European Maritime Safety Agency

IdM Guide

Access and Identity Management Guide

Document History

Title	**Access and Identity Management Guide**
Version	0.3 from 31/05/2011
ID	
Document Type	Technical Document

	In Process	In Review	Approved for Usage
Status	☒	☐	☐

Responsibility

Role	Names
Authors	Paulo Faustino
Processors	António Anciães
Reviewers	António Anciães
Approvers	António Anciães

Distribution

Role	Names
EMSA	ICT

Change History

Version	Date	Names	Comments
0.3	26/05/2011	Paulo Faustino	Initial draft
0.4	31/05/2011	António Anciães	Revision and minor changes
0.5	26/07/2011	Paulo Faustino	Added EMSA OIM Custom Interface

Table of Contents

Table of Figures

Table of Tables

Definitions, acronyms and abbreviations

Definition	Description
IdM	Identity Management which comprises Access and User Identity Management
OAM	Oracle Access Management
OIM	Oracle Identity Management
Webgate	Secured access entry point for applications

1. Introduction and objectives

This document describes EMSA Access and Identity Management. Its main objective is to document the technical solutions used by EMSA to implement Access Control and User Identity Management throughout EMSA applications.

The document is organized in several chapters:

- Chapter 1: Introduction and Objectives. This chapter;
- Chapter 2: EMSA Requirements for IdM. A quick specification of the requirements for implementing the IdM at EMSA.
- Chapter 3: General Overview of IdM Solution. Oracles vision of the IdM to be implemented at EMSA.
- Chapter 4: Deploying Applications with Single Sign-On. An explanation of what can be/has to be done to applications for them to be integrated with the Access Management.
- Chapter 5: Provisioning Applications. An explanation on how application users can be provisioned by Oracle Identity Management.

2. EMSA Requirements for IdM

The present text aims at documenting the Identity Management / Single Sign-on process, providing information on what was accomplished in the Proof of Concept (PoC) as well as providing an insight into what needs to follow. A first analysis divides the remaining work into another two different phases - making a total of three if we consider the PoC to also be a phase. These are:

- Phase 0 – The Proof of Concept can be defined as Phase 0, or a starting point, for the Identity Management / Single Sign-On project;
- Phase 1 – Implementation of the IdM / SSO system as a project including infrastructure and all adaptations needed to Liferay portal, Thetis, STCW and Clean Sea Net version 2 (CSN2);
- Phase 2 – Integration of other applications and EMSA corporate Active Directory (AD). It is possible that a new PoC be needed at this time for evaluating the Oracle Entitlement Server as a fine-grained access control solution to be used for implementing the access control that is not covered by the OIM / OAM solution;

Please note that what is described in this chapter is a tentative division of work and may not correspond to what may actually be implemented.

2.1. PHASE 0 – PROOF OF CONCEPT

This is a small analysis of the work executed during the PoC. It aims to identify what can be considered as a good practice (Right), what was done that shouldn’t be done (Wrong) and what needs to be bettered (Enhancements).

- Right
    - The use of a Reverse Proxy (Apache), though adding some apparently un-needed complexity to the overall solution, permits the implementation of SSO on non-Liferay applications. However, these applications have to be modified to recognize and correctly interpret the session Cookie generated by OAM (eventually connecting to OAM itself for this functionality).

- Wrong
    - Invocation of Thetis services for OIM provisioning should not be done indirectly through a JSP page (this was a workaround to surpass an OIM integration problem), but should be through a Web Service. This Web Service has to be implemented by the Thetis contractor.
    - Incomplete association of Users to Ports/Countries in Thetis when these are created through OIM. Thetis must provide richer services to user provisioning.

- Enhancements
    - “Design” the Login page to give it a more consistent look-and-feel with the rest of the EMSA user interface.
    - The OIM look-and-feel has to be tweaked to be more closely integrated with the overall EMSA theme.
    - Validate the SSO Cookie instead of simply checking for its existence.

2.2. PHASE 1 – IMPLEMENTATION OF IDM/SSO PROJECT WITH COMPLETE INTEGRATION

The goal of this phase is to provide the actual complete implementation of the IdM / SSO solution including all the adaptations to the Liferay portal, Thetis, STCW and CSN2.

Infrastructure

- High Availability
    - The final solution has to be compatible with the High Availability (HA) infrastructure at EMSA, and be run in an Active-Active Weblogic Cluster. This is true for both the OAM and OIM applications (as well as any solutions upon which these systems may depend – as described below).
    - The solution should also be compatible with BCF (Business Continuity Framework) to be implemented at EMSA.
    - Any database needed by the OAM and/or OIM should be RAC compatible as this is the HA solution adopted at EMSA for a database infrastructure.
    - Any other servers (Apache, LDAP, etc.) also have to be "clusterable" or at least provide an HA solution.

- Hardware architecture – The complete hardware architecture will be mentioned at a later time.

- Oracle Service Bus (OSB) is also a part of the overall software architecture at EMSA and should also be considered in the final solution. At the present moment it is not clear as to how SSO will be done on this module.

- In order for applications to be able to see all LDAP servers (AD, Open LDAP, etc.) in a similar coherent way, Oracle Virtual Directory (OVD) should be used to create a virtualization layer.

- Load Balancing
    - The final solution has to be compatible with the existing EMSA infrastructure as far as load balancing is concerned. This is presently done with F5 load balancers for external accesses and Apache Servers for internal accesses.

- The solution should be integrated with an SMTP Server to handle all mail related issues.

- The solution should be ready to cleanly integrate with OpenView to allow system monitoring.

- The solution should be prepared to integrate cleanly with a Syslog Server for log auditing.

IdM

- OAM
    - This software has to provide all the functionalities for duly managing the authentication of users accessing EMSA applications (those which are covered under the IdM/SSO "umbrella").
    - Due to the fact that EMSA has applications that are not directly deployed under Liferay as well as Liferay deployed applications, there is a need for three types of login interception/treatment:
        - Applications that are not deployed under Liferay should be intercepted by the Apache Reverse Proxy and the authentication process should correspond to the creation of an SSO session token that then needs to be validated inside the application.
        - Anonymous (or Guest) access to public pages in Liferay should not be authenticated. However, there should be a link available in the guests' home page allowing a login to the system. This login should then be serviced by the SSO code in the normal manner.
        - Users directly accessing an application that is directly deployed in Liferay should be presented a login page (go through the regular authentication process) if they have not already established a session.

        This will probably be the most common case and corresponds to the base tests made during the PoC.
    - At EMSA, there are two main classes of applications, those which run under Weblogic and those that do not. For those that do not run under Weblogic, a filter for treating the SSO session token must be developed for inclusion in such applications. Depending on the technology involved, this may or may not have to be done by the application contractor. For those applications that do run under Weblogic, a new Realm Provider has to be built that will obtain its information directly from the SSO session token instead of connecting to LDAP to obtain such information.

- OIM
    - This software has to provide all the necessary functionalities for provisioning users/user data in all EMSA applications (where applicable). As such, each specific "connection" has to be analysed and a specific connector has to be used / implemented (i.e. Liferay provisioning is done through a Web Service, LDAP is done through a specific connector, etc.).
    - Provisioning
        - Due to the fact that there are multiple applications to be deployed in the IdM/SSO scenario, there is no one standard way to provision users / user data to all the applications. To try to obtain maximum uniformity and maximum benefits from the tool, the provisioning models should follow a Role Base Access Control approach (RBAC). Each application will have to indicate the exact necessities for provisioning as well as specify the technological means through which this provisioning is to be done. The reason why such an approach is taken also has to be documented (for example, provisioning should be done through a Web Service instead of a direct access to a database table because of second level caching in the data access layer of the application).
    - Self Service Interface
        - Work flow approvals
        - Delegated Administration
    - Auditing
        - Logs
        - Reporting
    - Reconciliation with Authoritive sources

Integration

- Liferay
    - General user management should be blocked from Liferay because user provisioning is to be done by OIM. Specific user management details may still be done directly in Liferay but only by users with Liferay administration rights.
    - User first access challenge has to be disabled or alternatively the provision for this challenge has to be done in OAM / OIM.
    - Automatic importing of users / user data from LDAP has to be disabled as the user provisioning is solely to be done through OIM.
    - Even though Liferay is a "special" application (because it is a portal offering services to other applications), it should be considered as a separate application and have a specific LDAP group. This will allow a more uniform view of user provisioning amongst all applications.
    - Up until now the Community Version of Liferay has been used but in the production environment the Enterprise version will be used. This means that the IdM / SSO will need to be compatible with this latter referred version.

- Thetis
    - User Administration in Thetis should be removed as User Provisioning is to be done in OIM.

  - General authorizations are already done through JAAS so no changes are needed in this area.
  - Specific authorizations are still to be implemented directly inside Thetis as there is no current prevision for a "Fine Grain Access Control Infrastructure".
  - Due to application approval chain business requirements, provisioning will be done using delegated administration.

- STCW
  - Authentication / Authorization wise, no changes are needed to STCW because this application already uses standard JAAS functionalities.
  - The current version of STCW has a process which imports users that belong to STCWInspectors group. The correct form for STCW to obtain this information is for OIM to provision the data whenever a new STCW Inspector user is created. Once this is complete, STWC needs to delete the current process of importing Inspectors.

- Other
  - Each application/system is going to have to be analysed on a case-by-case basis to see what should be changed/removed/added. Without further study of each application/system to be included it is not possible to indicate which changes should be made.

The following diagram provides a "block view" of the possible architecture. In this diagram it is possible to see that all accesses are made through the Apache Server and Webgate module (acting as a reverse proxy). From here, if users are already authenticated, they may be permitted to access the web applications (Apache + Webgate -> Web Applications). If the users are not yet authenticated, they will be shown a Login Form from OAM for authenticating (Apache + Webgate -> OAM). After the users submit their credentials, these will be verified by OAM on the LDAP virtualization layer (OAM -> OVD) and if they are correct, a Session Token will be generated and returned to Apache for inclusion in all subsequent requests. Apache then redirects the user to the original URL requested. This authentication mechanism is used for all accesses that go through the Apache reverse proxy.

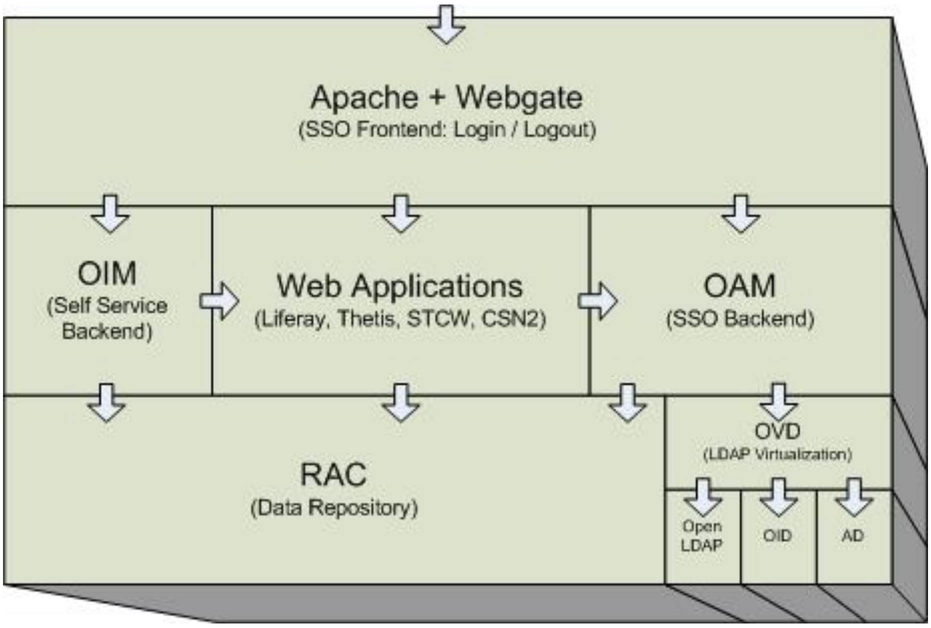


Figure 1: Block Diagram

In the event that the URL requested is part of the OIM self-service (Apache + Webgate -> OIM), there is a guarantee that users have already been authenticated and the corresponding functionality will be accessed. Depending on the action requested, OIM may

do provisioning work through a service interface (OIM -> Web Application) or directly to an application's database (OIM -> RAC).
If the URL requested corresponded to a web application (Apache + Webgate -> Web Applications), then the respective application may request Authorization information from OAM (Web Applications -> OAM). The exact process through which this is done will depend upon the application being deployed on Weblogic (in which case the request should be done through JAAS) or, if the application is not deployed on Weblogic (non java application), a call to the OAM API through custom code will need to be done. Please note that it may be possible for applications to access the Virtual LDAP layer instead of the OAM API but this is still an open issue.

The following diagram depicts the same information as the previous diagram, but through a more logical viewpoint. The machines depicted are purely "logical" and may not correspond to actual physical machines (they may be single, clustered or joined together depending on actual implementation constraints).

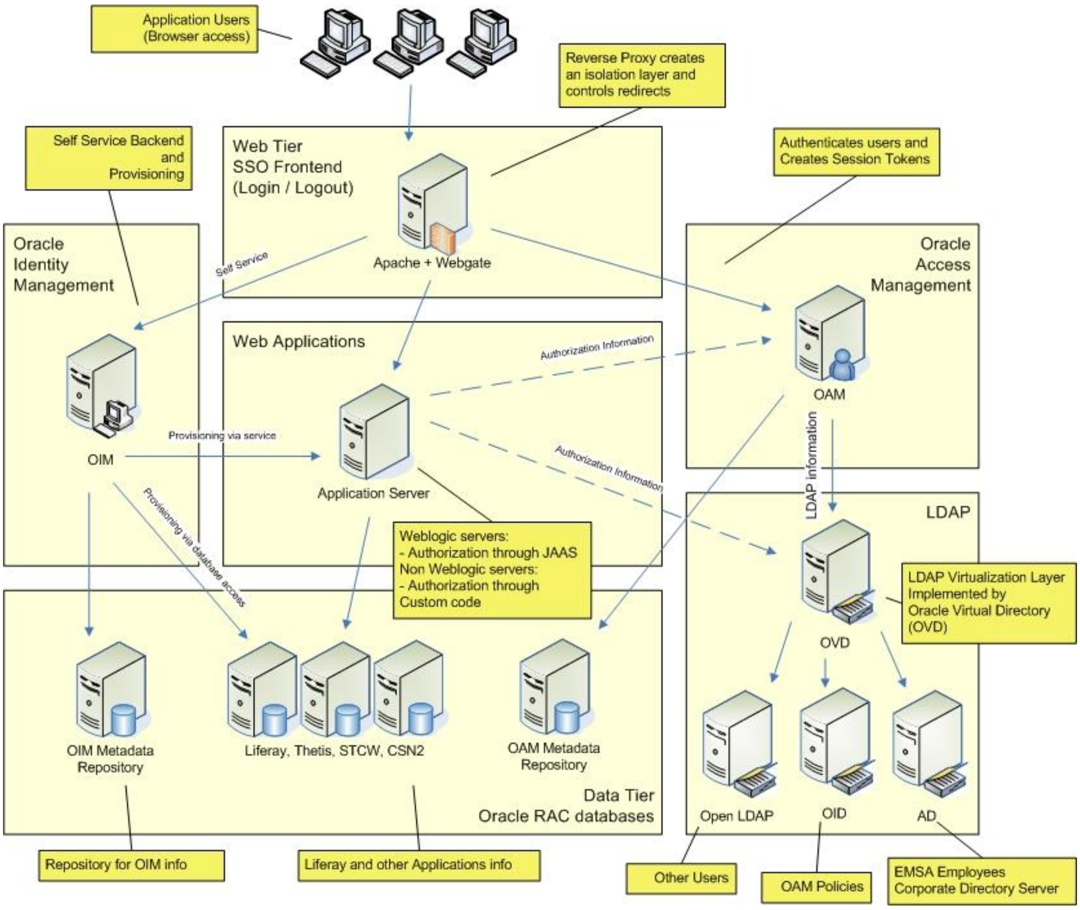

Application Users (Browser access)
Reverse Proxy creates an isolation layer and controls redirects
Self Service Backend and Provisioning
Web Tier
SSO Frontend
(Login / Logout)
Authenticates users and Creates Session Tokens
Oracle Identity Management
Self Service
Apache + Webgate
Oracle Access Management
Web Applications
Authorization Information
OAM
OIM
Provisioning via service
Application Server
Authorization Information
LDAP information
LDAP
Provisioning via database access
Weblogic servers:
- Authorization through JAAS
Non Weblogic servers:
- Authorization through Custom code
LDAP Virtualization Layer Implemented by Oracle Virtual Directory (OVD)
OVD
OIM Metadata Repository
Liferay, Thetis, STCW, CSN2
OAM Metadata Repository
Data Tier
Oracle RAC databases
Open LDAP
OID
AD
Repository for OIM info
Liferay and other Applications info
Other Users
OAM Policies
EMSA Employees Corporate Directory Server

Figure 2: Logical Overview

2.3. PHASE 2 – INTEGRATION WITH ACTIVE DIRECTORY (AD) AND IDM/SSO IN OTHER APPLICATIONS

This phase is comprised of the integration of the IdM / SSO infrastructure with Active Directory (AD) and also the complete adaptations to be executed on Safe Sea Net (SSN).

At this point, Oracle Virtual Directory (OVD) should be used (if it is not already being used) to permit the interconnection of Active Directory (AD) to the IdM / SSO infrastructure allowing EMSA employees to access applications (such as Thetis, STCW, etc.) by directly using their EMSA identities instead of having to provision specific application domain users.

Other applications will start to integrate into OIM / OAM solution. This will be an on-going process and will be studied in a case by case basis.

As was previously mentioned, it is possible that a new PoC be needed at this time for evaluating the Oracle Entitlement Server as a fine-grained access control solution to be used for implementing the access control that is not covered by the OIM / OAM solution. This is particularly important for the Safe Sea Net (SSN) system.

3. General Overview of IdM Solution

This chapter documents Oracles view of how the IdM solution can and should be implemented at EMSA. Please note that the information provided here may not correspond exactly to what is implemented.

