..

Configuring User Authentication Using LDAP

Date published: 2020-10-30
Date modified: 2022-09-21

https://docs.cloudera.com/

Legal Notice

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Enabling LDAP authentication.. 4

Using LDAPS.. 8

Complex matching logic for group queries using LDAPGroupQuery().............8

Enabling LDAP authentication

About this task

Cloudera Data Visualization by default uses local account (basic) authentication where users must be created manually through the UI using the default admin user.

This authentication method can be supplemented to also enable LDAP authentication so that corporate credentials can be used to login to ML Data Viz instead.

Before you begin

Prepare your installation by collecting the values of the following LDAP configuration parameters:

Configuration Item	Description
AUTH_LDAP_SERVER_URI	LDAP server URI for example, "ldap://ldap.example.com".
AUTH_LDAP_BIND_DN	Username DN (Distinguished User) of the bind user account. This needs to be the full DN for the Bind User, not just the bind username.
AUTH_LDAP_BIND_PASSWORD	Password of the bind user account.
AUTH_LDAP_USER_SEARCH	The DN of the subtree that contains users. Often an OU.
AUTH_LDAP_GROUP_SEARCH	The DN of the subtree that contains groups. Often an OU.
AUTH_LDAP_REQUIRE_GROUP	The DN of a group to which users must belong to have login privileges.
LDAP Group for Admins	The DN of the Admins group. Users in this group have admin access.

Procedure

1. Click the Gear icon in the upper right corner.
2. Click on Site Settings.
3. Locate the section for Advanced Settings. Here you have the following two options:

Option	
Option 1: Configuring LDAP authentication with a bind user	Bind User authentication with LDAP tends to be more flexible with user lookups/searches and also supports Group lookups whereas Direct Bind in Option 2 currently does not. Group lookups allow you to map users in ML Data Viz to Roles automatically so when they login they may have access to Dashboards and Datasets (This requires additional steps in the ML Data Viz Roles setup section). However this also requires you to request and maintain a Bind User which may take a little longer to setup at first. You will also occasionally need to update the Bind User password as it will likely expire. Here is the code for a simple search/bind approach that completes an anonymous bind, searches the OU for an object that matched the UID of the user's name, and attempts to bind using that DN and the user's password. The authentication fails unless the search returns exactly one result. If anonymous search is not possible, set AUTH_LDAP_BIND_DN to the DN of

Option

an authorized user, and AUTH_LDAP_BIND_PASSWORD to the password.

```
import ldap
from django_auth_ldap.config import
 LDAPSearch

AUTH_LDAP_BIND_DN = ""
AUTH_LDAP_BIND_PASSWORD = ""
AUTH_LDAP_USER_SEARCH = LDAPSearch(
"ou=users,dc=example,dc=com", ld
ap.SCOPE_SUBTREE, "(uid=%(user)s)")
```

Option 2: Direct Bind Approach

Direct Bind with LDAP explicitly passes the user's credentials to authenticate with the LDAP server. The advantage of Direct Bind is that doesn't require you to request and manage a Bind User account. However, the downside to Direct Bind is that Group lookups don't currently work for logged in users, which means automatic User-Role mapping can't be configured.

Here is the code for a simple direct bind approach:

```
AUTH_LDAP_BIND_AS_AUTHENTICATING_US
ER = True
AUTH_LDAP_USER_DN_TEMPLATE = "uid=
%(user)s,ou=users,dc=example,dc=
com"
```

4. If you're using a Bind User to authenticate you can store the LDAP_DN and LDAP_PASSWORD environment variables in the Project Settings section under the Engine tab for easier management.

