## Configuring Security

Security in SDK is provided by ACEGI and CSM. Users of the SDK need who intend to use security can do so in three steps

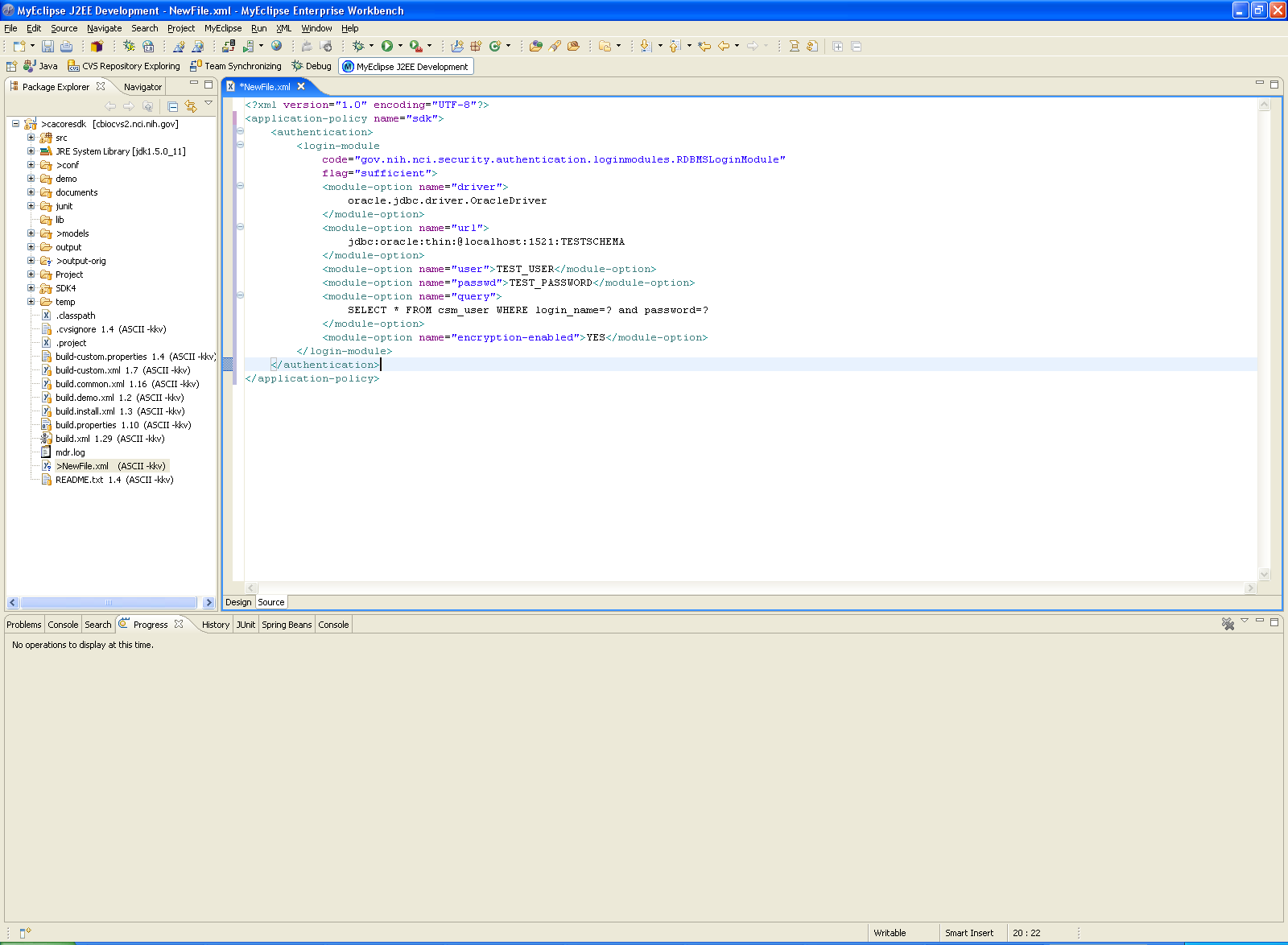
1. Configure security related properties. Users can do so by altering the configuration parameters in the deploy.properties file before generating the system.
2. Configure Application Server for JAAS based authentication configuration
3. Setup CSM database configuration for SDK based application

Table given below shows the properties that the user has to modify in order to correctly enable the security in SDK.

