" roles because of the way it is configured, it will be granted "Developer" access.

Group Based Access Control

When *group based access* control is disabled, a simple validation of the credentials supplied by the user is all that is required to allow access to the Admin UI, and the user will be granted "*Developer*" role permissions. Adding the dn of a group in to the configuration will enable *group based access* control and, following a successful validation of credentials, the LDAP server is queried to see if the user belongs to the desired group. Two slightly different approaches are used depending on the setting of the *groupsHoldMembers* flag.

By default, *groupsHoldMembers* is disabled. In this configuration, Aspire queries the LDAP server to get the user object using the query specified in the *userDNQuery* parameter. Once the user object has been found, the membership attribute (configured by the *memberAttr* parameter) is extracted and the values of this attribute are checked. If one is equal to the *adminGroupDN or devGroupDN groups* from the configuration, the user belongs to the group and is granted access. Otherwise access is denied.

When *groupsHoldMembers* is enabled, Aspire again searches LDAP for the user object. It then gets the membership attribute (configured by the *memberAttr)* parameter from the group (using the dn configured in the *adminGroupDN or devGroupDN* parameter) and looks for the dn of the user object. If found, the user belongs to the group and access is granted.

> In both of the above scenarios, the user must be a direct member of the group configured in the *adminGroupDN or devGroupDN*, not an indirect member (not in a group that is a member of the configured *adminGroupDN or devGroupDN*).

User Specific Access Control

Sometimes it is simpler to grant access to certain specific users instead of granting access to a whole group. In this case, the *adminUserDN and devUserDN* properties should be used. They should contain the distinguished name (dn) of each one the users to allow access with either "*Administrator*" or "*Developer*" roles.

The user specific access can be used combined with the group based access control.

The User DN Query

When a user logs in to the user interface, they are first validated against the LDAP server using the username and password they supplied. If *group based access* control is disabled, no further checks are performed and the user is granted access (assuming their username and password are valid). If *group based access* control is enabled, following successful validation by the LDAP server, Aspire then needs to establish the distinguished name of the user who logged on in order to determine if the user is in the appropriate group.

The dn of the user is found by performing a query in LDAP for the user, based on the user name used to login. The query entered in the configuration may contain 'parameters' that are then substituted. The following parameters are available:

fullname	The full username entered by the user in the login page
username	The name entered by the user in the login page, less any domain. Thus an entry of *myDomain\myUser* would become *myUser* in this parameter
domain	The domain from the username entered. An entry of *myDomain\myUser* would become *myDomain* in this parameter

Parameters are entered in the query by enclosing them in curly braces {}. For example, *(&(objectClass=person)(sAMAccountName={username}))* woul d become *(&(objectClass=person)(sAMAccountName=Administrator))*, if the user logged in with either *domain\Administrator* or *Administrator*.

When entering queries in the settings file, you will need to use a <![CDATA[]]> around the query or escape any special characters such as &.

Example Configuration for LDAP Access Control

Below is an example of a configuration allowing any valid LDAP user to log in to the Aspire interface.

```
<authentication>
  <type>Ldap</type>
  <ldap>
    <server>ldap://myLdapServer:389</server>
    <authentication>simple</authentication>
  </ldap>
</authentication>
```

Example Configuration for LDAP Group Access Control

Below is an example of a configuration allowing any validated LDAP user who is a member of the group with the distinguished nane *CN=Administrator s, CN=Builtin, DC=qa, DC=local* to log in to the Aspire interface with "*Administrator*" role and the *CN=Developers, CN=Builtin, DC=qa, DC=local* dn for the "Developer" role. In this configuration, once the user is validated, the *userDNquery* is used to locate the user object under the search base. The values of the attribute configured in the *memberAttr* parameter (default *memberOf*) are checked against the *adminGroupDN* and *devGroupDN* values and if it is found, the user is granted access according to the specified role.

```
<authentication>
  <type>Ldap</type>
  <ldap>
    <server>ldap://myLdapServer:389</server>
    <authentication>simple</authentication>
    <searchBase>dc=qa, dc=local</searchBase>
    <userDNQuery><![CDATA[(&(objectClass=person)(sAMAccountName={username}))]]></userDNQuery>
    <adminGroupDN>CN=Administrators, CN=Builtin, DC=qa, DC=local</adminGroupDN>
    <devGroupDN>CN=Developers, CN=Builtin, DC=qa, DC=local</devGroupDN>
  </ldap>
</authentication>
```

Below is an example of a configuration allowing any valid LDAP user who is a member of the group with dn *CN=Administrators, CN=Builtin, DC=qa, DC=local* to log in to the Aspire interface. In this configuration, the membership information is taken from group object (due to the *groupsHoldMembers* flag). In this case, once the user is validated, the *userDNquery* is used to locate the user object under the search base and establish the user's distinguished name (*dn*). Next, the group object is retrieved using its distinguished name (from the *adminGroupDN and devGroupDN* parameters). The values of the attribute configured in the *memberAttr* parameter (default *memberOf*) are checked against the user's *dn* and if found, the user is granted access.

In the following example, we also have two additional parameters: adminUserDN and devUserDN containing the DN of users that do not belong to either the Developers or Administrators groups on LDAP, but are still granted access according to role:

- *jdoe* will be granted access as "*Administrator*" role even though it doesn't belong to "*CN=Administrators, CN=Builtin, DC=qa, DC=local*"
- *aeinstein* will be granted access as "*Developer*" role even though it doesn't belong to "*CN=Developers, CN=Builtin, DC=qa, DC=local*" either

```
<authentication>
  <type>Ldap</type>
  <ldap>
    <server>ldap://myLdapServer:389</server>
    <authentication>simple</authentication>
    <searchBase>dc=qa, dc=local</searchBase>
    <userDNQuery><![CDATA[(&(objectClass=person)(sAMAccountName={username}))]]></userDNQuery>
    <adminGroupDN>CN=Administrators, CN=Builtin, DC=qa, DC=local</adminGroupDN>
    <devGroupDN>CN=Developers, CN=Builtin, DC=qa, DC=local</devGroupDN>
    <adminUserDN>UID=jdoe, CN=SomeOtherGroup, CN=Builtin, DC=qa, DC=local</adminUserDN>
    <devUserDN>UID=aeinsten, CN=SomeOtherGroup, CN=Builtin, DC=qa, DC=local</devUserDN>
    <groupsHoldMembers>true</groupsHoldMembers>
    <memberAttr>member</memberAttr>
  </ldap>
</authentication>
```