3.1. AUTHENTICATION AND AUTHORIZATION

EMSA applications that require user authentication and authorization should rely on a directory to store user credentials, roles and access privileges.

User directory technologies
• openLDAP
• Application Server embedded LDAP

Although the use of a database schema to cope with these functions is a common practice, it has several disadvantages and should be avoided.

EMSA is currently running a pilot project implementing a Single Sign-On mechanism. It is envisaged that over time all existing applications will be using this new SSO mechanism. For new application development, developers should focus on:

- Relying on SSO for authentication
- Using JAAS for in-app authorization
- Weblogic App Server needs to be configured accordingly (JAAS + OAM agent)
- Use an RBAC model
- All administration of security principals will be handled through the Oracle Identity Manager.

The following schemas give an overview of the current SSO implementation.

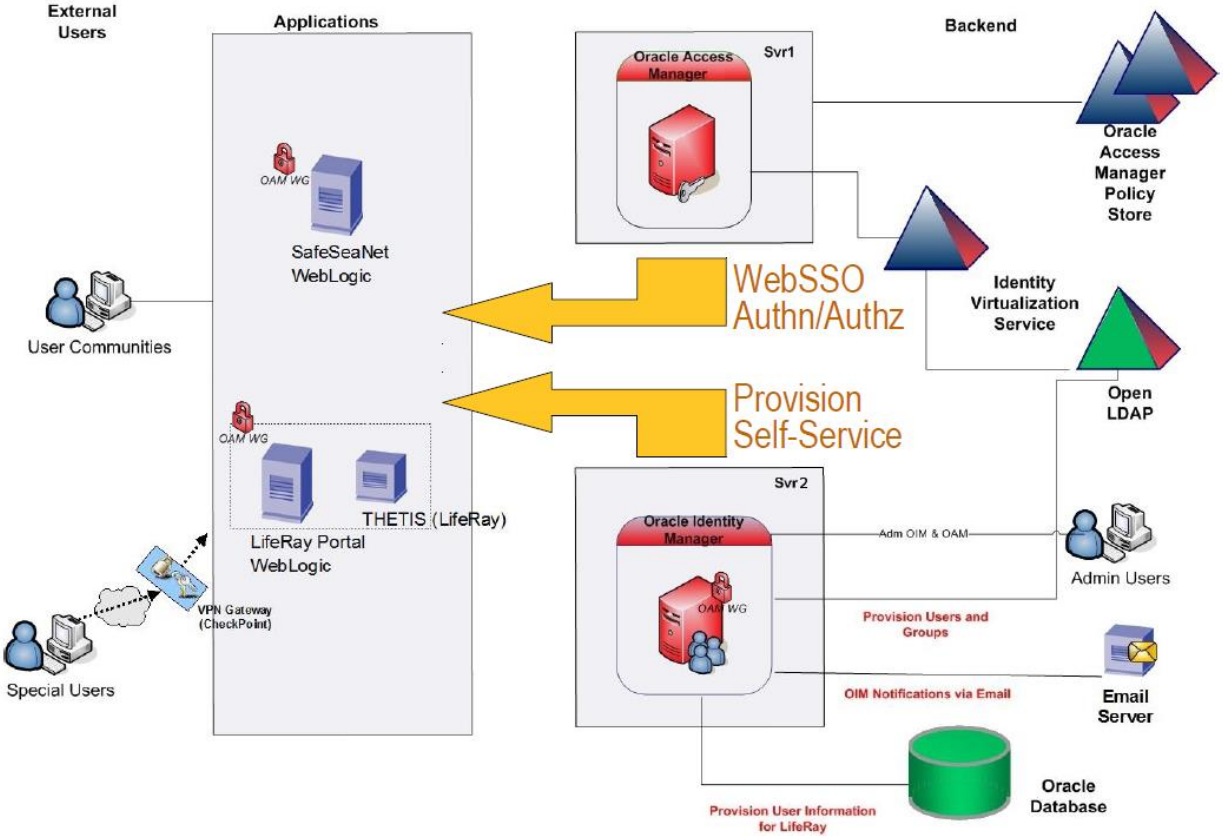


Figure 3: SSO high level diagram

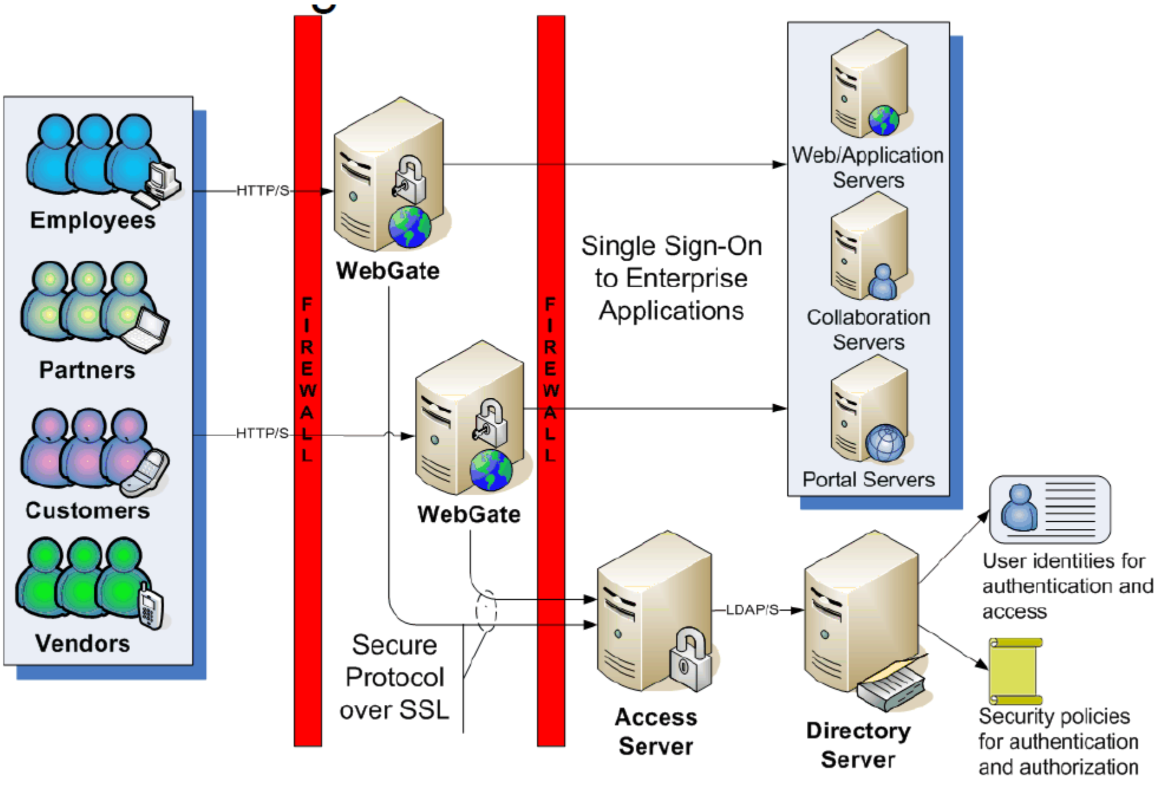


Figure 4: Access Manager Architecture

3.2. ADDITIONAL INFORMATION ON THE ORACLE PRODUCTS

The EMSA IdM solution uses a series of Oracle Products: Oracle Access manager, Oracle Identity Management, Oracle Internet Directory and Oracle Virtual Directory. Further information on any of these products may be found by following their respective links.

Oracle Access Manager 10gR3 (10.1.4.3.0)
http://www.oracle.com/technetwork/middleware/id-mgmt/index-090417.html?ssSourceSiteId=ocomen

Oracle Identity Management 10gR3
http://www.oracle.com/technetwork/middleware/id-mgmt/overview/index-098451.html?ssSourceSiteId=ocomen

Oracle Internet Directory 11g R1 (11.1.1.3 – 11.1.1.5)
http://www.oracle.com/technetwork/middleware/id-mgmt/overview/index-082035.html?ssSourceSiteId=ocomen

Oracle Virtual Directory 11g R1 (11.1.1.3 - 11.1.1.5)
http://www.oracle.com/technetwork/middleware/id-mgmt/index-093158.html?ssSourceSiteId=ocomen

4. Deploying Applications with Single Sign-On

Most of the EMSA applications are not prepared to be integrated with IdM and may need some changes. This chapter documents how these changes can be done by using a "generic" application such as the Java Pet Store reference application as a "Guiney pig". It also documents changes already made to some of the applications (such as RuleCheck) as well as documenting the necessary changes needed for others.

4.1. JPETSTORE

In the EMSA test environment, a well-known reference application – the Java Pet Store – has been deployed that allows for investigation and development of the Single Sign-On solution. One of the goals of deploying such an application in this environment was to assess the difficulties involved in adapting a web application to the Single Sign-On system.

Before going into the details of the necessary changes, we will first explain how the "normal" (unchanged) application works. The Java Pet Store application simulates an on-line shop for selling animals. There is "public" access to the application in which you can browse the existing information and you can even put items into a "shopping cart". If you decide to checkout your order, containing items in the shopping cart, you will have to log-in to the application to be identified. Only users that have been previously registered (provisioned) to the application may checkout orders. Likewise, if you wish to change your user attributes (password, address, phone, etc.) you must also be logged in.

Pre-emptive Authentication

A first interesting approach, while still not the desired one because of not fulfilling the previous "public user" functional requirements, will allow us to demonstrate how to perform authentication through Single Sign-On with minimum changes to the application. In this first approach, the whole application has been registered as "protected" in OAM (Oracle Access Management). This has the effect of the user/password being requested even before the first screen of the application is shown. After the initial logging in to OAM, there is no further need for identifying the user. If a user had already been authenticated in OAM prior to accessing any application screen, he will not be prompted to do so again (Single Sign-On). Please note that the only noticeable change in the application is the fact that the login form is never shown to the user.

Technical Considerations

We have indicated that the jPetStore application is now performing Single Sign-On with minimal changes to the application. We will now proceed to explain the actual changes made.

The first change is made to the web.xml file. Two new sections have to be added to this file to allow intercepting of HTTP requests by a filter (see change two for information on the actual filter developed).

web.xml

```
  …
  <display-name>JPetStore</display-name>
  <description>Online Pet Store Sample Application</description>

  <filter>
    <filter-name>AutoLogin</filter-name>
    <description>This is a test auto-login filter.</description>
    <filter-class>
      com.ibatis.jpetstore.presentation.filter.AutoLoginFilter
    </filter-class>
  </filter>

  <filter-mapping>
    <filter-name>AutoLogin</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
   ...
```

The second change is the creation of the actual filter code. You can see an example of this code following.

```
AutoLoginFilter.java

package com.ibatis.jpetstore.presentation.filter;

import java.io.IOException;
import java.util.Enumeration;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import com.ibatis.jpetstore.presentation.AccountBean;

public class AutoLoginFilter implements Filter {

  /** Flag for debugging purposes */
  private boolean debugging = false;

  /**
   * This is part of the Filter specification.
   * It can be used to initialize data.
   */
  public void init(FilterConfig arg0) throws ServletException {
    // Empty Statement
  }

  /**
   * This is part of the Filter specification.
   * It can be used to perform "garbage collection" of data objects.
   */
  public void destroy() {
    // Empty Statement
  }

  /**
   * This is part of the Filter specification.
   * This is called whenever there is an access to some URL of the PetStore
   * application.
   * The SM_USER http header variable is created by Oracle SSO and provides the
   * identification of the user accessing the application. If this does not exist,
   * then the user has not passed
   * through the Single Sign On entry point.
   * @param request The Http Servlet Request
   * @param response The Http Servlet Response
   * @param chain A Filterchain
```

```
    * @throws IOException Can be thrown if something goes wrong
    * @throws ServletException Can be thrown if something goes wrong
    */
  public void doFilter(ServletRequest request, ServletResponse response,
                       FilterChain chain) throws IOException, ServletException {
    if (request instanceof HttpServletRequest) {
      HttpServletRequest req = (HttpServletRequest) request;
      HttpSession session = req.getSession();

      String userId = req.getHeader("SM_USER");
      if (userId != null) {
        if (debugging) {
          System.out.println("In AutoLoginFilter ... userId(from SM_USER)=" +
                             userId);
          System.out.println("Remote User Name: " + req.getRemoteUser());
          String name = (req.getUserPrincipal() == null) ? "NONE-DEFINED" :
                                                   req.getUserPrincipal().getName();
          System.out.println("Principal Name: " + name);
          System.out.println("Authentication Type: " + req.getAuthType());
          System.out.println("Is a Manager(THETISReader): " +
                                                  req.isUserInRole("THETISReader"));
        }

        AccountBean accountBean =
                             (AccountBean) session.getAttribute("accountBean");
        if (null == accountBean || !accountBean.isAuthenticated()) {
          String incomingUrl = req.getRequestURI();
          if (debugging) {
            System.out.println("------ Incoming URL = " + incomingUrl + " ------");
          }
          if (incomingUrl.startsWith("/jpetstore/shop/checkout")
                || incomingUrl.startsWith("/jpetstore/shop/signonForm")
                || incomingUrl.startsWith("/jpetstore/shop/editAccountForm")) {
            String newUrl = "signon.shtml?username=" + userId + "&password=" +
                                                                          userId;
            if (debugging) {
              System.out.println("------ NewURL = " + newUrl + " ------");
            }
            RequestDispatcher request_Dispatcher =
                                  request.getRequestDispatcher(newUrl);
            request_Dispatcher.forward(request, response);
          }
        }
      }
      if (debugging) {
        showHeaders(req);
      }
    }

    chain.doFilter(request, response);
  }

  /**
   * Prints out the http HEADERS
   * @param req An HttpServletRequest object from which the headers can be
   * obtained
   */
  private void showHeaders(HttpServletRequest req) {
    String headers = null;
    Enumeration<String> e = req.getHeaderNames();
    while (e.hasMoreElements()) {
      headers = e.nextElement();
      if (headers != null) {
        System.out.println("HEADER (" + headers + ")=" + req.getHeader(headers));
      }
    }
  }

}
```

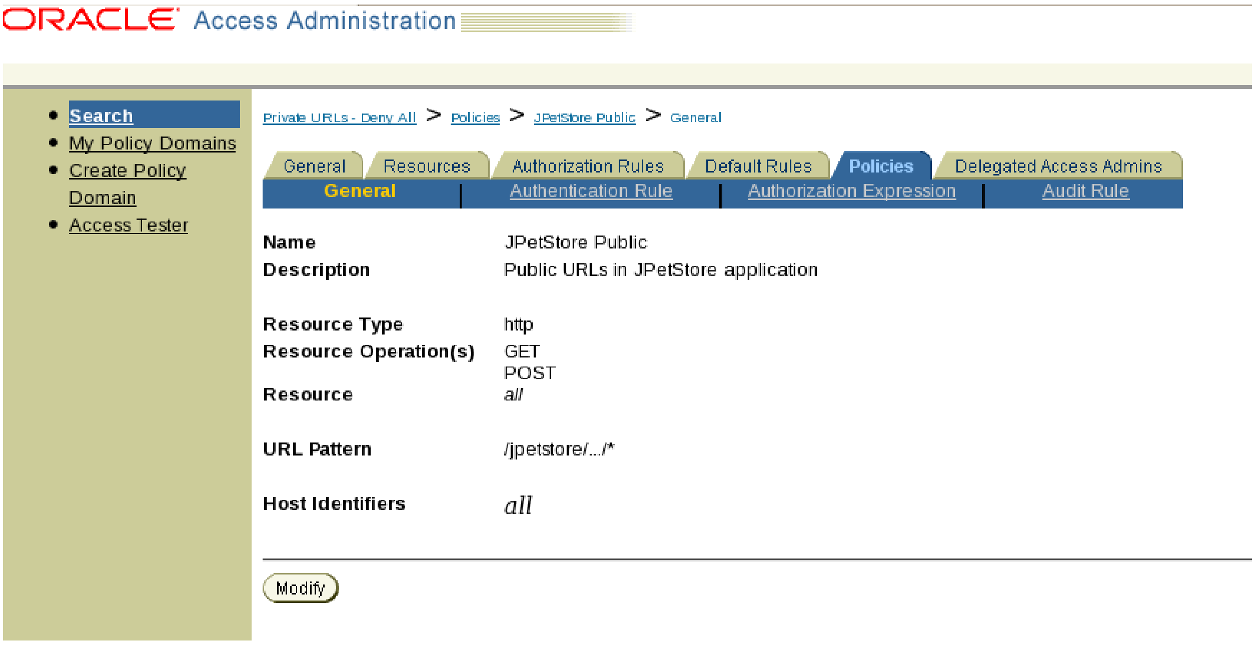
From inspection of the previous file we can conclude that three URLs were intercepted (the signonForm, the checkout and the editAccountForm). All three of these URLs have now been internally (internal to the server) redirected to the sign-on URL with additional parameters for the username and password. There are two comments to be made about this URL: first – it is always just internal to the server so there is no problem in sending the username and password as http GET parameters because the internal redirection can never be intercepted, and second – due to the fact that the user's password is never known outside of OAM, we need either to pass the username twice (serving as password) or pass a constant dummy password. This has to be consistent with the provisioning process followed.

Public and Private access to the application

As we have previously stated, the pre-emptive authentication scheme is not our target. As such, we now need to make some changes to the OAM to be able to comply completely with the full functional requirements. It is important to point out that there will not be the need to make any more changes to the jPetStore application, but the previously performed changes are still necessary for this stage.

Technical considerations for granting public access

As a result of the previous section, the jPetStore application is a protected resource which will require user authentication to be accessed. However, the functional requirements state that there is a part of the application that has public access. Though there might be (and probably are) other ways of granting public access to an application under OAM, we have chosen to configure OAM the following way.

