ORACLE INSURANCE

Oracle Health Insurance Back Office

SOAP Service Layer (SVL)

Installation & Configuration Manual

Version 1.50

Part number: G49637-01

March 15, 2026

CHANGE HISTORY

Release	Version	Changes
10.12.2.0.0	1.8	• Added additional paragraph regarding securing web service access.
10.12.2.2.0	1.9	• ohibo.properties changes for the latest releases up to 2012.02 have been added
10.12.3.0.0	1.10	• Made clear to delete retired deployment before updating again.
10.13.1.1.0	1.11	• Updated a number of screen prints.
10.13.1.4.0	1.12	• Added new ohibo.properties for the latest releases up to 10.13.1.4
10.13.2.1.0	1.13	• Web service consumers are now referenced using this correct term.
10.13.3.0.0	1.14	• The situation as needed for the properties file for 10.13.3 has been documented. Please be aware of the new quotations and provider contract properties (the latter were introduced in a patch set on 10.13.2) and the rename of the collective agreement to group contract.
10.13.3.0.0	1.15	• When calling alg_security_pck.svl_grants the grantee parameter name should be passed named as this routine is overloaded. • The folder that covers publishing the services now contains also some simple instructions where the WSDL files can be found in the .ear file.
10.13.3.3.0	1.16	• Added changes in the properties file for patch set release 10.13.3.3.
10.14.1.0.0	1.17	• Added changes in the properties file for major release 10.14.1.0.0.
10.14.1.3.0	1.18	• Added changes in the properties file for patch set release 10.14.1.3.0.
10.14.2.0.0	1.19	• Introduction of secure deployment by default. Rewrote most of Ch. 3 ('Installation of SVL provider services'). Described security configuration. • Added Appendix A ('Removing a WLS domain') • Added Appendix B ('Compare version information in EAR files')
10.15.1.0.0	1.20	• Some additional information added regarding the security configuration of the Service Layer web services. • Added Appendix C ('Managing security policies using WSLT') • Added Appendix D ('fixear.sh') • Added Appendix E ('polman.py')
10.15.3.0.0	1.21	• Added changes in the properties file for patch set release 10.15.1.1. • Added changes in the properties file for patch set release 10.15.1.3. • Added changes in the properties file for major release 10.15.3.0
10.16.1.0.0	1.22	• Added missing changes in the properties file for major release 10.16.1.0 • Added a small paragraph about sizing/load impact
10.16.2.0.0	1.23	• Adapted for FMW 12.2.1.1.0 • Removed old properties file definitions, which break the links above.
10.16.2.3.0	1.24	• Some minor adjustments in SVL domain configuration. • Replaced Log4J configuration by Java Util logging configuration
10.17.1.0.0	1.25	• Updated reference from FRS12211 to FRS12212 • Changed grant instructions
10.17.2.0.0	1.26	• No changes
10.17.2.3.0	1.27	• Added changes in the properties file relevant starting with patch set release 10.17.2.2. • Added JDK version specific information regarding JSSE configuration.
10.18.1.0.0	1.28	• Revised introduction and document title • Revised Architectural Overview
10.18.1.2.0	1.29	• Use setUserOverrides.sh instead of modifying startManagedWebLogic.sh and Server Start arguments. Support for FMW 12.2.1.3. Numerous small changes and updates.
10.18.1.3.0	1.30	• Corrected error in setUserOverrides.sh: removed double quote after disableCaptureStackTrace. • Added warning about patch **28278427**
10.18.2.0.0	1.31	• Rewrote explanation of setUserOverrides.sh
10.18.2.2.0	1.32	• Extended SVL properties file with vecozoverdragsrecht properties.
10.18.2.3.0	1.33	• Introduction of properties file templates
10.19.1.0.0	1.34	• No changes. Republished with different part nr.
10.19.1.4.0	1.35	• Change in properties template is described • Updated description of data source creation • Chapter 6 regarding web service consumers has been defined as 'deprecated'
10.19.2.0.0	1.36	• No changes, republished with new part number.
10.20.1.0.0	1.37	• No changes, republished.
10.20.3.0.0	1.38	• Adapted for changes due to DB 19c and FMW 12.2.1.4 certification

Release	Version	Changes
10.20.4.0.0	1.39	• Added new recommended Initial Capacity configuration option for data sources in Creating a Data Source paragraph
10.21.1.0.0	1.40	• Removed sections about Service Consumers because implementation has changed. See document "Service Consumer Installation & Configuration Manual" • New part number.
10.21.2.0.0	1.41	• Added descriptions about Security Models
10.21.7.0.0	1.43	• Added Statement Timeout
10.22.1.0.0	1.44	• No changes, republished with new part number.
10.23.1.0.0	1.45	• No changes, republished with new part number.
10.23.6.0.0	1.46	• Clarified impact of certification of Forms & Reports Services 12.2.1.19.0
10.24.1.0.0	1.47	• No changes, republished with new part number.
10.25.1.0.0	1.48	• No changes, republished with new part number.
10.26.1.0.0	1.49	• No changes, republished with new part number.
10.26.3.0.0	1.50	• Changed for WLS 14.1.2 and WebLogic Remote Console

RELATED DOCUMENTS

A reference in the text (**doc[x]**) is a reference to another document about a subject that is related to this document.
Below is a list of related documents:

Doc[1]	Object Authorisation within OHI Back Office (docs.oracle.com)
Doc[2]	Oracle Health Insurance Back Office HTTP Service Layer - Installation and Configuration Guide (docs.oracle.com)
Doc[3]	Oracle Health Insurance Back Office - Service Consumer Installation & Configuration Manual (docs.oracle.com)
Doc[4]	Oracle Health Insurance Back Office - Service Callout Installation & Configuration Manual (docs.oracle.com)
Doc[5]	Oracle Health Insurance Security Guide (docs.oracle.com)
Doc[6]	Oracle WebLogic Remote Console Online Help

Contents

1 Introduction

Attention: OHI Back Office releases 10.26.3.0.x to 10.26.8.0.x are certified against Fusion Middleware 12.2.1.4.0 AND 14.1.2.0.0. In WebLogic 14.1.2.0.0, the Admin Console has been removed, and is replaced by the WebLogic Remote Console. This document will assume an installation on Fusion Middleware 14.1.2.0.0. For installation of SVL on Fusion Middleware 12.2.1.4.0, see version 1.49 of this document, as delivered with OHI 10.26.1.0.0.

OHI Back Office web services are used to integrate with existing applications or provide a back end to bespoke self-service portals for insurance members.

OHI Back Office provides two types of web services:

- Business services (aka. SVL services) – this document
  generic object-oriented SOAP/HTTP operations on core OHI Back Office data, with 'find' and 'get' operations to retrieve data and 'write' operations to update/add data.

- Use Case services (aka HSL services) – see Doc[2].
  REST operations to support typical use cases for Dutch healthcare payers. Examples: requesting a new policy, adding an insured member, changing insured products, changing payment method etc.

The default security for all web services is Basic Authentication over SSL.

SVL services are deployed to WebLogic Server to be accessed over SOAP/HTTP. Service operations can also be called using PL/SQL since the service functionality is implemented in PL/SQL.

This document describes the generic technical details regarding the SVL services, how to install and update them and how to change configuration settings.

1.1 Provider web services and web service consumers

There used to be two types of SVL components:

- **provider web services**
  SOAP/HTTP services built by OHI Back Office to be called ('consumed') by client applications in the surrounding environment.

- **web service consumers**
  Java classes built by OHI Back Office to 'consume' third party SOAP/HTTP services from within the OHI Back Office database.

The implementation of the web service consumers is very different from that of the (provider) web services. Since OHI version 10.19.1.3.0 the web service consumers functionality has been implemented in a different way, using Advanced Queing in the database, exposed as JMS Queues. See the "Service Consumer Installation & Configuration Manual" Doc[3]. Since OHI version 10.25.2.0.0 the web service consumers have an alternative implementation, using Oracle Application Express (APEX) functionality instead of JMS Queues. See the "Service Callout Installation & Configuration Manual" Doc[4].

This document now only describes the provider web services. These web services use SOAP/HTTP technology, which implies that WSDL is used to describe the interface and that XML is used to serialize objects.

1.2 PL/SQL and SOAP interface

SVL services are primarily invoked through SOAP/HTTP as document-style web services in a Service Oriented Architecture.
However, they can also be called from PL/SQL. Using the PL/SQL interface can be attractive if your client code is a PL/SQL script or package.

The Java classes which provide the SOAP/HTTP interface are a light weight wrapper around the functional implementation in PL/SQL.

1.3 Licenses

No license is required to use the 'Vecozo-specific' provider web services. The PreAuthorization service is an example of this.

Customers are required to have the appropriate license for using all other SVL provider web services.

Separate licenses can be obtained for the Claims-related functions and for the Policy-related functions within SVL.

Customers with a Connect to Back Office license are currently permitted to install and use the provider web services component of the Service Layer. This is valid until further notice.

A separate license is required for the use of the get function to obtain Composite Relation Details (part of the Relation Service). This function can only be used when the so-called Connector option for the Oracle Service Cloud has been purchased.

The underlying PL/SQL service of a provider web service may not be used when no Connect to Back Office or Service Layer license is obtained for the provider web service.

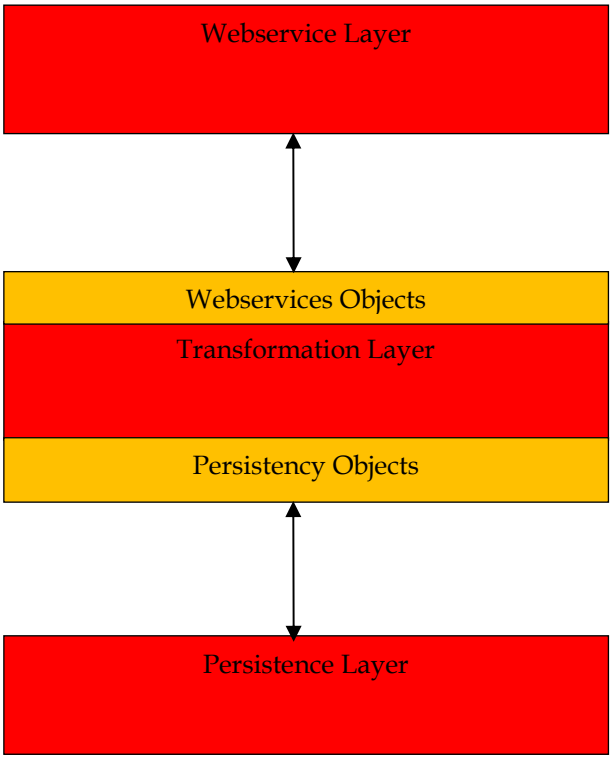
For further information, please consult your OHI sales representative.

2 Architectural overview

This chapter gives a high level architectural overview of the current SVL provider web services.

