How to configure FIXICC authorization through LDAP

FIXICC Agent configuration for LDAP

FIXICC Agent could be configured to authorize users and authenticate their roles with LDAP. It uses Apache Shiro security framework for user authorization and authentication. This framework could be easily adopted for working with LDAP. Below you can find a sample and description of such configuration. This sample uses a security part of Apache Isis framework, which helps to not only authorize users and check their roles but also assign certain permissions for them.

Example of FIXICC Agent security configuration file (security.properties)

Here is a sample configuration for FIXICC Agent (security.properties):

```
[main]
contextFactory = org.apache.isis.security.shiro.IsisLdapContextFactory
contextFactory.url = ldap://localhost:10389
contextFactory.authenticationMechanism = simple
contextFactory.systemAuthenticationMechanism = simple
contextFactory.systemUsername = uid=admin,ou=system
contextFactory.systemPassword = secret

ldapRealm = org.apache.isis.security.shiro.IsisLdapRealm
ldapRealm.contextFactory = $contextFactory
ldapRealm.userDnTemplate = uid={0},ou=users,o=fixicc

ldapRealm.searchBase = o=fixicc
ldapRealm.groupObjectClass = groupOfUniqueNames
ldapRealm.uniqueMemberAttribute = uniqueMember
ldapRealm.uniqueMemberAttributeValueTemplate = uid={0}

ldapRealm.rolesByGroup =\
  user_role: read_only_user,\
  anonimouse_role: read_only_user,\
  admin_role: admin

ldapRealm.permissionsByRole=\
   read_only_user = SessionsList,ServerStatus,SessionParams,SessionsParametersSubscription,\
SessionStatus,SessionsSnapshot,SessionStat,GeneralSessionsStat,MeasurementPointList,\
MeasurementPointStatistic,LatencyAlertSubscription,AverageReceivedStat,AverageSentStat,\
AverageValidateStat,ReceivedStat,SentStat,ProceedStat,Help; \
   admin = *

securityManager.realms = $ldapRealm
```

Such configuration allows resolving user via LDAP server ldap://localhost:10389. It also describes 2 roles with their permissions for FIXICC users: read_only_user and admin. This approach in configuration requires to describe certain permissions in configuration file.

Security properties description

Now let's describe each property from this config file:

- ***contextFactory*** defines a factory for accessing a source of users data. Here we are using the instance from Isis framework - org.apache.isis.security.shiro.IsisLdapContextFactory. It allows defining a way how to connect to LDAP and authorize users.
  - ***contextFactory.url*** defines an address of used LDAP server as URL(RFC 2255). An LDAP URL is defined by the following grammar:
    scheme "://" [host [":" port]] ["/" [dn ["?" [attributes] ["?" [scope] ["?" [filter] ["?" extensions]]]]]]
    where:
    scheme: Indicates the URL scheme. This class library supports either the traditional "ldap" for normal LDAP connections or "ldaps" for SSL connections (see SSL Support section).
    - host: Name of the LDAP server.
    - port: The LDAP server's port number.
    - dn: Identifies the base object for the operation.
    - attributes: Comma separated list of attributes to be returned. If not specified then the default is to return all attributes.

        - scope: The scope of a search, one of "base", "sub" or "one", representing base object, subtree or one level, respectively. If not specified then the default is "base".
        - filter: The search filter. If not specified then the default is "(objectclass=*)".
        - extension: This provides the LDAP URL with an extensibility mechanism, allowing the capabilities of the URL to be extended in the future. The only extension supported by this toolkit is "bindname".
    - ***contextFactory.systemAuthenticationMechanism***. Authentication mechanism for service user. Service user is used for connecting to LDAP server and accessing the rest of data. To authenticate such a system user LDAP server could use such the mechanisms:
        - Anonymous (property value “none”) — no security information is provided.
        - Simple (property value “simple”) — the client provides a clear text name and password.
        - Simple Authentication and Security Layer (SASL) (property value depends of the underlying mechanism. For example, “CRAM-MD5” or “GSSAPI”) — the client and server negotiate an authentication system based on a challenge and response protocol that conforms to RFC2222.

