Oracle® Transportation Management

Security Guide

Release 6.3

Part No. E38419-09

April 2015

Oracle Transportation Management Security Guide, Release 6.3

Part No. E38419-09

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Contents

Send Us Your Comments

Oracle Transportation Management Administration Guide, Release 6.3

Part No. E38419-09

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: otm-doc_us@oracle.com

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, contact Support at https://support.oracle.com or find the Support phone number for your region at http://www.oracle.com/support/contact.html.

Preface

This document contains recommendations on how to make your software installation more secure. All of these recommendations should be evaluated carefully and implemented based on your own unique needs and the dictates of your own internal security procedures and guidelines. This guide applies generally to Oracle Transportation Management, Global Trade Management, Fusion Transportation Intelligence and Oracle In-Memory Logistics Command Center. There are additional Logistics Command Center-specific security considerations detailed in the "Security" section of the Logistics Command Center Administration Guide.

Terminology

Term	**Definition**
Machine	The physical (or virtual) server
Web Tier	The web server (OHS and Tomcat)
Application Tier	The application server (WebLogic)
Web Tier Instance/Application Tier Instance	A specific instance in the Web or Application Tier
Applications	The applications covered by this Guide – Oracle Transportation Management (OTM), Global Trade Management (GTM), and Fusion Transportation Intelligence (FTI).
Properties	Customizable settings that require modification to the glog.properties configuration file or a data-driven property set. See *Section 4 – Advanced Configuration: Custom Properties* of the Administration Guide for more information.

Change History

Date	**Document Revision**	**Summary of Changes**
12/2012	-01	Initial release.

Date	Document Revision	Summary of Changes
03/2013	-02	Added additional SSO properties. Added a note referencing the Configuring FTI with OAM (SSO) section in the OTM Integration Guide. In the Command-line Utilities section, this note has been added: These utilities are provided as-is and may be removed at any time. Removed several command line utilities that are no longer supported. Added section about disabling manual user login from login page. Added older version default users, and a user that was missed. Made it clearer that the change password instructions given were for system and guest users.
06/2013	-03	New preface.
08/2013	-04	Updated OBIEE section: The Business Intelligence Role field on the OTM User Manager controls the role a user will have within OBIEE/FTI. Since, these user accounts are shared by OTM and OBIEE/FTI any change to the Business Intelligence Role field in the OTM User Manager will require a restart of the OBIEE server for the changes to be effective within OBIEE/FTI. GL_USER EFFECTIVE_DATE and EXPIRATION_DATE OTM Tomcat Servlet Container User ID and password property and system property additions. Corrected the system and guest change password instructions. Updated section that mentions Business Intelligence Role field to use new Business Intelligence Application and Business Intelligence Role grids. LDAP configuration information was added to the Security Guide Forbidden characters for application user passwords.
12/2013	-05	Updated Object Lock Cleanup process definition. Log-in history updates. Added Oracle HTTP Server & Tomcat Integration section.