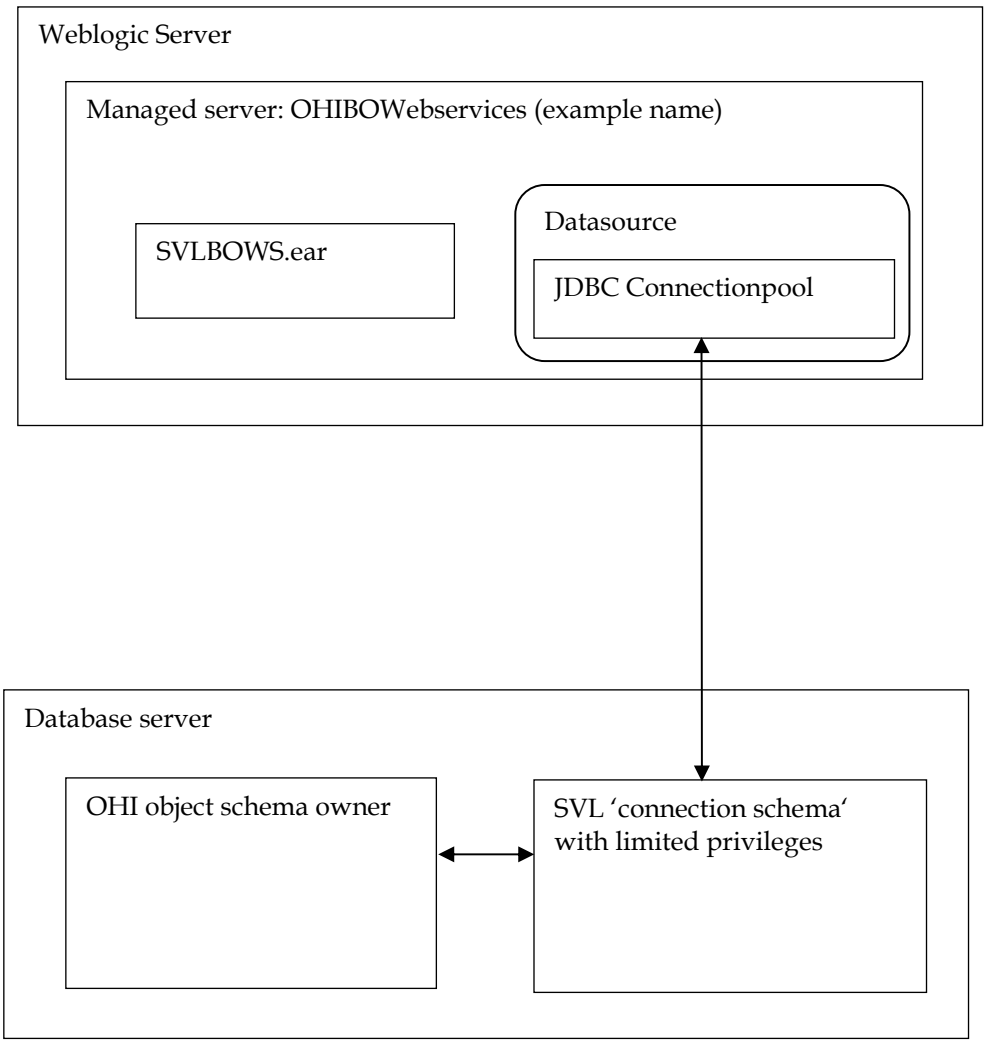
The Java classes of each SVL provider web service are stored as a WAR file. The SVL services are then bundled into a single EAR file and deployed to WebLogic Server.

Each SVL service has the following architecture:



The persistence layer is used to map Java objects to SQL types and vice versa when calling the PL/SQL services in the OHI Back Office database.

The schema below shows how the deployed SVL service connects to the OHI Back Office database.
The SVL 'connection schema' is a separate database account with limited access rights which is used to call the PL/SQL implementation of the SVL service.



3 Installation of SVL provider web services

This chapter describes the steps to (re)install the SVL provider web services.

This chapter contains the following parts to separate the various work areas:

1. Sizing/load aspects
2. Database installation
3. WLS preparation
4. Security configuration
5. (Re)deployment of the SVL application (SVLBOWS.ear)
6. Security aspects
7. Miscellaneous

3.1 Sizing/load aspects

From the "Introduction" and the "Architectural overview" chapters it should be clear that the actual functionality of the services is offered by the PL/SQL implementation in the database.

The SOAP interface as implemented in Java within the application server is a very light weight pass through layer for the request and response messages. It only validates and transforms the actual XML request call to a PL/SQL call and transforms the result from the PL/SQL routine back to an XML response message.

As a result of this choice the load on the application server is very limited. The processing on the application server is typically less than 10% of all processing involved. As a rule of thumb you may assume that when you have a heavy load situation, where 10 CPU threads are involved on the database server handling all the incoming web service requests, you should not need more than 1 CPU thread busy on the application server handling these web service calls.

Most of the simpler service operations on a well-sized and well-performing production environment should not take more than 0.1 up to 0.5 second in total elapsed time when measured on the WebLogic Server. Of this elapsed time most of the time should be spent by the database server handling the call, as mentioned before.

More complicated calls and service calls that return large data sets may take more time, but usually should not exceed response times of more than a few seconds. As an example a typical premium calculation call should be executed within a second and a large set of claim lines (several hundreds) should usually be returned within 5 to 10 seconds.

An exception to this rule is processing a large provider contract write request, this may take minutes to process (on the database server by the PL/SQL implementation).

These response times are based on production experiences with the OHI services as observed until early in 2018.

3.1.1 Deployment choices

Of course the overall load of the OHI application and the portion of the load that is related to the web service calls is customer specific and may change over time. When all insured members use a healthcare payer website that directly calls the OHI web services quite some load may be expected during the commercial season. Offloading choices to standby databases and potentially caching may reduce this load.

It is expected that the actual load of the application functionality that calls the OHI web services and the related database load still widely exceed the web service application server load given only the low level pass through functionality implemented on the application server.

Knowing this, the application server that is used for the OHI user interface processes (implemented through Oracle Forms and WebLogic Server) may be an obvious and valid choice for the deployment of the application server part of the web services. When the application server load of the service calls grows a lot over time, additional processing power may be required. Monitoring the load of the Forms processes and the Service Layer processes will show whether this might be needed at some moment.

An advantage of deploying the web services on the same application server is that existing WebLogic Server licenses for Oracle Forms can be used for the web services. The OHI web services are typically certified for the same WebLogic Server version that is certified for using the Oracle Forms user interface.

Of course requirements like high availability and fail over may influence the deployment choices as well as the use of a service bus. This may lead to re-using existing infrastructure and licenses for other Oracle products using the same WebLogic Server technology stack, provided the same certified versions of these technology products are used.

3.2 Database installation

The database installation for the Service Layer consists of the creation of a separate account (or even several) with Service Layer access privileges. All functional Service Layer database objects are owned by the OHI Back Office schema owner and should have been installed as part of the database part of the OHI Back Office release installation.

The separate database account(s) must be created separately as part of the Service Layer installation, provided you have a license for the Service Layer.

Before creating the account(s), check if you will be able to use the Service Layer:

You should find a database object (package) SVL_UTILS_PCK in the OHI Back Office schema owner. If not, something went wrong with the installation of the Service Layer code as part of the OHI Back Office release installation. If this is the case, please contact the OHI Support department.

If the package is present in your database, you can continue with the database part of the installation.

The use of a separate database account / schema owner for accessing the Service Layer components is required for improved security. This account needs to receive the necessary object privileges.

One or more of these accounts can be created. It is an option to use this account also as schema owner for custom code development. If you choose to do that, please

follow the directions as described in Doc[1]. We advise to use separate accounts for these purposes, though.

The following steps are needed to setup a Service Layer database account:

1. Create a schema owner, for example SVL_USER. Determine the password policy, temporary tablespace, etc. according to your company standards but beware there is no interactive login which might show expiration messages for the password, due to an enforced password policy.

2. Grant create session system privilege to this account.

3. Grant the Service Layer object privileges: logon as the OHI Back Office schema owner, enable server output, and run "alg_security_pck.svl_grants(pi_owner => '<your OHI schema owner>', pi_grantee => '<your account>')", for example:

```
execute
alg_security_pck.svl_grants
(pi_owner   => 'OZG_OWNER'
,pi_grantee => 'SVL_USER')
```

This command does not have to be repeated after each new deployment of a new .ear file. During the database installation of OHI patches any existing grantees of the SVL objects receive any required additional grants. However, if you run into ORA-01403 errors during a web service execution your first check should be to run this command in SQL*Plus, enabling server output before running, and see whether missing grant privileges were granted.

3.3 WLS Preparation

When the database account has been created and granted successfully, a WebLogic Server environment (software home) must be prepared for deploying the SVL application.

We expect that you are familiar with the WebLogic concepts like 'Domain', 'Managed Server', 'Cluster', etc.

These are your options:

- Use the same WebLogic environment which is used for servicing the OHI Back Office user interface and batches. In that case you are required to create a new WebLogic domain (with a new Admin Server) to run the SVL services, in order to prevent interference with the GUI application.

- Deploy the web services in a separate WebLogic environment (possibly on a separate server). This has the advantage that you can separately upgrade or patch the different WebLogic environments, or implement a workload distribution.

  Deploy the web service applications in multiple environments for better scalability. Be sure to deploy the SVL services only once in a Managed Server or a cluster of Managed Servers.

- For testing purposes you may want to have multiple versions within the same domain. In that case you should have a separate Managed Server for each deployment.

Some remarks about installing in a separate WebLogic environment:

- The OHI Back Office GUI application (Forms) installation requires a WebLogic Server "Infrastructure" installation. That means the domain created for Forms needs to have its own database schemas with OPSS and Audit database tables (created by RCU, the Repository Configuration Utility). For the Service Layer domain these schemas are not required, provided you do not select more components during the domain configuration than described.

- When installing in a separate WebLogic Server environment, use a different Installer: use the "Generic" installer instead of the "FMW Infrastructure" installer. When installing in a separate WebLogic environment make sure the correct components are installed when creating the Domain. You need at least:

    - `Weblogic Advanced Web Services for JAX-WS Extension - 14.1.2.0 [oracle_common]`

    - `Weblogic JAX-WS SOAP/JMS Extension - 14.1.2.0 [oracle_common]`

      When you have not installed these components your web services will respond with 'There are error messages.' All info in the functionalFaultType will contain question marks (???).

The instructions in the following paragraphs cover the setup of a new domain including the setting up of Managed Servers, a machine definition, data sources, etc.

This will support the following scenarios:

- Creating a separate domain with a single Managed Server
- Creating a separate domain with a cluster of 2 Managed Servers
- Adding a Managed Server to an existing domain

3.3.1 Using setUserOverrides.sh

To pass Server Start arguments to the WebLogic servers (like memory settings for the JVM, debug options and other Java options), create a file `$DOMAIN_HOME/bin/setUserOverrides.sh`

See the following documents on My Oracle Support for details:

KB581904	How to Customize Environment Parameters using 'setUserOverrides.sh' File
KB364001	How to Pass Unique Start Arguments to Different Managed Servers Using setUserOverrides.sh Script in WebLogic 12c

3.3.2 Requirements

The following requirements/limitations must be taken into account:

- ✓ A certified WebLogic Server version including JAX-WS (SOAP/JMS) extensions. The web services must be deployed on a single Managed Server or a cluster of Managed Servers (the 'target').