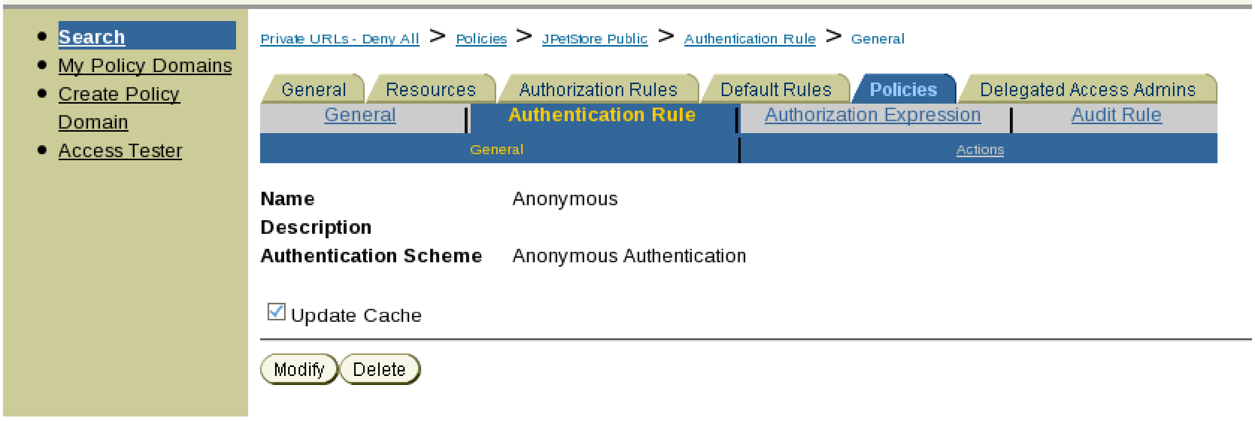


Name	JPetStore Public
Description	Public URLs in JPetStore application
Resource Type	http
Resource Operation(s)	GET POST
Resource	*all*
URL Pattern	/jpetstore/.../*
Host Identifiers	*all*

Figure 5: jPetStore public policy

In Oracle Access Manager, access the Policy Manager Application. Under the "Private URLs" policy domain, we will add another policy to the ones already existing in this domain. We have called this new Policy "JPetStore Public" and it consists of an http policy on GETs and POSTs, for all resources and all host identifiers, with the "/jpetstore/…/*" URL pattern.

In order for this policy to work correctly, the "Authentication Rule" associated to it must be that of "Anonymous Authentication" without any specific "Actions".



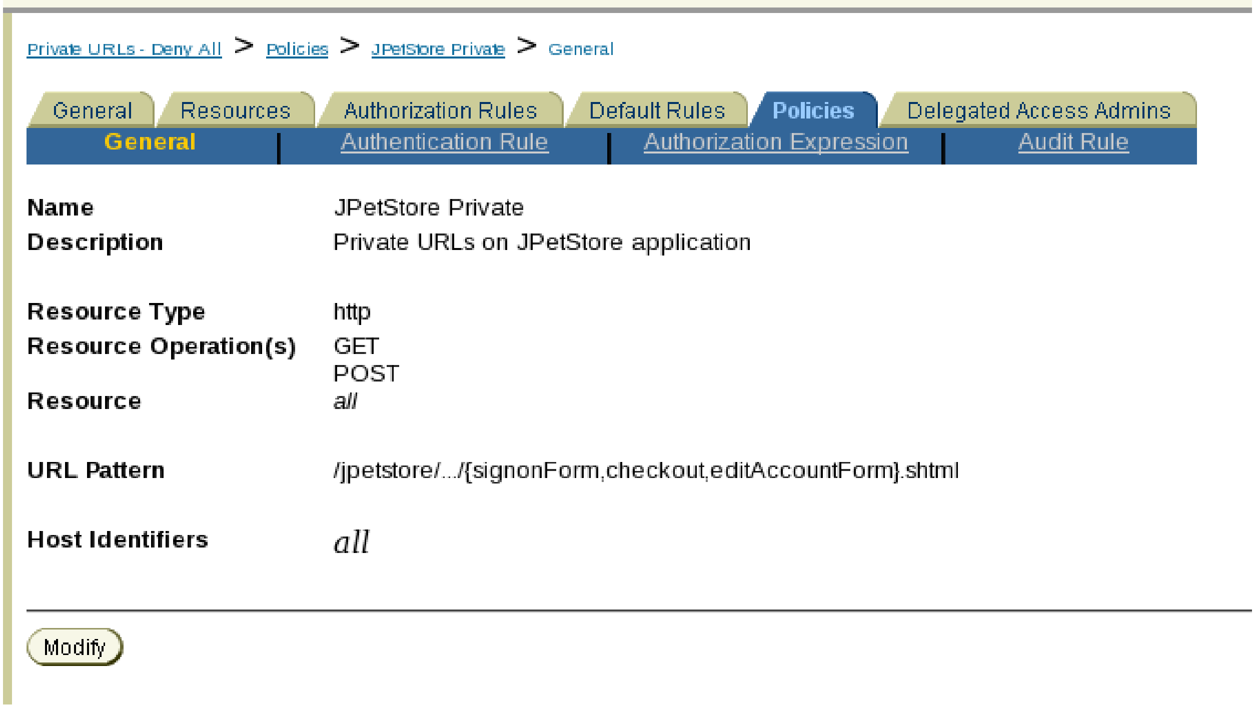
Name Anonymous
Description
Authentication Scheme Anonymous Authentication

☑ Update Cache

Figure 6: jPetStore public policy - anonymous access

Once these changes are made, and once the OAM cache has been duly updated, no more user authentication is needed to access the application. There should now be no Authentication Form presented to the user whenever he accesses the jPetStore application, whatever the operation performed within. This, however, is not what is intended as the user will now have to perform an application login (answering to an application login form – not the OAM one) whenever we tries to access the "private" area of the application (accessing the user account or checking out an order). This means that we now need another policy to protect what we have just un-protected.

Technical considerations for protecting parts of a public access application



Name JPetStore Private
Description Private URLs on JPetStore application

Resource Type http
Resource Operation(s) GET
POST
Resource *all*

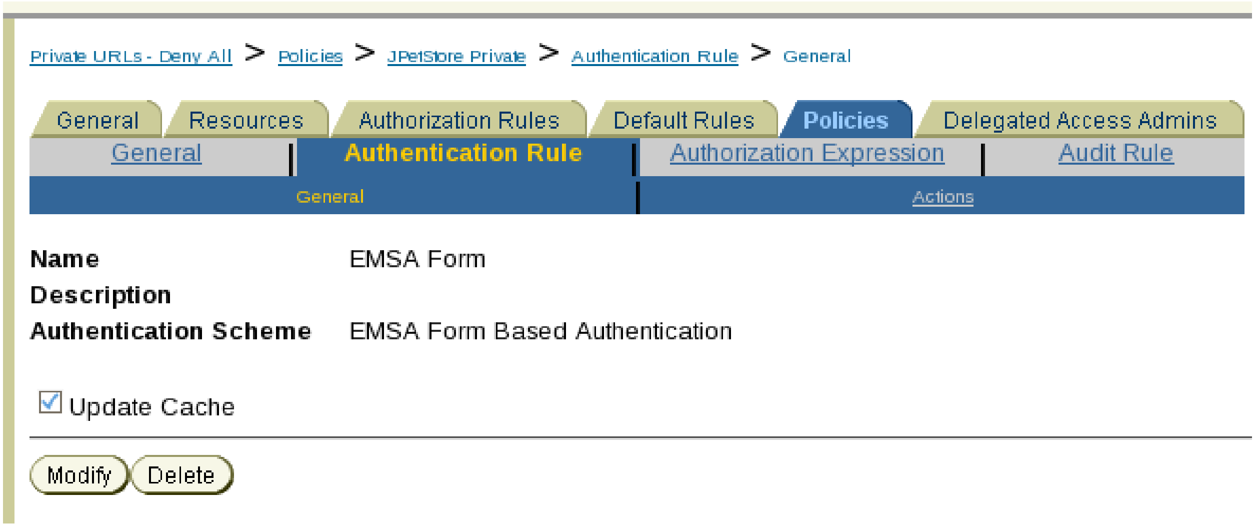
URL Pattern /jpetstore/.../{signonForm,checkout,editAccountForm}.shtml

Host Identifiers *all*

Figure 7: jPetStore private policy

Once again, as a result of the previous section, the jPetStore application is not fully compliant to the requirements. We will now have to add a new policy (still under the "Private URLs" policy domain) to protect the "checkout" and "editAccount" functionalities. This new policy will have the following values:

An http policy on GETs and POSTs, for all resources and all host identifiers, with the "/jpetstore/.../{signonForm,checkout,editAccountForm}.shtml" URL pattern.

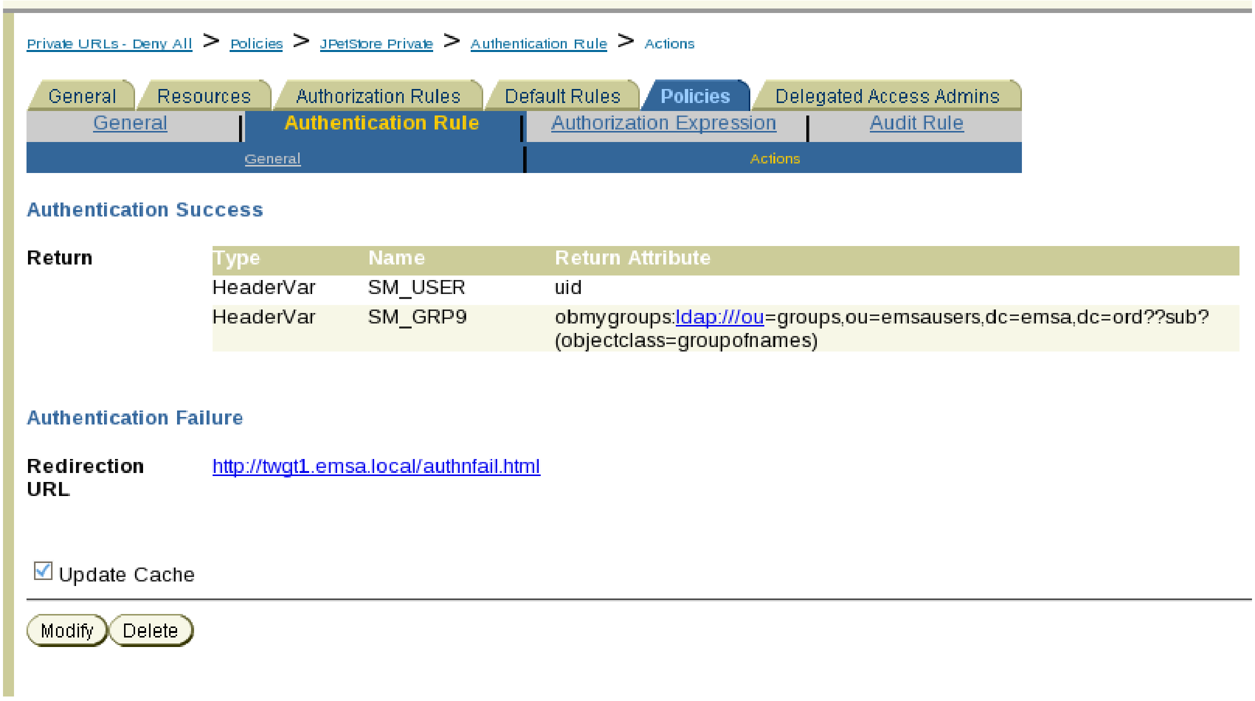


Name	EMSA Form
Description	
Authentication Scheme	EMSA Form Based Authentication

☑ Update Cache

Figure 8: jPetStore private policy - Form based authentication

In this case, the “Authentication Rule” is now “EMSA Form Based Authentication” and the corresponding Actions are the creation of 2 “Authentication Success” – “Returns” consisting of Type “HeaderVar”, Name “SM_USER” and “SM_GRP9” with Return Attributes of “uid” and “obmygroups : ldap : /// ou = groups, ou = emsausers, dc = emsa, dc = ord ?? sub ? (objectclass = groupofnames)” (without the spaces) respectively. Further to these “Returns”, the “Authentication Failure” – “Redirection URL” should be pointing to the html page that signals an authentication error “http://twgt1.emsa.local/authnfail.html”.



Authentication Success

Return

Type	Name	Return Attribute
HeaderVar	SM_USER	uid
HeaderVar	SM_GRP9	obmygroups:ldap:///ou=groups,ou=emsausers,dc=emsa,dc=ord??sub?(objectclass=groupofnames)

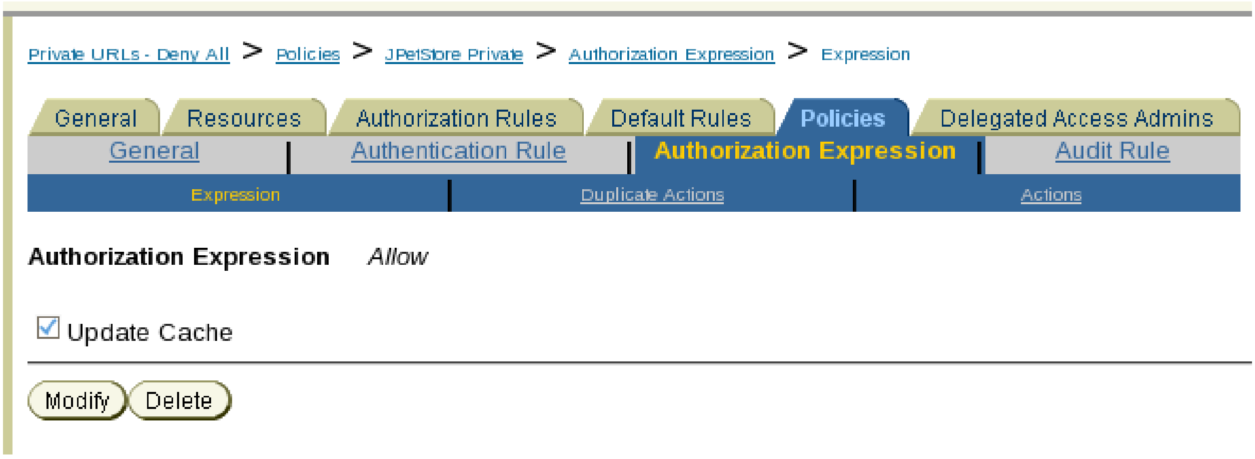
Authentication Failure

Redirection URL http://twgt1.emsa.local/authnfail.html

☑ Update Cache

Figure 9: jPetStore private policy - Authentication role

The final step is to configure the “Authorization Expression” by creating an Allow expression

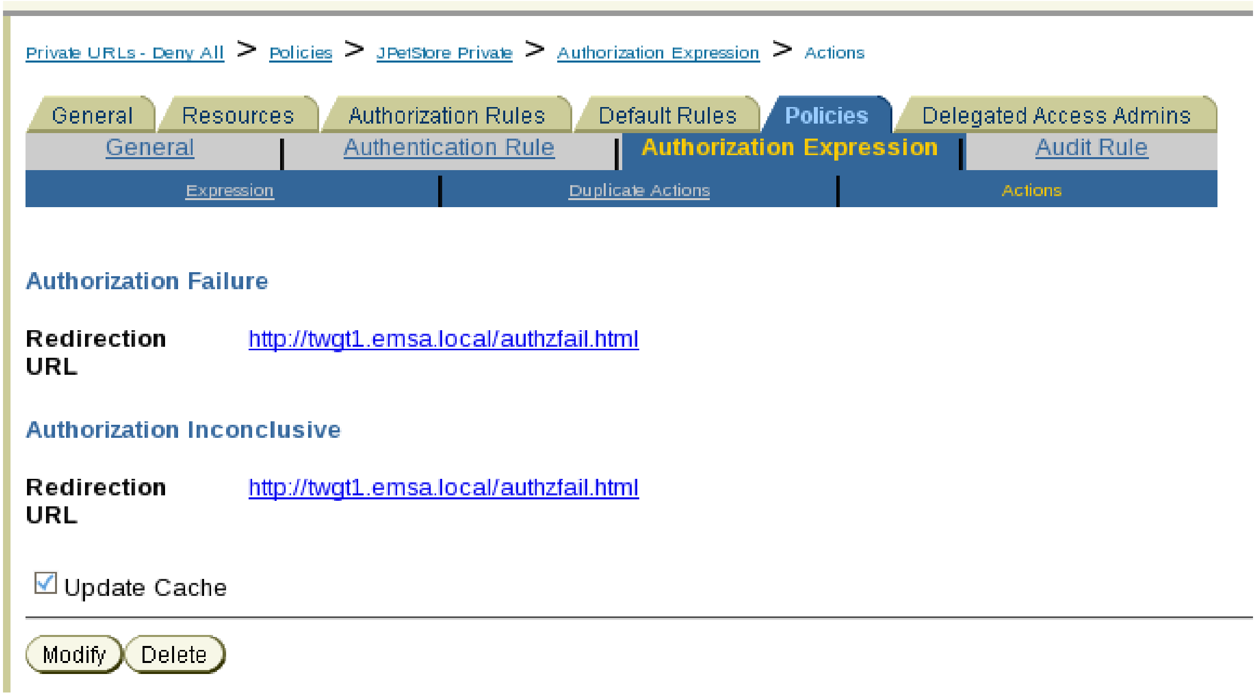


Authorization Expression *Allow*

☑ Update Cache

Figure 10: jPetStore private policy - Authorization Expression

with "http://twgt1.emsa.local/authnfail.html" as the "Redirection URL" for both the "Authorization Failure" and "Authorization Inconclusive" Actions.



