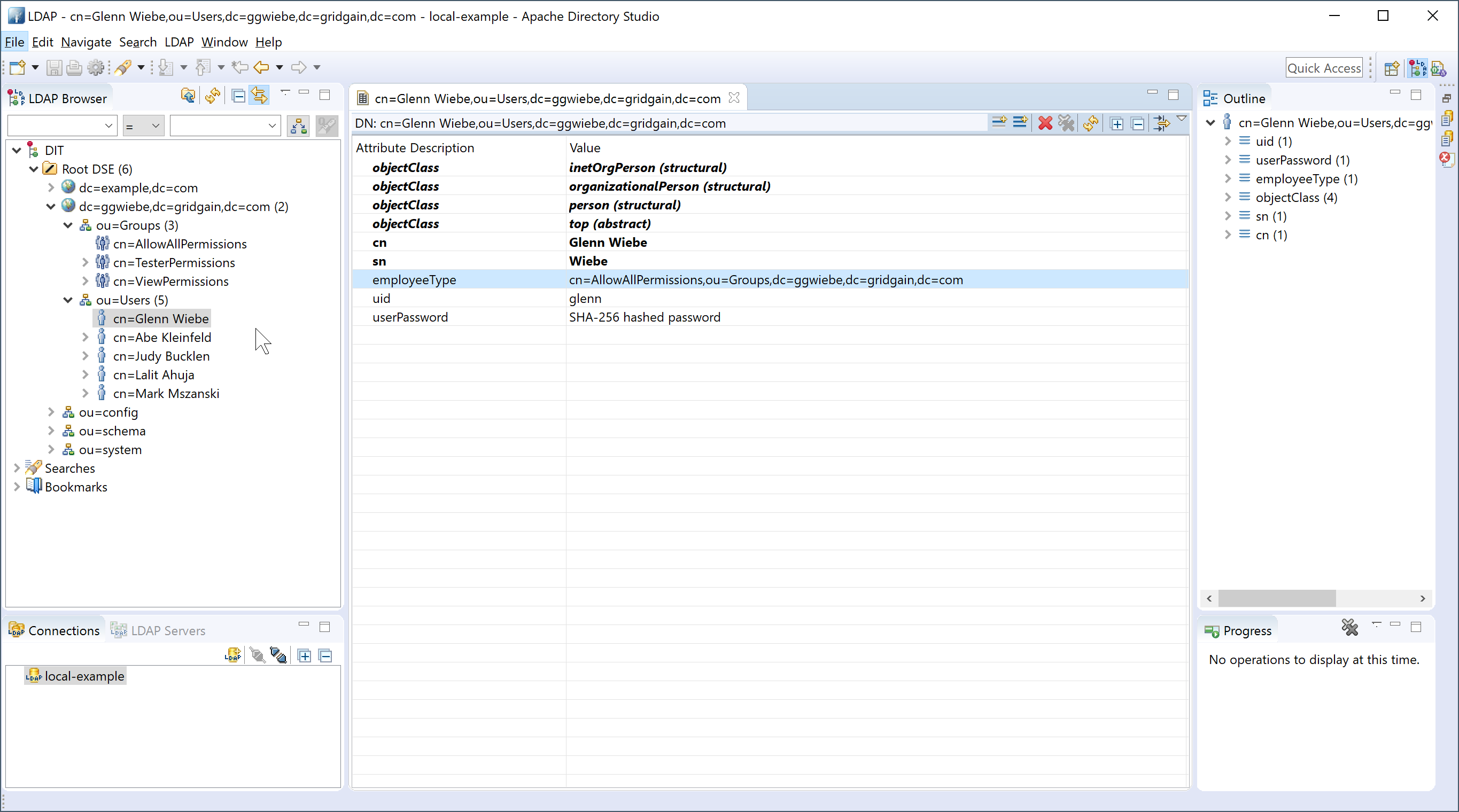
## Custom LDAP Authentication & Authorization using ApacheDS

This document describes the setup to run a custom authenticator based off ApacheDS and leveraging group membership only.

### ApacheDS Setup

This solution leverages the out-of-the-box setup of ApacheDS for authentication as well as an alternative approach for storing user membership. ApacheDS does not support the virtual memberOf attribute and as such this solution will use a static group definition AND an alternate attribute for referencing these groups: employeeType.

The following screen shot shows the configuration of a custom domain: ggwiebe.gridgain.com with a Users and Groups organizational unit:



There are three groups in this example defined as:

* cn=AllowAllPermissions
* cn=TesterPermissions
* cn=ViewPermissions

The user who’s canonical name is “Glenn Wiebe” has:

* A uid attribute of the particular userid: “glenn”
* A userPassword set to “Glenn001”
* A employeeType attribute set to the DN of one of the above defined groups.

