ORACLE INSURANCE

Oracle Health Insurance Back Office

Security Aspects

version 1.7

Part number: E54856_01

February 12, 2014

CHANGE HISTORY

Release	Version	Changes
10.12.2.0.0	1.6	• Improved description regarding client side access for web services.
10.12.2.0.0	1.6	• Added note about how to disable SQL injection in the forms interface.
10.13.3.0.0	1.7	• Updated references to other documents

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 with in brackets their internal code. Most documents are available on docs.oracle.com, some of them on Beehive Online:

- **Doc[1]**: Object Authorization within Oracle Health Insurance (internal id cta13533)
- **Doc[2]**: OHI Back Office General Users Manual API (cdo14527)
- **Doc[3]**: Reading, Writing and Authorizing Oracle Health Insurance Application Files (cta13531)
- **Doc[4]**: Oracle Health Insurance Installation, Configuration and DBA Manual (cta13508)
- **Doc[5]**: Oracle Health Insurance Data Marts - Administrator reference.

The following Oracle standard documentation deals with the database-specific security issues. This documentation can be downloaded from the Oracle Technology Network (http://otn.oracle.com).

- Oracle Database 2 Day + Security Guide 11g Release 2 (Part No. E10575-01)
- Oracle Database Security Guide 11g Release 2 (Part No. E10574-03)
- Oracle Database Administrator's Guide 11g Release 2 (Part No. E10595-05)
- Oracle Database Concepts 11g Release 2 (Part No. E10713-03)
- Oracle Database Advanced Application Developer's Guide 11g Release 2 (Part No. E10471-03)
- Oracle Database Net Services Administrator's Guide 11g Release 2 (Part No. E10836-01)
- Oracle Fusion Middleware Securing Oracle WebLogic Server 11g Release 1 (Part No. E13707-06)
- Oracle Fusion Middleware Configuring Log Files and Filtering Log Files for Oracle WebLogic Server 11g Release 1 (Part No. E13739-05)

CONTENTS

1 Introduction

A large amount of confidential information owned by healthcare business relations is stored in the OHI Back Office and OHI Business Intelligence databases. Examples of this are personal data and treatment history. This data must not only be correctly entered and stored but because of its confidential nature it must also be properly safeguarded.

Two developments have led to this document being written about Oracle Health Insurance security:

- The increasing use of OHI Back Office and OHI Business Intelligence in larger and more decentralized organizations where sight of both products' users is reduced.

- The modernization project where OHI Back Office functionality outside of the existing screen application can be used by means of custom interfaces and applications, as a result of which it is not always clear who sees and changes the data.

Security features on all fronts: organization (people), infrastructure (building, working spaces, hardware, networks) and systems (system programs, applications, interfaces) only works if the measures taken in the various sub-areas are mutually supporting. For example, there is no point in safeguarding a computer system if people can enter the building without a pass. Therefore, an integral security policy is required in order to implement coherent security across all sub-areas. As this is the responsibility of the client organization this component falls outside of the scope of this document.

This document deals with the security aspects that have a direct relationship with the administration and use of the OHI Back Office and OHI Business Intelligence applications and the Oracle components that are used for this.

Since OHI Business Intelligence is an option to OHI Back Office and it's Extraction-, Transformation- and Load steps are administered through OHI Back Office the main focus of this document will be on OHI Back Office. A separate paragraph is included that considers the security aspects of OHI Business Intelligence.

1.1 How this document is organized

Drawing up a security architecture begins with listing the 'assets'. An asset could be anything, ranging from a building or an employee to a software component. For this document we restrict ourselves to Oracle software components.

Every 'asset' in the security architecture should meet specific conditions. These conditions can be split into categories, also known as 'domains' or 'security services'. This document uses the term security services.

The following security services are distinguished:

- Identification/authentication/authorization
- Access control
- Data (transport) integrity & data (transport) confidentiality
- Non-repudiation
- Network segregation
- Malicious software control
- Service disruption prevention
- Security audit and alarms

- IT security management

The approach adopted in this document is to describe how the relevant security services are applied per Oracle component.

1.2 Relevant Security Services

Of these security services the following are of particular importance:

- *Identification/Authentication* is needed to determine the identification and authenticity of individuals or IT resources. In addition, *authorization* is needed to be able to assign rights or access to individuals or to IT resources. Examples of this are using (one-time) passwords, certificates, authentication servers (Kerberos for example), SSO (Single Sign On) and smartcards.

- *Access control* protects the access to IT-resources using access control mechanisms. This includes, for example, Windows NT access control, Oracle Corporation DBMS access control, UNIX access control, access control mechanisms based on Kerberos, etcetera.

- *Data (transport) integrity* guarantees the integrity of data during storage, transport and processing. Access control is a frequently used means of guaranteeing the integrity of data. Checksums, encryption, etcetera can also be used, for example.

- *Security audit and alarms* concerns logging and monitoring security related events, analyzing these events and generating alarms if this is considered necessary. Most IT-systems have features for event logging (which can usually be configured). On the other hand, network components often have limitations in the field of logging (generally due to limited storage capacity).

The following have not been examined:

- *Data (transport) confidentiality* guarantees the confidentiality of data during storage, transport and processing. This can be resolved using the same means as data (transport) integrity.

- *Non-repudiation* relates to guaranteeing that transmission and receipt of a message (transaction, e-mail, for example) cannot be repudiated by either the sender or the recipient. This service is not applicable for the core functionality within OHI Back Office. But it may apply to interfaces that are based on API usage or web service call-ins or call-outs.

- In order to be able to set up security in networks *network segregation/segmentation* is used to be able to address the specific security characteristics per segment (dial-in connection, firewall, internal network, vpn). This is the network administrator's responsibility.

- *Malicious software control* includes preventing and/or detecting software (components) that are capable of performing actions that are not foreseen in the formal design and specifications. Examples of this are computer viruses, bugs or 'backdoors' designed by programmers.

- *Service disruption prevention* is employed to guarantee continuity of data processing. In addition to physical security attention is given to the capacity and stability of the environment and measures are applied to improve this.