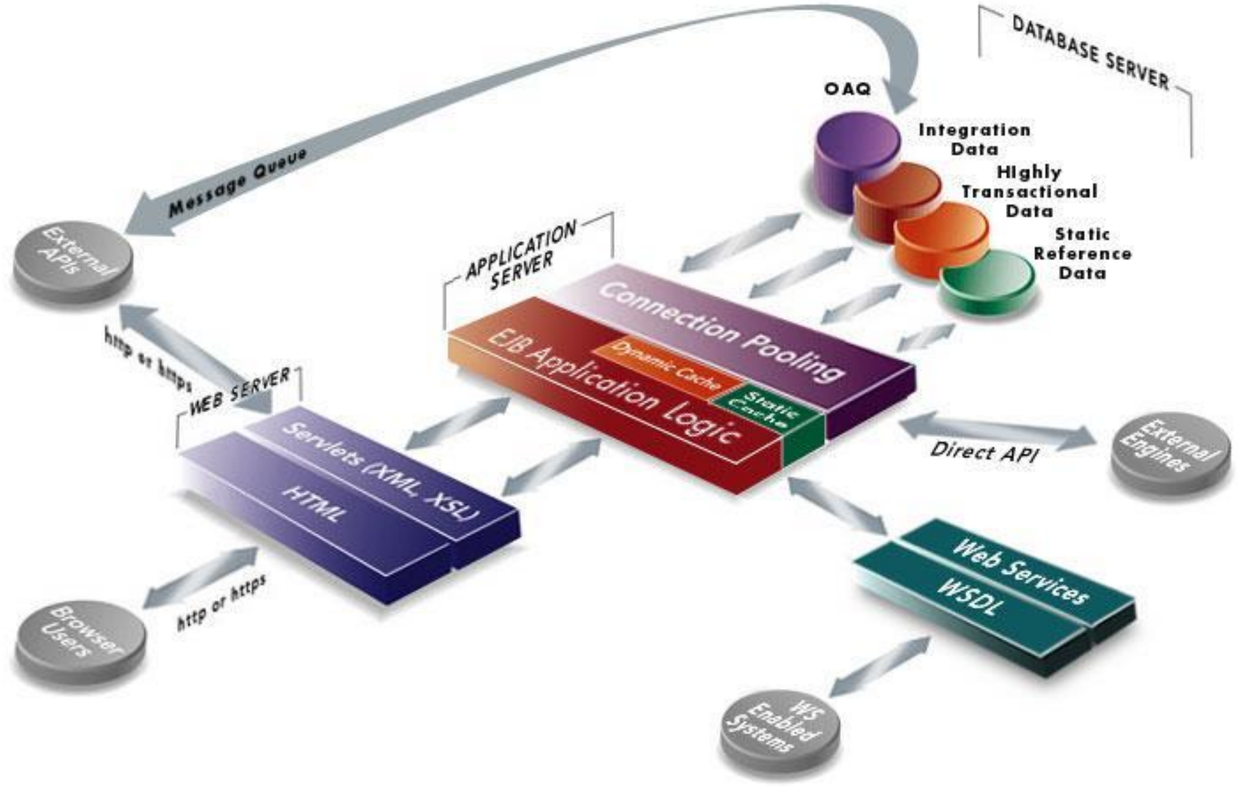
Date	Document Revision	Summary of Changes
05/2014	-06	Added new WSS Policy template to disable Web Service Security. Corrected example path for custom policies. Added Disabling Manual Login for ADMIN Application Users section. Added information on how property sets work. Added section: Securing Command Line Tools
08/2014	-07	Revised discussion of external BI Publisher farms. Added the new Database user: dir_xml_user Removed 2 options from User Access section. Removed references to XSQL, now unused installation file. Added namespace to webservice sample security token XML Added Encrypting the Signed Servlet Password Stored in tomcat-users.xml section. 6.3.5 Review Changes
01/2015	-08	Added new users for Cloud implementations, the DBA.OPS user. Explained concepts of Reserved Users and Reserved User Roles. Added information about sending attachments via email, and for external virus checking. Added section: Content Use Cases in OTM Added section: Disabling Virus Checking Fixed path for the web.xml file for OTM Added information about new INTEGRATION User Role. Add information about INTEGRATION and External Integration Access Control Lists 6.3.6 Review Changes and additions

Date	**Document Revision**	**Summary of Changes**
3/2015	-09	Added section: Diagnosing LDAP Communication Added section: Custom Action Control Added section: OTM Web Services Control Added: glog.security.sso.loginBackdoorName Added note about Integration Stack Trace control. Added section about Installing a Trusted SSL Certificates in the OTM Application Server

1. Overview

Architectural Overview

Oracle Transportation Management (OTM), Global Trade Management, (GTM) and Fusion Transportation Intelligence (FTI) are complex, multi-tier applications that share a common infrastructure.



The main components are the Application tier (backed by Oracle WebLogic), the Web tier (backed by Oracle HTTP Server and Tomcat), and the Database tier (backed by Oracle RDBMS). FTI, used by both OTM and GTM, sits on top of Oracle Business Intelligence, Enterprise Edition (OBIEE). There are many touch points, both within these layers and between them and the outside world. All of these pieces must take security into account in order to protect the data within the system and the operational integrity of the system itself.