For the latest information on certification of specific OHI Back Office versions against WebLogic 12.2.1.3 and 12.2.1.4, please check the Certification information in My Oracle Support.

- ✓ The SVL web services may not be deployed on a Managed Server which is also used for hosting the OHI GUI application (Forms). The Managed Server may not belong to a cluster used for deploying the GUI application.

- ✓ One deployment can only service one single OHI Back Office environment (it connects to a specific connection pool which accesses a specific OHI Back Office 'instance').

If the SVL application must be deployed more than once (for servicing different OHI Back Office environments) each deployment should be on its own Managed Server or Cluster.

SVL can be deployed on the same Managed Servers as HSL and PSL plus OHIJET .

3.3.3 Creating a domain

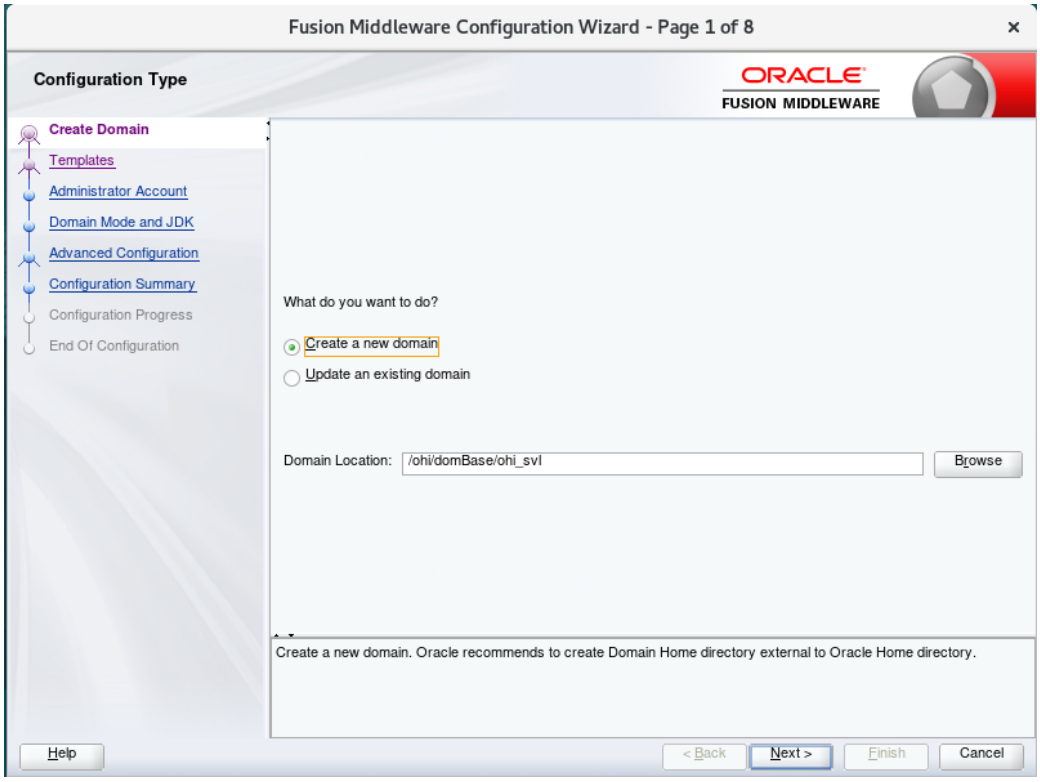
Before creating a Domain, be sure to understand the difference between a "FMW Infrastructure" and a "Generic" WebLogic installation, and the consequences. Make sure the environment variable DOMAIN_HOME is not set.

If you create the new WebLogic Domain from the same software home as the Forms Domain, you have to choose the same "Domain Mode" (Development or Production), to avoid errors during start-up of the new Managed Server(s).

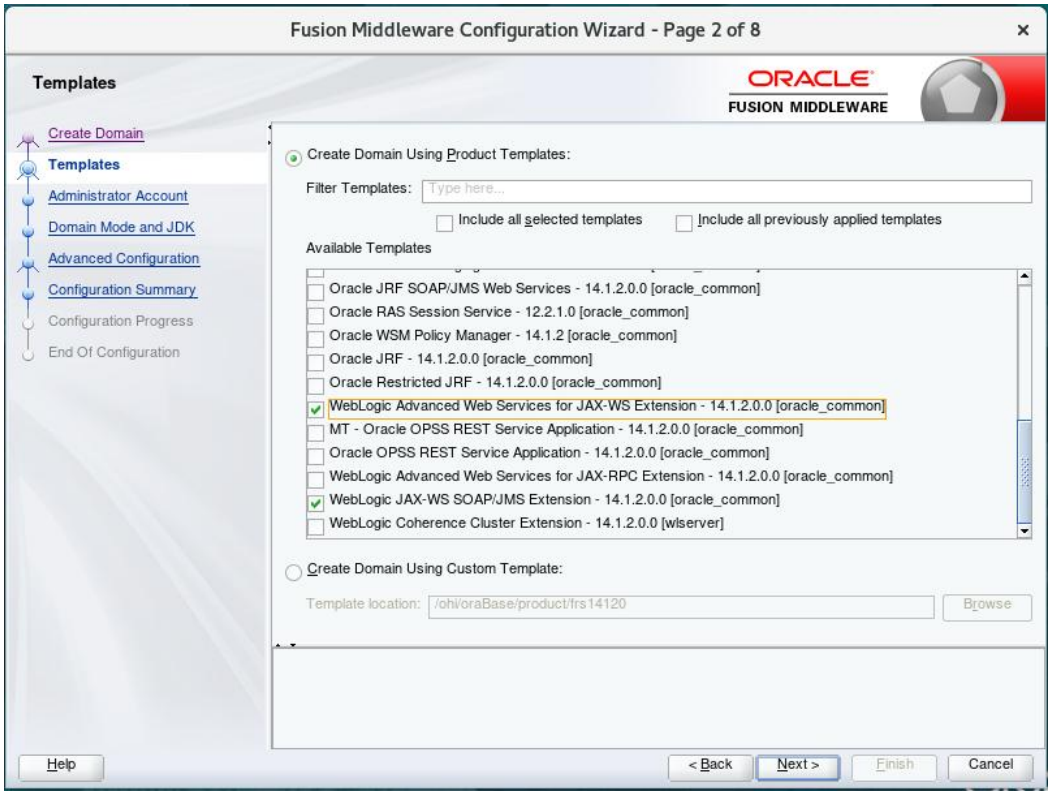
For creating a new WebLogic domain please use the Configuration Wizard (typically in the common/bin folder of the WebLogic Server home, so for example `$MW_HOME/oracle_common/common/bin/config.sh`)

This is a Graphical User Interface and requires tiger-vncserver on the Linux application server, and e.g. tigervnc-client on your PC. Other X11 services can be used.

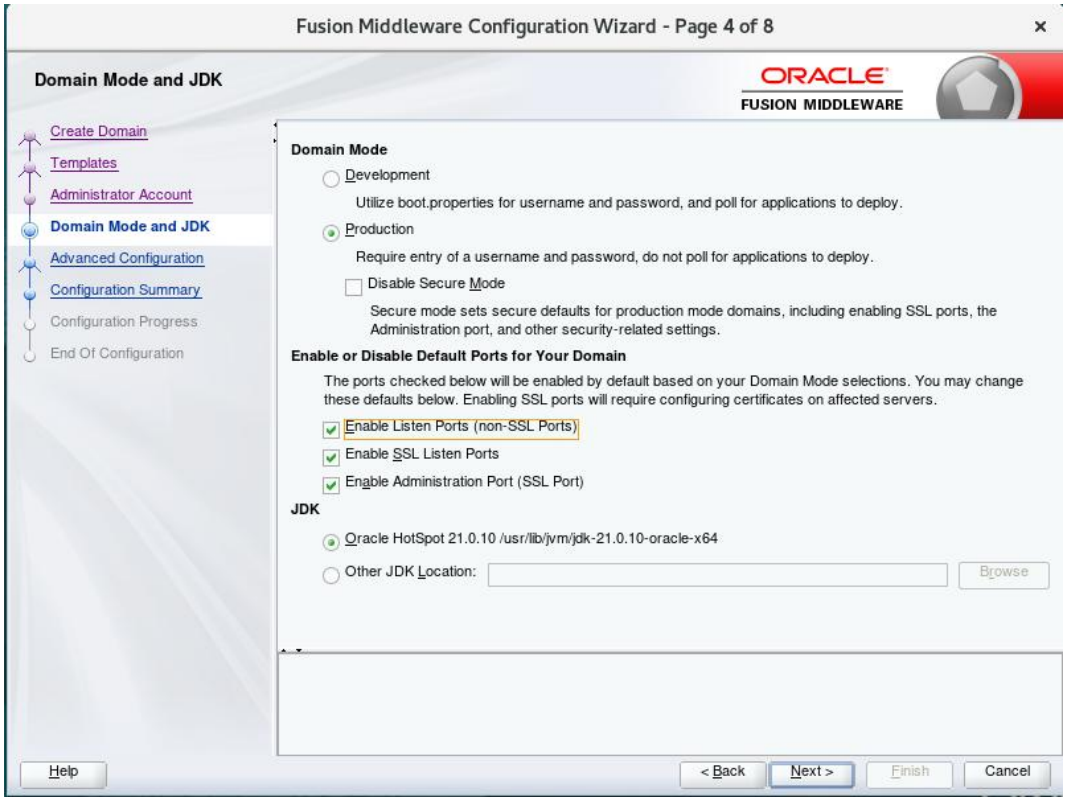
Specify the domain location. This is inside the WebLogic Home by default, but you should specify a location outside the WebLogic Home. The last part of the location will be the Domain Name.



