# Chapter 1

# Assignment

In this chapter we will present the class that has been assigned to us. First, the lines of code to be analysed are presented without any comment. Afterwards, a brief description of the class role and function is presented. Obviously this is made by us and it is based only on the code and on the documentation provided by the authors.

```
1  /*
2   * DO NOT ALTER OR REMOVE COPYRIGHT NOTICES OR THIS HEADER.
3   *
4   * Copyright (c) 1997-2013 Oracle and/or its affiliates. All rights reserved.
5   *
6   * The contents of this file are subject to the terms of either the GNU
7   * General Public License Version 2 only ("GPL") or the Common Development
8   * and Distribution License("CDDL") (collectively, the "License").  You
9   * may not use this file except in compliance with the License.  You can
10  * obtain a copy of the License at
11  * https://glassfish.dev.java.net/public/CDDL+GPL_1_1.html
12  * or packager/legal/LICENSE.txt.  See the License for the specific
13  * language governing permissions and limitations under the License.
14  *
15  * When distributing the software, include this License Header Notice in each
16  * file and include the License file at packager/legal/LICENSE.txt.
17  *
18  * GPL Classpath Exception:
19  * Oracle designates this particular file as subject to the "Classpath"
20  * exception as provided by Oracle in the GPL Version 2 section of the License
21  * file that accompanied this code.
22  *
23  * Modifications:
24  * If applicable, add the following below the License Header, with the fields
```

```
 * enclosed by brackets [] replaced by your own identifying information:
 * "Portions Copyright [year] [name of copyright owner]"
 *
 * Contributor(s):
 * If you wish your version of this file to be governed by only the CDDL or
 * only the GPL Version 2, indicate your decision by adding "[Contributor]
 * elects to include this software in this distribution under the [CDDL or GPL
 * Version 2] license."  If you don't indicate a single choice of license, a
 * recipient has the option to distribute your version of this file under
 * either the CDDL, the GPL Version 2 or to extend the choice of license to
 * its licensees as provided above.  However, if you add GPL Version 2 code
 * and therefore, elected the GPL Version 2 license, then the option applies
 * only if the new code is made subject to such option by the copyright
 * holder.
 */

package com.sun.enterprise.iiop.security;

import com.sun.corba.ee.org.omg.CSI.ITTAnonymous;
import com.sun.corba.ee.org.omg.CSI.ITTPrincipalName;
import com.sun.corba.ee.org.omg.CSI.ITTX509CertChain;
import com.sun.corba.ee.org.omg.CSI.ITTDistinguishedName;
import com.sun.enterprise.common.iiop.security.AnonCredential;
import com.sun.enterprise.common.iiop.security.GSSUPName;
import com.sun.enterprise.common.iiop.security.SecurityContext;

import java.net.Socket;
import java.util.Set;
import java.util.HashSet;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.List;
import java.util.ArrayList;

import java.security.PrivilegedAction;
import java.security.AccessController;
import javax.security.auth.Subject;
import java.security.cert.X509Certificate;
import javax.net.ssl.SSLSession;
import javax.net.ssl.SSLSocket;
// GSS Related Functionality

import com.sun.enterprise.deployment.EjbDescriptor;
import com.sun.enterprise.deployment.EjbIORConfigurationDescriptor;
import org.omg.CORBA.ORB;
import com.sun.enterprise.security.auth.login.common.PasswordCredential;
import com.sun.enterprise.security.auth.login.common.X509CertificateCredential;

import com.sun.enterprise.util.Utility;
```

```
74  import com.sun.corba.ee.spi.ior.IOR;
75  import com.sun.corba.ee.spi.ior.iiop.IIOPAddress;
76  import com.sun.corba.ee.spi.ior.iiop.IIOPProfileTemplate;
77  import com.sun.corba.ee.spi.transport.SocketInfo;
78  import com.sun.corba.ee.org.omg.CSIIOP.*;
79  import org.ietf.jgss.Oid;
80  import java.util.Enumeration;
81  import sun.security.x509.X500Name;
82  import com.sun.enterprise.security.SecurityServicesUtil;
83  import com.sun.enterprise.security.auth.login.LoginContextDriver;
84  import com.sun.enterprise.security.auth.login.common.LoginException;
85  import com.sun.enterprise.security.auth.realm.Realm;
86  import com.sun.enterprise.security.common.ClientSecurityContext;
87  import com.sun.enterprise.security.common.SecurityConstants;
88  import com.sun.enterprise.security.ssl.SSLUtils;
89  import com.sun.enterprise.util.LocalStringManagerImpl;
90
91  import java.util.logging.*;
92  import com.sun.logging.*;
93  import java.util.Arrays;
94  import org.glassfish.api.admin.ProcessEnvironment;
95  import org.glassfish.api.admin.ProcessEnvironment.ProcessType;
96  import org.glassfish.api.invocation.ComponentInvocation;
97  import org.glassfish.enterprise.iiop.api.GlassFishORBHelper;
98  import org.glassfish.enterprise.iiop.api.ProtocolManager;
99
100
101 import org.jvnet.hk2.annotations.Service;
102 import org.glassfish.api.invocation.InvocationManager ;
103 import org.glassfish.hk2.api.PostConstruct;
104 import javax.inject.Singleton;
105
106 import javax.inject.Inject;
107
108 /**
109  * This class is responsible for making various decisions for selecting
110  * security information to be sent in the IIOP message based on target
111  * configuration and client policies.