Example

Example configuration for Bind User with LDAP_DN and LDAP_PASSWORD project environmental variables in Cloudera Internal EDH:

```
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupT
ype, ActiveDirectoryGroupType

# Connection options
#AUTH_LDAP_START_TLS = True  # Optional for LDAPS but normally not needed
AUTH_LDAP_SERVER_URI = "ldap://ad-readonly.sjc.cloudera.com:389"

# Bind user setup
AUTH_LDAP_BIND_DN = os.getenv('LDAP_DN')
AUTH_LDAP_BIND_PASSWORD = os.getenv('LDAP_PASSWORD')

# Required Group for all users to access application
#AUTH_LDAP_REQUIRE_GROUP = "CN=All_Staff_WW,OU=Groups,DC=cloudera,DC=local"

# Group for specifying super admins
#AUTH_LDAP_USER_FLAGS_BY_GROUP = {
```

```
#  "is_superuser": ["CN=cloud_spend_analysts,OU=Groups,DC=cloudera,DC=loc
al"]
#}

# User and group search objects and types
AUTH_LDAP_USER_SEARCH = LDAPSearch("CN=users,DC=cloudera,DC=local",
 ldap.SCOPE_SUBTREE,"(sAMAccountName=%(user)s)")

AUTH_LDAP_GROUP_SEARCH = LDAPSearch("OU=Groups,DC=cloudera,DC=local",
 ldap.SCOPE_SUBTREE,"(objectClass=group)")

# Map LDAP attributes to Django
AUTH_LDAP_USER_ATTR_MAP = {
"first_name": "givenName",
"last_name": "sn",
"email": "mail"
}

# Cache settings
# Note this may cause a delay when groups are changed in LDAP
AUTH_LDAP_CACHE_GROUPS = True
AUTH_LDAP_GROUP_CACHE_TIMEOUT = 3600*4  # Cache for 4 hours
REMOTE_GROUP_CACHE_TIMEOUT = 3600*4

# Group Settings
AUTH_LDAP_GROUP_TYPE = ActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True
AUTH_LDAP_MIRROR_GROUPS = False

# Some optional TLS/SSL options when enabling LDAPS

#AUTH_LDAP_GLOBAL_OPTIONS = {
#ldap.OPT_X_TLS_CACERTFILE: "/etc/bla.cert",        # Point to CA Cert file
#ldap.OPT_X_TLS_REQUIRE_CERT: ldap.OPT_X_TLS_NEVER, # Disable cert checking
#}

AUTH_LDAP_CONNECTION_OPTIONS = {
ldap.OPT_DEBUG_LEVEL: 1,  # 0 to 255
ldap.OPT_REFERRALS: 0,  # For Active Directory
}

# If there is no Bind User you can use these settings, but it's not the pr
eferred way
#AUTH_LDAP_BIND_AS_AUTHENTICATING_USER = True
#AUTH_LDAP_USER_DN_TEMPLATE = "cloudera\%(user)s"

# The backend needed to make this work.
AUTHENTICATION_BACKENDS = (
'arcweb.arcwebbase.basebackends.VizBaseLDAPBackend',
'django.contrib.auth.backends.ModelBackend'
)
```