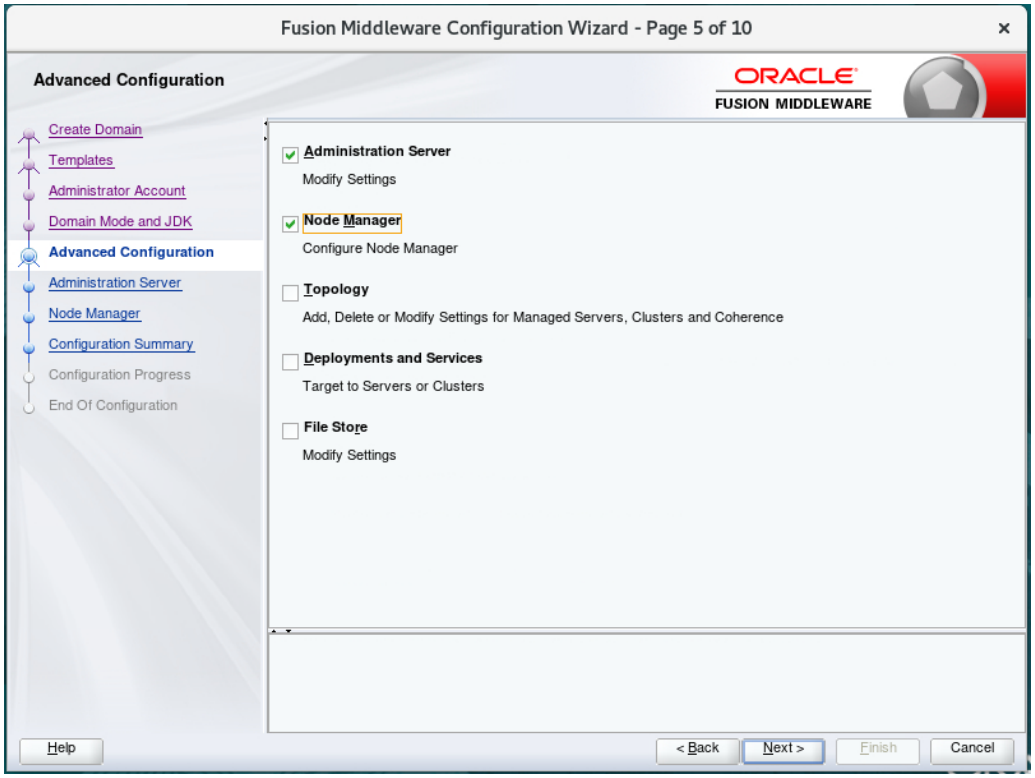
When creating a new domain select at least the options as shown below.



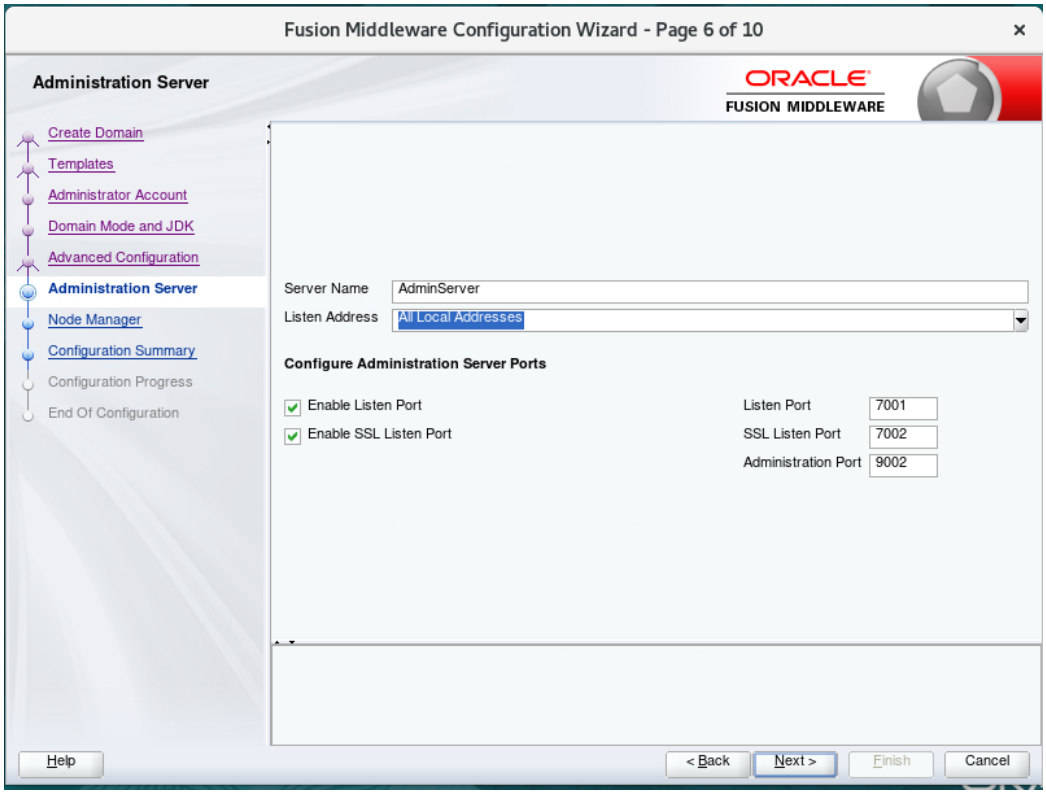
In the next screens, specify the *username* and *password* for the domain administrator account. When prompted for developer or production mode choose *production mode* and pick a JDK.



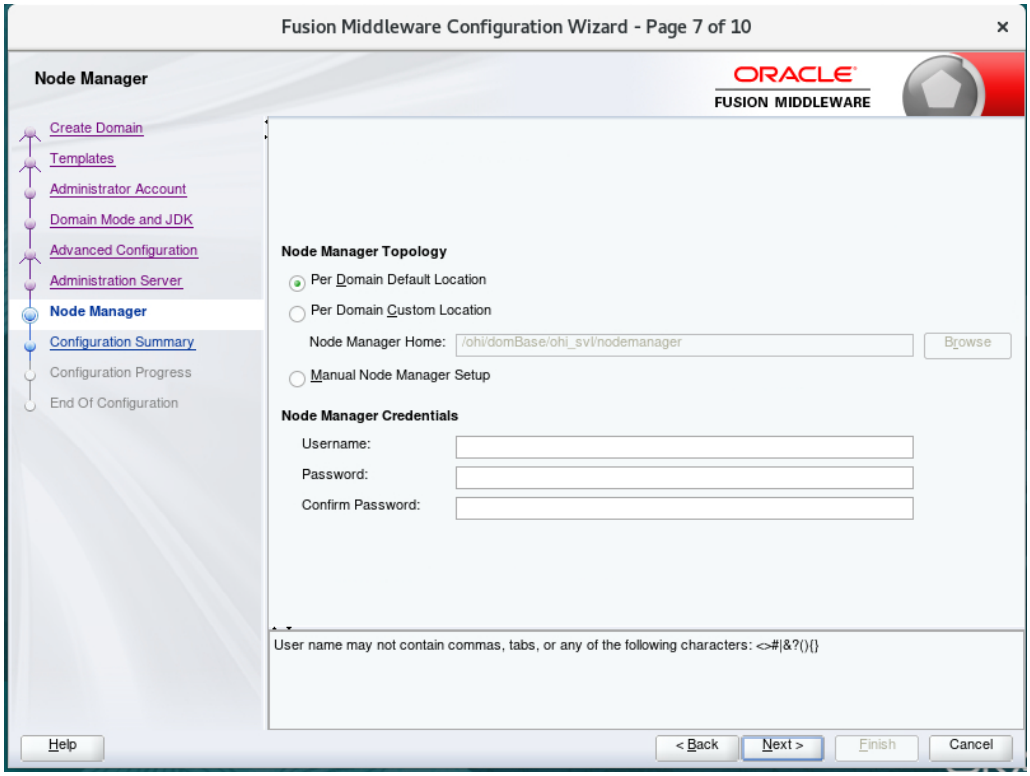
In this documentation we choose to configure only the Administration Server and the Node Manager using the wizard. The Administration Server can be used as the starting point for additional configuration options you may want to choose later:

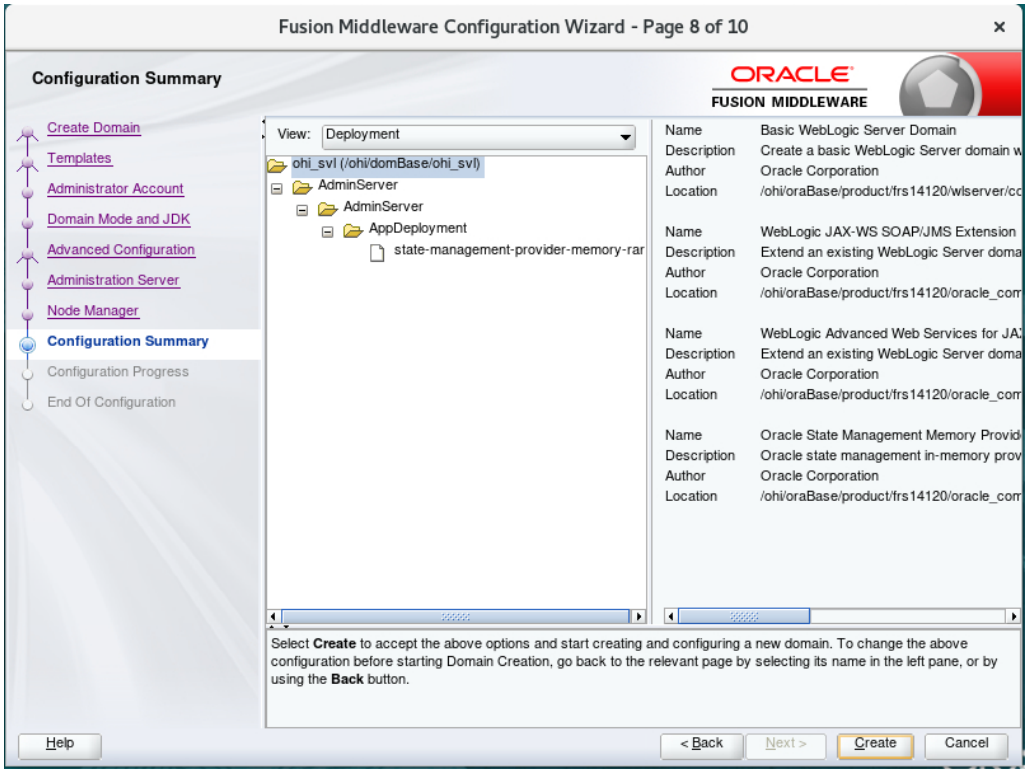


For the Administration Server free port numbers must be specified. Enable SSL to support secure connections. An example using non default ports is shown below.



In the next screen, enter a username and password for the Node Manager:





Finish the domain creation by the Configuration Wizard.

3.3.4 Creating Managed Server(s)

The other WebLogic objects can be created with the wizard too. Here, we use the Administration Server to create one or more Machines, a Cluster (optionally) and one or more Managed Servers.

If you start a Node Manager you can use the WebLogic Remote Console to start the Managed Servers later on. You need to associate the machine with the Node Manager so that the Node Manager can start the Managed Server within the domain of the machine definition.

Start the Administration Server (of the existing or newly created domain) using the startWebLogic.sh script (this is present in the root folder of the domain folder, which you created through the Configuration Wizard).

When the Administration Server is running, either:

- deploy the Hosted WebLogic Remote Console according to paragraph 4.9. *DEPLOYING THE HOSTED WEBLOGIC REMOTE CONSOLE* in the "Oracle Health Insurance Back Office Installation, Configuration and DBA Manual"

  or

- download and install the local WebLogic Remote Console according to paragraph 4.1.2 in the "Oracle Health Insurance Back Office Installation, Configuration and DBA Manual".