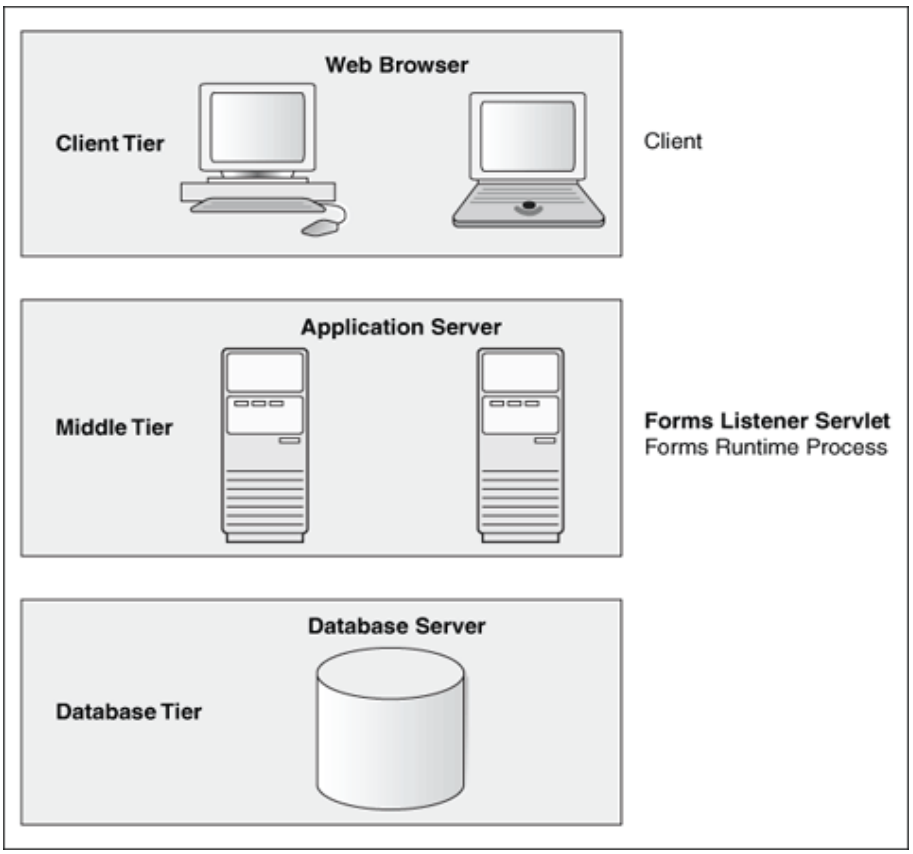
- Tools and techniques for managing the various security services are described under *IT Security Management.* Examples of this are, for example, setting up the administration organization using ITIL components, setting up

the security administration and security officer function, and also administration tools such as Oracle Enterprise Manager etcetera.

As mentioned above the implementation of security services on the infrastructure is the responsibility of the administration organization.

1.3 Relevant Oracle Components for OHI Back Office

A standard OHI Back Office implementation can be shown schematically as follows:



The following Oracle Components are currently used in a standard OHI Back Office implementation for OLTP and batch processing:

- Oracle 11g R2 database
- Net Services
- Oracle WebLogic application server
- Forms Server
- Batch Scheduler (including server side calling of SQL*Loader, SQL*Plus, Oracle Reports Runtime)

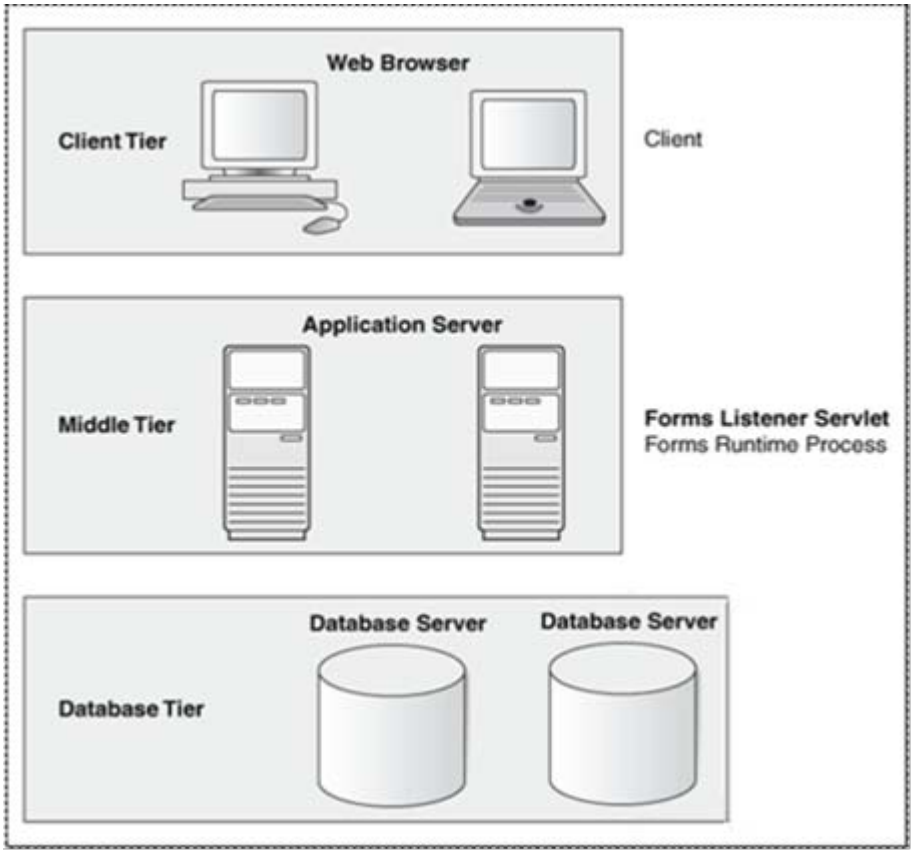
The components that are used for custom-made solutions are very much client-dependent. A possible summary, incomplete, follows:

- PL/SQL API's
- Web Services
- Client tools for desktop use such as SQL*Plus, SQL*Loader, PLSQL/Developer, SQL Developer, TOAD, Microsoft Excel

- Applications running under Oracle WebLogic. Example: web services developed for some OHI Back Office clients.

1.4 Relevant Oracle Components for OHI Business Intelligence

A standard OHI Business Intelligence implementation can be shown schematically as follows:



In viewing the schema above one should keep in mind that OHI Business Intelligence is installed as an option on top of OHI Back Office, so the only real extra component in this schema is the second database (server) on which the OHI Business Intelligence database including its Oracle Warehouse Builder runtime service runs.

The components that are used for custom-made solutions are very much client-dependent. A possible summary, incomplete, follows:

- Client tools for desktop use such as SQL*Plus, SQL*Loader, PLSQL/Developer, SQL Developer, TOAD, Microsoft Excel

- OLAP/Analytics tools such as Oracle Business Intelligence Enterprise Edition, SAP Business Objects, IBM Cognos.

2 Oracle 11g database (OHI Back Office)

All OHI Back Office and OHI Business Intelligence data is stored in a single schema in a single 'instance' of an Oracle 11g database. This means that optimal consistency of the data can be guaranteed. Ongoing developments in the field of authorization and integrity monitoring contribute to the consistency and security of the valuable data – our clients' working capital in fact - being effectively protected.