112  * Note: This class can be called concurrently by multiple client threads.
113  * However, none of its methods need to be synchronized because the methods
114  * either do not modify state or are idempotent.
115  *
116  * @author Nithya Subramanian
117
118  */
119
120 @Service
121 @Singleton
122 public final class SecurityMechanismSelector implements PostConstruct {
```

```
123
124     private static final java.util.logging.Logger _logger =
125        LogDomains.getLogger(SecurityMechanismSelector.class, LogDomains.SECURITY_LOGGER
               );
126
127     public static final String CLIENT_CONNECTION_CONTEXT = "ClientConnContext";
128     //public static final String SERVER_CONNECTION_CONTEXT = "ServerConnContext";
129
130     private  Set<EjbIORConfigurationDescriptor> corbaIORDescSet = null;
131     private  boolean sslRequired = false;
132
133     // List of hosts trusted by the client for sending passwords to.
134     // Also, list of hosts trusted by the server for accepting propagated
135     // identities.
136     //private static String[] serverTrustedHosts = null;
137
138     private static final LocalStringManagerImpl localStrings =
139         new LocalStringManagerImpl(SecServerRequestInterceptor.class);
140
141     // A reference to POAProtocolMgr will be obtained dynamically
142     // and set if not null. So set it to null here.
143     private  ProtocolManager protocolMgr = null;
144
145     @Inject
146     private SSLUtils sslUtils;
147
148     private GlassFishORBHelper orbHelper;
149
150     //private CompoundSecMech mechanism = null;
151     private ORB orb = null;
152     private CSIV2TaggedComponentInfo ctc = null;
153
154     @Inject
155     private InvocationManager invMgr;
156
157     @Inject
158     private ProcessEnvironment processEnv;
159     /**
160      * Read the client and server preferences from the config files.
161      */
162     public SecurityMechanismSelector() {
163     }
164
165     public void postConstruct() {
166         try {
167             orbHelper = Lookups.getGlassFishORBHelper();
168       // Initialize client security config
169       String s =
170     (orbHelper.getCSIv2Props()).getProperty(GlassFishORBHelper.ORB_SSL_CLIENT_REQUIRED)
```

```
        ;
171       if ( s != null && s.equals("true") ) {
172     sslRequired = true;
173       }
174
175       // initialize corbaIORDescSet with security config for CORBA objects
176       corbaIORDescSet = new HashSet<EjbIORConfigurationDescriptor>();
177       EjbIORConfigurationDescriptor iorDesc =
178               new EjbIORConfigurationDescriptor();
179       EjbIORConfigurationDescriptor iorDesc2 =
180               new EjbIORConfigurationDescriptor();
181       String serverSslReqd =
182                     (orbHelper.getCSIv2Props()).getProperty(GlassFishORBHelper.
                          ORB_SSL_SERVER_REQUIRED);
183       if ( serverSslReqd != null && serverSslReqd.equals("true") ) {
184     iorDesc.setIntegrity(EjbIORConfigurationDescriptor.REQUIRED);
185     iorDesc.setConfidentiality(
186           EjbIORConfigurationDescriptor.REQUIRED);
187     iorDesc2.setIntegrity(EjbIORConfigurationDescriptor.REQUIRED);
188     iorDesc2.setConfidentiality(
189           EjbIORConfigurationDescriptor.REQUIRED);
190       }
191       String clientAuthReq =
192     (orbHelper.getCSIv2Props()).getProperty(GlassFishORBHelper.ORB_CLIENT_AUTH_REQUIRED
          );
193       if ( clientAuthReq != null && clientAuthReq.equals("true") ) {
194     // Need auth either by SSL or username-password.
195     // This sets SSL clientauth to required.
196     iorDesc.setEstablishTrustInClient(
197           EjbIORConfigurationDescriptor.REQUIRED);
198     // This sets username-password auth to required.
199     iorDesc2.setAuthMethodRequired(true);
200     getCorbaIORDescSet().add(iorDesc2);
```