**LDAP Authentication in HS2**

**(HIVE-7193)**

There are two important functional deficiencies in the current LDAP Authentication provider implementation.

**Group membership**: This enables HiveServer to enforce group membership for users. The authentication request will succeed if the user belongs to one or more of the groups listed in the hive configuration file. If the user does not belong to atleast on of the groups listed, the user authentication fails.

To support group membership based authentication, two new parameters will be added.

**hive.server2.authentication.ldap.groupDNPattern** :

This value represents a pattern for “distinguishedName”(DN) for groups in the directory. This value could be single DN if the LDAP Group entities are co-located or could be a COLON separated list of all DN patterns if the groups are scattered across different trees.

Each DN pattern can contain a “%s” in it that will be substituted with the group name (from the group filter) by the provider for group search queries.

Ex1:

*<property>*

*<name>*

*hive.server2.authentication.ldap.groupDNPattern*

*</name>*

*<value>CN=%s,OU=Groups,DC=apache,DC=org</value>*

*</property>*

This indicates that all LDAP Group entries are under the following directory root “OU=Groups,DC=apache,DC=org”

The LDAPAtnProvider replaces the %s with the group name in the ldap queries to locate the group entry.

For example:

If group by name “group1” is being queried for, it uses

CN=group1,OU=Groups,DC=apache,DC=org

Ex2:

*<property>*

*<name>*

*hive.server2.authentication.ldap.groupDNPattern*

*</name>*

*<value>*

*CN=%s,OU=Groups,DC=apache,DC=org:*

*uid=%s,CN=Users,DC=apache,DC=org*

*</value>*

*</property>*

The above pattern advices the LDAPAtnProvider that LDAP group entities can exist in two separate trees in the directory and can have different attributes in their DNs.

**hive.server2.authentication.ldap.groupFilter**:

This value represents the group name filter that is to be enforced by the LDAPAtnProvider. All individual groups are represented using a COMMA separated list. The user MUST belong to one or more of these groups for the authentication request to succeed.

Ex:

*<property>*

*<name>hive.server2.authentication.ldap.groupFilter</name>*

*<value>group1,group2</value>*

*</property>*

**User search list:** This enables HiveServer to restrict access to a specified list of users. If the user, being authenticated, is not part of this userlist, access will be denied. To support this feature, two new parameters will be added.

**hive.server2.authentication.ldap.userDNPattern**:

This value represents a pattern for “distinguishedName” (DN) for users in the directory. This value could be a single DN if the LDAP User entities are co-located within a single root or could be a COLON separated list of all DN patterns if the users are scattered across different trees/forests in the directory.

Each DN pattern can contain a “%s” in it that will be substituted with the username (from the user filter) by the provider for user search queries.

Ex1:

*<property>*

*<name>*

*hive.server2.authentication.ldap.userDNPattern*

*</name>*

*<value>*

*CN=%s,CN=Users,DC=apache,DC=org*

*</value>*

*</property>*

In the example above, all Users are co-located under a single root “*CN=Users,DC=apache,DC=org*”. To search for user “foo”, LDAPAtnProvider attempts to find the user with DN like “*CN=foo,CN=Users,DC=apache,DC=org*”

Ex2:

*<property>*

*<name>*

*hive.server2.authentication.ldap.userDNPattern*

*</name>*

*<value>*

*CN=%s,OU=Users,DC=apache,DC=org:*

*uid=%s,CN=UnixUsers,DC=apache,DC=org*

*</value>*

*</property>*

The above pattern advices the LDAPAtnProvider that LDAP user entities can exist in two separate trees in the directory and can have different attributes in their DNs.

**hive.server2.authentication.ldap.userFilter**:

This is a COMMA separated list of usernames to grant access to. The Atn provider grants access if the user being authenticated is part of this list, denies access otherwise.

Ex :

*<property>*

*<name>*

*hive.server2.authentication.ldap.userFilter*

*</name>*

*<value>*

*hive-admin,hive,hivetest,hive-user*

*</value>*

*</property>*

**Custom Query String**

**hive.server2.authentication.ldap.customLDAPQuery:**

There are several LDAP implementations available for commercial use, with no standard attributes within each implementation. If either of the above filters do not meet the requirements for some unforeseen reasons, HiveServer can use a user-specified LDAP Query string to execute against the LDAP server. The returned result will then be used to adjudicate a GRANT/DENY decision to the authenticating user. To support this configuration, a new property has been introduced.

Ex:

*<property>*

*<name>hive.server2.authentication.ldap.customLDAPQuery</name>*

*<value><![CDATA[(&(objectClass=person)(|(memberOf=CN=Domain Admins,CN=Users,DC=ent,DC=cloudera,DC=com)(memberOf=CN=Administrators,CN=Builtin,DC=ent,DC=cloudera,DC=com)))]]>*

*</value>*

*</property>*

The above query returns “All users that is a member of the one of the groups (Domain Admins or Administrators)”. This offers a lot more flexibility that allows hive users to get customize the LDAP configuration for their implementation of LDAP.

**Order of precedence**:

The group membership parameters can be used in conjunction with the user lists to enforce a stricter access. LDAP Atn provider adjudicates decision according to the following criterion

1. If hive-site.xml contains “*hive.server2.authentication.ldap.customLDAPQuery”,* ONLY the results of this query are used to adjudicate an authentication decision. All other property values (groupDNPattern, groupFilter, userDNPattern, userFilter) are ignored entirely.
2. if GroupFilter and UserFilter are both specified in the configuration file, access is granted IFF the user being authenticated satisfies BOTH conditions, access is denied otherwise. So the user has to be listed in the UserFilter AND the user MUST also belong to one of the groups listed in the GroupFilter.
3. If ONLY one of the filters ( (userDNPattern + userFilter) || (groupDNPattern + groupFilter) ) is specified, a decision is adjudicated based on where the user being authenticated satisfies the specified filter.
4. If neither GroupFilter nor UserFilter is specified in the configuration file, the provider attempts to search for the user in the LDAP directory within the *baseDN* tree. Access is granted if user has been found, denied otherwise. **IMPORTANT:** This implementation is a little more stringent compared to the prior implementation. In the prior implementation, if the baseDN was not provided, authentication would be granted if the provider is to be able to the bind to LDAP with the user.