2.1.1 Developments relating to the OHI Back Office database

In the summer of 2002 the first steps were taken in extending and hardening the database part of the OHI Back Office application by introducing a 'robust business rule implementation layer'. The aim was to make the application more open for manipulation from sources other than the 'reference programs' or 'checked custom-work' developed by Oracle. This enables and eases the development of automated interfaces of whatever kind to increase efficiency within organisations.

To do this the business rules had to be implemented centrally in the database (previously they were spread and duplicated across the application code due to technical limitations). The result has been an increasingly 'Open' healthcare database that enforces the data always being handled in accordance with the business rules. This has also helped in being able to enforce and check the integrity of the data against the business rules.

Part of the implementation of the 'robustness layer' is an improved privileges structure, where the OHI Back Office user is only assigned privileges on the database objects after the application assigns them for the duration of the session. Database roles and custom code schema owners will also be assigned only limited privileges for realizing interfaces or custom functionality. After all, as the assigned privileges can only be used for changes where the data integrity is guaranteed, the requirement for listener settings or firewalls to deny users access to the database other than by the user interface has lapsed. This new privileges structure was realized in release 2003.03.

2.2 Identification/Authentication in OHI Back Office

The data that is stored in the OHI Back Office database can only be accessed using a valid username and password combination, which is defined in the database. When access control has been correctly arranged at operating system level this cannot be circumvented.

The identification/authentication of OHI Back Office users has been implemented as a two-step procedure:

- All users are created as Oracle database users so that they can login on the application screen. The creation of the user does not directly assign the user any rights on the application or the database!
- Moreover, the application administrator must set up every user as an 'OHI user' so that when the application starts the secure application database role OZG_ROL can be assigned dynamically as well, which means that the user is assigned the necessary database object privileges only for the duration of the session. The database role OZG_ROL has additional privileges for enabling and disabling business rules, needed for supporting specific user interface functionality. This makes it important to well protect the enabling of this role so it can only be used by authorized code components.

The database is accessed using the following accounts:

- Using a personal username and password.
- BATCH account (not necessarily named as such)
- OZG_BATCH account (not necessarily named as such)
- Sysoper and sysdba accounts (database accounts with these OS privileges)

2.2.1 Personal username and password

This includes all interactive users of the OHI Back Office application, as well as users of the APIs, custom application and individual users of client tools that access the database with their personal account.
For these users the database administrator creates accounts in the Oracle database, which can be used to login to the OHI Back Office database regardless of the tools used or the systems.

Risks / measures

- Risk: default passwords are not changed.
  Examples: SYS, SYSTEM, OZG_OWNER
  Action: expire passwords, setup a policy for enforcing a certain complexity for passwords and lock accounts during longer periods in which they are not needed.
- Risk: accounts created by the system (DBSNMP, OUTLN etc.) can be accessed.
  Action: lock these accounts unless they are actually used.
- Risk: accounts for employees who have left remain extant.
  Action: lock immediately and delete these accounts.
- Risk: passwords for OHI Back Office users remain unchanged, facilitating misuse by third parties.
  Action: define and implement a password policy. This includes expiring passwords after a period of time and implementing criteria (length, different to old passwords, use of a mix of numbers/characters, etc.) that must be met by new passwords to reduce the risk of guessing each other's passwords.
- Risk: accounts are used to connect to the database using a different environment than the standard user interface in an attempt to try to exploit the additional privileges as assigned to the OZG_ROL role.
  Action: use the Back Office Parameter that identifies the IP address(es) of the application server(s), which may enable the secure application role. This prevents users in trying to misuse their account by enabling the secure application role while not using the user interface.
- Risk: accounts that are used for custom code being defined also as an OHI user enabling such an account to enable the secure application role.
  Action: make sure these custom code accounts are not defined as an OHI user so they cannot enable the OZG_ROL role, limiting the custom code account to only the allowed role privileges assigned to custom code roles (OZG_ROL_DIRECT or likewise privileges directly assigned to the custom code account)
- Risk: custom code and other interface accounts that are configured once in an interface definition and normally will have no expiration policy because this can cause interface disruption. Typically passwords of these accounts are in some configuration file. Examples: special interface scripts, connection pools.
  Action: administer these accounts on a special list and change their password

on a regular basis to prevent misuse; also make sure the configuration files are stored with minimum access rights and use encryption in tooling whenever possible.

2.2.2 BATCH account

Used exclusively by OHI Back Office's batch scheduler and all batches that are not an Oracle Reports module.
This is a special database account for the UNIX user 'batch'. The operating system user can connect without a password by having access to a secure password store (a wallet) that stores the password in an encrypted format.

Risks / measures:

- Risk: individuals/accounts that can get hold of or access to the secure password store file can logon and gain OZG_ROL like privileges on the OHI Back Office objects.
  Action: use operating system access control to minimize access to the secure password store files; monitor use by user 'batch' and consider using another database account with a less obvious name.
  Action: implement a logon trigger for the database BATCH account so that only permitted servers are allowed to start a database session. This is described in detail in **doc[5]**.

- Risk: The default operating system password for the batch operating system account remains unchanged and use by third parties cannot be excluded.
  Action: set password expiry and complexity enforcement at operating system level.

2.2.3 OZG_BATCH account

Used exclusively by OHI Back Office's Oracle Reports modules.

This is a special database account for these modules because the underlying technology is not able to use the secure password store. The reports connect through a dedicated 'OHI guest' account with a known fixed password. The database account has no object privileges or roles received and needs to activate the secure application role OZG_ROL through a secured routine to get access to the OHI database objects.

Risks / measures:

- Risk: anyone who gets hold of this username can logon to the database and attempt to get access to the OHI objects by trying to enable the secure application role (at logon no privilege is present in contradiction to the BATCH account).
  Action: use the Back Office Parameter that identifies the IP addresses of the application servers, which may enable the secure application role. This prevents users in trying to misuse their account by enabling the secure application role while not using the user interface.
  Action: implement a logon trigger for the database OZG_BATCH account so that only permitted servers are allowed to start a database session. This is described in detail in **doc[5]**.

2.2.4 Risk: Sysoper and Sysdba accounts

Used exclusively by the 'Oracle' operating system user(s) on the database server for DBA activities. Please follow advice as given in the documentation for setting up your Oracle software environment on the server. Apply security measures for limiting access as much as possible, without compromising the management facilities.

Risks / measures:

- Risk: the operating system password for the user 'oracle' remains unchanged and use by third parties cannot be excluded.
  Action: set password expiry and complexity enforcement at operating system level.

- Risk: misuse by an administrator who abuses the access rights by viewing or changing data because of the extreme privileges assigned to these accounts.
  Action: implement audit logging on the OS and the database level, consider to use the option to redirect logs to a separate environment and combine these measures with a strong separation of duties. This will make sure illegal misuse of privileges will be logged while that logging cannot be influenced without noticing this.

