LDAP Cache

The *LDAP Cache* is a stage that caches groups from an LDAP server and other information for use in Group Expansion Service. After the information has been cached, it can be retrieved using the generic group expansion client, which responds to group expansion request messages.

The cache can also be used by connectors that require a list of external groups from LDAP to properly populate their own group expansion caches.

LDAP Cache	
Factory Name	com.searchtechnologies.aspire:aspire-ldap-cache
subType	default
Inputs	A Job containing a group expansion request
Outputs	LDAP and group information via the LDAPGroupExpansionServer.java and ExternalGroupServer.java interfaces

Information held by the cache

The *LDAP Cache* holds three sets of information:

- Group Expansion cache
  - A map of user names against the LDAP or Active Directory groups to which they belong.
- LDAP cache
  - A map of username against LDAP or Active Directory attributes for the user. Note that all attributes are held as strings and any binary attributes will be encoded using Base32.
- User/Group cache
  - A list of users and groups found in LDAP or Active Directory.

All information is persisted to disk, allowing the cache to be populated from the time Aspire starts.

Cache warming

All three of the caches are refreshed at the same time. A scheduler periodically send an Aspire job to the cache component which begins the process of rebuilding the caches. Once the caches are complete, they are *swapped in* to use for cache requests. While the process of rebuilding is ongoing, the last set of caches which built successfully will still be in use.

The rebuilding process will gather all of the users and groups from LDAP or Active Directory, either by connection to a server or by running the configured script and reading its output.

Cache warming request

In order to have the *LDAP Cache* (re)populate its caches, it is sent an Aspire job containing and AspireObject with the following format:

```
<doc actionProperties="cacheGroups"/>
```

Cache warming algorithm

During configuration, the administrator supplies the names of a number of attributes that define how to link groups to users, including whether user objects hold references to the groups to which they belong or if the groups hold reference to the users that are members.

This information is then used to construct a map of users against the groups to which they belong. As ldap information is downloaded from the server, or read from a script response, it is inserted in to a temporary cache with a key taken from the attribute specified in the *user key* or *group key* attribute. At the end of this stage, we have a map of the unique identifiers against the LDAP or Active Directory object themselves.

What happens next depends on whether user objects hold references to the groups to which they belong or if the groups hold reference to the users that are members. If user objects hold references to groups, then each user is processed in turn. The group name is taken from the value of the *group name* attribute and the value(s) of the (repeated) attribute given as the *group mapping* are collected. These values hold the unique identifier of the groups to which the user belongs. These identifiers are looked up in the temporary cache to get the group objects and the names of these are found by looking at the value of the *group name* attribute of each group. Finally (as we now have a user name and a set of group names) the user and groups are added to the group expansion cache.

If groups hold references to users that belong to it, the process is similar, but instead the groups are processed in turn. Their *group mapping* attribute is found to give the identifiers of the member user objects. These users are located from the temporary cache and the names obtained from *user name* attributes. Then the group and users are added to the group expansion cache.

When the group expansion cache is complete, we can extract information to fill the other caches – the cache of users against ldap information and the external user and group list.

Note: The above assumes that the attributes are populated with values. With both server and script processing, you must ensure that the attributes you choose are present and populated to make the expansion work correctly.

Using a script

The ldap cache can be configure to use a script to obtain its user and group information. This script could be a Windows batch file, Power Shell or executable, or Linux shell script or executable.

When the cache rebuild begins and the ldap cache is configured to use a script, the cache component creates an empty temporary file and passes the file name to the script. The script can return either xml or json and should write its output the given file. The format of the xml or json is given below.

The script can do whatever is required to gather the users and groups (for example connect to a database, ldap server or other repository) and then write the information to the file. Once the script completes, the ldap cache component will read the information from the file and load the groups. If the script encounters an error, it can report this back to the component via the file.

Assuming that the processing of the file completes successfully, the temporary file is deleted. If the ldap cache encounters an error while processing the file, the file is left on disk to aid debugging.

File format

The format of the file holding results when a script is used can be found here

Specifying Group Membership

When using a script, it is possible for you to return information that defines the groups to which a user belongs, rather than have the cache calculate it for you. Details on how to do this can be found here

Getting information from the cache

User/Group Cache & LDAP information

This component implements the LDAPGroupExpansionServer.java interface, which is an extension of the GroupExpansionServer.java interface. To retreive the information from the user group cache, use the GroupExpansionServer.java interface. If you want to get LDAP attributes for the user, use the LDAPGroupExpansionServer.java interface.