- Log in in the Hosted WebLogic Remote Console.
- Connect to the newly created Admin Server.
- Select the "Edit Tree".

- Select “Environment”.
- At this point you should decide whether or not to make the Managed Servers part of a Cluster. If no Cluster exists, you can create one first, using the “Clusters” option.
- Select “Servers”
- Click on “New”
- Enter a name for the new Managed Server. For easy reference you may want to include the domain name and the OHI environment name in the name of the Managed Server, for example “MS_SVL1420_OHIPRD”.
- Click on “Create”.
- In the next screen, in the tab “General”:
    - Click on “Show Advanced Fields”
    - Select a Cluster or create one using the dots behind the field “Cluster” (if desired)
    - Create a Machine using the dots behind the field “Machine”
        - Enter a name
        - Select Type = “Unix Machine”

      and select the new Machine in the main screen.
    - In the “Listen Address” field, fill in the Fully Qualified Domain Name (FQDN) of the application server. Using “localhost” or an IP number may result in errors in SSL/TLS communication.
    - (At least initially, until SSL has been configured and tested) keep the “Listen Port Enabled” and specify a “Listen Port”.
    - Set the “SSL Listen Port Enabled” and specify a “SSL Listen Port”.
    - Set the “WebLogic Plug-In Enabled” on

  Make sure the listen address is the actual listen address that is used by the Node Manager. This is passed as first parameter to the `$WL_HOME/server/bin/startNodeManager.sh` shell script. The correct value can be found as ListenAddress in the file nodemanager.properties.

  This address can be changed in the file nodemanager.properties which is located in the <domain home>/nodemanager folder. This is necessary when you have a node manager per domain.
- Optionally, change the settings in the tab “Logging”.

You need to create a boot.properties file for the new Managed Server for the domain in the domain home Managed Server ../data/nodemanager.

Because you are running in Production Mode, you need to create the file yourself, in the $DOMAIN_HOME/servers/AdminServer/security folder. This file is used when the AdminServer is started by the script startWebLogic.sh. If the file is not present,

the script prompts for the username/password. The same goes for the Managed Servers when you start them through a script.

3.3.5 Creating a data source

The SVL application needs a data source to connect with the OHI Back Office database.

To create a data source in the Weblogic Remote Console:

- Connect to the newly created Admin Server.
- Select the "Edit Tree".
- Select "Services"
- Select "Data Sources"
- Click on "New"
- Enter a name for the data source that reflects its purpose. For example, you may want to reference the database name: DS_OHI_SVL_PRD.
- Enter a JNDI name. The JNDI name will be used in the properties file for starting the SVL application.
- Select a Managed Server as Target, e.g. MS_SVL1420_OHIPRD
- Select "Data Source Type" = "Generic Data Source" for single instance databases. If the data source connects to a RAC data source it is more useful to choose "GridLink Data Source", as this can respond to instance state changes.
- The following fields appear for the different types of data source:

	Generic Data Source	GridLink Data Source
Database Driver	Oracle's Driver (Thin) for Service connections; Versions: Any. Do not use the XA driver.	Oracle's Driver (Thin) for GridLink Connections, Versions: Any
Global Transactions Protocol	N.A.	One-Phase Commit
Listeners	N.A.	The SCAN address of the RAC database listener
Service Name	N.A.	The Service Name of the RAC database
Protocol	N.A.	TCP
FAN Enabled	N.A.	<empty>
ONS Nodes	N.A.	<empty>
ONS Wallet File	N.A.	<empty>

ONS Wallet Password	N.A.	<empty>
Database Type	Oracle	N.A.
Database Name	The (service) name of the database	N.A.
Host Name	The name of the database server	N.A.
Port	1521	N.A.

- For "Database User Name", enter the user name of the Service Layer database account you created in paragraph 3.2 Database installation (e.g. SVL_USER)
- For "Password", enter the password of the Service Layer database account.
- Click on "Create".

The next page shows the results. You can test the connection using the "Test Configuration" icon.

- Open the "Connection Pool" tab, and then the sub-tab "General".
- Consider setting the "Initial Capacity" to 0. During the setup of new connections a health check, if you have configured this (for more information see the OHI Release Installation manual), claims a shared lock that might stall (patch) sessions and vice versa. Setting this option to zero implies no connections are set up after the connection pool is initialized but only on demand.

- Open the "Connection Pool" tab, and then the sub-tab "Advanced".
- Ensure that the option 'Wrap Data Types' is off. This setting is needed for passing CLOB objects to and from the database. If it is checked, it slows down execution.
- Consider activating the "Statement Timeout"'. This is the time after which a statement currently being executed will time out. This can be used to limit the impact of run-away queries (e.g., if a bad execution plan is chosen or a wildcard search is not selective). Especially in scenarios where the requestor (e.g., OSB) stops listening after a certain period – and possibly retries the same operation several times – continued execution of these long running queries can overload the database. This can have serious effects on the performance of other web services. In those cases, it is better to cancel the query (after a period that is a little longer than the timeout of the requestor).

  Queries that are cancelled will result in a technical fault:

  ```
  ORA-01013: user requested cancel of current operation
  ```
  or
  ```
  ORA-03111: break received on communication channel
  ```

  Note that it is possible to specify different data sources for different web services (see properties file for the SVL web services). This can be used to specify different timeouts for different web services. You could create a data source with a long timeout for web services that usually take longer (e.g. pxprovidercontract) and one for web services that are normally quick.

  Be aware that the 'Statement Timeout' is only about database processing. Any processing of output by the Middle Tier is not included in this period.

- Click on “Save”
- You may have to restart your Managed Server to activate this setting.

3.4 Security Configuration

All SVL provider web services are configured to use a default security policy (policy:Wssp1.2-2007-Https-BasicAuth.xml). The default policy enforces basic authentication (against WebLogic, so with a WebLogic username and password) and SSL encryption.

You need to decide on a Security Model for the SVL Web Services. See Comparison of Security Models for Web Applications and EJBs for an introduction of the different Security Models. Your choice will depend on the security standards set by your company.

A very short summary:

- **DD Only** (Deployment Descriptors Only): choose this option if you want to use the default policy and allow any authorized WebLogic user to execute the SVL Web Services
- **Custom Roles:** choose this option if you want to use the default policy and create your own roles to restrict access to the web services. These roles can be created using the WebLogic Remote Console. Unless you create and assign roles, all requests will be refused with HTTP 403 “Unauthorized”.
- **Custom Roles and Policies**: choose this option if you want to overrule the default policy of each web service and create your own roles, eg. to limit access to certain of the SVL Web Services.

See paragraph *Security Aspects* for details on how to override the default policies and roles.

The following steps are needed to set up minimal security for the SVL application:

- Set up security realm
- Configure user lockout
- Enable SSL
- Configure key store
- Configure logging level

3.4.1 Set up a security realm

Create a security realm if this has not already been done (normally realm ‘myrealm’ will already be present).

- Log in in the Weblogic Remote Console.
- Connect to the newly created Admin Server.
- Select the “Edit Tree”.
- Select “Security”.

- Select “Realms”.
- Select “myrealm”.

The security realm ‘myrealm’ will be used to configure the security at application level. If there are no other security realms, this will be the default security realm.

3.4.2 Create a WebLogic user

Unlesss you want to use the default “weblogic” user to make the web service requests (which is not a good idea), you need to create a dedicated user in this security realm, and add the username and password to each SVL web service request.

- Log in in the Weblogic Remote Console.
- Connect to the Admin Server of your web services domain.
- Select the “Edit Tree”.
- Select “Security”.
- Select “Realms”.
- Select “myrealm”.
- Select “Authentication Providers”
- Select “DefaultAuthenticator”
- Select “Users”
- In the screen on the right, click on “New”.
- Enter values for “Name”, “Description” and “Password”. For name, use e.g., WLS_SVL_USER.
- Click on “Create”

3.4.3 lockout

While setting up the SVL web services you may want to disable user lockout. In a production environment you should enable user lockout to discourage fraudulent use.

- In the “Edit Tree”, navigate to the Security Realm “myrealm” (see above).
- Open the “User Lockout” tab.
- Set “Lockout Enabled“ to On.
- Choose values for the other fields.

If you choose Security Model “Custom Roles” or “Custom Roles And Policies”, create your custom roles and or policies now. See paragraph 3.6.3 *Restricting access with custom roles.*

3.4.4 Enable SSL

The SVL services are preconfigured to use a default policy which uses SSL. Therefore, you need to enable SSL for every Managed Server to which you deploy the SVL services application. Check if SSL was enabled when the Managed Server was created.

- In the "Edit Tree", go to the Managed Server configuration.
- Open the "General" tab.
- Set "SSL Listen Port Enabled" = On
- Set "SSL Listen Port" to a value that is free.

3.4.5 Setting up a key store

For testing purposes, you may want to use the Demo keystore that is provided.

Note that in a production environment it is not safe to use the demo keystore.

You should create your own Custom Identity Key Store and import your certificates.

To register your Custom Identity Key Store:

- In the "Edit Tree", go to the Managed Server configuration.
- Open the "Security" tab.

For more information about configuring keystores please read the WebLogic documentation. As a starter you can use this: Configuring Keystores

It contains references to pages that describe in more detail how to obtain private keys, digital certificates, etc.

You should take action and not rely on the demo keystore!

3.4.6 Configure Managed Server logging level

The standard logging level for a Managed Server regarding security issues is intentionally non-informative to discourage fraudulent users.

A typical security-related error message looks like:

Got 'Unknown exception, internal system processing error.'

If you are trying to setup the SVL application to work with SSL and basic authentication in a non-production environment you can configure verbose logging with the following Server Start argument for the Managed Server:

```
-Dweblogic.wsee.security.debug=true
```

Create or edit a file $DOMAIN_HOME/bin/setUserOverrides.sh and add the following text:

```
#!/bin/bash
echo Adding Settings from UserOverrides.sh

# global settings (for all servers)
# this will decrease start up times
JAVA_OPTIONS="-Djava.security.egd="file:/dev/./urandom" ${JAVA_OPTIONS}"
export JAVA_OPTIONS
CONFIG_JVM_ARGS="-Djava.security.egd=file:/dev/./urandom ${CONFIG_JVM_ARGS}"
```

```
export CONFIG_JVM_ARGS

# specify additional java command line options for the Admin Server
#if [ "${SERVER_NAME}" = "${AS_NAME}" ]
#then
#
#fi
#export JAVA_OPTIONS

# specify additional java command line options for specific servers
if [ "${SERVER_NAME}" = "MS_SVL1420_OHIPRD" ]
then
  # add settings for SVL
  # Custom Setting for MS_SVL1420_OHIPRD to set debug level for SSL
  JAVA_OPTIONS="-Dweblogic.wsee.security.debug="true" ${JAVA_OPTIONS}"
fi
export JAVA_OPTIONS
```

Replace the server name MS_SVL1420_OHIPRD with your server name.