2.3 Authorization

When a new account is created in the database it initially only has the right to logon (the CREATE SESSION privilege is required in order to do this). All rights to select or manipulate tables or to execute database procedures and packages must be explicitly assigned to the account. Granting privileges on OHI Back Office objects is done either directly or via a role (preferred option).

Up to and including release 2003.02 of the OHI Back Office application a single application role was used at database level to which all OHI Back Office database object privileges were assigned. By default users had the privileges of this powerful role (OZG_ROL).

With effect from 2003.03.0.0000 all users lost the OZG_ROL except the BATCH account (who effectively receives this role indirect). This role gives access rights on the objects used throughout the application (screens, reports, batches that perform changes, external integration media). The application users currently obtain the role dynamically, starting with release 2003.03, after security checks regarding access have been executed.

In special cases specialized users can still receive SELECT privileges (OZG_ROL_SELECT role) or the capability to execute DML operations on the 'rule protected' tables using the OZG_ROL_DIRECT role where the integrity is fully safeguarded in the database.

The role OZG_ROL may never be granted directly to any account in the database!

This will prevent unauthorized use as far as possible.

This topic is described in detail in **Doc[2]**.

2.4 Access control

Only users who have an account that was created for them can logon to the database.

The 'normal' users cannot access any objects outside of their own schema without the authorization of the schema-owner. See 2.2 for authorization.

2.5 Data (transport) integrity

The data integrity is safeguarded in a number of ways:

- The database files can only be accessed by the 'root' and 'oracle' operating system accounts. Administration of these accounts and access to the database files is the administration organization's responsibility.
  Obviously, only 'administrators' should have access to these accounts.

- Up to and including release 2002.04 the complex controls for changing data (the business rules) were primarily implemented in the application programs. As part of the technical 'make-over' of the server component of the application all checks (business rules) were fully transferred to the database. The application programs still only perform these checks partially for reasons of efficiency or user-friendliness. Making the OHI Back Office database 'more robust' has been spread across a series of releases due to the significant effort that this entailed.

2.6 Security audit and alarms

The features for monitoring and logging have been implemented at two levels in the OHI Back Office database:

- By default the Oracle database has many features for configuring database auditing. Information on this can be found in chapter 17 of the 'Oracle Database Concepts 11g Release 2' manual.

- The OHI Back Office application provides a change-logging mechanism for the 'business rule enabled' tables. By default the user and time of creation and last change are stored within each functional record. A setting can be activated per table to implement a more detailed form of change–logging. Depending on the requirements and the application features desired this would make it possible to trace each change in detail including the order and the common transaction in which these changes were implemented.

Additional events and alarm signalling can be realized on this basis.

3 Net Services

'Net Services' is the name of Oracle's connectivity software, which was previously known as SQL*Net.

Now there are only 2 ways to access the database using a username/password combination:

- by direct inter-process communication on the database server machine itself. In this case you must have been granted access to the software concerned at operating system level.
- via an Oracle database listener process which can be used to create a connection to a database.

The first option requires the use of the same 'software tree' as the one in which the database is running and is therefore generally only used for administrative purposes (starting, closing down and securing data in the database).

The second option enables the user to access the Oracle database from any computer that has network access to the database server(s) where the database resides on. One can only connect to the database if the correct logon details are available. In doing this the Net Services component is only a transparent medium between the user and the database.

3.1 Identification/Authentication

Not applicable to users.

The listener can be configured in such a way that a password is requested on a call to the UNIX server. This will prevent the Net Services being stopped / started without authorization.

3.2 Authorization

Not applicable, after all, the user always logs on to the database.

3.3 Access control

Access to the listener can be screened by, for instance, firewalls or by the configuration options in the listener and this can be used, for example, to guarantee that only certain machines can create a connection to the database using the listener. See the Net Services documentation also for the features that are provided by the Oracle programs in this area.

3.4 Data (transport) integrity

The traffic between the client and the TNS listener can be encrypted. The TNS listener configuration has to be changed to achieve this.

3.5 Security audit and alarms

There are various options for logging the use of Net Services at operating system level. In doing so, logging can be configured on both the client side and on the server side.

4 Oracle WebLogic Server application server

The Oracle WebLogic Server application server comprises many components; the following being relevant to the OHI Back Office application.

- Oracle HTTP Server (based on Apache)
- Forms Server (discussed in Chapter 5)

The functions of the HTTP Server are:

- Handling HTTP requests for static HTML.
  Examples of this are static HTML pages, images and JAR files.
- Pass on requests for the forms server listener servlet.

4.1 Identification/Authentication

There are a number of ways to configure the Apache HTTP server to perform identification / authentication. These are not used currently.

4.2 Authorization

The Apache HTTP server can be extensively configured to grant access to resources.

Examples here are:

- The option to display directory listings.
- The option to view files in a directory.
- The option to call modules (e.g. PERL, CGI, etc.).

Doc[5] describes the way in which the HTTP server should be configured for use with OHI Back Office and in doing so restrictions are already applied in the authorization structure.

4.3 Access control

The Apache Server can be configured to grant access to specifically named servers / IP addresses.

By default no restrictions are applied to access to the Apache server. It is worth considering restrictions in larger organizations.

4.4 Data (transport) integrity

The integrity and confidentiality of data can be improved by using encryption. To do this requests are routed through the SSL port. The drawback is the increased server load as encryption is paired with a great deal of processing work.
This mechanism is not used by default. Please note that implementing SSL does not entail changes to the OHI Back Office programs.