Authorization Failure

Redirection URL http://twgt1.emsa.local/authzfail.html

Authorization Inconclusive

Redirection URL http://twgt1.emsa.local/authzfail.html

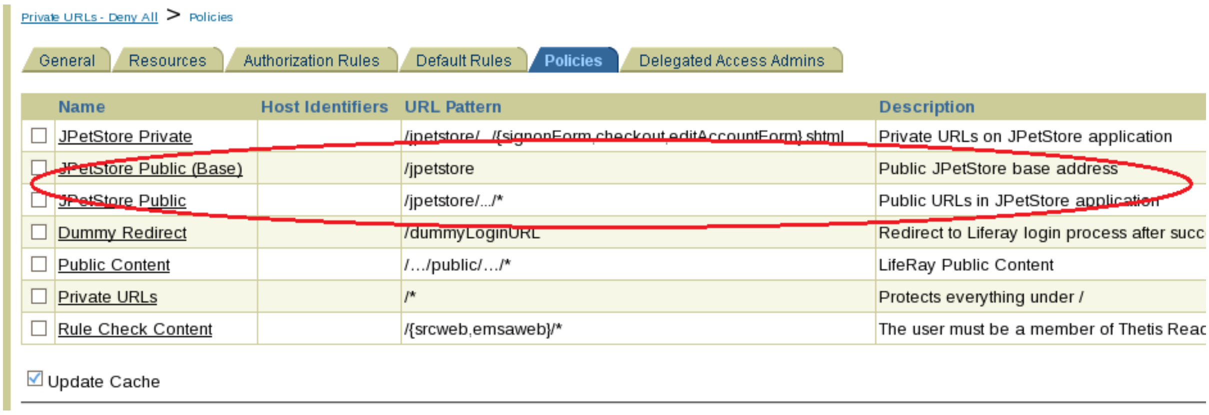
☑ Update Cache

Figure 11: jPetStore private policy - Redirections

At this point, after having saved all the necessary changes made, there user should be presented the EMSA login form for authenticating whenever he accesses the SignonForm, Checkout or EditAccountForm URLs.

Final notes on configuration

Due to our current limited knowledge of the Oracle products, we have had the need to configure another policy, exactly the same as the previously mentioned "JPetStore Public", associated to the public URL for the application "/jpetstore". It is possible that the two public URLs, "/jpetstore/…/*" and "/jpetstore", might be merged into a single expression but we do not currently possess the knowledge to do so.



Private URLs - Deny All > Policies

	Name	Host Identifiers	URL Pattern	Description
☐	JPetStore Private		/jpetstore/.../{signonForm,checkout,editAccountForm}.shtml	Private URLs on JPetStore application
☐	JPetStore Public (Base)		/jpetstore	Public JPetStore base address
☐	JPetStore Public		/jpetstore/.../*	Public URLs in JPetStore application
☐	Dummy Redirect		/dummyLoginURL	Redirect to Liferay login process after succ
☐	Public Content		/.../public/.../*	LifeRay Public Content
☐	Private URLs		/*	Protects everything under /
☐	Rule Check Content		/{srcweb,emsaweb}/*	The user must be a member of Thetis Reac

☑ Update Cache

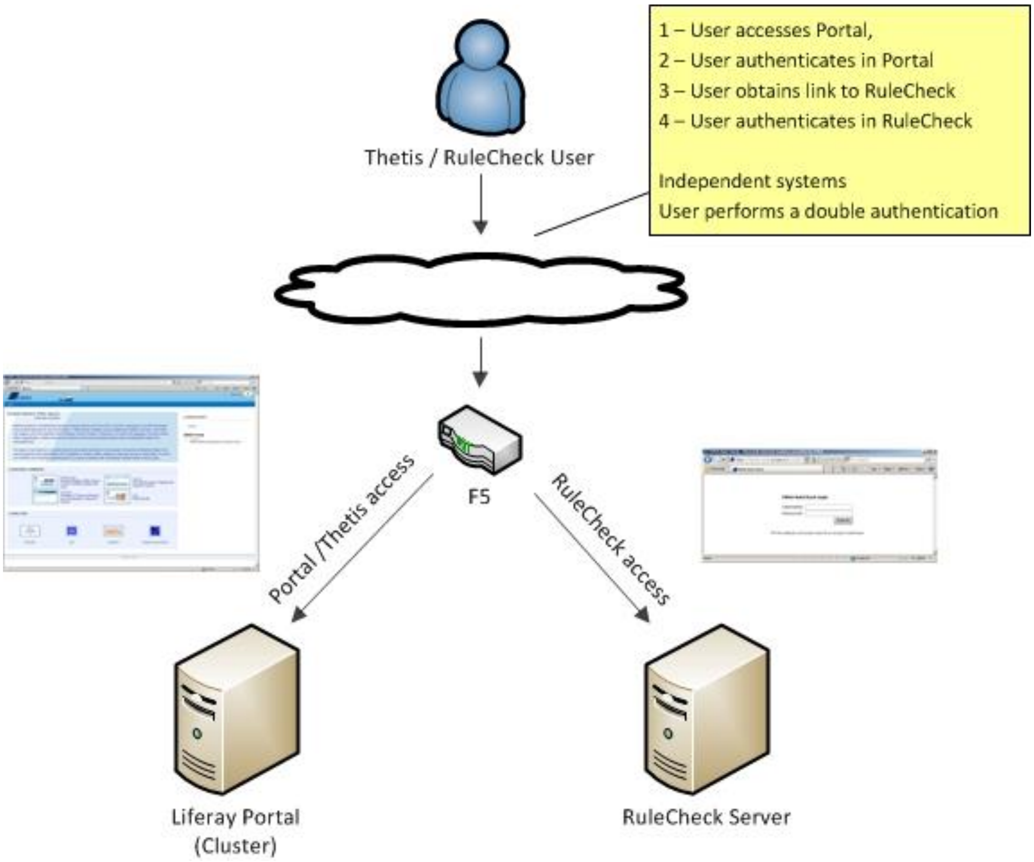
Figure 12: jPetStore public policies

4.2. RULECHECK

There are currently two versions installed of the RuleCheck application: a protected version (self-protected from within the application packaging) and a non-protected version. The latter is the version that is of interest for Single Sign-On purposes.

Accessing RuleCheck prior to protection by Oracle Access Manager

Before the implementation of the Oracle Access Management - Single Sign-On, when a user wanted to access RuleCheck, he would either start by following the link that he could obtain from the Portal (if the URL was not previously known), or he could go directly to the URL of the RuleCheck application if it were already known.



1 – User accesses Portal,
2 – User authenticates in Portal
3 – User obtains link to RuleCheck
4 – User authenticates in RuleCheck

Independent systems
User performs a double authentication

Figure 13 - Non SSO access

In the first step, the user would be prompted to present his credentials to access the Portal. Having been authenticated in the Portal, the user could then obtain the link to the RuleCheck application. Upon following this link, he would be directed to the RuleCheck application (see second step description).

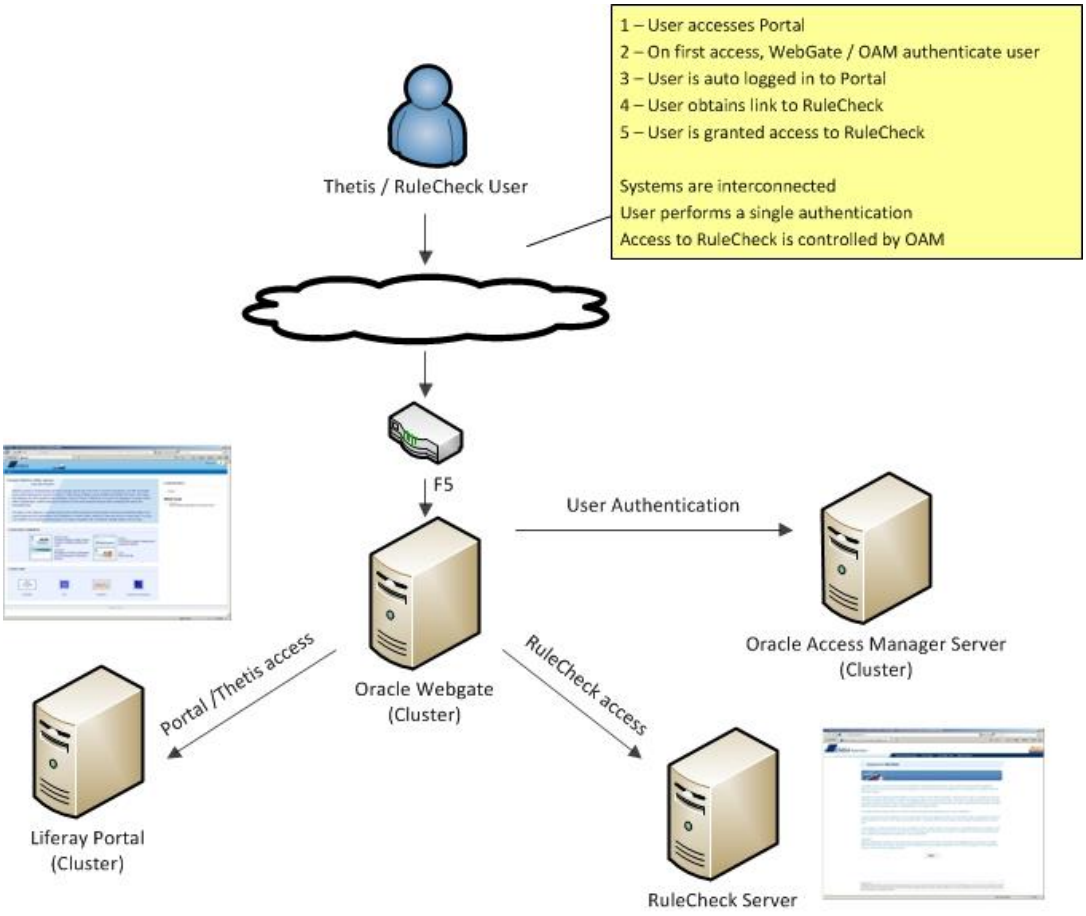
In the second step, either with a known URL or after having obtained the URL from the first step, the user would be prompted to supply a set of credentials for accessing the RuleCheck application. Once authenticated, the user has access to the full set of functionalities provided by the RuleCheck application.

The points to retain are:

- The two Systems (Portal and RuleCheck) are completely independent of one another and no physical relation exists between them,
- Due to the previous fact, the authentication process is duplicated with the user having to provide access credentials twice, once for each system. There is no relation between the two user identities (even though they may be apparently the same – for example: have the same user name and even the same password).

Accessing RuleCheck as a resource protected by Oracle Access Manager

After the implementation of the Oracle Access Management - Single Sign-On, when a user wants to access RuleCheck, there will be a different behaviour than that previously described. Similar to what was previously mentioned, the user would either start by following the link that he could obtain from the Portal (first step – if the URL was not previously known), or he could go directly to the URL of the RuleCheck application if it were already known (second step).



1 – User accesses Portal
2 – On first access, WebGate / OAM authenticate user
3 – User is auto logged in to Portal
4 – User obtains link to RuleCheck
5 – User is granted access to RuleCheck

Systems are interconnected
User performs a single authentication
Access to RuleCheck is controlled by OAM

Figure 14 - SSO protected access

The differences from the previous case start when the user tries to access either the Portal or the RuleCheck application. Because both systems will be part of a common protected resource, when trying to access either of them, the request will be intercepted by the Oracle Webgate. The Webgate will check to see if the user has already previously logged-in to the system. If not, the request will be redirected to the OAM server (Oracle Access Management server) who will present a login screen, receive the user credentials and authenticate him. After having done this once, the Webgate will detect that the user has already logged-in and will forward the user's requests directly to the systems (if the user has permissions to access those systems).

In this situation, if the user has requested to go to the Portal, he will be automatically logged-in to the Portal by the SSO solution. This means that he will not have to supply credentials to do so, the SSO solution will have done this for him. From here, he can obtain the URL to the RuleCheck application.

Alternatively, the user might have been trying to access the RuleCheck application directly (without having gone to the Portal first). In either case, the RuleCheck application will not request for credentials because it knows that the user has already been authenticated and also that only users that have the correct permissions will be able to access the RuleCheck application (since the Webgate intercepts all requests to the RuleCheck URL, it will not allow access if the user's profile does not coincide with the one necessary for accessing the application). Under this solution, only the duly authenticated users who have permissions to access RuleCheck will be able to do so.

The points to retain here are:

- The two systems are now related to each other due to the fact that they belong to the same protected resource (domain emsa.europa.eu),
- The user needs only perform a single authentication (at the time of first access) when accessing any of the protected systems,
- The user is effectively unique – has a single set of credentials that identifies him in all the protected systems,
- The protection for the RuleCheck application is performed by the Oracle Access Manager and not by the application itself.

Technical Considerations

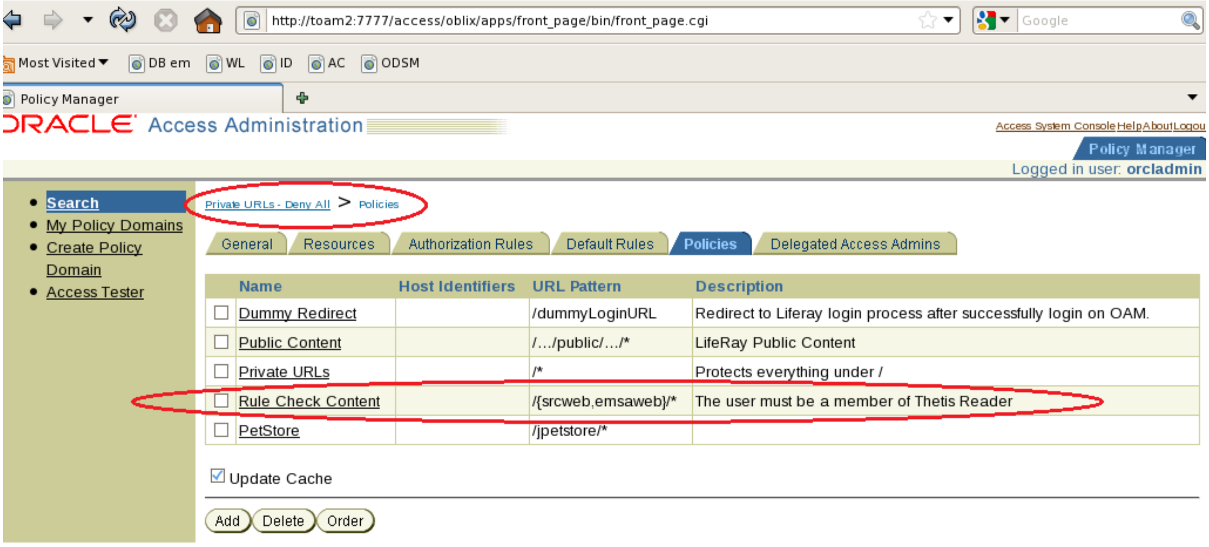
The non-protected version of RuleCheck (whenever accessed directly) will not request any user credentials, thus allowing anyone to access it. The correct form of access to this application is thru the Single Sign-On Webgate which will redirect requests accordingly whilst interacting with Oracle Access Manager to enforce access security. The configuration file (as of the 3rd of May 2011) for the Webgate is as follows:

httpd-rulecheck.conf

```
#-----> RuleCheck
  ProxyPass /srcweb http://twls33.emsa.local:7031/srcweb
  ProxyPassReverse /srcweb http://twls33.emsa.local:7031/srcweb

  ProxyPass /css http://twls33.emsa.local:7031/css
  ProxyPassReverse /css http://twls33.emsa.local:7031/css

  ProxyPass /js http://twls33.emsa.local:7031/js
  ProxyPassReverse /js http://twls33.emsa.local:7031/js

  ProxyPass /images http://twls33.emsa.local:7031/images
  ProxyPassReverse /images http://twls33.emsa.local:7031/images

  ProxyPass /emsaweb http://twls33.emsa.local:7031/emsaweb
  ProxyPassReverse /emsaweb http://twls33.emsa.local:7031/emsaweb
```

Basically, the previous file redirects all the application URLs to their correct location on the application machine.
Besides the existence of the previous file, it is also necessary to include a section for RuleCheck in the httpd-vhosts.conf file (also as of the 3rd of May 2011). This declares a new virtual host for the application:

```
httpd-vhosts.conf
…
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# RULECHECK
# twgt1.emsa.local : 7081  (RuleCheck on 7081)
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<VirtualHost *:7081>

    ServerAdmin root@localhost
    UseCanonicalName On
    ServerName twgt1.emsa.local:7081
    DocumentRoot /var/www/html/
    ErrorLog /etc./httpd/logs/rulecheck-7081.log
    ProxyRequests Off
    ProxyPreserveHost On

    #First Location to redirect to index.html if we try to access to /
    <LocationMatch "/">
      RedirectMatch ^/$ /index_rulecheck.html
    </LocationMatch>

    # Don't bother looking for favicon.ico
    Redirect 404 /favicon.ico
    # Don't bother sending the custom error page for favicon.ico
    <Location /favicon.ico>
        ErrorDocument 404 "No favicon"
    </Location>

Include conf/extra/httpd-rulecheck.conf

</VirtualHost>
…
```



	Name	Host Identifiers	URL Pattern	Description
☐	Dummy Redirect		/dummyLoginURL	Redirect to Liferay login process after successfully login on OAM.
☐	Public Content		/…/public/…/*	LifeRay Public Content
☐	Private URLs		/*	Protects everything under /
☐	Rule Check Content		/{srcweb,emsaweb}/*	The user must be a member of Thetis Reader
☐	PetStore		/jpetstore/*	

☑ Update Cache

Figure 15: RuleCheck private policy

The Webgate alone is not enough to guarantee security. Oracle Access Manager has been configured to have a policy that only allows ThetisReaders to access the RuleCheck URLs (please note that at the current time of writing, due to a limitation in the LDAP server, it is not possible to have the correct settings of ThetisReaders OR RuleCheckUsers. This will be corrected in the future when a new version of the LDAP server supporting this functionality becomes available):

An important note is that the non-protected RuleCheck application has not been modified in any way whatsoever; the access protection comes from the Oracle systems alone.

As a final note, it should be noted that at the current moment, the Liferay Portal is still not being protected by the OAM solution. This means that effectively the only application that is under the Single Sign-On protection is RuleCheck. In the previous diagram, this corresponds to the F5 talking directly to the Liferay Portal Cluster instead of going through the Oracle Webgate (as depicted).

4.3. STCW

The EMSA test environment does not have latest version of the STCW application portlets because of changes needed to be made (manually) in the weblogic.xml files. These changes are:

weblogic.xml – OLD version (obsolete with SSO)

```
…
  <security-role-assignment>
    <role-name>liferay_users</role-name>
    <externally-defined />
  </security-role-assignment>
…
```

which should be changed to

weblogic.xml – NEW version (to be used with SSO)

```
…
  <security-role-assignment>
    <role-name>liferay_users</role-name>
    <principal-name>liferay_users</principal-name>
  </security-role-assignment>
…
```

However, even though this change allows access to the portlets in Liferay, it does not allow for full functionality of these same portlets. There is currently a problem with STCW in which the user that has logged into Liferay is obtained and used as a parameter in the calls to the business components. The actual problem is when trying to obtain the password to pass which is no longer available under single sign-on.