Example

Example configuration for Direct Bind in Cloudera Internal EDH:

```

import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupT
ype, ActiveDirectoryGroupType

# Connection options
#AUTH_LDAP_START_TLS = True  # Optional for LDAPS but normally not needed
```

```
AUTH_LDAP_SERVER_URI = "ldap://ad-readonly.sjc.cloudera.com:389"

# Bind user setup
#AUTH_LDAP_BIND_DN = os.getenv('LDAP_DN')
#AUTH_LDAP_BIND_PASSWORD = os.getenv('LDAP_PASSWORD')

# Required Group for all users to access application
#AUTH_LDAP_REQUIRE_GROUP = "CN=All_Staff_WW,OU=Groups,DC=cloudera,DC=local"

# Group for specifying super admins
#AUTH_LDAP_USER_FLAGS_BY_GROUP = {
#  "is_superuser": ["CN=cloud_spend_analysts,OU=Groups,DC=cloudera,DC=local"
]
#}

# User and group search objects and types
#AUTH_LDAP_USER_SEARCH = LDAPSearch("CN=users,DC=cloudera,DC=local",
ldap.SCOPE_SUBTREE,"(sAMAccountName=%(user)s)")

AUTH_LDAP_GROUP_SEARCH = LDAPSearch("OU=Groups,DC=cloudera,DC=local", ldap
.SCOPE_SUBTREE,"(objectClass=group)")

# Map LDAP attributes to Django
AUTH_LDAP_USER_ATTR_MAP = {
"first_name": "givenName",
"last_name": "sn",
"email": "mail"
}

# Cache settings
# Note this may cause a delay when groups are changed in LDAP
AUTH_LDAP_CACHE_GROUPS = True
AUTH_LDAP_GROUP_CACHE_TIMEOUT = 3600*4  # Cache for 4 hours
REMOTE_GROUP_CACHE_TIMEOUT = 3600*4

# Group Settings
AUTH_LDAP_GROUP_TYPE = ActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True
AUTH_LDAP_MIRROR_GROUPS = False

# Some optional TLS/SSL options when enabling LDAPS

#AUTH_LDAP_GLOBAL_OPTIONS = {
#ldap.OPT_X_TLS_CACERTFILE: "/etc/bla.cert",        # Point to CA Cert file
#ldap.OPT_X_TLS_REQUIRE_CERT: ldap.OPT_X_TLS_NEVER, # Disable cert checking
#}

AUTH_LDAP_CONNECTION_OPTIONS = {
ldap.OPT_DEBUG_LEVEL: 1,  # 0 to 255
ldap.OPT_REFERRALS: 0,  # For Active Directory
}

# If there is no Bind User you can use these settings, but it's not the p
referred way

AUTH_LDAP_BIND_AS_AUTHENTICATING_USER = True
AUTH_LDAP_USER_DN_TEMPLATE = "cloudera\%(user)s"

# The backend needed to make this work.
AUTHENTICATION_BACKENDS = (
'arcweb.arcwebbase.basebackends.VizBaseLDAPBackend',
'django.contrib.auth.backends.ModelBackend'
)
```

Using LDAPS

About this task

If you plan to configure authentication using LDAPS instead of LDAP there are 3 extra steps in the configuration that need to be considered.

Procedure

1. Update the LDAP Server URI and port to use LDAPS protocol.

```
                        AUTH_LDAP_SERVER_URI = "ldaps://ad-readonly.sjc
.cloudera.com:636"
```

2. Uncomment this section and add a valid path to a SSL certificate file.

```
                        AUTH_LDAP_GLOBAL_OPTIONS = {
                        ldap.OPT_X_TLS_CACERTFILE: "/path/to/bla.cert", #
Point to CA Cert file

                        ldap.OPT_X_TLS_REQUIRE_CERT: ldap.OPT_X_TLS_NEVER,
 # Disable cert checking
                        }
```

3. [Optional] Enable TLS if not already running

   Note: Test setting up LDAPS with Steps 1 and 2 and restart ML Data Viz without this turned on first to avoid unnecessary debugging.

Complex matching logic for group queries using LDAPGroupQuery()

You can use LDAP to restrict user access to Data Vizualization resources by using multiple require groups.

About this task

Note: This feature depends on the LDAPGroupQuery() class, which is available starting with the django_auth_ldap release 1.2.12.

Procedure

To use a complex group query, implement the LDAP authorization requirement group, as we demonstrate in the following code snippet:

```
AUTH_LDAP_REQUIRE_GROUP = (
(LDAPGroupQuery("cn=enabled,ou=groups,dc=example,dc=com") |
LDAPGroupQuery("cn=also_enabled,ou=groups,dc=example,dc=com")) &
~LDAPGroupQuery("cn=disabled,ou=groups,dc=example,dc=com")
)
```

What to do next

For more information, see the django-auth-ldap reference documentation.