The security model for such a system is many-layered and oftentimes components of it are built on top of lower level components, the whole of which provides a bulwark against today's IT dangers, both deliberate and unintentional. For example, first the physical machines are secured, then the operating system is secured, then the base components (e.g. WebLogic or the RDBMS) are secured individually, and then they are secured in relation to one another. The applications themselves are secured, and any configuration and log files are secured via the operating system, then each entry into and out of the application are secured, etc.

General Security Principles

Overall Goals of Security

There are two main thrusts to securing your systems: preventing unauthorized access, and keeping the system up and running. Both are important aspects to consider, and both can be compromised by both deliberate acts and accidental failures.

Preventing unauthorized access consists of the following broad pieces:

- **Authentication**: is the person or process that is attempting to access the system who or what they say they are?
- **Authorization**: is the person or process allowed to be doing what they are attempting to do?
- **Data Access**: is the person or process restricted in what data it/they can access?
- **Auditing**: is there a way to tell that some aspect of security has been compromised?

The chapters ahead will take a look at each of these pieces in depth.

Ensuring that the application stays up and running is vitally important, of course, and is therefore an essential part of security. Deliberate attempts to bring a system down are called Denial of Service attacks, and the base components along with the application itself are configured by default to guard against these attacks. Performance problems can also bring a system down, which has the same effect as someone maliciously targeting the system, so this document will on occasion point out ways in which performance can be affected.

Finally, there are security issues that do not fall cleanly into either of these broad categories, but they will be talked about and addressed as well.

General Principals

The following principals are fundamental to any software security plan.

Keep Software Up-to-Date

One of the foundations of good security practice is to keep all software versions and patches up-to-date. Critical Patch Updates (CPUs) should always be applied whenever they become available. Typically Oracle releases these four times a year. These should be applied to keep your system as secure as possible. It is **highly recommended to apply any Oracle Critical Patch Update** that is released. Additionally, later versions of the software will usually include performance and stability fixes, and should be evaluated periodically.

Minimize the Attack Surface

Presenting a small profile to a potential assailant will lessen the opportunities they have to attack the application. This can be accomplished by turning off unneeded services or by simply limiting access via the network using something like a firewall. In the first case, if a service is not running, newly discovered vulnerabilities in it cannot be exploited. Similarly, if a machine is behind a firewall then most avenues of attack will be cut off just because the network ports are unreachable, requiring all access to be made by well-known, monitored routes.

Follow the Principle of Least Privilege

The principal of least privilege states that users should be given the least amount of privilege to perform their job responsibilities. Over-ambitious granting of responsibilities, roles, grants, etc., especially early on in an organization’s life cycle when people are few and work needs to be done quickly; often leaves a system wide open for abuse. User privileges should be reviewed periodically to determine relevance to current job responsibilities.

Monitor System Activity

System security stands on three legs: good security protocols, proper system configuration, and system monitoring. Auditing and reviewing audit records address this third requirement. Each component within a system has some degree of monitoring capability. Follow audit advice in this document and regularly monitor audit records.

Obfuscation

Hiding something is not a secure solution in and of itself, but it can certainly be part of an overall solution. For example, using a non-standard network port for HTTP traffic would not ensure the safety of your data being transmitted over the network. However using a non-standard network port in-conjunction with HTTPS would be that much better, because a potential attacker would not know where to start looking for SSL-specific security vulnerabilities (e.g. if you used port 9477 instead of 443).

Keep Up-to-Date on the Latest Security Information

Oracle continually improves its software and documentation. Check this document regularly for revisions as well as the Oracle Technology Network Security Topics.

2. Secure Installation and Configuration

Installation Overview

This section outlines the planning process for a secure installation and describes several recommended deployment topologies.

Understand Your Environment

To better understand your security needs, ask yourself the following questions:

Which resources am I protecting?

Many resources in the production environment can be protected: data in the database, physical servers and the availability, performance, and integrity of the application itself. Pre-production and test systems also have data which needs to be protected, often every bit as sensitive as the production data itself. Consider the resources you want to protect when deciding the level of security you must provide.