          More information about authentication mechanism you can find here: http://docs.oracle.com/javase/tutorial/jndi/ldap/auth_mechs.html
    - ***contextFactory.systemUsername***. Distinguished Name of service user.
    - ***contextFactory.systemPassword***. Password for service user.
    - ***contextFactory.authenticationMechanism***. Authentication mechanism for users. See description for ***contextFactory.systemAuthenticationMechanism*** property.
- ***ldapRealm*** defines a realm for LDAP data.
  A Realm is a component that can access application-specific security data such as users, roles, and permissions. The Realm translates this application-specific data into a format that Shiro understands so Shiro can in turn provide a single easy-to-understand Subject programming API no matter how many data sources exist or how application-specific your data might be.
  Here we are using instance of com.epam.fixicc.agent.security.ldap.IsisLdapRealm from Isis framework. In addition to standard LDAP Shiro realm it also allows assigning permissions for authenticated users.
    - ***ldapRealm.contextFactory***. Link real context factory with defined above instance.
      In general, it’s possible also to define basic LDAP connection parameters directly for realm. But in this case we can lost some advantages of additional attributes.
    - ***ldapRealm.userDnTemplate***. This property allows specifying the LDAP server's User DN format.
      During an authentication attempt, if the submitted principal is a simple username, but the LDAP directory expects a complete User Distinguished Name (User DN) to establish a connection, the userDnTemplate property must be configured. If not configured, the property will pass the simple username directly as the User DN, which is often incorrect in most LDAP environments (maybe Microsoft ActiveDirectory being the exception).

      User DN formats are unique to the LDAP directory's schema, and each environment differs - you will need to specify the format corresponding to your directory. You do this by specifying the full User DN as normal, but you use a {0} placeholder token in the string representing the location where the user's submitted principal (usually a username or uid) will be substituted at runtime.

      For example, if your directory uses an LDAP uid attribute to represent usernames, the User DN for the jsmith user may look like this:

      uid=jsmith,ou=users,o=fixicc

      in which case you would set this property with the following template value:

      uid= uid={0},ou=users,o=fixicc
    - ***ldapRealm.searchBase***. Search base for user groups.
    - ***ldapRealm.groupObjectClass***. Group object class.
    - ***ldapRealm.uniqueMemberAttribute***. Attribute name in group for mapping user with this group.
    - ***ldapRealm.uniqueMemberAttributeValueTemplate***. value template for mapping user with group.
    - ***ldapRealm.rolesByGroup***. Property for mapping LDAP groups with roles. If no group-to-role mapping is provided, then the group names are used as role names with no translation.
    - ***ldapRealm.permissionsByRole***. Property for mapping permissions for roles.
- ***securityManager.realms***. Directly link our realm to security manager. It’s possible to define few realms and link them to security manager.

Our sample configuration means that:

- FIXICC Agent uses user with DN uid=admin,ou=system to perform a search in the domain specified by searchBase
- simple authentication type is used for authenticate service user
- users accounts are searched like uid=xxx,ou=users,o=fixicc, where xxx is a user login
- users have, at minimum, a uid and userPassword attributes
- simple authentication type is used to authenticate users
- groups are searched under o=fixicc
- each group has an LDAP objectClass of groupOfUniqueNames
- each group has a vector attribute of uniqueMember
- each value of uniqueMember is in the form uid=xxx, with xxx being the uid of the user
- the group membership is looked up using the specified system user uid=admin,ou=system
- groups looked up from LDAP can be mapped to logical roles
- if no group-to-role mapping is provided, then the group names are used as role names with no translation

LDAP server configuration

Example of LDAP server configuration can be found here: LDAP server configuration guide

More information:

- Apache Shiro Reference Documentation
- Using Shiro with an LDAP Server
- An example of using Shiro to perform Active Directory authentication over SSL