When startup times of your service calls are important and the security of the connection is less important you may consider to specify an alternative for retrieving cryptographically strong random numbers (included above):

```
JAVA_OPTIONS="-Djava.security.egd="file:/dev/./urandom" ${JAVA_OPTIONS}"
```

Restart the Managed Server to get the new verbose messages later on. After you have deployed the services and are testing them (through for example SoapUI as described later), you might get a message like this in the Response message:

weblogic.xml.crypto.wss.WSSecurityException: Timestamp validation failed.

This would indicate that you forgot to add a timestamp when calling the SVL application.

3.5 (Re)deployment of the SVL Application

For deploying the SVL application you need to obtain an EAR file. This file is typically named SVLBOWS.ear. It should reside in the $OZG_BASE/java directory on the application server containing the OHI Back Office software release and you can copy it to another location if required.

The WebLogic Remote Console can deploy files located on your local PC and on the Application Server where the WebLogic Remote Console runs (for the Hosted WebLogic Remote Console).

To automate deployments, you can use java weblogic.Deployer. See A weblogic.Deployer Command-Line Reference.

Note that you cannot use an older EAR file with a newer OHI Back Office release and vice versa.

The following scenarios are discussed:

- Deploy to a single Managed Server
- Deploy to multiple Managed Servers
- Deploy to a WebLogic cluster
- Deploy for DTAP (development, test, acceptance, production)

3.5.1 Deploy to a single Managed Server

..3.5.1.1 *Deploy EAR file*

- Log in in the Weblogic Remote Console.
- Connect to the Admin Server of the SVL web services Domain.
- Select the “Edit Tree”.
- Select “Deployments”
- Select “App Deployments”
- Click on “New”
- Enter a name that includes the OHI environment name, to help identify the deployment, e.g. SVL_VOHI.
- Select the Managed Servers that are the Target(s)
- If you deploy the ear file from your laptop, set “Upload” to On, and select the ear file using the popup after “Source”.
- If you deploy an ear file that is already located on the application server, set “Upload” to Off. Enter the file name (including the directory) in “Source Path”.
- Set “Security Model” to the desired option. Select the Security Model as discussed above. The default for SVL is “DD Only”.
- Set “Staging Mode” to “Stage” so the ear file will be available on the Managed Server for redeployment.
- Set “On Deployment” to “Start Application”.
- Click on “Create”.
- In the next page, review the settings. At this moment the version of the .ear file is also shown (can contain up to 4 digits like any application source).
- If needed, change values and click on “Save”.
- Click on the Shopping Cart, top right, and select “Commit Changes”.
- Go to the “Monitoring Tree” (via “Home” or via the icon in the left margin)
- Click on”Deployments”
- Click on “Application Management”
- Check the “State” of your deployed application. This should be “Active” if the svl.properties file has been specified and can be found.
- In the “Monitoring Tree”, click on “Application Runtime Data”.
- Click on your deployed application.
- Check the “Health State”. This should be “Okay”.

..3.5.1.2 *Specify configuration file*

Before using the web services implement the following actions as described below. These actions have to be executed only once. There is no need to repeat them when you update the deployment or delete and install it again.

Add two lines to the file $DOMAIN_HOME/bin/setUserOverrides.sh you created earlier. Add the lines to the part for the SVL Managed Server, as indicated below:

```
# specify additional java command line options for specific servers
if [ "${SERVER_NAME}" = "MS_SVL1420_OHIPRD" ]
then
  # add settings for SVL
  # Set debug level for SSL
  JAVA_OPTIONS="-Dweblogic.wsee.security.debug="true" ${JAVA_OPTIONS}"
  # Set location for SVL properties file
  JAVA_OPTIONS="-Dapp.properties="/u01/app/oracle/product/OHI/vohi/svl_ws.properties" ${JAVA_OPTIONS}"
  # Disable stack traces in soap faults
  JAVA_OPTIONS="-Dcom.sun.xml.ws.fault.SOAPFaultBuilder.disableCaptureStackTrace="false" ${JAVA_OPTIONS}"
fi
export JAVA_OPTIONS
```

- Make sure to keep the parts with ${JAVA_OPTIONS} on the same line
- This example uses a properties file with the custom name svl_ws.properties which is located in the $OZG_BASE folder of your OHI Back Office application server environment, but you can also specify the default name: the $OZG_BASE/conf folder and svl.properties.

  The contents of this file are discussed in chapter 4 *Configuration files for provider web services.*
- The second line adds a setting to disable stack traces in soap faults (which is a security measure)

When completed, (re)start the Managed Server. This can be done from the WebLogic Remote Console, or from the command line with the following commands:

```
cd $DOMAIN_HOME/bin
./startManagedWebLogic.sh MS_SVL1420_OHIPRD http://<FQDN>:7016
```

The example above contains the Managed Server's name as first parameter and the listen address of the Admin Server of the domain as second parameter.

Check in the <ManagedServer>.out file in the logs directory of your Managed Server whether the command line contains the arguments as specified above.

If the file specified by svl.properties cannot be read, messages as below will show up:

```
ERROR: logfile could not be set because of: null
```

3.5.2 Deploy to multiple Managed Servers

You may deploy the application to more than one target.

Example: if you choose to target the application to Managed Servers MS1 and MS2 running on the same application server, the application will be available on separate end points. The URLs of these end points will only differ in port number. You will have to configure a load balancer or Oracle HTTP server to distribute the load and/or do failover.

If you choose this rather unlikely scenario, be aware that each Managed Server should have different Server Start argument values (for svl.properties).

3.5.3 Deploy to a WebLogic cluster

You may deploy the application on all the Managed Servers of a WebLogic cluster. This may be needed for better scalability. Be aware to use some form of load balancing to allow the use of a single end point by the client applications.

The best way to implement this type of deployment depends on your specific situation.

If you are planning a load balanced environment with multiple Managed Servers in a WebLogic cluster, it is vital that the configuration of every Managed Server is aligned with the others.

3.5.4 Deploy for multiple environments (DTAP)

If you use several OHI-related environments to support the various DTAP (Develop-Test-Accept-Production) stages you may want to have different versions of the SVL application running at the same time.

To implement this you need to:

- Create a Managed Server for each of the DTAP stages.
- Create a data source for each of the DTAP OHI Back Office databases and deploy that data source only to the corresponding Managed Server.
- Create an svl.properties file for each Managed Server.
- Configure each Managed Server to start up with the appropriate svl.properties. Add multiple tests for Managed Server names in `setUserOverrides.sh` to specify a different file name for each Managed Server.
- Deploy the appropriate version of the SVL application to its corresponding Managed Server and give it a unique deployment name to identify its deployment.

3.5.5 Publishing and testing the deployed services

After you have deployed the web services, perform a few small initial tests, using curl or wget, or see paragraph 3.6.4 *Testing with SoapUI* to test the isAlive request operation.

If you have not tested before with security enabled, please read on in the following paragraph how to test with security enabled. When you are sure the deployment succeeded you can proceed with publishing the WSDL's to your developers and testers.

The WSDL files for the SVL provider services can be found in various locations:

- (after patching OHI Back Office): $OZG_BASE/help/nl_nl/Output/OHIHelp/Webservices/Soap_webservices/Generated_docs/XSD_WSDL.zip
- (after patching OHI Back Office with a major release): $OZG_BASE/java/SVLBOWS.ear. This file contains <web application>.war files for the different web services. Locate the WSDL and XSD files in the WEB-INF/wsdl folder of the WAR archive.

You can extract files from EAR or WAR files with jar (similar parameters as 'tar') or use a zip utility (also on a Windows based platform).

- (after deployment): via the service URL. The format of the service URL is <servername>:<port>/<web application name>. The WSDL will contain the https based address for the service.

- (after deployment): via the "Context Root URL" in the WebLogic Remote Console. Navigate to e.g.: Monitoring Tree -> Environment -> Servers -> <MS_SVL1420_OHIPRD> -> Deployments -> Application Runtimes -> <SVL_VOHI_vxxx> -> Component Runtimes -> <MS_SVL1420_OHIPRD> _/<OHIBOWebservicePxRelation_v1234>. Beware: the tree view on the left does not show all Component Runtimes. The list on the right does. Copy the value of the "Context Root URL" to a browser. In the resulting page, click on the link for the WSDL.

-

3.6 Security Aspects

The SVL services provide an additional access channel to retrieve and change OHI Back Office data.

Your SVL services deployment must be sufficiently secure to prevent exposing sensitive data or enabling unauthorized changes to the OHI Back Office data. Therefore, access should be limited to trusted systems and interfaces. Otherwise people in your organization might be tempted to try to misuse the functionality provided by the SVL services.

Please consult the 'Oracle Health Insurance Security Guide Doc[5] for more information about OHI Back Office security aspects.

As a minimal policy to reduce the risk of unauthorized access and network sniffing, all SVL provider web services are configured to use a default policy (policy:Wssp1.2-2007-Https-BasicAuth.xml). This policy requires HTTPS communication and a username/password combination. This is a WebLogic username.

It is your responsibility as an administrator to secure the SVL services within your organization.

This paragraph provides some pointers to get started:

- Using the default security policy
- Overruling the default policy
- Restricting access with custom roles
- Testing with SoapUI

3.6.1 Using the default security policy (authentication)

To check the default security policy, retrieve the actual WSDL from the WebLogic Remote Console (see above).

Note that this information is visible only if the SVLBOWS application is active.

View the WSDL to examine the policy:

This means the default security policy requires you to send a username, password as well as a timestamp (!) to authenticate a call. When using SoapUI for testing, as described later, this will become more clear.

3.6.2 Overruling the default policy (authentication)

You may replace the default policy with a stronger policy.

Currently, this is not supported by the WebLogic Remote Console. Use WLST scripting to set a different WS Policy for the endpoint(s) of the SVL Web Applications.

For testing purposes, on non-production environments only: if you experience problems in calling your web services, you could temporarily remove the security policy. That way, you can test if the Web Service works properly without a security policy enabled. Of course you should only do this temporary and re-enable it again as soon as possible.

You may also opt for using specialized security solutions like Oracle Web Services Manager. In such a situation you may disable the policy and leave the security implementation to such a product.

3.6.3 Restricting access with custom roles (authorisation)

The default policy allows any valid WLS user to access the SVL services. This includes the 'weblogic' user (!).

You can restrict access at the service (or even operation) level by creating and granting global roles in the Weblogic Remote Console.

See Security Policies and Roles in Oracle WebLogic in the Remote Console Online Help.

- Create one or more global roles in the security realm used for the SVLBOWS application. For example, SvlAccessRole (to be used for all SVL services) or SvlPxRelationRole (to be used for a single SVL service).

- Grant each service (operation) to the appropriate global role. Decide if you want to implement fine-grained access (multiple roles, grant at service operation level) or coarse-grained access (one role, grant at service level) or anything in between.