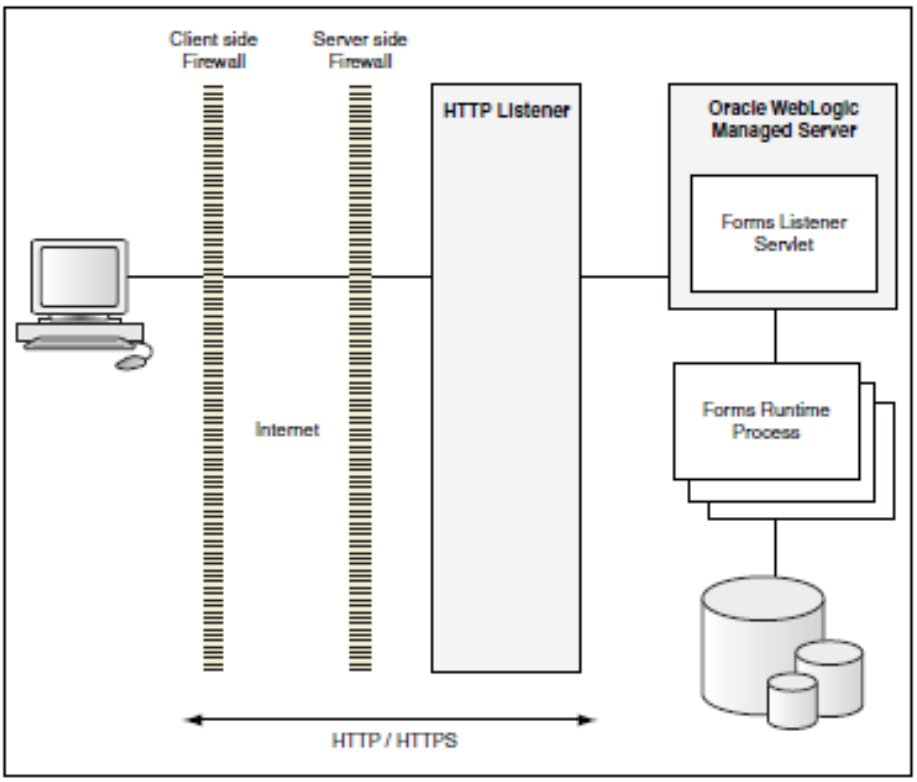
4.5 Security audit and alarms

The Apache listener has extensive and easily configurable features for logging administration activity and HTTP requests from users.

5 Forms Server

The screens of the OHI Back Office application are supplied as 'web forms' using the Oracle WebLogic Server application server for running the Forms/Reports/Discover/Portal Edition – the security aspects of this are described in chapter 4.
The forms server listener is central to the implementation. This ensures that a forms runtime process is started for each new session. The forms server listener is implemented as a so-called forms server listener servlet.
In web forms the presentation and control is via a Java applet that is executed by a Java plug-in in the browser. This plug-in communicates with the dedicated forms runtime process through HTTP(S).

Schematic this looks like:



The database tier and application tier can both physically run on a single machine (for demonstration purposes all three tiers can be enabled to run on even a notebook).
The architecture above makes it clear that there is no need for direct communications from desktops with the database tier or the database (listener), which provides simple options to have the database listener process being accessed from the application tier only (see the 'Net Services' chapter also).

5.1 Identification/Authentication

The user identifies as soon as the forms runtime session is started. The opening screen (with the menu) is only displayed after this identification has been executed successfully.

5.2 Authorization

Authorization is arranged as follows:

- The user identifies itself using a username/password combination.
- After logging on the user is assigned a database role dynamically.

- After logging on the application menu is created dynamically for the user.
- In specific cases the information that is shown to the user is determined dynamically when the information is displayed based on the authorization.

5.3 Access control

In addition to the mandatory logon from within the application access to the forms application can be set at application level in Apache (see chapter 4).

To prevent SQL injection, which can be used by highly skilled users to get access to unauthorized data, it is advised to define the environment variable FORMS_RESTRICT_ENTER_QUERY in the forms environment settings file and assign it the value TRUE:

FORMS_RESTRICT_ENTER_QUERY=TRUE

For more information regarding the environment settings file please look for the 'envFile' string in the 'OHI Installation, Configuration and DBA manual'.

5.4 Data (transport) integrity

Sending a HTTP request to the Apache server starts a session. There is no standard encryption when this is done (although HTTPS can be setup). As no sensitive information (such as username/password for example) is sent with this there is, therefore, no direct security risk.

As soon as a session has been started the applet (started from within the browser) communicates with the forms server listener.

By default message encryption is not used. All that needs to be done to activate encryption of the forms messages is to set environment setting FORMS_MESSAGE_ENCRYPTION to 'true'.

In a previous version a socket mode as well as HTTP mode communication was possible. Nowadays only HTTP mode is supported. This has the following advantages:

- Additional security features from firewall/proxy. This means that forms can also be used outside of your own network.
- Option to use HTTPS (secure HTTP). Encryption of HTTP/HTML traffic and better features to protect against unauthorized use by means of certificates. The result of this is a CPU load that may be so heavy that the purchase of separate encryption-hardware is recommended.

5.5 Security audit and alarms

By default the forms server listener logs all requests for new sessions in $MW_BASE/user_projects/domains/frs_d1/servers/WLS_FORMS/logs/access.log. Here you can view the moments in time a forms session is started and the IP address used.

The HTTP requests from the browser for starting the forms executable, retrieving the Java classes, images, etc. are logged in $ORACLE_INSTANCE/diagnostics/logs/OHS/ohs1/access_log.

6 Batch Scheduler

An important part of the OHI Back Office functionality is performed using batch background processes. Examples here are: reports using Reports Runtime, scripts for reading-in (using SQL*Loader) and processing external integration media or running change (processing) batch runs (using SQL*Plus), (XML) file producing batches which may also change the database, web services consuming batches.

The OHI Back Office Batch Scheduler has been developed to start this kind of background processes and monitor whether they are running.

The batch scheduler (a Pro*C program) runs on the application server using a separate BATCH database account for which the password is stored in a secure external password store. In doing so there is no requirement to know the requesting user's password for the processing programs to be started.

By default this special BATCH account is named 'BATCH' with historically seen (but not necessarily) a similar named UNIX account.

6.1 Identification/Authentication

Identification/authentication is at two levels:

- The OS user logs-in as UNIX user 'batch' using a password to be set by the administrator.
- The Oracle user BATCH logs in using '/@<batch_alias>' and is identified as BATCH if he can logon through the alias which identifies the password to use as stored encrypted in the secure application store. More details for setting this up are described in **doc[5]**.

Risk: OS accounts/individuals with access to (a copy of) the secure password store can use the legitimate account and gain rights on the OHI Back Office objects. For this it is important to limit access on the file system to the secure password store files. Next to that a logon trigger that rejects sessions from unauthorized systems such as desktop users can be implemented to prevent this.

The implementation of a logon trigger is described in **doc[5]**.

6.2 Authorization

Authorization is at two levels:

- OS level:
  The user batch has hardly any application runtime files of his/her own, but can, however, start the batch scheduler. And all produced output and log files are stored with the batch user as owning OS account.
- Database level:
  The OZG_ROL is assigned (indirectly) to the Oracle user BATCH and this role has far-reaching privileges on the application and tables.

6.3 Access control

You can only logon as UNIX user 'batch' if you have the correct password or if you have super user (root) rights.

Risks / measures:

- Administrators can log-in as user 'batch'.
  Action: ensure that as few people as possible can log on to the OHI Back Office system as root. In addition, pay close attention to files such as .rhosts, .host.equiv etc. in the home directory of batch which could be used to log on from other systems.

6.4 Data (transport) integrity

The 'data' that is used/produced by the batch scheduler comprises:

- Batch scheduler process log, stored in $OZG_LOG/batch_$ORACLE_SID_<server>.log.
  This file contains in normal mode no sensitive information apart from the times at which certain jobs have been run. When 'verbose' mode is used the complete start command is logged per job. For Oracle Reports jobs this includes the name of the database account that is used to logon.

