Available login modules

In Apache Geronimo, all login modules are in the package **org.apache.geronimo.security.realm.providers**. The following list shows the available login modules:

- org.apache.geronimo.security.realm.providers.#PropertiesFileLoginModule
- org.apache.geronimo.security.realm.providers.#SQLLoginModule
- org.apache.geronimo.security.realm.providers.#LDAPLoginModule

PropertiesFileLoginModule

The **PropertiesFileLoginModule** module keeps user and group databases in property file format.

class = org.apache.geronimo.security.realm.providers.PropertiesFileLoginModule;
Options:

- usersURI - URI for the users properties file. The content of the file is user=password entries.
- groupsURI - URI for the groups properties file; The content of the file is group=

These URIs are server-relative and resolved by the Geronimo components.

```
<gbean name="properties-login" class="org.apache.geronimo.security.jaas.LoginModuleGBean">
        <attribute name="loginModuleClass">org.apache.geronimo.security.realm.providers.
PropertiesFileLoginModule</attribute>
        <attribute name="serverSide">true</attribute>
        <attribute name="options">
            usersURI=var/security/users.properties
            groupsURI=var/security/groups.properties
        </attribute>
        <attribute name="loginDomainName">geronimo-properties-realm</attribute>
</gbean>
```

Back to Top

SQLLoginModule

This login module authenticates a user against relational database.

class = org.apache.geronimo.security.realm.providers.SQLLoginModule
Options:

- userSelect - prepared SQL statement to select a user. This query must return 2 columns: user and password. Parameter to the query is user name.
- groupSelect - prepared SQL statement to select a group. This query must return 2 columns: user and group. Parameter to the query is user name.
- jdbcURL - JDBC connection URL
- jdbcUser - JDBC user id; used by the login module to log into the database
- jdbcPassword - JDBC password; used by the login module to log into the database
- jdbcDriver - JDBC drive class name
- dataSourceName - data source name, for example: SystemDatabase. If dataSourceName is specified, it will take precedence over other connection-related parameters.
- dataSourceApplication - application module in which data source is deployed. This is the value of the configurationId for the deployment module. For the system-wide data source use the value of **null**.

SQL Login configuration example

```
<gbean name="sql-login"
        class="org.apache.geronimo.security.jaas.LoginModuleGBean">
        <attribute name="loginModuleClass">org.apache.geronimo.security.realm.providers.SQLLoginModule<
/attribute>
        <attribute name="serverSide">true</attribute>
        <attribute name="options">
                dataSourceName=SystemDatasource
                dataSourceApplication=null
                userSelect=select user, password from user where user=?
                groupSelect=select user, group from groups where user=?
        </attribute>
</gbean>
```

Back to Top

LDAPLoginModule

This module keeps user and group information in the LDAP directory. See the Configuring LDAP article for the complete LDAP deployment working example.

Tip: The key to working with the LDAP module is: KNOW YOUR LDAP SCHEMA.

class = org.apache.geronimo.security.realm.providers.LDAPLoginModule
Options:

- initialContextFactory - the class name of the initial context factory. Usually `com.sun.jndi.ldap.LdapCtxFactory`.
- connectionURL - LDAP connection URL, such as ldap://localhost:1389 . Note that usual LDAP port is **389**.
- connectionUsername - this is the DN used by the login module itself for authentication to the directory server.
- connectionPassword - this is credential (password) that is used by the login module to authenticate itself to the directory server
- connectionProtocol - security protocol to use. This value is determined by the service provider. An example would be **SSL**.
- authentication - security level to use. Its value is one of the following strings: "none", "simple", "strong". If this property is unspecified, the behavior is determined by the service provider.
- userBase - the base DN for the user search.
- userSearchMatching - filter specification how to search for the user object. RFC 2254 filters are allowed. In addition you can pass parameter to the search filter instead of the literal value. For example: this is RFC 2254 filter spec: (cn=Babs Jensen). If you want to parameterize the value of the CN attribute type, specify (cn = {0}). This integer refers to the parameter number. Parameter value is the user name. This query must return exactly one object.
- userSearchSubtree - Defines directory search scope for the user. If set to true, directory search scope is SUBTREE, if set to false, directory search scope is ONE-LEVEL.
- roleBase - the base DN for the group membership search.
- roleName - LDAP attribute type that identifies group name attribute in the object returned from the group membership query. Note that group membership query is defined by the `roleSearchMatching` parameter. Often group name parameter is **cn**.
- roleSearchMatching - filter specification how to search for the role object. RFC 2254 filters are allowed. In addition you can pass parameter to the search filter instead of the literal value. For example: (uniqueMember = {0}). This integer refers to the parameter number. This parameter is the DN of the authenticated user. Note that if role membership for the user is defined in the member-of-like attribute (see **`userRoleName`** parameter) you may not need to search for group membership with the query.
- roleSearchSubtree - Defines directory search scope for the role. If set to true, directory search scope is SUBTREE, if set to false, directory search scope is ONE-LEVEL.
- userRoleName - LDAP attribute type for the user group membership. Different LDAP schemas represent user group membership in different ways. Examples are: memberOf, isMemberOf, member, etc. Values of these attributes are identifiers of groups that a user is a member of. For example, if you have: memberOf: cn=admin,ou=groups,dc=foo, specify **memberOf** as the value for the `userRoleName` attribute. Be aware of the relationship between this parameter and group membership query. Sometimes (often) they will return the same data.

LDAP GBean configuration example

```
<gbean name="ldap-login"
        class="org.apache.geronimo.security.jaas.LoginModuleGBean">
        <attribute name="loginModuleClass">org.apache.geronimo.security.realm.providers.LDAPLoginModule<
/attribute>
        <attribute name="serverSide">true</attribute>
        <attribute name="options">
                initialContextFactory=com.sun.jndi.ldap.LdapCtxFactory
                connectionURL=ldap://localhost:1389
                connectionUsername=uid=admin,ou=system
                connectionPassword=secret
                connectionProtocol=
                authentication=simple
                userBase=ou=users,ou=system
                userSearchMatching=uid={0}
                userSearchSubtree=false
                roleBase=ou=groups,ou=system
                roleName=cn
                roleSearchMatching=(uniqueMember={0})
                roleSearchSubtree=false
                userRoleName=
        </attribute>
        <attribute name="loginDomainName">ldap-realm</attribute>
</gbean>
```

Back to Top