- Create WebLogic users for the SVL services.
- Grant the appropriate role(s) to each user.
- Restart the Managed Server to ensure that all changes are processed.
- Verify that the new access rules are now in place (for example using SoapUI).

3.6.4 Testing with SoapUI

SoapUI is a tool for testing web services which can be downloaded from http://www.soapui.org.

You have to test and use the OHI Web Services with a client that uses TLS 1.2 or higher.

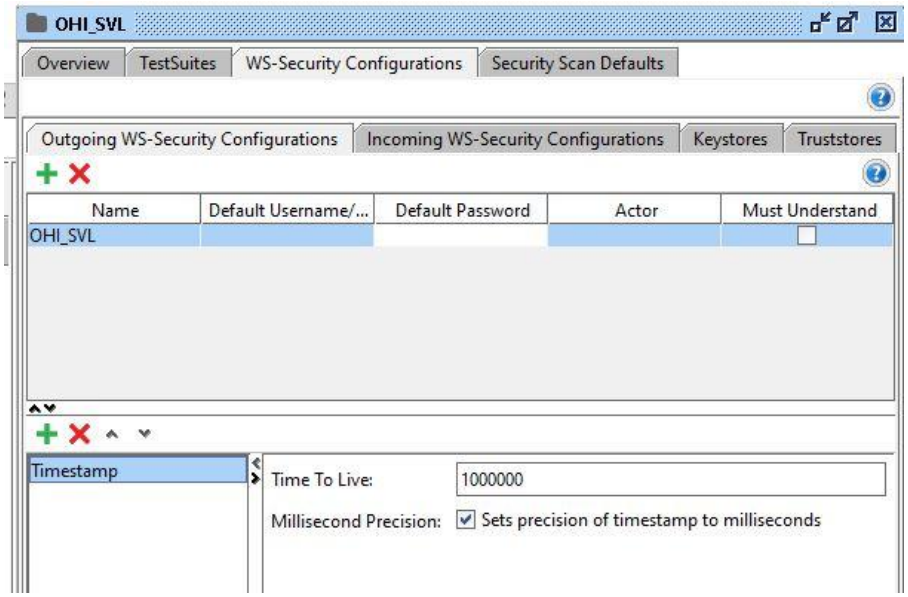
SoapUI is not only useful for testing the functionality of the SVL services, but it is also suitable for testing their security settings.

The following procedure should work if you deployed the SVL application using default security policies:

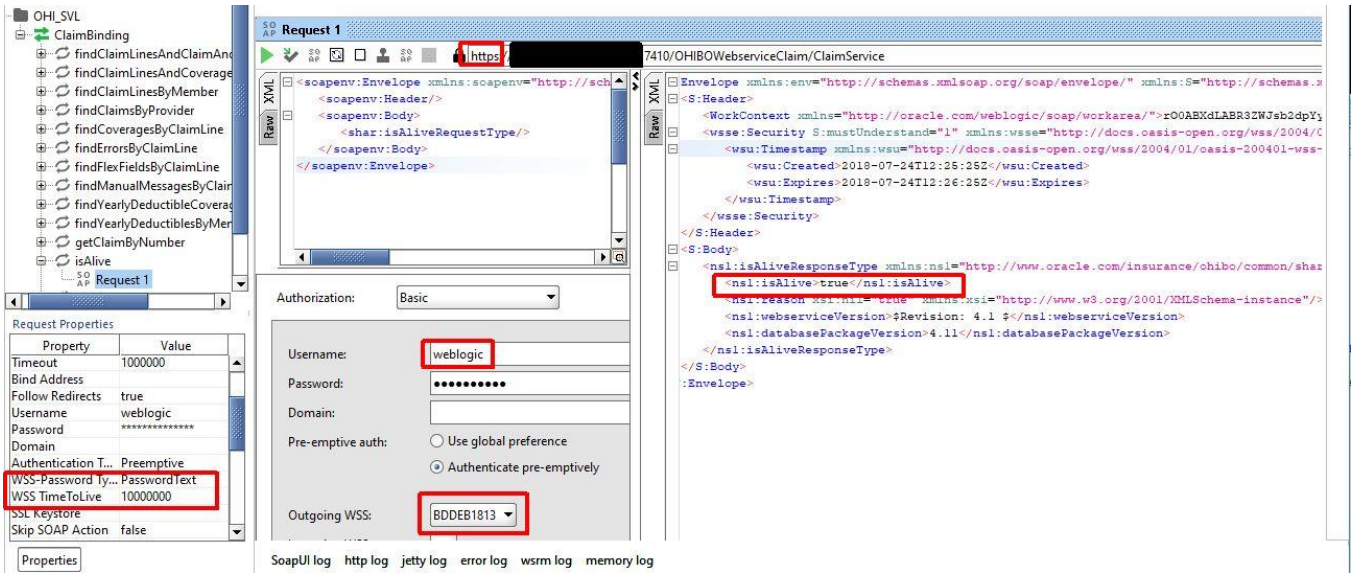
- Create a new SOAP project based on the service WSDL.
- Create requests (these should be SSL requests)
- Create an 'Outgoing' WS-security configuration at project level to include a timestamp and a time to live for the timestamp (e.g. 1000ms)
- Open request for 'isAlive' operation.
- Set the properties in the "Properties" panel in the lower left corner, or in the "Auth" panel (visible after you have opened the actual request message window, change Authorization from None to Basic):
    - Username= <user defined in weblogic>
    - Password= <password defined in weblogic>
    - Authentication Type = Preemptive
    - WSS-Password Type=PasswordText
    - WSS TimeToLive= 10000000
- Send the "isAlive" request. The request should succeed because 'weblogic' is a valid WLS user.
- If your response is empty and returns within a few milliseconds your calling configuration is not ok (leaving for example the password empty results in a HTTP/1.1 200 OK in the raw message)
- If you want to get some HTTPS related information from the SSL requests in the .out or .log file of the Managed Server, you need to enable additional debugging (by default only HTTP requests will show up for example in the access log file). Please add the following to the Managed Server Start arguments in DOMAIN_HOME/bin/setUserOverrides.sh:

```
JAVA_OPTIONS="-Dssl.debug="true" ${JAVA_OPTIONS}"
```

Example of a WS-security configuration with 1000ms 'Time To Live' for the Timestamp setting (this window is accessed on the SoapUI project level by double clicking or choosing the right mouse menu 'Show Project View'):



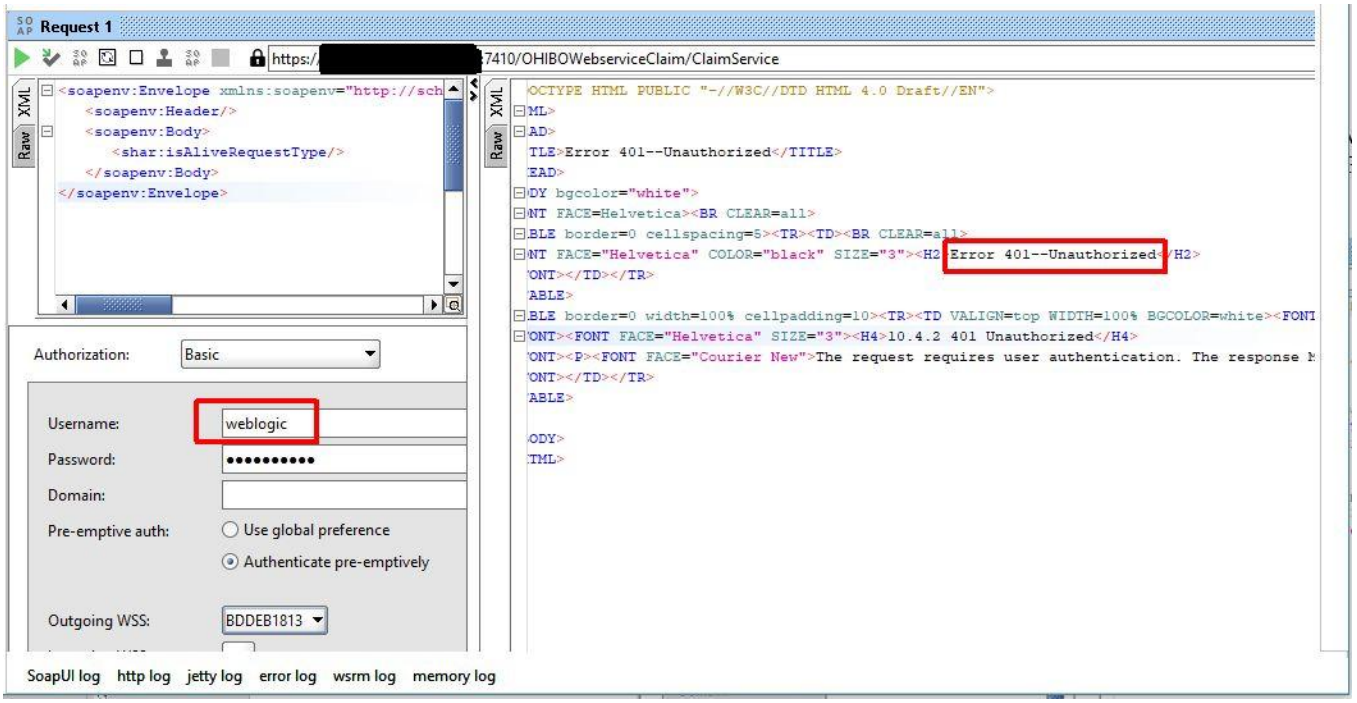
Testing the isAlive operation with an authorized WLS user (created 'scott' and linked to the Claim service through group 'AppTesters'):



Note:

- SSL connection
- Preemptive authentication type
- Basic Authentication
- Outgoing WS-security configuration (choose the configuration you created at the project level)
- Received a valid response.

Finally, an example with the non-authorized WLS user 'weblogic' (we did not assign 'weblogic' to the fictitious 'SvlPxRelationRole' needed to access the PxRelation service):



Note:

- Same authorization but different user credentials as in the example with 'scott'.
- Error message indicates that access was denied to the service.

4 Configuration files for provider web services

The previous chapter mentioned a properties file svl.properties. This chapterprovides details for that file.

4.1 Properties file template

With the OHI Back Office release installation, a properties file template called *svl.properties.template* is distributed to the $OZG_BASE/conf/Back-Office directory. Each OHI Back Office release may overwrite this template with an updated version. The svl.properties.template can be used as an example to create your own svl.properties file (for example in $OZG_BASE/conf).

Please note that all values are examples. You should consider if these values are appropriate for your installation and replace them with your own values if needed. Values indicated with <<SOME_NAME>> in the templates are placeholders and must be replaced. This notation is intended to make scripted deployment easier.

Also make sure not to set log level to FINE, FINER or FINEST in production mode, use SEVERE or WARNING instead.

The properties are described in more detail in the next section.

4.2 Back Office web services properties file

The Back Office properties file for the SVL web services is specified in the Server Start argument of the Managed Server with:

```
-Dapp.properties=<filename>
```

See paragraph 3.5.1 *Deploy to a single Managed Server* to set the Server start argument using the file setUserOverrides.sh.