- Files that are created by the batch scheduler are stored in a user-specific directory structure (namely $OZG_OUT/<user> and/or $OZG_LOG/<user> where <user> is the username of the requesting database user acount).
  OHI Back Office users can, in principle, submit a request to browse the output of the batch scheduler (jobs started by others if necessary) using the script. It is up to the administration organization to set up the HTTP server/OS privileges in such a way that only the owner of the batch scheduler output can retrieve it.

- Status information in table ALG_SCRIPT_AANVRAGEN (script requests) and related tables.
  Integrity is controlled by the database.

Risks / measures:

- Users can use the script requests to view output of any confidential process (namely, via the forms built-in *show_document* or directly via URLs).
  Actions:
  Set up authorizations on batch output (described in **doc[4]**) and purge output directories regularly (for example, by setting up a cron job that deletes all output files older than 2 day a number of times each day).

6.5 Security audit and alarms

The UNIX 'last batch' command can be used to find out who has been logged-in as 'batch' in the last x days. An example of the output is shown below:

```
batch    pts/1      vlan441dhcp473.n Tue Sep  2 07:24 - 07:31  (00:06)
batch    pts/16                      Mon Sep  1 16:47   still logged in
batch    pts/18                      Mon Sep  1 12:08 - 18:23  (06:15)
batch    pts/16                      Mon Sep  1 10:01 - 16:47  (06:45)
batch    pts/1      vlan441dhcp484.n Fri Aug 29 15:18 - 16:00  (00:42)
batch    pts/14                      Fri Aug 29 12:03   still logged in
batch    pts/8                       Fri Aug 29 11:59   still logged in
batch    pts/1      vlan441dhcp223.n Thu Aug 28 18:46 - 18:52  (00:05)
batch    pts/8                       Thu Aug 28 16:51 - 17:14  (00:23)
batch    pts/18                      Thu Aug 28 15:47 - down   (04:27)
```

Please note that most UNIX systems can be configured such that the log involved can be purged periodically.

In addition, database auditing can, if required, be used to monitor the behaviour of the database user 'BATCH'.

7 APIs

The adding of the 'robust business rule layer' of OHI Back Office (where, amongst other things, the business logic has been transferred from the screens to the database) has created the possibility of creating API packages on top of this layer.
The APIs that have been developed are database PL/SQL routines that can only be used by logged-in database users with sufficient authorization.
The most likely use of the APIs is to have the API calls execute under a generic account. This account is then assigned the necessary rights by means of a standard grant privilege routine.
In order to subsequently determine the origin of changes the source id (system identification, should be known in table ALG_BRONNEN) and the OHI officer username (must be known within the OHI Back Office application) must be specified.
Please note that almost all but not all parts of the database provide this rule layer implementation.

You can read more about APIs in **doc[3]**. This document references the 'classical' API routines. Currently there is a small but growing set of plsql routines which service the 'service layer' web services. The plsql services are the implementation of this functionality.

7.1 Identification/Authentication

At two levels:

- Database user (password required)
- OHI officer (no password required)

7.2 Authorization

In order to execute DML statements in the database the Oracle database user must obtain the correct authorization for using the APIs. The OZG_ROL_SELECT and OZG_ROL_DIRECT are needed for this. The role OZG_ROL may not be used!
This is covered in more detail in **doc[2]**.

7.3 Access control

API packages are used via a Net Services connection, i.e. in principle you can logon as an API user from any computer in the network. See 'access control' in chapter 3 also.

7.4 Data (transport) integrity

The API's that have been developed are based exclusively on the parts of the OHI Back Office database that are contained in a robust business rule layer. This means that the integrity of the data is automatically safeguarded.

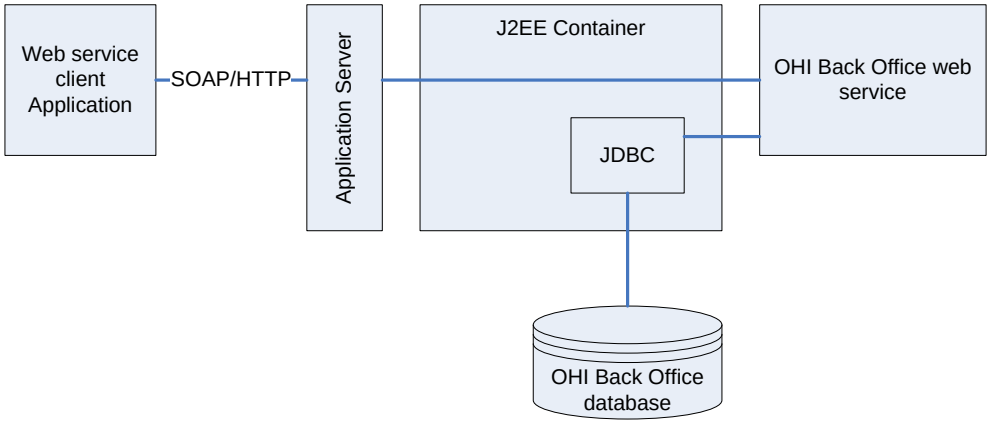
7.5 Security audit and alarms

In the vast majority of cases API packages will be used using a generic API account. If the API packages are used as an interface between an external front-office system and the OHI Back Office database this creates a risk that it will subsequently be impossible to trace who in the client organization was responsible for each change. An example of this is the possibility that individual users could input incorrect data due to inexperience.
This is why we always recommend that an originating source is specified. Defining a specific officer username and source system can do this. Where there are larger numbers of external users it is recommended that an officer is appointed either for

each person or for every organizational unit (e.g. branch).
Please see chapter 2 for additional information.

8 Web Services

Java web services that manipulate/select the application data have been developed. The web services implement a technical interface for at this moment a limited set of operations on primary policy and claims data. The application architecture for synchronous use is outlined below:



As can be seen from the diagram above the web services connect to the database via JDBC and execute the necessary (PL/)SQL calls. Please note that the 'web service' client in the diagram does not represent a user with a browser, but a remote application or a component in the middleware.

Currently (2012) there are 2 types of web services. The longer existing web services are historically offered as 'OHI Connect to BackOffice'. They will be indicated as the 'classic' web services. The newer web services, which also are part of the OHI Connect to BackOffice product option, are indicated as the 'service layer'. In the long run only the last type of services will remain.

8.1 Identification/Authentication

8.1.1 Server-side Access

The web service creates a database connection to the database.

For the 'classic' services it is recommended that an existing personal account is not used here but that a user is created specifically, to which the OZG_ROL_SELECT / OZG_ROL_DIRECT roles are assigned.

For the newer web services it is required to have a new separate database account setup and grant only a subset of object privileges to this account. This account with a limited set of privileges is used for connections.