WideScope is studying the problem and has suggested two possibilities:
Still on the presentation layer, build a SAML token which can be passed to the business layer for this to acknowledge user authentication, and alternatively
a work-around consisting of using a “fixed” user (a “system” user with fixed – database stored - credentials) to access the business layer.

4.4. THETIS

Details will be added in the next version of this document

5. Provisioning Applications

This chapter aims at providing an overview of the user provisioning process for applications deployed at EMSA. It describes a generic tier model that shows how parts of provisioning can be done at different levels and it also describes two possible work-flows that can be applied to the provisioning process.

5.1. PROVISIONING TIER MODEL

One of the goals of having an Identity Management solution in EMSA is to have a common way (as much as possible) of provisioning users to applications. This essentially means that whatever can be found common to all possible applications should be done in a single point in Identity Manager leaving any particularities up to the specific applications that need these particularities. These particularities can be done in custom forms inside Identity Manager and/or directly inside the applications. This approach leads us to a three tier provisioning model that is depicted in the following diagram.

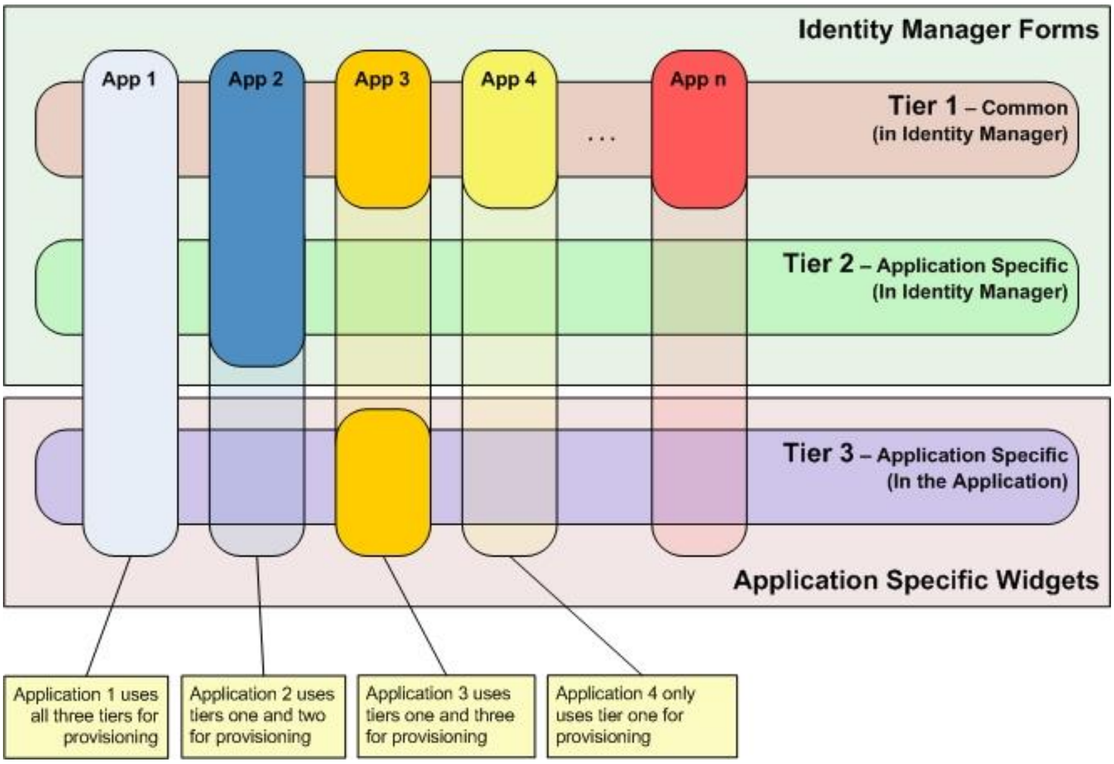

Identity Manager Forms
App 1
App 2
App 3
App 4
...
App n
Tier 1 – Common (in Identity Manager)
Tier 2 – Application Specific (In Identity Manager)
Tier 3 – Application Specific (In the Application)
Application Specific Widgets
Application 1 uses all three tiers for provisioning
Application 2 uses tiers one and two for provisioning
Application 3 uses tiers one and three for provisioning
Application 4 only uses tier one for provisioning

Figure 16 - Tier Model for Provisioning

In the previous figure, the first tier corresponds to the Identity Manager forms[1] that will be filled with the data that is common to (most of) the applications. Due to the fact that all users accessing EMSA applications need to exist in an LDAP server (or any other form of data storage that can be seen as such), the basic LDAP provisioning will be done automatically by this first tier. Depending on the work-flow involved (work-flows are seen later in this document), and also on the various applications' specific needs, parts of this data may be replicated to other forms in Identity Manager that are application specific. Likewise, further information can be added to LDAP by the following tiers if such is needed. This leads up the next tier in the model.

---

[1] The creation of the Forms in the first tier will be the responsibility of the SSO/IdM project and will not be imputed to existing applications.

The second tier might be, in some cases, an optional tier depending on each specific application[2]. Whenever an application needs certain information that is not part of the common layer (first tier in this model) but is still sufficiently application independent as far as the user interface is concerned (i.e. does not need a specific custom user interface to obtain the data), then this information can be filled in through custom forms still inside the Identity Manager. At this point all of the common data is available to the forms from the previously filled in first layer.

It is convenient to point out that for the data from both tier 1 (common) and tier 2 (application specific), to be passed on to the applications, these need to provide specific services[3] (in the form of Web Services, Database Stored Procedures or any other types of remote services) to be invoked from Identity Manager. These services should be as much as possible based upon normalized, secure standards (for example Web Services) that can use existing standard Identity Manager Adapters. Any form of service that does not comply with a standard adapter may have the added overhead of having to develop an adapter for use in Identity Manager.

The third tier, which like the second tier, may or may not be necessary, is completely application specific. In this tier, the user interface for gathering the provisioning information is to be done with native application widgets (there can be no use of Identity Manager – otherwise it would still be tier two and not three). This tier, if needed, will most probably correspond to the most complex user interface and most complex business logic for the provisioning of a user in the application (once again, otherwise it could be done in one of the other two tiers). The implementation of this tier should be done in such a manner that it could be seamlessly (at least apparently from the users' point of view) called from the provisioning process done inside Identity Manager (for example by providing a URL link that could be embedded in the Identity Manager User interface).

Having described the Tier Model, we will now concentrate on possible Work-flows using these various tiers.

5.2. PROVISIONING WORK FLOWS

The previously mentioned three tier model can be associated to various work-flows for provisioning. The two which are foreseen to be the most common ones will be described in some level of detail. These are a *Role Oriented Work-flow* in which one user will be provisioned to multiple Applications[4] according to a *Policy* (most probably to be used by administrators managing applications), and also a *System Oriented Work-flow* in which a

---

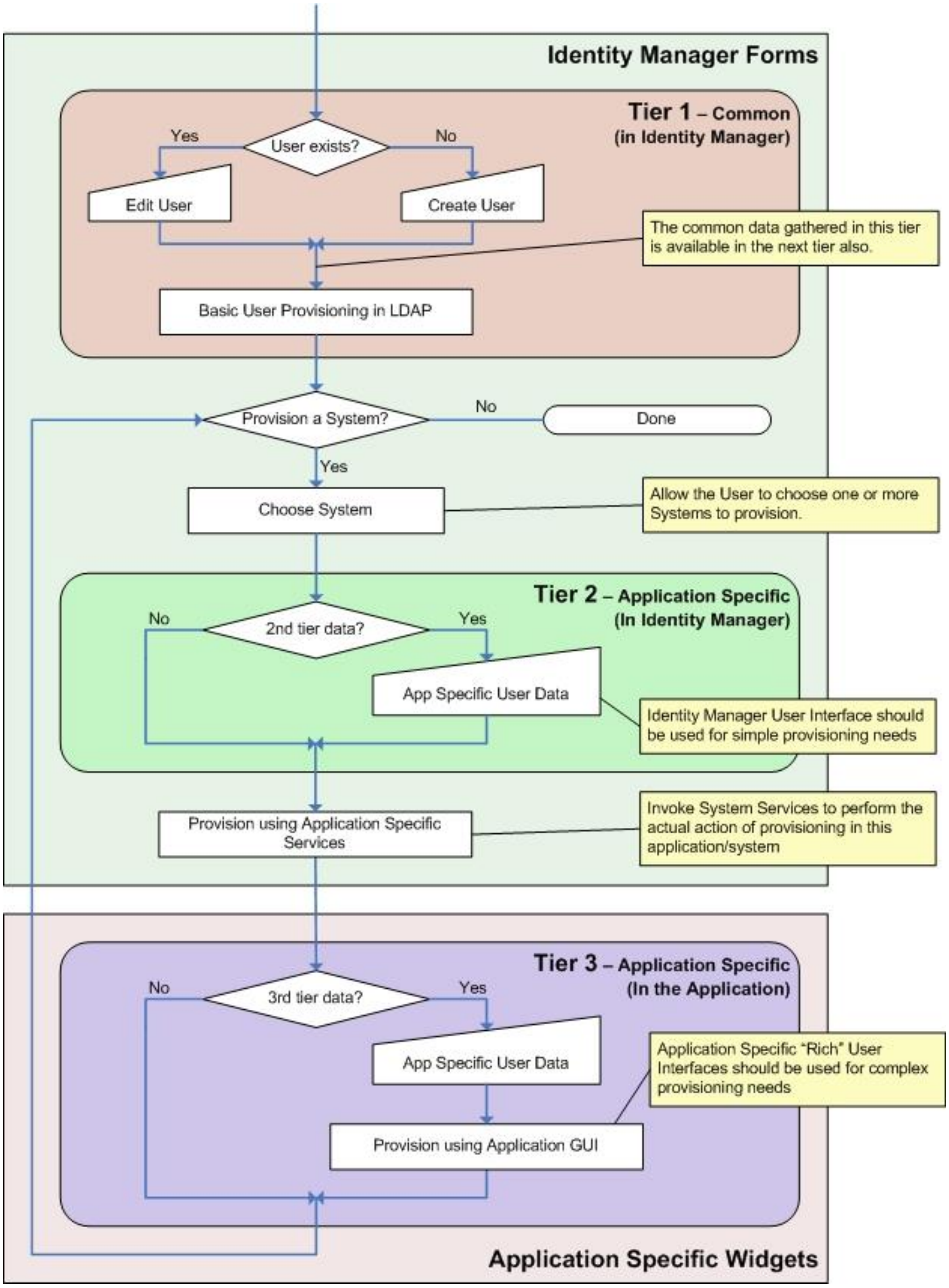
[2] Any Forms that may be necessary in the second tier will be imputed to the application needing the information there gathered. However, if these forms are constructed during the development phase of the SSO/IdM project, there may be some help provided from the contractor of this project to ease the burden of creation of these forms as well as to help accelerate the whole process. Any application using this process that is developed after the terminus of the SSO/IdM project will have to develop these forms on its own account.

[3] The services needed for provisioning user information in the applications are the sole responsibility of the team developing such applications. These services should be developed according to the specifications for provisioning general information. If there is a need for tier two forms, then besides having to create the services for provisioning, these should be duly documented.

[4] For the purpose of this document, an Application is interpreted as something that provides a contextualized set of goals for users, for example THETIS or STCW or CSN. Any other software that provides a more generic set of features is considered a System (see next footnote on this subject).

user is provisioned in a single System[5] at a time (most probably to be used for and by EMSA users).

System Provisioning



The common data gathered in this tier is available in the next tier also.

Allow the User to choose one or more Systems to provision.

Identity Manager User Interface should be used for simple provisioning needs

Invoke System Services to perform the actual action of provisioning in this application/system

Application Specific "Rich" User Interfaces should be used for complex provisioning needs

Figure 17 - System Provisioning Work-flow

The previous figure shows a possible work-flow associated with provisioning a user in one System at a time. This provisioning can/will typically be done by an administrator of the

---

[5] In this particular context, the use of the word System can be analogous to Application. It is being used instead of the word Application because we tend to think of an LDAP server more as a System than as an actual application, for example. Another such example would be the Liferay Portal. Even though it is an application, due to its nature of supporting portlets and hence "applications" inside it, it can be seen as a system.

system or it can be delegated to another user, permissions allowing. This will probably not be the preferred way to provision a user to an Application as the best way to do so would be to associate him with a given Role in the Application (that will be seen later on in this document).

The flow starts by performing a lookup of the user by way of some unique identifier. If the user is found (if he already exists), then the form shown will include this users' data and this data can be changed. If the user does not exist yet, then a clean form will be presented where all the relevant information should be filled in. This first step of providing the relevant "general" information about the user to be provisioned is done inside the first tier. Once the information is complete, the user can then be provisioned to LDAP (the reason for this has already been explained previously and is related to the fact that all EMSA applications use some form or another of an LDAP server to identify the users). If the user who is being provisioned just needs to be provisioned in the LDAP system, then the flow can end here. If more than this simple LDAP provisioning is needed then the following steps are also used.

At this point it is possible to choose any of the existing systems (subject to the permissions of the user performing the tasks of provisioning) and perform a specific provisioning for this system. This may correspond to having either an Identity Manager form or an application (system) specific interface for gathering more information for executing the actual provisioning work (corresponding to tiers two and three respectively). The actual process of provisioning the user in the selected system may use (re-use) information gathered in the common tier one form and it may also imply changing or adding information in the LDAP system. This is totally dependent on the system being provisioned.

Having completed the provisioning of one system by this process, the user can choose to provision other systems to which he may have access, or he can choose to end the provisioning session in which case all the changes done will be "committed" to the various systems.

This work-flow can be repeated as many times as wanted for the same user performing corrections on data in already provisioned systems or provisioning new systems for the user.

Role Oriented Provisioning

The Role Oriented Provisioning is based upon the concept that a single user may perform functions in one or more applications. In order for him to do so, it may be necessary to give him permissions in more than one system. The logical grouping of such permissions (provisioning) in various systems is called a *Policy*. An example of one possible policy could be the role of a Thetis Inspector. In this case, the user has to first be provisioned in the LDAP system, then he has to be further provisioned in the Liferay system (to be made a Liferay User) and finally he has to be given the correct permission set in the Thetis Application to be able to act as an Inspector. It might also be possible for this Inspector to be given further permissions at the application level to allow him to only see a certain subset of data, for example according to his country.

Similar to the System Provisioning Work-flow, the first step is to identify the existence of the user (in which case a data edit can be performed), or to create a new user (by submitting new data). This first step is exactly the same as in the previously described work-flow (Role Oriented), having been already explained.

From this point on, the difference between this work-flow and the previous one becomes apparent. In this flow there is the possibility to decide upon the Role Policy to apply to the user (instead of applying permissions to actual systems) followed by a tight loop performing the provisioning on the various systems/applications associated with the selected policy.

After the provisioning for a specific Role has been applied, it is possible to select another Role Policy to apply to the user (if this user being provisioned is to have access to various applications), or terminate the process for that user and proceed to provision another user.

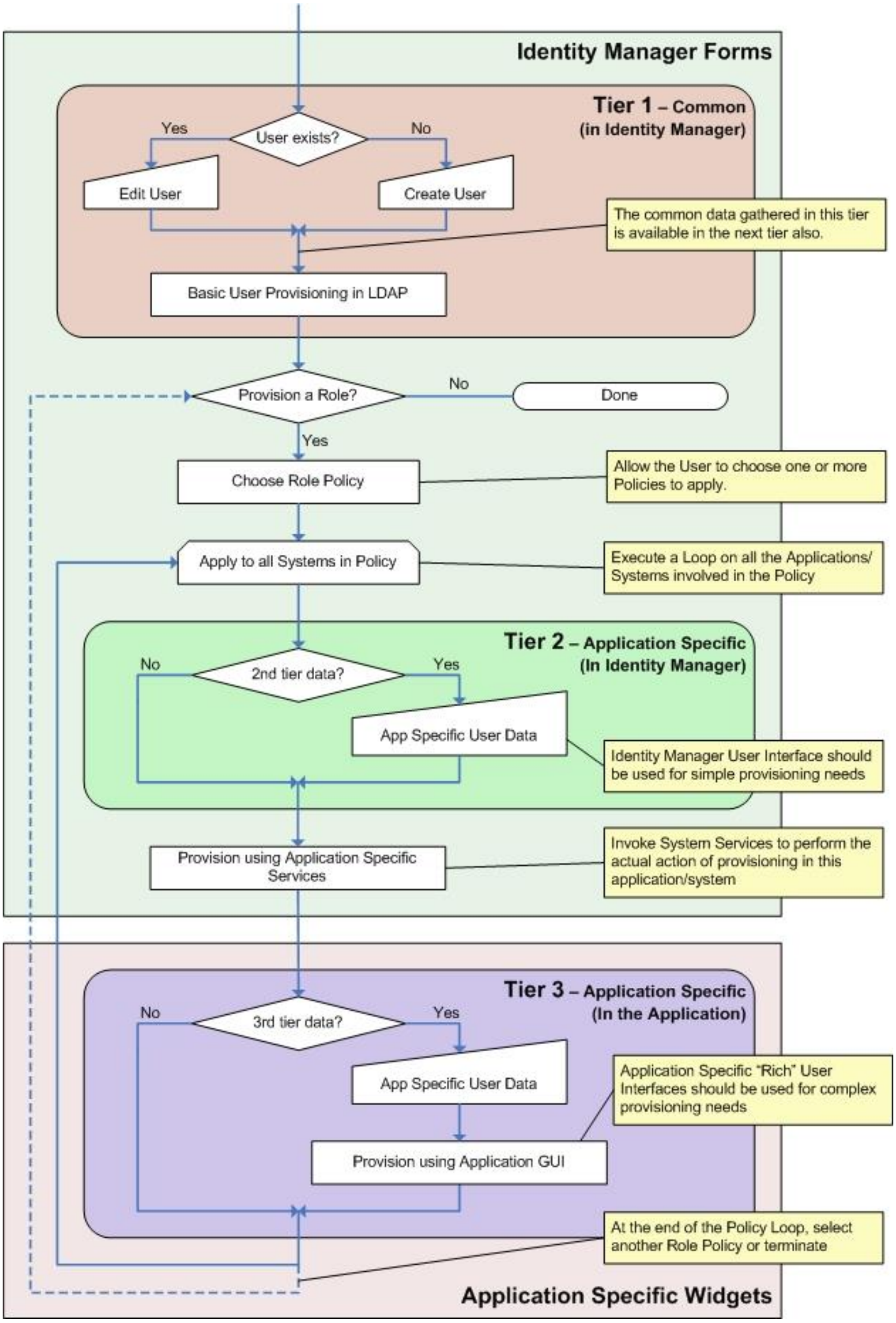

Identity Manager Forms
Tier 1 – Common
(in Identity Manager)
User exists?
Yes
No
Edit User
Create User
The common data gathered in this tier is available in the next tier also.
Basic User Provisioning in LDAP
Provision a Role?
No
Done
Yes
Choose Role Policy
Allow the User to choose one or more Policies to apply.
Apply to all Systems in Policy
Execute a Loop on all the Applications/ Systems involved in the Policy
Tier 2 – Application Specific
(In Identity Manager)
No
2nd tier data?
Yes
App Specific User Data
Identity Manager User Interface should be used for simple provisioning needs
Provision using Application Specific Services
Invoke System Services to perform the actual action of provisioning in this application/system
Tier 3 – Application Specific
(In the Application)
No
3rd tier data?
Yes
App Specific User Data
Application Specific "Rich" User Interfaces should be used for complex provisioning needs
Provision using Application GUI
At the end of the Policy Loop, select another Role Policy or terminate
Application Specific Widgets

Figure 18 – Role Oriented Provisioning Work-flow

5.3. IMPORTANT DEFINITIONS

Now that the provisioning model has been defined, it is time to define how actual EMSA applications can use this model; but before doing so, we will state a few concepts by providing their definitions so that there are no doubts about what is being said and how the existing infrastructure at EMSA supports these concepts.