The file (suggested name 'svl.properties') contains a number of properties.

The properties for specifying logging functionality are generic for all SVL web services and are specified only once.

Other settings can be set per web service and may differ for a certain web service. A web service can support several web service operations and typically references a single WSDL. The properties are named below and you should adapt the values to the needs of your organisation.

The logging properties are not specific for each web service, so specify the logging settings for the deployment as a whole (using Java Util Logging, in short JUL):

1. `common.logging.filename`
   A value to specify the directory + filename that will be created for logging.

2. `common.logging.loglevel`
   The severity level for the logging. This can be SEVERE, WARNING, INFO, CONFIG, FINE, FINER or FINEST (not all these levels are actually used within this web services implementation).

3. `common.logging.loglimit`
   The maximum size of the file, in bytes. If this is 0, there is no limit. The default (used when the value is omitted or an invalid value specified), is 1000000 (which is about 1 MB). Logs larger than 1MB will then roll over to the next log file.

4. `common.logging.logcount`
   The number of log files to use in the log file rotation. The default is 1 which produces a maximum of 1 log file, meaning that when the maximum size is reached the file is emptied and not saved to a 'rotation file' resulting in recent log information being deleted. We advise to specify a value of 2 or higher for that reason.

5. `common.logging.logappend`
   A value to specify to append to the logfile or not when the application is restarted. Possible values are True and False (the default is True).

Additionally, you need to set 2 properties for each web service:

1. A value for the application user that will be used when web service operations are executed. This determines the user identity that is used for logging changes to the data, and which also determines the language of this user to be used for messages.
   This is the `<service>.callcontext.usercontext.user.username` property. The value must be the Oracle username of a registered Back Office user (in Dutch: "functionaris"). The examples in this document use SVL_FUNC_USER.

   NOTE: Do not use the technical account (SVL_USER) that is used for the data source.

   NOTE: Do not use the name of a Back Office user ("functionaris") that also has a database account.

   NOTE: Do not use the value "MANAGER". Records created and updated by SVL functionality should be recognizable as such. Using MANAGER will make it impossible to distinguish those records from records created or updated by batch procedures and conversion scripts that might run with the MANAGER account.

   We strongly advise to have a separate administrative user that is known as a Back Office user ("functionaris") but does not have a database account, so changes through the web service are recognized when looking at who created or changed a record.

2. A value for the data source within WebLogic that will be used for connecting to the database. This is the `<service>.datasource.jndiname` property. This can (and generally will) be the same data source for all web services but you can use different data sources. This can be used for example to implement different availability per web service or even prevent a web service from being used. The value must be the JNDI name of an existing DataSource. Please note: you need to specify 2 forward slashes to indicate a forward slash in the JNDI name.

BEWARE: Be sure you do not add any space after the values, before an end of line character. This may lead to malfunction of the web services.

4.2.1 Keeping svl.properties up to date

As new web services are released through (patch) releases of OHI Back Office, the installation instructions will tell you to change the svl.properties file if required.

Also, an updated properties file template will be released, as described in the previous section 'Properties file template'.

4.2.2 PX services

Web service names starting with "px" are write services (i.e modifying data) based on new standards for implementing the OHI BO web services. The request message passed is (part of) a 'pixel' photo of how the 'end situation' after the write activity should look (the 'demanded' situation is described and the service needs to determine intelligently what changes need to be executed to finally reach the situation described in the 'photo'). Pixel photo is abbreviated to 'px', to indicate this new type of services.

5 OHI release upgrade and provider web services

When you need to redeploy the provider web services (the .ear file) because a new version is delivered in an OHI release this is relatively simple. Please follow the steps below:

- Check your web service properties file (typically svl.properties) and implement necessary changes for your release. For information about the contents please see the previous Chapter.
- Log in in the Hosted Weblogic Remote Console.
- Connect to the Admin Server of the web services Domain.
- Select the "Edit Tree".
- Select "Deployments"
- Select "App Deployments"
- If you already have a version of the SVLBOWS deployment, mark the check box in front of that deployment and delete it.
- Click on the Shopping Cart, top right, and select "Commit Changes".
- Now execute the steps described in paragraph 5 *(Re)deployment of the SVL Application.* Determine whether the same source path still applies (typically a new version is delivered in the $OZG_BASE/java folder of your environment but your organisation may have additional distribution methods implemented).

After this the deployment state of the web services should be Active again (be sure the Managed Server(s) is/are running, otherwise start it/them to get this result).

If not, check whether your OHI database environment and deployed version are correct, meaning that their version levels correspond with each other.

Appendix A – Removing a WLS domain

If you want to restructure your environment or recreate a domain you can remove an existing domain.

In order to do this make sure all servers for the domain are stopped and make sure there is no Node Manager process running which 'guards' this domain.

Next perform the following actions:

- Completely remove your domain directory including all contents.
- Remove any reference in start and stop scripts to this domain.
- If you created Repository schemas using the Repository Creation Utility, remove them by runinning the rcu utility again.
- Remove, if present, the domain from the <WebLogic home>\oracle_common\common\nodemanager\nodemanager.domains.
- Remove the domain from the domain-registry.xml file which is located in the Middleware home folder ($MW_HOME).

For more information please use the standard WebLogic documentation.

6 Appendix B – Compare version information in EAR files

As of release 10.14.2.0.0, OHI Back Office versions the SVL webservices operations, not the XML Schema Definitions or the provider web services.

The SVLCMPRV.pl script compares two EAR files and lists version differences between web service operations.

The script can be used from release 10.14.2.0.0 onwards.

6.1 Invocation

SVLCMPRV.pl resides in the $OZG_BASE/sh directory

SVLCMPRV.pl should be run from the command prompt. The script requires two .ear files as parameters.

For example, to compare versions 10.15.1.3.0 and 10.15.3.0.0:

```
perl $OZG_BASE/sh/SVLCMPRV.pl \
$OZG_PATCH/10.15.1.3.0/java/SVLBOWS.ear \
$OZG_PATCH/10.15.3.0.0/java/SVLBOWS.ear
```

6.2 Operation

SVLCMPRV.pl extracts each EAR file to a temporary folder before comparing the version numbers of the operations defined in the WSDL files.

A higher version number means that at least one definition in the type hierarchy used by the operation has changed since the previous version.

Example:

- Assume that for release 10.15.3.0.0 a region code has been added to the PxAddressType. The PxAddressType is part of the type hierarchy for PxAbstractRelationType, used in the request message for the PxRelationService.WriteRelation operation.
- If the previous version for PxRelationService.WriteRelation was ‘v1’ (10.15.1.3.0 SVLBOWS.ear), then the new version will be ‘v2’.
- If we compare the SVLBOWS.ear files for these two versions we should see that PxRelationService.WriteRelation has ‘v1’ for 10.15.1.3.0 and ‘v2’ for 10.15.3.0.0

A comparison between web service operations may yield the following results:

- An operation is present in the 10.15.1.3.0 SVLBOWS.ear but not in the 10.15.3.0.0 SVLBOWS.ear file. This may mean the operation has been removed or renamed.
- An operation is present in the 10.15.3.0.0 SVLBOWS.ear but not in the 10.15.1.3.0 SVLBOWS.ear. This may mean the operation has been added or renamed.
- A change in revision number is reported. This may mean that the message or the underlying XML schema definition has changed.
- If no change has been made to the operations and/or revision numbers the message: INFO: No differences found will be given.

6.3 Output

The output is written to the console. Sample output:

```
INFO: BrokerService.findBrokerRequestTypeMessage(v1) only occurs
in [$OZG_OPL/10.15.3.0.0/java/SVLBOWS.ear].brokerservice.wsdl
INFO: BrokerService.findBrokerResponseTypeMessage(v1) only occurs
in [$OZG_OPL/10.15.1.3.0/java/SVLBOWS.ear].brokerservice.wsdl
INFO: BrokerService.getBrokerDetailsByBrokerCodeRequestTypeMessage
[$OZG_OPL/10.15.1.3.0/java/SVLBOWS.ear].brokerservice.wsdl:v1
[$OZG_OPL/10.15.3.0.0/java/SVLBOWS.ear].brokerservice.wsdl:v2
```

7 Appendix C - Managing security policies using WLST

All web services in the SVLBOWS application are configured with a default security policy (policy:Wssp1.2-2007-Https-BasicAuth.xml).

The WebLogic Remote Console does not support managing the security policies on a per service basis.

Since there are currently over 20 web services in the SVLBOWS ear application, managing the security policies for the SVL web services manually is a burden if you have a DTAP landscape with many environments for development and testing.

To automate the administration of WLS environments, the WebLogic Scripting Tool (WLST) can be used.

This chapter aims to help you manage the SVL provider web service policies through WLST and explains some of the restrictions that apply.

7.1 Relevant WLST commands

Generally, an administrator records a manual session to generate a WLST script for later editing.

This is not possible for managing web service policies, so you need the WLS documentation to find the required WLST commands.

The relevant commands are:

- listWebServices(application, detail=true)
  Lists the web services and security policies of an application.
- listWebServicePolicies
  Lists the service policies attached to a given service of an application.
- attachWebServicePolicy
  Attach a service policy to a given service of an application.
- detachWebServicePolicyDetach a service policy from a given service of an application.

7.2 Requirements

7.2.1 WLS version

The WLST commands for listing, detaching and attaching web service security policies are available by default in the WebLogic 12 Infrastructure installation. See Oracle® Fusion Middleware WLST Command Reference for Infrastructure Components for documentation.

7.2.2 OWSM needed to attach policies using WLST

If you use WLST to attach a security policy to a web service you will use the attachWebServicePolicy command. For this command you must have OWSM installed (and have a license to use OWSM).

Notes:

- this requirement does not apply to detaching security policies.

- you can only attach OWSM security policies with WLST
- you can not manually attach (Weblogic) security policies using the WebLogic Remote Console.
- See Securing Web Services and Managing Policies with Oracle Web Services Manager for details.

7.3 Restrictions in WLST

The WLST command for attaching a security policy to a web service is attachWebServicePolicy.

Unfortunately it can only attach OWSM policies. This was a design decision by the WebLogic development team.

It has the following implications:

- You can detach the default WLS security policy (policy:Wssp1.2-2007-Https-BasicAuth.xml) using WLST.
- You cannot re-attach policy:Wssp1.2-2007-Https-BasicAuth.xml or any WLS policy using WLST.
- If you want to attach any policy using WLST you must configure OWSM into your domain and select a valid OWSM policy.

Note that you can not re-attach policy:Wssp1.2-2007-Https-BasicAuth.xml and other non-OWSM policies manually using the WebLogic Remote Console.

7.4 Tips and Tricks

7.4.1 Activate application before running WLST commands

Be aware that WLST can only access a running application.

This requires that:

- The AdminServer is running
- The Managed Server for the SVLBOWS application is running
- The SVLBOWS application is active