The standard connection pooling mechanism in WebLogic is used for the connection, which means that the connect-string (including username/password) has been defined in the data-sources.

Risks / measures:

- Connect-string defined in configuration file. This information is, in principle, only available to administrators. It is stored encrypted.
  Actions: use password expiry and change the password frequently, install WebLogic under an administrator-account ('oracle' for example), implement the production platform for WebLogic as a separate (UNIX) server (separate from user and acceptance test servers).

8.1.2 Client-side Access

A web services client does not need to give a username/password when calling the web service. Implementing a security solution is the task of specialized products for this goal and by externalizing the security an organisation is free in choosing the preferred solution. This means it is sufficient to specify a valid web service request to execute the request as long as the network and infrastructure do not prevent this.

Risks / measures:

- Too many users can call services. This entails the risk of misuse.
  Actions:
  1) apply (network) access control so that authorized servers can only call the web services.
  2) for the 'service layer' services you can implement http header security/authentication; this option is supported by WebLogic; be sure to use ssl (https) for service connections
  3) use specialized tooling like for example Oracle Web Services Manager or a different security solution

8.2 Authorization

8.2.1 Server-side access

The comments under 7.2 apply to this section as the web services are a similar layer as the APIs.

8.2.2 Client-side access

Currently the web services are by default not protected with a username/password. See the comments on access control.

8.3 Access control

The WebLogic application server provides the option to make services available to a select group of systems.
Applying access control to the calling servers likewise (which themselves function as a client of the web service) makes the risk of unauthorized use easily manageable.

As indicated in the 'client-side access' paragraph the new services can be enabled with http header authentication to prevent unauthorized access.

For more information, please read the instructions regarding the use of connection filters within WebLogic as present in the WebLogic Server documentation.

8.4 Data (transport) integrity

HTTP is used for data-transport between client and service. HTTPS can be used if necessary (for use over the Internet for example) – this does, however, result in a performance penalty.

8.5 Security audit and alarms

The WebLogic server offers extensive functionality for logging and auditing purposes. The standard documentation contains more information.

9 Other Client Tools

Where the OHI Back Office application has been made more robust data can be changed in the database using manipulating SQL (DML) or APIs. The robustness layer ensures that the data remains consistent and continues to comply with the built-in business rules. The technology employed to access the database (Net Services, ODBC, JDBC, etc.) is irrelevant in this case.

9.1 Identification/Authentication

By means of the database user's username/password.

Risks / measures:

- There is a risk of scripts / registry settings existing on client machines when using certain tools where the username/password for the connection to OHI Back Office can have been saved.
  Actions: apply password expiry. Identify scripts / settings with username/password. If necessary and possible restrict the use of saved username/password combinations using a central configuration file.

9.2 Authorization

Authorization is transparent as the client tool user has to logon as a database user. The risk of users obtaining improper rights on database objects is not an issue provided that the correct database privileges have been set and the schema owner's passwords are not known. See the comments in chapter 2 also.

9.3 Access control

The overwhelming majority of PC's already have ODBC software installed that facilitates access to the OHI Back Office database from MS-Excel, for example. Locking client tools is impractical. On the other hand, it is recommended that an investigation is carried out into whether the access control features of Net Services can be used to prevent unwanted access by client tools. If necessary, consideration should be given to setting up a (Net Services) proxy.

9.4 Data (transport) integrity

Enforced encryption should be considered (see the comments in 3.4).

9.5 Security audit and alarms

The use of client tools itself cannot be logged. However, the Net Services log can be used to discover when users were logged in. The V$SESSION view can be used to take a snapshot which shows who is logged in, using which IP address, which tool, etc.

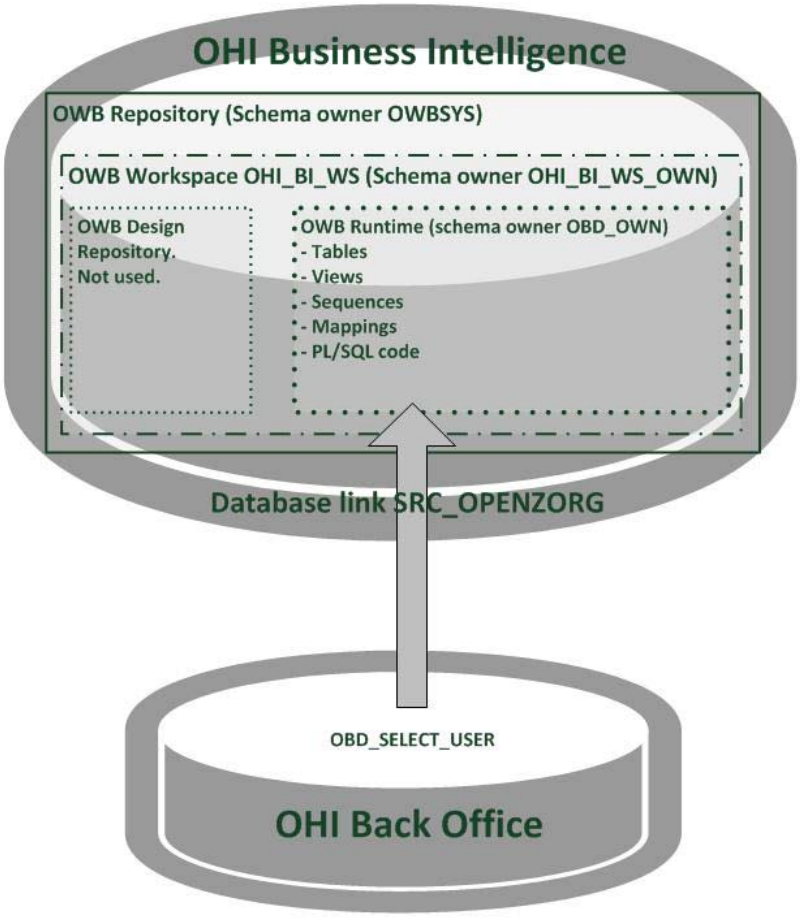
Moreover, using database auditing makes it possible to audit Oracle sessions.

10 Oracle 11g database (OHI Business Intelligence)

All OHI Back Office and OHI Business Intelligence data is stored in a single schema in a single 'instance' of an Oracle 11g database. This means that optimal consistency of the data can be guaranteed. Ongoing developments in the field of authorization and integrity monitoring contribute to the consistency and security of the valuable data – our clients' working capital in fact - being effectively protected.