Basic Definitions

Following is a list of simple definitions that need be fully comprehended.

- **Role-Based Access Control** – Role-Based Access Control (RBAC) is a logical architecture commonly used to implement control of accesses to protected resources (functionality, services, systems, …);
- **User** – An individual member having identification credentials to access a protected resource;
- **Permission** – A privilege that grants access rights to one specific resource;
- **Role** – A set of permissions granting access rights across their resource scope (functionality, service, system, …);
- **Functionality** – A specific resource to be protected;
- **User Group** – A collection of users;
- **Community** – Collections of Users or User Groups who have a common interest.

Relationships

Having defined the basic concepts, following is a list of how these concepts are related to each other.

- A **User** may belong to one or more **User Group**s;
- One **Functionality** has only one **Permission**;
- A **Role** is composed by **Permissions** and one **Permission** may belong to more than one **Role**;
- A **Role** can be assigned to one or more **User Groups**;
- A **Community** is composed by **Users** and/or **User Groups**.

As a side note, we will also state that the following relationships are also possible to implement in the RBAC model, but these complicate a lot the management of the model. Best practices and experience show that these relationships must be considered **only in very exceptional circumstances**; they are possible relations but not desirable ones. It is highly recommended that no EMSA application make use of these concepts.

- A **Permission** may be assigned to a **User** and a **User** can have more than one **Permission**;
- A **Role** can be assigned to a **User** and a **User** can have more than one **Role**.

Besides not applying the extended relations, further restrictions will be imposed (suggested but not enforced) limiting the cardinality of Roles and User Groups. These further restrictions will be discussed in due time.

5.4. RBAC IMPLEMENTATION IN THE EMSA INFRASTRUCTURE

In the scope of the Single Sign On / Identity Management project, we can state that all the applications to be considered are Web Applications. Furthermore, we can also state that all these web applications are to be run under a common “umbrella” which is a Portal environment which will run on Weblogic JEE (Java Enterprise Edition) Application Servers. The Portal environment used at EMSA is based upon the Liferay Enterprise Portal implementation. An LDAP Server supports both the Portal as well as the web applications.

We will now describe how each piece of infrastructure implements/uses the previously mentioned RBAC concepts (basic definitions and relations).

5.4.1. LDAP

An LDAP server allows for the creation of a tree structure of Distinguished Names; DN's in LDAP terminology. It does not directly implement the notion of User Groups or Roles (or even Users for that matter). However, through the use of the DN syntax, one can just about map anything inside the LDAP tree structure. Roles and User Groups can be obtained by associating specific attributes to a DN (whose direct meanings can be interpreted as a Role or User Group) or they can be obtained by answering questions like "in what groups X is a member of" for Roles or "who are the members of that group" for User Groups.

The semantics of use of LDAP at EMSA are[6]:

- The "top level" of the structure is defined as "dn: dc=emsa,dc=europa,dc=eu";
- The **groups** concept is defined as "dn: ou=groups,dc=emsa,dc=europa,dc=eu";
- The **users** concept is defined as "dn: ou=users,dc=emsa,dc=europa,dc=eu";

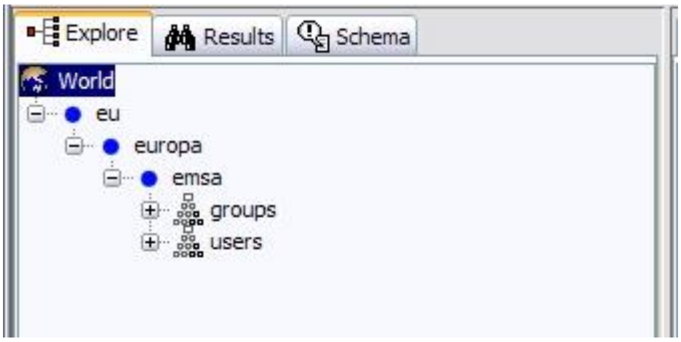


Figure 19 - LDAP top level tree view

- Under the **groups'** branch, further organizational units (ou) should be created to represent a specific application (or parts of and extensions to applications). Examples of such units are:
  - "dn: ou=CSN,ou=groups,dc=emsa,dc=europa,dc=eu"
  - "dn: ou=THETIS,ou=groups,dc=emsa,dc=europa,dc=eu";
- Inside (or underneath) a specific application group should come the actual names of meaningful groups. Once again, examples of such names are:
  - "dn: cn=CSNSysAdmins,ou=CSN,ou=groups,dc=emsa,dc=europa,dc=eu"
  - "dn: cn=STCWReader,ou=STCW,ou=groups,dc=emsa,dc=europa,dc=eu"

---

[6] In order to simplify the description of the semantics, only the **dn:** field will be presented. Any other existing attributes will only be presented if they bring added value to the description.

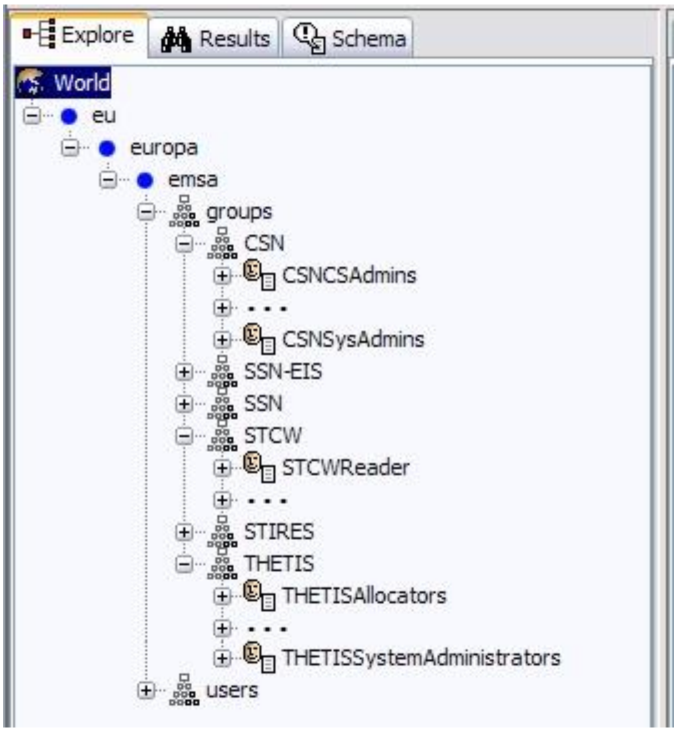


Figure 20 - LDAP groups

- Under the **users** branch, two organizational units are possible; one to be used for actual physical users – people, and another to be used for either system administrators or other external systems (the interpretation of which depends on the application) – systems. The resulting distinguished names are:
  - “dn: ou=people,ou=users,dc=emsa,dc=europa,dc=eu”
  - “dn: ou=system,ou=users,dc=emsa,dc=europa,dc=eu”
- Inside the **people** branch are all the physical application users, such as:
  - “dn: uid=user1,ou=people,ou=users,dc=emsa,dc=europa,dc=eu”
  - “dn: uid=user88,ou=people,ou=users,dc=emsa,dc=europa,dc=eu”
- Inside the **system** branch are the system administrators or external systems (depending on the interpretation performed by the applications)
  - “dn: uid=IMO,ou=system,ou=users,dc=emsa,dc=europa,dc=eu”
  - “dn: uid=ssouser,ou=system,ou=users,dc=emsa,dc=europa,dc=eu”

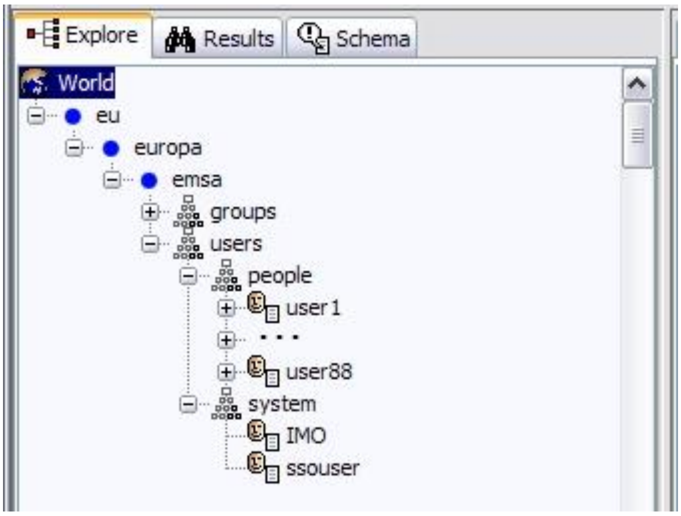


Figure 21 - LDAP users

- Due to the fact that the concept of a role is not directly implemented in the EMSA semantics (for example through the use of an organization unit such as ou=role),

such a concept should be achieved by associating users to groups through the **member** attribute. By using the first question previously described ("in what groups X is a member of"), one can conclude that in this way it is possible to infer **roles** from this structure (assuming that the name of the role is the same as the name of the group for ease of use). The only "restriction" applied here is that the name of the role be the same as the name of the LDAP group supporting the role.
- Applications that require only global authentication should create a group named **members** under the applications own group name, such as in the example "dn: cn=members,ou=CSN,ou=groups,dc=emsa,dc=europa,dc=eu" and then associate the actual users with this group (for example, by means of "member: uid=user1,ou=people,ou=users,dc=emsa,dc=europa,dc=eu").
- Applications that need to implement role authorizations should associate the users with the name of the group that represents the desired role, as shown in "dn: cn=STCWReader,ou=STCW,ou=groups,dc=emsa,dc=europa,dc=eu" with "member: uid=user88,ou=people,ou=users,dc=emsa,dc=europa,dc=eu".

5.4.2. Weblogic JEE Application Server

After having seen how the basic concepts mapped into the LDAP server, we continue by mentioning that, at the present moment, a **Global Role** needs to be created in the Weblogic JEE Application server. This global role is named **liferay_users** and contains a "logical or" of all the roles to be accessed by the Liferay Portal. Please note that as was defined in the LDAP text, roles here mean the same as the LDAP groups supporting them (with the same name). An example of such a global role might be "THETISAllocators or THETISInspectors or THETISNationalAdministrators or THETISProcessors or THETISReaders or THETISSupervisors or THETISSystemAdministrators or THETISStatisticAnalysers or THETISNationalStatisticAnalysers or LIFERAYAdmins or STCWReaders or STCWInspectors or STCWAssistants or STCWSupervisors".

One objective of the SSO / IdM project is, if possible, to eliminate the need for such a global role.

5.4.3. Liferay Enterprise Portal

The third step in our tour of the underlying infrastructure is a look at how the basic concepts map into the Liferay Portal. The Liferay portal implements the following concepts: Communities, User Groups, Roles and Users. It also has other concepts, like Organizations, which we will not discuss as they do not map into our RBAC model. Likewise, the portal implements the concept of a portlet which we will consider as a resource in our RBAC model (or Functionality if you like). We will now have a look at each individual concept and discuss it in more detail.

- Users – In Liferay, a User represents a person and has a set of attributes, such as First, Middle, Last Name, Screen Name, E-mail, etc. While it is possible to directly associate Permissions to Users, it is highly recommended not to do so as there are other ways to allow access to resources. There is a "one-to-one" relation between the users in Liferay and the users created in LDAP (even though it is possible for users to exist on only one of either side of the relation).
- User Groups – As the name suggests, this is an aggregator for joining Users. It allows a means for performing some operations on a variable number of users without having to do the same actions on each user individually. Whilst it is possible to assign Permissions to User Groups, as it was for users, this should also not be done. Like the relation between Liferay Users and LDAP users, there is also a "one-to-one" relation between Liferay User Groups and LDAP groups. Please note that in this case the names need not necessarily be exactly the same (case wise) because Liferay is case insensitive.

- Roles – A role is a way through which Liferay will grant user access to certain resources. A role is logically connected to a User Group (by associating the User Group to the Role) and should maintain a similar name to facilitate human reading/interpretation. This means that any User belonging to the User Group associated with the Role will have access to the resource protected by the Role. In this particular case, there is no direct connection between a Liferay Role and LDAP even though a logical association may be made through the similarity in the names.
- Communities – In Liferay, a Community is created to allow various Pages (we have called them resources in previous bullets and they are the Functionalities in the RBAC model) to be joined together thus providing a single point of configuration for a specific interest. One such example could be the Private Pages (Portlets) for the CSN application. Another such example could be the Public Pages for the STCW application. Whenever access restrictions need to be applied (such as in the private pages of a community), Roles can be associated to a Functionality (Page) in a Community.

We have defined some basic concepts on the RBAC model. We have also explained how this model fits into the EMSA infrastructure. The next section will be about defining the requirements for provisioning users in the EMSA infrastructure for the THETIS, STCW and CSN applications.

5.5. PROVISIONING OF EMSA APPLICATIONS

This sub-chapter provides generic information on the provisioning of the EMSA applications.

5.5.1. Provisioning of jPetStore Application

JPetStore has an internal representation of users. This consists of various database tables where information on the users is kept. In the jPetStore application deployed in the EMSA test environment, the provisioning process has not been touched, which goes to say “users” have not been provisioned through OIM (Oracle Identity Management). A full integration with the complete suite of Oracle products deployed for this solution would require deeper changes to the application in order to support the provisioning through OIM, namely:

- Removal of the “Register Now” link from the Login page;
- Change in the user Account form to only allow viewing of the attribute values (no edit allowed);
- Creation of new services to allow “remote” calls to the user provisioning methods (preferably through web services).

An implication of the provisioning process not being integrated is that it is now necessary to put the password exactly the same as the username.

5.5.2. Provisioning of RuleCheck Application

RuleCheck “users” do not have a specific provisioning process so there was no work done in OIM (Oracle Identity Management) for this application, at least at the present moment. In the future there may be a provisioning process for RuleCheck users but this will most certainly only be the creation of users/association to the correct groups in LDAP.

5.5.3. Provisioning of STCW Application

STCW “users” do not have a specific provisioning process so there was no work done in Oracle Identity Management.

5.5.4. Provisioning of THETIS Application

The generic process for provisioning applications has been discussed. We will now proceed to discuss the intended process for specific applications starting with THETIS.

Provisioning a new user

For a user (with the correct credentials) to be able to create a new user, he will have to start by accessing a link to OIM available in the Thetis theme. This link will contain a parameter that indicates that it is the Thetis application that is performing the request (… `?app=THETIS&` …), and will open an OIM form where the user can search for the existence of the new user to be created. Even though this step seems to be redundant, after all we do want to create a new user, it is necessary because the user may be "new" in Thetis but already registered (already exists) in another context (for example CSN or STCW). If the user is not found in this initial search, then it effectively is a new user and must be provisioned in LDAP, Liferay, etc. followed by the actual Thetis provisioning. On the other hand, if the user is found in the initial search, then all that is needed is the Thetis provisioning.

Provisioning for an existing user

For a user (once again with the correct credentials) to be able to update an existing user, he will have to start by accessing a link to OIM available in the Thetis theme. This is similar to the creation process but differs in that the link provided has not only a parameter indicating the Thetis application, but also a parameter indicating the ID of the user to be provisioned (… `?app=THETIS&userId=`*`User`* …). This will lead to an OIM form that contains the result of the search for the indicated user ID. It is thus possible to confirm the existence of the user, and proceed to the actual Thetis provisioning, or to discover that the indicated user does not exist after all and take the appropriate steps to correct the situation.
An alternative option is to access the link without the use of the ID parameter. This case results in something similar to the initial creation process in which a search form is presented where the user can apply a search criterion. After having found the desired user, further provisioning can be done on this user as if his ID had been passed initially.

Provisioning Thetis

Independently of being a new or existing user, after having passed the previously mentioned search form, the provisioning user will be allowed to change/fill in (update and create respectively) the new users' common data in a first form (by common data we mean data that is common to all EMSA applications like userId, name, etc.). Once the common data has been filled in, the next step is to define more information about the user (for example, roles).
Due to the fact that the provisioning user arrived at the OIM screens through a link in Thetis, the application is already known at this time. If it were not known, then a screen showing the possible applications to provision would be presented (the actual applications available would depend on the permissions held by the provisioning user) and one would be chosen.
Since we know that we are provisioning Thetis, a "second" form will be presented where the user can register specific information only present in Thetis along with the association of application "roles" to the user.