When a group expansion client connects to this component, it will add both the user/group in formation and the LDAP information.

External groups

Connectors requiring a list of external groups will use the ExternalGroupServer.java interface.

Configuration

The LDAP cache component takes the following configuration flags

Element	Type	Default	Description
useScript	boolean	false	Configure the cache to be populated by running a script and reading its output from a file
script/file	string	**[Required (script)]**	The name of the script to run to populate the file (if using a file)
script/file/@json	boolean	false	If true and using a script, treat the output written to the file as JSON. If false, treat the output written to the file as XML
ldap/component	string	**[Required (ldap)]**	when not using a script, the path to LDAP server (component) that is used to get the LDAP ssers & groups
useSearchBase	boolean	false	If true, the searchBase will be the base domain of the LDAP server. If false, multiple domains will be searched.
searchBase	string	**[Required (ldap)]**	The base directory in the LDAP for searches. Normally this is the domain of the LDAP server.
scope	int	2	The scope of the LDAP for searches. 0 = search base only, 1 = search base and immediate children, 2 = subdirectory
ldap/users/query	string	**[Required (ldap)]**	The LDAP query used to find all users to be cached
ldap/users /attribute	string	<all>	The LDAP attributes to be retrieved and stored in the cache for users
ldap/groups /query	string	**[Required (ldap)]**	The LDAP query used to find all users to be cached
ldap/groups /attribute	string	<all>	The LDAP attributes to be retrieved and stored in the cache for groups
attributes/user /@key	string	dn (a pseudo attribute representing the object dn)	The attribute in LDAP that is the unique key for the user
attributes/user /@name	string	sAMAccountName	The attribute in LDAP that holds the account name
attributes/group /@key	string	dn (a pseudo attribute representing the object dn)	The attribute in LDAP that is the unique key for the group

attributes/group/@name	string	sAMAccountName	The attribute in LDAP that holds the account name
attributes/group/@mapping	string	memberOf	The attribute in LDAP that holds the groups for a user, or users for a group
groupsHoldMembers	boolean	false	If true, group objects reference their members (typically via a *uniqueMember* attribute). If false, user objects reference their groups (typically via a *memberOf* attribute).
userGUID	boolean	false	If true, the user GUID (binary attribute objectGUID) will be part of the returning attributes.
lowerCase	boolean	false	Tells if group names retrieved from LDAP should be changed to lower case.

Example configuration

Using LDAP

```
<component factoryName="aspire-ldap-cache" name="LdapCache" subType="default">
  <debug>true</debug>
  <ldap>
    <component>/LDAP_Cache/LDAPConnection</component>

    <useSearchBase>true</useSearchBase>
    <searchBase>dc=search,dc=local</searchBase>
    <scope>2</scope>
    <users>
      <query>(&(objectClass=user)(objectClass=organizationalPerson)(!(objectClass=computer)))</query>
      <attribute>cn</attribute>
      <attribute>sn</attribute>
      <attribute>c</attribute>
      <attribute>l</attribute>
      <attribute>title</attribute>
      <attribute>description</attribute>
      <attribute>telephoneNumber</attribute>
      <attribute>givenName</attribute>
      <attribute>memberOf</attribute>
      <attribute>sAMAccountName</attribute>
      <attribute>mail</attribute>
    </users>
    <groups>
      <query>(objectClass=group)</query>
      <attribute>sAMAccountName</attribute>
      <attribute>cn</attribute>
      <attribute>mail</attribute>
      <attribute>member</attribute>
    </groups>
  </ldap>
  <attributes>
    <user key="dn" name="sAMAccountName"/>
    <group key="dn" mapping="member" name="sAMAccountName"/>
  </attributes>
  <groupsHoldMembers>true</groupsHoldMembers>

  <userGUID>false</userGUID>
  <lowerCase>false</lowerCase>
</component>
```

Using a script

```
<component factoryName="aspire-ldap-cache" name="LdapCache" subType="default">
  <debug>true</debug>
  <useScript>true</useScript>
  <script>
    <file json="false">c:\ldap\download.bat</file>
  </script>
  <attributes>
    <user key="dn" name="sAMAccountName"/>
    <group key="dn" mapping="member" name="sAMAccountName"/>
  </attributes>
  <groupsHoldMembers>true</groupsHoldMembers>

  <userGUID>false</userGUID>

  <lowerCase>false</lowerCase>
</component>
```