|  |  |  |
| --- | --- | --- |
| **Property Name** | **Default Value** | **Description** |
| **SECURITY\_ENABLED** | false | Used to enable or disable security within the generated system during code generation.  This applies to all of the SDK interfaces, including:   * Web Interface (GUI) * Java API Interface (local and remote clients) * Web Service Interface |
| **CSM\_PROJECT\_NAME** | sdk | Used as a prefix when creating the CSM security configuration file name. CSM configuration should have the same application name configured |
| **INSTANCE\_LEVEL \_SECURITY** | false | Used to enable/disable CSM instance level security.  Only relevant if the SECURITY\_ENABLED property is set to ‘true’ |
| **ATTRIBUTE\_LEVEL \_SECURITY** | false | Used to enable/disable attribute level security.  Only relevant if the SECURITY\_ENABLED property is set to ‘true’ |
| **CSM\_USE\_JNDI \_BASED\_CONNECTION** | no | Indicates whether a JNDI DB connection should be used for the CSM database.  If USE\_JNDI\_BASED\_CONNECTION=yes, then the DB\_JNDI\_URL property value is used to obtain the DB connection and retrieve data |
| **CSM\_DB\_JNDI\_URL** | java:/SDK | The DB JNDI URL value for the CSM database.  This property is irrelevant/ignored if CSM\_USE\_JNDI\_BASED\_CONNECTION=no |
| **CSM\_DB\_CONNECTION\_URL**  **CSM\_DB\_USERNAME**  **CSM\_DB\_PASSWORD** | none | The CSM database connection properties.  A sample DB\_CONNECTION\_URL value: jdbc:oracle:thin:@cbiodb30.nci.nih.gov :1521:CBTEST  These values are purposely blank. SDK users should provide appropriate values for their CSM database instance within the local.properties file located in the root folder of the SDK distribution. |
| **CSM\_DB\_DIALECT** | org.hibernate.dialect.OracleDialect | The Hibernate Database dialect used when connecting to the CSM database.  Typical values include:   * org.hibernate.dialect.OracleDialect * org.hibernate.dialect.MySQLDialect |

### JAAS based authentication configuration

Applications dependent on JAAS[[1]](#footnote-2) based login can configure their login procedure in many ways. Since caCORE SDK uses ACEGI and CSM as underlying security technologies, users of the SDK would have to perform configuration as recommended by them. For SDK generated local-client, users will get the database based JAAS configuration prepared by SDK. Users of the web application will have to configure the application server container. Paragraph below provides an example of how to configure JAAS based authentication in JBoss server. Users can refer to CSM Developer’s Guide for detailed information on how to configure JAAS based security in different Application Servers and other configuration options.

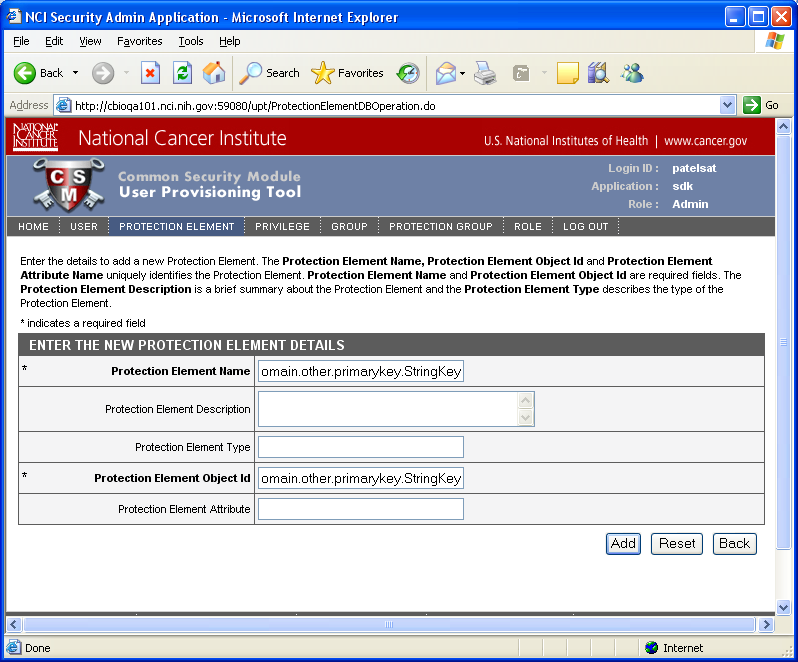


Users of the SDK will have to make an entry in the <jboss-home>/server/default/conf/login-config.xml similar to the code snippet shown above. CSM will read the entry from the server’s login configuration and perform authentication using the configuration.

#### Configuring CSM

caCORE SDK 4.0 requires CSM 4.0 for security. Users of caCORE SDK will have to setup a CSM database schema and configure it with CSM’s User Provisioning Tool (UPT). Note that if the SDK users are planning to use instance level security then they will be required to put CSM tables on the same database schema where the tables for domain class resides. SDK users can refer to the CSM Developer’s Guide to get more information on how to install CSM on a particular database and how to use User Provisioning Tool (UPT) for configuring the security schema.

While configuring the security schema with UPT, user of the SDK is required to create Protection Element for each domain object in the SDK generated system. The protection element should have the name as fully qualified name of the domain object. Security implementation of SDK uses the name of the domain object as a key to be searched in the CSM configuration to determine access privileges. Figure shown below demonstrates addition of the StringKey class as a protection element in the CSM.



Once all the protection elements are created, user of the SDK can create users, user groups and assign them READ privileges to appropriate protection elements based on the security needs of the application.

1. <http://java.sun.com/j2se/1.4.2/docs/guide/security/jaas/tutorials/GeneralAcnOnly.html> [↑](#footnote-ref-2)