Provisioning Services

Typically provisioning of users is not as trivial as filling in the first level form in OIM and choosing an application to associate to the user. Normally there will be a need for further work to be done inside the actual application to which the user is being granted access. The way to promote the interaction between OIM and the applications will be through Web Services. Each and every application will have to implement at least one Web Service which will allow OIM to execute CRUD services in the application. The exact format for this service is still to be defined at the present moment, but should look something like:

- A first section which will contain all of the mandatory parameters for the creation/edition of a "generic" user;
- A field indicating what type of CRUD operation is to be performed by the application (still belonging to the first section);
- A second section containing a Hashmap of "key – value" pairs that is meaningful to the application (that contains the application specific data for the user). Please note that a possible Hashmap entry can be another Hashmap whenever complex structures are needed (such as for defining roles).

Whilst the first section is mandatory and will be defined by EMSA/OIM, the second section is optional and it will be the responsibility of the contractors of each application to provide relevant information as to the keys needed for their application. A complete specification of the previously mentioned Web Service can be found in the annexes.

Provisioning Permissions

It has been mentioned various times previously that the user performing the provisioning needs to have the privileges necessary to do so. The permission set for provisioning Thetis is as follows:

- ThetisSystemAdministrator: Can create/update/delete any user inside Thetis;
- ThetisNationalAdministrator: Can create/update/delete users who belong to the same Country as this user;

The complete set of permissions is defined in the annexes.

User self-service is to be done by the actual user and allows for changing of personal information (as long as this information is changeable, i.e. userId is not changeable).

It should be possible to revoke (delete) a user in one of two ways:

- Globally. The users' access is removed from all applications;
- Per application. The user loses access to the particular specified application.

Please note that a revocation is <u>permanent</u> and <u>cannot be undone</u>.

If there is a need to temporarily disable access for a user to a given application (or all at once), this can be done in OIM by issuing a Disable/Enable command. Either of the two operations can be undone by performing the complementary operation.

5.6. PROVISIONING OF EMSA APPLICATIONS BY TIERS

After having presented all of the definitions and auxiliary information, we will now present the requirements for provisioning users in EMSA applications according to the information previously presented.

5.6.1. Tier 1

As has been previously mentioned, tier1 corresponds to the common information about users that is to be gathered in a form that is to be created for this purpose in Oracle Identity Manager (OIM). The following table provides information about the logical names of fields and indication if they are mandatory in certain applications.

Table 1 - Tier1 Mandatory fields

	Mandatory

	THETIS	STCW	CSN
userId	X	X	X
firstName	X	X	X
lastName	X	X	X
initials	X	X	
expiryDate[7]			
memberState[8]	X	X	X[9]
phone			X
address			X
password	X	X	X
email	X	X	X
status	X	X	X

The next table provides information about the logical names of fields (for relating to the other tables) and mapping of those fields in application database tables (where applicable), LDAP and Liferay. Please note that the STCW application has no need for a database table to hold any of this information and as such is not included in the following table.

Table 2 - Tier1 Field Mappings

	Tables		LDAP Mapping[10]	Liferay Mapping
	THETIS “USER_”	CSN “SIB_USERS”		
userId	USER_ID	ID_USER	uid	screenName
firstName	FIRST_NAME	NAME	givenname	firstName
lastName	LAST_NAME	LAST_NAME	sn	lastName
initials	INITIALS		initials	customAttrib(TBD)
expiryDate	EXPIRY_DATE		expiryDate	[11]
memberState	COUNTRY_ID	?	memberstate	country
phone	PHONE	TELEPHONE_NUMBER	telephone number	customAttrib(TBD)
address	ADDRESS	?	postalAddress	street1

---

[7] To be checked by SSO. If expired, login is denied. Pending questions: How is it updated? Who is responsible for updating?

[8] “EMSA” should be included as a member state.

[9] There is a need for discussion with the CSN Project team to see if MemberState = “Organization (e.g. coastal state)”. If yes, then the field is mandatory. If not, then it is Application specific.

[10] The Distinguished Name for new users should follow the pattern “dn: uid=XXX,ou=people,ou=users,dc=emsa,dc=europa,dc=eu” where XXX is the data from the **userId** field.

[11] There is a need to check if there is a field for the same type of information.

password	PASSWORD	?	userPassword	password
email	EMAIL	?	mail	emailAddress
status			status	active
			cn=givenname + sn	customAttrib(TBD)

As was previously stated elsewhere in this document, EMSA has chosen to simplify the RBAC model by assuming that a Role will have the same name as a User Group. What this means is that an application using Role XX will actually be looking at the group XX in LDAP (indirectly through JAAS). The next table describes the various applications Roles which can be found in LDAP under the distinguished name "dn: cn=*RoleName*,ou=*App*,ou=groups,dc=emsa,dc=europa,dc=eu" where *RoleName* is the name of the Role and *App* is the name of the application respectively.

Table 3 - Role Names

THETIS	STCW	CSN
THETISAllocators	STCWAdministrators	CSN DC System administrator
THETISExternalSystems	STCWInspectors	CSN DC Service Desk
THETISInspectors	STCWReaders	CSN DC Maritime Services Support Operator
THETISNationalAdministrators	STCWSupervisors	CSN DC Financial Officer
THETISNAtionalStatisticAnalysers	STCWAssistants	CSN DC Authorising Officer
THETISProcessors		Coastal State Administrative Representative
THETISReaders		Coastal State Operational Representative
THETISStatisticsAnalysers		Coastal State User Group Representative
THETISSupervisors		Coastal State System User
THETISSystemAdministrators		Coastal State Planning Representative
		Service Provider System
		Satellite Operator System User
		Service Provider Administrative Representative
		Satellite Operator Administrative Representative
		Guest User

Once defined the Users and Roles, there is a need for relating these roles to the actual users. This relation will be presented in three parts: indications on how the relationships are stored in the LDAP, a first table defining the Rules for giving Roles to Users and a second table containing attributes and mappings that define the relationships.

LDAP

In LDAP, a relation between a user and a role is actually stored as a relation between a user and a group because of the direct correspondence of Role to an LDAP group. The solution is to simply declare any given user as a "Member Of" a group and this will be interpreted as the user having the Role (with the same name as the group). An example of such an entry in LDAP might be "dn: cn=STCWReader, ou=STCW, ou=groups, dc=emsa, dc=europa, dc=eu" for the name of the Role and "member: uid=user1, ou=people, ou=users, dc=emsa, dc=europa, dc=eu" for the name of the user to whom the role is granted.

Rules and Attribute Mappings

Please note that there is only detailed information about how THETIS performs these relationships in its database. STCW has no need for database tables to contain the relationships because the LDAP structure contains all the necessary information. CSN does in fact internally use some database tables ("SIB_ROLES", "SIB_PRIVILEGES" and "SIB_X_GROUPS_PRIVILEGES") but the actual relationships between these tables has not been provided.

Table 4 – Application Rules for Roles

THETIS Rules

Only system users can be THETISExternalSystems
System users can be or not THETISExternalSystems
Every user is a THETISReader
Only users with member state can have Office Roles (THETISAllocators, THETISInspectors, THETISProcessors, THETISSupervisors, THETISNationalAdministrators)
Only users with member state can be THETISNationalStatisticAnalysers
Only users without member state can be THETISStatisticsAnalysers

STCW Rules

Not defined

CSN Rules

Not defined

Table 5 – Database Table Mappings for Roles

THETIS Attribute Mappings

<table>
<tr><td colspan="2">Roles:
THETISAllocators, THETISInspectors, THETISProcessors, THETISSupervisors</td></tr>
<tr><td>Field name</td><td>Database table "OFFICEROLE"</td></tr>
<tr><td>Role Type</td><td>ROLE_TYPE</td></tr>
<tr><td>Office</td><td>OFFICE_ID</td></tr>
<tr><td>PSC</td><td>PSC</td></tr>
<tr><td>Ropax</td><td>ROPAX</td></tr>
<tr><td>User</td><td>USER__ID</td></tr>
<tr><td colspan="2">Roles:
THETISNationalAdministrators, THETISNationalStatisticAnalysers, THETISStatisticsAnalysers, THETISSystemAdministrators</td></tr>
<tr><td>Role name</td><td>Database table "USER_" (column name in USER_ table)</td></tr>
<tr><td>THETISNationalAdministrators</td><td>NATIONAL_ADMINISTRATOR</td></tr>
<tr><td>THETISNationalStatisticAnalysers</td><td>NATIONAL_STATISTICAL_ANALYSER</td></tr>
<tr><td>THETISStatisticsAnalysers</td><td>STATISTIC_ANALYSER</td></tr>
<tr><td>THETISSystemAdministrators</td><td>SYSTEM_ADMINISTRATOR</td></tr>
</table>

STCW Attribute Mappings

Not necessary due to the fact that there are no database tables involved in the user provisioning process.

CSN Attribute Mappings

Not defined

The final step necessary to provision at the Tier 1 level is the definition of application services that can be used to execute the actual user provisioning. The following table will define existing services for such provisioning.

Table 6 – Provisioning Services

THETIS

Normal Users	Users are created in the database with the roles associations, and then the user is created in the LDAP, and last the LDAP groups relationships are created
System Users	Currently the only way to create a system user is running the databuilder. The user is created in the Thetis database and after that it's created in LDAP. If it's an external system, a new attribute "member" is created in the THETISExternalSystem LDAP group.
Service	UserRemoteBean
Methods	addUser
	editUser
	delete
	changeStatus
	addSystemUser
	editSystemUser

STCW

Not necessary

CSN

The user provisioning for a CSN-DC user is done using the OIM Web Console. The following entities (*IT resources* in the OIM terminology) are involved in the provisioning process:

- LDAP
- CSN-DC user database

The provisioning is carried out by operating the *OIM web console* (whose behaviour has been previously designed using the *OIM design console*). When provisioning a new CSN-DC user, the provisioning operator (e.g. the CSN-DC sys admin) will launch the OIM web console and fill in the appropriate form. Upon accepting the data entered into the console, the provisioning process will be triggered which consists in:

- Writing the user info into the LDAP
- Writing the user info into the CSN-DC user database, using the JAVA event handler define by ACS which implements a web service hosted by the CSN-DC

The CSN-DC user database has all the details necessary for the users, in order to define the user authorization profile to be used during the CSN-DC application usage. For example this database knows if a given user is able to access a certain function on the WUP, POR, Alerting, etc.

The CSN-DC user database I formed by 4 tables:

- SIB_USERS: defines the users details
- SIB_ROLES: define the roles used into the CSN-DC

- SIB_PRIVILEGES: lists all the possible privileges
- SIB_X_GROUPS_PRIVILEGES: implements the association between roles and privileges

Please note that in the CSN-DC the concepts of role and groups are exactly the same. In fact there are a number of roles which corresponds exactly to the groups, e.g. CSN-DC System Administrator, CSN DC Service Desk, etc. Therefore the concept of role is used here which indicates clearly that the role is mapped into an authorization profile, i.e. what the user can do and cannot do. The SIB prefix only indicates that the info is used with the ACS SIBILLA integration layer.

5.6.2. Tier 2

As of the present moment it is not clear as to the existence of Tier 2 forms in OIM. Further study has to be conducted in order to find out if, and how, tier 2 forms are to be created in OIM.

Table 7 – Tier 2 definitions

THETIS

Not known

STCW

Not necessary

CSN

Not known

5.6.3. Tier 3

Even though intuitively Tier 3 processing will be necessary, at the present moment there is not enough information to describe exactly what applications will use this approach or how this will be used.

Table 8 – Tier 3 definitions

THETIS

Not known

STCW

Not necessary

CSN

ACS is still finalizing this. The mapping between users and roles is clearly defined by the CSN-DC tables described above. Please note that being the CSN-DC a generally stateless web application, for performance and availability reasons it has been decided to write into the CSN-DC database also some information that is in the LDAP, e.g. e-mail, telephone number. The correct alignment between the 2 databases is granted by the provisioning process described above (the JAVA web service which communicates between the OIM and the CSN-DC database).

Field	**Semantics**	**Note**
ID_user	Unique ID of the user	Primary key
Role_ID	Association to roles	Association to roles. A user can be associated to multiple roles. A role can include multiple users.
Name	First name	
Last name	Last name	
e-mail	e-mail	

Telephone number	Telephone number	This shall be used for alerting.

SIB_USERS table description

Field	**Semantics**	**Note**
ID_role	Unique ID of the role	Primary key
Role_name	Short name for the role	e.g. csndc_admin
Role_description	Descriptive name for the role	e.g. CSN DC System Administrator
Role_enabled		The role can be enabled or disabled, in case is no longer used

SIB_ROLES table description

Field	**Semantics**	**Note**
ID_privilege	Unique ID of the privilege	Primary key
Privilege	Privilege name	e.g. Alert configuration admin
Category	Functional group	e.g. Alerting, POR, etc.
Priv_alias	Alias for the privilege	Used internally in the SW, e.g. *oas_cs_rules_show*

SIB_PRIVILEGES table description

Field	**Semantics**	**Note**
Role_ID	ID of the role	
Privilege_ID	ID of the privilege	
Privilege_Type	Enumerable value	Possible values (**TBC**): • 0 de-activated • 1 activated • 2 read-only This list may be expanded if necessary. The system can work in positive logic or negative logic. In positive logic, means that all permissions are granted by default and only those that are set to 0 are negated. In negative logic, is the other way round, no permission is granted to the users, with the exception of those for which a value of 1 is granted. The choice of the logic to apply will depend on the application. If most permission are granted to the users the negative logic is preferred, which is simpler to adopt. In any case this is transparent to the OIM configuration, but is just part of the pre-configuration of the SIBILLA framework to be used by the CSN-DC.

SIB_X_GROUPS_PRIVILEGES

The pre-configuration of the tables, with the exception of the SIB_USERS are not managed by the OIM. In fact they define what a certain user in the CSN-DC can or cannot do and as such are managed and configured using the CSN-DC application.

The OIM will only insert/update entries into the SIB_USERS tables, when a user is provisioned/de-provisioned. As already indicated above, this link with the SIB_USERS table is completely managed by the CSN-DC Event Handler.

6. EMSA OIM Custom Interface

EMSA needs to access OIM directly from links placed in some third party applications. The implementation of an OIM custom interface has been created in order to provide a full User Management functionality in OIM - Create and Edit User fully adapted to the customer needs.

Create User

The procedure followed by EMSA to create a new OIM user and provision the necessary resources involves, apart from filling up the OIM user information, selecting a group that will trigger the corresponding Access Policy that automates the resource provisioning to the new user. All this is done in one single step.

In order to avoid creating more than one account in OIM for the same user, a search process is launched before creating the user, using the **First Name** and **Last Name** as search criteria. If some results are returned, they are shown in a table (User ID, First Name, Last Name and Email) so the user can decide whether the user already exists or creates a new one. If no results are returned by the search process, the user is created automatically and the resources provisioned.

The groups that a user is able to manage must comply with the security model defined in OIM defined for EMSA.

Edit User

The third party applications are responsible to display the list of users that can be edit by the logged user. By clicking on the link of the user to be edited, the View Details window will be shown, including, as well as the OIM User information the groups assigned to it.

The user information and the group assignation can be edited at one single step. When modifying the group the resources associated to the old group will be revoked and the new resources will be provisioned.

Apart from editing, a user is also able to enable/disable and delete OIM users.

6.1. APPLICATION GENERIC APPROACH

For every application that accesses the Custom Interface there may be different fields for the User Forms. This means that, apart from the system fields common to all the applications, there may be some OIM User Defined Fields that may appear in the user forms for one application but not in the others. These will be referred to as application-specific fields.

Note: <u>One field can only belong to one application.</u>

In order to achieve this behaviour we have made the following customizations:

The applicationFields.properties file

Every time the need arises for adding a new field to the Create OIM User Form in an application, the new field must be included in this file.

This .properties file contains two entries for each application-specific OIM User Defined Fields. One entry is used to specify the application name and the other one to specify if the field is optional or not.

The syntax for each entry is "application" or "optional" followed by the OIM User Defined Fields column name. The value for the application name is the Resource Object Name and it must match the application passed as a parameter in the Custom Interface URL. The value for optional must be "true" or "false".

You can find this file under the same directory as the other Message Resource Bundles: WEB-INF/classes.

$OIM_HOME/xellerate/OIMApplications/WLXellerateFull.ear/xlWebApp.war/WEB-INF/classes

Below there is an example of how this file should look like:

```
#--------------------------------------------------------
#-------------------------- Thetis ----------------------

application.USR_UDF_THTS_INITIALS=Thetis User
optional.USR_UDF_THTS_INITIALS=false


application.USR_UDF_THTS_PHONE=Thetis User
optional.USR_UDF_THTS_PHONE=true


application.USR_UDF_THTS_ADDRESS=Thetis User
optional.USR_UDF_THTS_ADDRESS=true
```

- OIM User Defined Fields column name:
    - `USR_UDF_THTS_INITIALS`
    - `USR_UDF_THTS_PHONE`
    - `USR_UDF_THTS_ADDRESS`
- Resource Object Name:
    - `Thetis User`
- Optional Attributes:
    - `USR_UDF_THTS_PHONE`
    - `USR_UDF_THTS_ADDRESS`
- Required Attributes:
    - `USR_UDF_THTS_INITIALS`

6.2. ACCESSING THE CUSTOM INTERFACE

The Custom Interface will be accessed through links placed in the third party applications.

Create User

The URL to access the Create User Form window will look as follows:

http://twgt1.emsa.local/xlWebApp/createUserCustom.do?method=**New+User**&application=***Thetis+User***&userAction=new

In the previous link, *Thetis+User* has to be changed to the correct resource name for the application in which the link is published.

Edit user

The URL to access the Edit User Form window will look as follows:

http://twgt1.emsa.local/xlWebApp/createUserCustom.do?method=viewUserDetails&loginID=***ORACLEUSER1***&application=***Thetis+User***&userAction=new

In the previous link, *ORACLEUSER1* has to be changed to the correct user identification per call to the edit method. See also the comment made on the "Create User" link about the application name (*Thetis+User*).