### Custom Authenticator Module

A sample custom authenticator has been included in this solution. The class of this authenticator is rbc, as seen in the Java class definition:

**package** rbc;

**public** **class** GridGainAuthenticator **implements** Authenticator, AuthenticationValidator {

Essential to the process is:

* an authenticate() method that will check the supplied credentials against the LDAP server
* a getLdapGroups() method that will look up the groups for the authenticated user
* a getLdapPermissionSet() method that will build up the permissions based on all the groups in which the authenticated user is found

This module as currently compiled uses the following settings:

* Bind authorization = none
* LDAP user “memberOf”, or membership attribute: “employeeType”

The module as currently constructed is attached here:



The source project for this module is attached here:



\*\*\* NOTE: there is an additional jar file and xml file in the root folder of this project that is different than described in this document. These may be used, but are notably different and would require different configuration settings not documented here.

### Custom Authenticator Server Configuration

In order to invoke the rbc.GridGainAuthenticator, one must configure the server node to use it. This is performed by configuring a GridGain plugin in the Spring Bean configuration xml file (e.g. YourCluster-server.xml)

<property name="pluginConfigurations">

<list>

<bean class="org.gridgain.grid.configuration.GridGainConfiguration">

<property name="securityCredentialsProvider">

<bean class="org.apache.ignite.plugin.security.SecurityCredentialsBasicProvider">

<constructor-arg ref="user\_cred"/>

</bean>

</property>

<property name="authenticator">

<bean class="rbc.GridGainAuthenticator">

<property name="ldapUrl" value="ldap://localhost:10389/"/>

<property name="ldapDomain" value="ggwiebe.gridgain.com"/>

<property name="ldapBindAuth" value="none"/>

<property name="ldapSearchLocation" value="OU=Users,DC=ggwiebe,DC=gridgain,DC=com"/>

<property name="ldapSecuritySearchFilter" value="uid={USERNAME}"/>

<!-- <property name="ldapTrustStore" value="/opt/gdd/datagrid/ignite/config/gdd\_ldap\_truststore"/> -->

<!-- <property name="ldapTrustStorePassword" value="{{ldapTrustStorepwd}}"/> -->

<property name="ldapPermissions">

<map>

<entry key="CN=AllowAllPermissions, OU=groups,DC=ggwiebe,DC=gridgain,DC=com" value-ref="allow.all.permissions"/>

<entry key="CN=AdminPermissions, OU=groups,DC=ggwiebe,DC=gridgain,DC=com" value-ref="admin.permissions"/>

<entry key="CN=TesterPermissions, OU=groups,DC=ggwiebe,DC=gridgain,DC=com" value-ref="tester.permissions"/>

<entry key="CN=ViewPermissions, OU=groups,DC=ggwiebe,DC=gridgain,DC=com" value-ref="view.permissions"/>

</map>

</property>

</bean>

</property>

</bean>

</list>

</property>

Where the references define the actual permissions granted to those groups. Here are the four permission sets defined and mapped to specific groups:

<bean id="allow.all.permissions" class="org.apache.ignite.plugin.security.SecurityBasicPermissionSet">

<property name="defaultAllowAll" value="true"/>

</bean>

<bean id="admin.permissions" class="org.apache.ignite.plugin.security.SecurityBasicPermissionSet">

<property name="defaultAllowAll" value="false"/>

<property name="systemPermissions">

<list>

<value>ADMIN\_OPS</value>

<value>ADMIN\_VIEW</value>

<value>ADMIN\_QUERY</value>

<value>ADMIN\_CACHE</value>

<value>ADMIN\_VIEW</value>

<value>CACHE\_CREATE</value>

<value>CACHE\_DESTROY</value>

</list>

</property>

</bean>

<bean id="view.permissions" class="org.apache.ignite.plugin.security.SecurityBasicPermissionSet">

<property name="defaultAllowAll" value="false"/>

<property name="systemPermissions">

<list>

<value>ADMIN\_VIEW</value>

</list>

</property>

</bean>

<bean id="tester.permissions" class="org.apache.ignite.plugin.security.SecurityBasicPermissionSet">

<property name="defaultAllowAll" value="false"/>

<property name="systemPermissions">

<list>

<value>ADMIN\_VIEW</value>

<value>ADMIN\_QUERY</value>

</list>

</property>

</bean>