10.1.1 OHI Business Intelligence architecture

OHI Business Intelligence is created using Oracle Warehouse Builder (OWB). When OWB is installed the OWBSYS schema is created, as well as the OWBSYS_AUDIT schema.
Since OWB can be used to manage multiple datawarehouses by creating separate workspaces for these datawarehouses a separate workspace needs to be created for OHI Business Intelligence. The workspace OHI_BI_WS is used with its owner OHI_BI_WS_OWN.
Inside this workspace there is a Design component and a Runtime component. The Design component is not used since this would normally be the part where the OHI Business Intelligence objects are created.
The specific objects are deployed to the runtime component and are owned by the application owner. In the example below this user is called OBD_OWN, the preferred user name for this owner. The OWB runtime component keeps track of all the OHI Business Intelligence objects. For instance it registers how many rows are inserted, updated, merged and deleted per mapping per batchrun and it can also be used to monitor running (E)xtraction, (T)ransformation and (L)oad processes.



These batches are started using the OHI Back Office batch scheduler. Only the Extraction batch job pulls OHI Back Office data into OHI Business Intelligence over

the src_openzorg database link which points to the OBD_SELECT_USER in the OHI Back Office database. At the end of every E, T or L run checks are in place to confirm that everything is loaded correctly. These checks also make use of the src_openzorg database link.

10.2 Identification/Authentication in OHI Business Intelligence

OHI Business Intelligence has 4 specific users in the OHI Business Intelligence database which are:

- OWBSYS – The OWBSYS schema contains all Warehouse Builder repository metadata. The following grants need to be issued by this user:
    - grant select on wb_rt_service_nodes to ohi_bi_ws_own;
    - grant select on wb_rt_audit to <application owner>;
    - grant select on wb_rt_errors to <application owner>;
    - grant select on wb_rt_warehouse_objects to <application owner>;
    - grant execute on wb_workspace_management to <application owner>;
- OWBSYS_AUDIT – The OWBSYS_AUDIT schema is used by the Warehouse Builder Runtime (Control Center) Agent to access the heterogeneous execution audit tables of the OWBSYS schema.
- OHI_BI_WS_OWN – The owner of the OHI Business Intelligence Workspace (OHI_BI_WS).
- <application owner> (preferably OBD_OWN) – The owner of the runtime objects.
- SYSOPER and SYSDBA accounts

The data in the OWBSYS, OWBSYS_AUDIT schema's is specific to Oracle Warehouse Builder and can only be accessed using the schema owner.

The data in OHI_BI_WS_OWN schema can be accessed by the schema owner. It can also be accessed by the <application owner> if the correct workspace has been set using the OWBSYS.wb_workspace_management package.

The data in the <application owner> schema is only accessible to the application owner. It is up to the customer on how this data is protected.

To be able to read data from OHI Back Office a database link (SRC_OPENZORG) is created in the OHI Business Intelligence database that connects to the OBD_SELECT_USER in the OHI Back Office database. This user only has specific privileges as mentioned in the OHI Business Intelligence Administrator reference (**Doc[6])** .

10.2.1 Risks / Measures

This includes all the aforementioned accounts, as well as custom application and individual users of client tools that access the database with their personal account. For these users the database administrator creates accounts in the Oracle database, which can be used to login to the OHI Business Intelligence database regardless of the tools used or the systems.

Risks / measures

- Risk: default passwords are not changed.
  Examples: SYS, SYSTEM, OBD_OWN
  Action: expire passwords, setup a policy for enforcing a certain complexity for passwords and lock accounts during longer periods in which they are not needed.

- Risk: accounts created by the system (DBSNMP, OUTLN etc.) can be accessed.
  Action: lock these accounts unless they are actually used.

- Risk: passwords for OHI Business Intelligence users remain unchanged, facilitating misuse by third parties.
  Action: define and implement a password policy. This includes expiring passwords after a period of time and implementing criteria (length, different to old passwords, use of a mix of numbers/characters, etc.) that must be met by new passwords to reduce the risk of guessing each other's passwords.

- Risk: custom code and other interface accounts that are configured once in an interface definition and normally will have no expiration policy because this can cause interface disruption. Typically passwords of these accounts are in some configuration file. Examples: special interface scripts, connection pools.
  Action: administer these accounts on a special list and change their password on a regular basis to prevent misuse; also make sure the configuration files are stored with minimum access rights and use encryption in tooling whenever possible.

10.3 Authorization

Authorization is transparent as the client tool user has to logon as a database user. The risk of users obtaining improper rights on database objects is not an issue provided that the correct database privileges have been set and the schema owner's passwords are not known.

10.4 Access control

Only users who have an account that was created for them can logon to the database.

The 'normal' users cannot access any objects outside of their own schema without the authorization of the schema-owner.

10.5 Data (transport) integrity

The data integrity is safeguarded in a number of ways:

- The database files can only be accessed by the 'root' and 'oracle' operating system accounts. Administration of these accounts and access to the database files is the administration organization's responsibility.
  Obviously, only 'administrators' should have access to these accounts.

- On several points in the loading process of OHI Business Intelligence there are load-checks to make sure the correct data is loaded into the datawarehouse.

10.6 Security audit and alarms

The features for monitoring and logging have been implemented at two levels in the OHI Business Intelligence database:

- By default the Oracle database has many features for configuring database auditing. Information on this can be found in chapter 17 of the 'Oracle Database Concepts 11g Release 2' manual.

- Every run (Extraction, Transform and Load) has a specific AUDIT_ID attached to it. This AUDIT_ID can be used to identify what went wrong in which specific batch on what specific moment in time. Special views and tables are created to maintain a good functional view of what is happening inside the OHI Business Intelligence database.

Additional events and alarm signalling can be realized on this basis.

11 Appendix A – Tips

A number of practical tips are given below.

- Apply password expiry to Oracle accounts and UNIX accounts.
- Ensure that every PC is fitted with a password deactivated screen saver so that the PC locks itself after 5 or 10 minutes of inactivity.
- Ensure that terminals/consoles in the system room are on 'login' by default.
- Minimize physical and network access to the UNIX servers.
- Ensure that administrators cannot logon as 'root' or 'batch' as a matter of principle.
- Make a list of all configuration files where Oracle username/password combinations are defined.
  In general, except for WebLogic, the wallet and earlier named exceptions, they do not exist in the standard application, but can exist in custom components.
- Implement Oracle recommended Critical Patch Update patches (CPU patches).
- List all custom and administration scripts that use Oracle username/password combinations.
  Minimize their use.
- Have an active policy for protection of your workstations by checking and/or enforcing security software to run on it and having recent patches applied.
- Check the system for report-output containing confidential information that can be accessed by unauthorized individuals. Delete report-output with obsolete information.

The My Oracle Support site (http://support.oracle.com) can be used to stay up to date on new developments in relation to security. Unexpected 'security leaks' can also be reported here. For OHI Back Office specific matters administrators will be informed of relevant developments in this area via the Support site or by the technical team at OHI Back Office Development.