Edit my account (only fields common to all applications)

The URL to access the Edit my account User Form window will look as follows:

http://twgt1.emsa.local/xlWebApp/createUserCustom.do?method=viewUserDetails&**loginID**=&**application**=&userAction=new

Please note the *loginID* and *application* variables must exist in the link and be empty for this to work.

6.1. STEPS FOR CREATING A NEW CUSTOM ATTRIBUTE

Before we can add any attribute to the custom form, we must create them on OIM.

1. Go to User Defined Fields Definition and search for the Form Name <u>Users</u>
2. Add the new field.
   <u>Note:</u> Do not change its visibility to false or make it required here.
3. Save the Form
4. Edit applicationFields.properties file as described below on "Steps for Deployment"

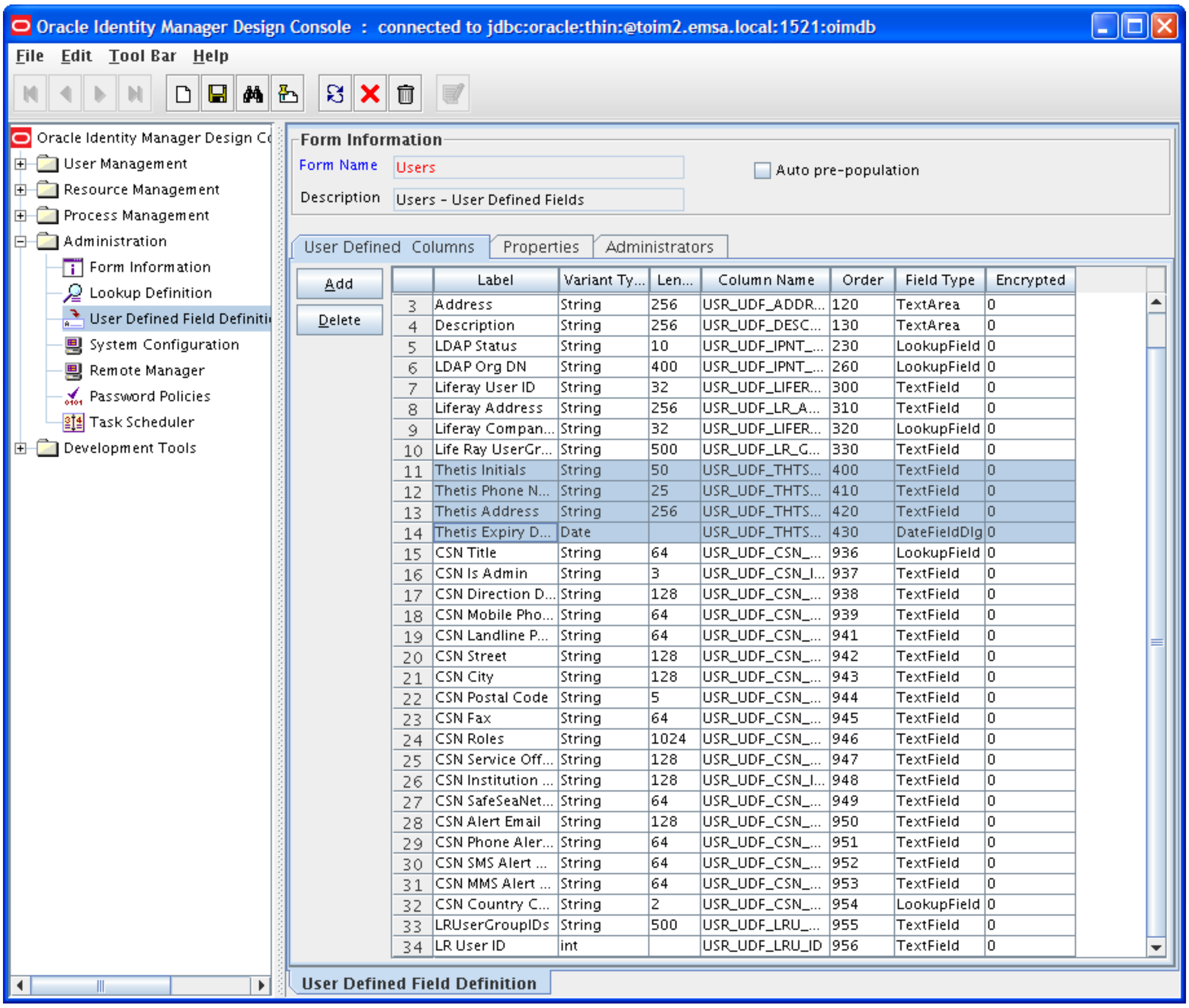
6.2. STEPS FOR DEPLOYMENT

In order to deploy the OIM Custom Interface on OIM, we must follow the steps described below:

1. Stop Weblogic Managed Server
2. Backup the current deployment file WLXellerateFull.ear in the directory $OIM_HOME/xellerate/OIMApplications and $OIM_HOME/xellerate/webapp/xlWebApp.war
3. Extract to a temporary place $OIM_HOME/xellerate/webapp/xlWebApp.war (for example: $OIM_HOME/xellerate/tmp)
4. Modify the $OIM_HOME/xellerate/tmp/xlWebApp/WEB-INF/classes/applicationFields.properties file as needed
5. Make xlWebApp.war:
   a. cd $OIM_HOME/xellerate/tmp/xlWebApp
   b. jar cvf ../xlWebApp.war *
6. Copy the newly created war to $OIM_HOME/ xellerate/webapp
7. Run patch_weblogic to re-deploy OIM with the new modifications:
   a. $OIM_HOME/xellerate/setup/patch_weblogic.sh tabcd1234 tabcd1234
   b. And check the logs on $OIM_HOME/xellerate/logs/patch_weblogic.log
8. Restart the Weblogic server:
   a. /etc/init.d/oracle-weblogic restart
9. **Note:** Sometimes we need to restart weblogic twice.

6.3. SCREEN SHOTS

OIM User Defined Fields



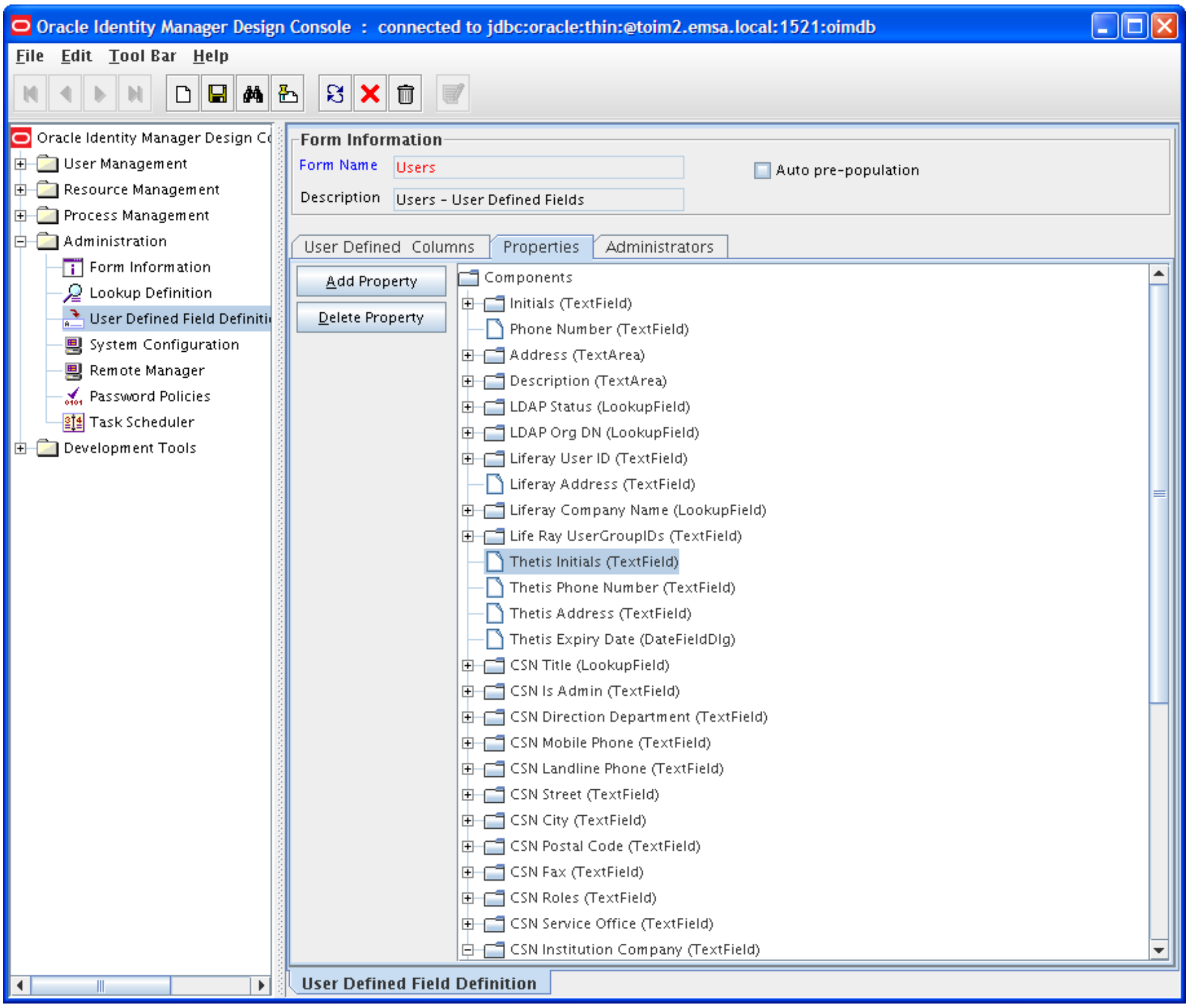
Form Name: Users

Description: Users - User Defined Fields

☐ Auto pre-population

	Label	Variant Ty...	Len...	Column Name	Order	Field Type	Encrypted
3	Address	String	256	USR_UDF_ADDR...	120	TextArea	0
4	Description	String	256	USR_UDF_DESC...	130	TextArea	0
5	LDAP Status	String	10	USR_UDF_IPNT_...	230	LookupField	0
6	LDAP Org DN	String	400	USR_UDF_IPNT_...	260	LookupField	0
7	Liferay User ID	String	32	USR_UDF_LIFER...	300	TextField	0
8	Liferay Address	String	256	USR_UDF_LR_A...	310	TextField	0
9	Liferay Compan...	String	32	USR_UDF_LIFER...	320	LookupField	0
10	Life Ray UserGr...	String	500	USR_UDF_LR_G...	330	TextField	0
11	Thetis Initials	String	50	USR_UDF_THTS...	400	TextField	0
12	Thetis Phone N...	String	25	USR_UDF_THTS...	410	TextField	0
13	Thetis Address	String	256	USR_UDF_THTS...	420	TextField	0
14	Thetis Expiry D...	Date		USR_UDF_THTS...	430	DateFieldDlg	0
15	CSN Title	String	64	USR_UDF_CSN_...	936	LookupField	0
16	CSN Is Admin	String	3	USR_UDF_CSN_I...	937	TextField	0
17	CSN Direction D...	String	128	USR_UDF_CSN_...	938	TextField	0
18	CSN Mobile Pho...	String	64	USR_UDF_CSN_...	939	TextField	0
19	CSN Landline P...	String	64	USR_UDF_CSN_...	941	TextField	0
20	CSN Street	String	128	USR_UDF_CSN_...	942	TextField	0
21	CSN City	String	128	USR_UDF_CSN_...	943	TextField	0
22	CSN Postal Code	String	5	USR_UDF_CSN_...	944	TextField	0
23	CSN Fax	String	64	USR_UDF_CSN_...	945	TextField	0
24	CSN Roles	String	1024	USR_UDF_CSN_...	946	TextField	0
25	CSN Service Off...	String	128	USR_UDF_CSN_...	947	TextField	0
26	CSN Institution ...	String	128	USR_UDF_CSN_I...	948	TextField	0
27	CSN SafeSeaNet...	String	64	USR_UDF_CSN_...	949	TextField	0
28	CSN Alert Email	String	128	USR_UDF_CSN_...	950	TextField	0
29	CSN Phone Aler...	String	64	USR_UDF_CSN_...	951	TextField	0
30	CSN SMS Alert ...	String	64	USR_UDF_CSN_...	952	TextField	0
31	CSN MMS Alert ...	String	64	USR_UDF_CSN_...	953	TextField	0
32	CSN Country C...	String	2	USR_UDF_CSN_...	954	LookupField	0
33	LRUserGroupIDs	String	500	USR_UDF_LRU_...	955	TextField	0
34	LR User ID	int		USR_UDF_LRU_ID	956	TextField	0

Figure 22 - User Defined Fields Definitions



Oracle Identity Manager Design Console : connected to jdbc:oracle:thin:@toim2.emsa.local:1521:oimdb

Form Information

Form Name: Users

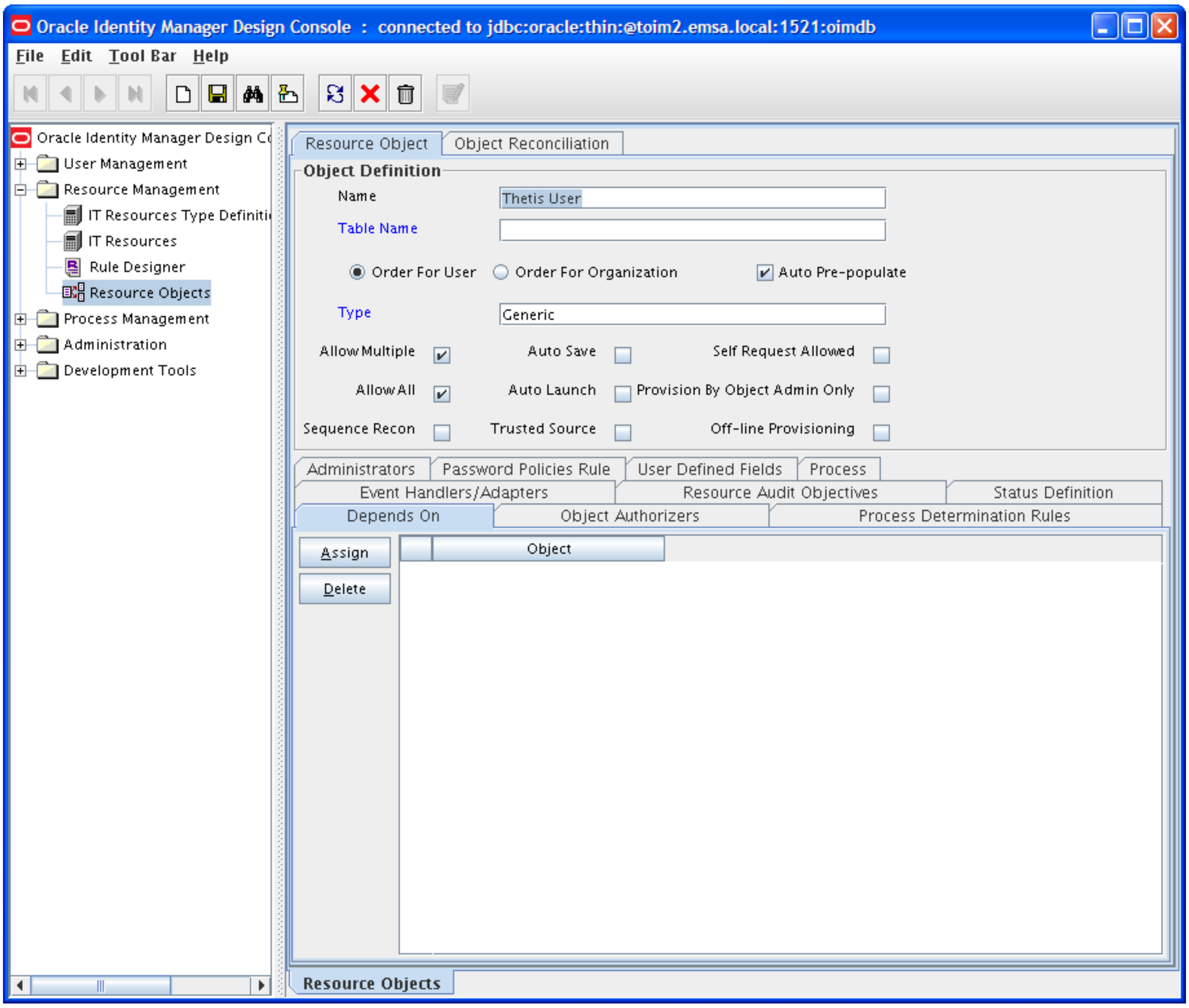
Description: Users - User Defined Fields

☐ Auto pre-population

File
Edit
Tool Bar
Help
Oracle Identity Manager Design C
User Management
Resource Management
Process Management
Administration
Form Information
Lookup Definition
User Defined Field Definiti
System Configuration
Remote Manager
Password Policies
Task Scheduler
Development Tools
User Defined Columns
Properties
Administrators
Add Property
Delete Property
Components
Initials (TextField)
Phone Number (TextField)
Address (TextArea)
Description (TextArea)
LDAP Status (LookupField)
LDAP Org DN (LookupField)
Liferay User ID (TextField)
Liferay Address (TextField)
Liferay Company Name (LookupField)
Life Ray UserGroupIDs (TextField)
Thetis Initials (TextField)
Thetis Phone Number (TextField)
Thetis Address (TextField)
Thetis Expiry Date (DateFieldDlg)
CSN Title (LookupField)
CSN Is Admin (TextField)
CSN Direction Department (TextField)
CSN Mobile Phone (TextField)
CSN Landline Phone (TextField)
CSN Street (TextField)
CSN City (TextField)
CSN Postal Code (TextField)
CSN Fax (TextField)
CSN Roles (TextField)
CSN Service Office (TextField)
CSN Institution Company (TextField)
User Defined Field Definition

Figure 23 - User Defined Field Properties

Resource Object Name



Name: Thetis User

Table Name:

● Order For User ○ Order For Organization ☑ Auto Pre-populate

Type: Generic

Allow Multiple ☑ Auto Save ☐ Self Request Allowed ☐

Allow All ☑ Auto Launch ☐ Provision By Object Admin Only ☐

Sequence Recon ☐ Trusted Source ☐ Off-line Provisioning ☐

Figure 24 - Resource Object Name