From whom am I protecting the resources?

Resources belonging to application should be protected from everyone on the internet. Do you have service providers that will be accessing a Supplier Portal? What level of access do you want to give to employees? What resources should they be able to access? Should the system administrators be able to access all data? You might consider giving access to highly confidential data or strategic resources to only a few well trusted system administrators. Perhaps it would be best to allow no system administrators access to the data or resources.

What will happen if the protections on strategic resources fail?

In some cases, a fault in your security scheme is easily detected and considered nothing more than an inconvenience. In other cases, a fault might cause great damage to companies or individual clients that use the software. In analyzing the risks vs. rewards for implementing a specific piece of security, you should look at the following:

- How sensitive is the data?
- How easy would it be to capture/alter the data?
- How easy would it be to detect that a security breach has happened?
- How easy or hard will it be to make use of the captured data?
- What are the penalties associated with the increased security?

In almost all cases, the default position should be to use the most secure option available. Sometimes though, the risks (usually manifesting as worsening performance) do not warrant the gain. Understanding the security ramifications of each resource will help you protect it properly.

Application Components and Network Topologies

OTM, GTM, and FTI are all composed of many different components, and can be used by many users in a variety of roles. Some of the users will be internal to your company’s network, while others will be external. Data will be exchanged between the application and both internal and external systems. Each access path should be looked at individually and decisions made appropriately as to what activity will be permitted or blocked, and what controls will be put in place to enforce those decisions.

For example, if your company has both internal and external users, you may give them each different access. External users may be segregated onto a separate web tier instance, accessible only via

HTTPS, which itself is separated from the corporate WAN via firewalls, perhaps isolated in a DMZ. Internal users, on the other hand, may have a completely different setup, accessible via HTTP, with access from their web tier instance to the DB instance unrestricted, and so on.

Firewalls, reverse proxy servers, SSL and the like all help protect the systems and the data, but all of them can, potentially, also contribute to performance and connectivity issues. Pre-production and test systems may not need to be protected from external threats in the same manner as a production system, but the data inside them may.

It is difficult to make general recommendations in this area as each client's needs and the dictates of corporate security groups vary tremendously. All of the products covered in this guide, though, have been designed and developed to be as flexible and configurable as possible in this regard, so most normal (and not so normal) situations should be able to be handled by the software. That said here are some general recommendations:

- Always use HTTPS for external traffic originating from or being transmitted to the internet.
- If you trust your corporate WAN, don't use HTTPS for internal traffic (and conversely, if your corporate IT group treats your WAN as hostile, do use HTTPS, even internally).
- Machines running web & integration tiers exposed to the internet should be separated from the application machines via firewalls and related technology, and direct access to the database should be disabled from such machines when possible.
- Machines running application & database tiers should not be directly accessible from outside the WAN (i.e. they should not be exposed to the internet). Resources on these machines (e.g. inbound Web Services) can be exposed externally via the OTM web tier by using a reverse proxy (see the OTM Administration Guide under Advanced Configuration: UI/Web Server).
- There should be as few barriers as possible between the application tier and the database tier.

The key is to be aware of the potential effects of various security protocols and devices, both positive and negative, and use as many measures as are needed to secure your systems and data, but not any more than that; the benefits usually outweigh the risks, unless taken to the extreme. Seek advice from your internal network experts and make sure to follow the mandates of your internal security practices/directives.

Whatever you do, make sure to document it, and make sure to keep the document up-to-date! This really cannot be stressed enough; if you have a connectivity issue and you expect Oracle Support to help resolve it, you will need to have a current network diagram available to give to Support. If this is a production system, time will be of the essence, and the time needed to pull together the right people to create one on the fly could be critically detrimental.

Application Deployment Topographies

As said earlier, OTM, GTM, and FTI are all composed of many different components, and they can be deployed in many different configurations. Which configuration to use in different situations is not really a security issue, and thus falls outside of the scope of this document. The security implications of the various configurations will, of course, be discussed.

Deploying to a Single Machine vs. Multiple Machines

There is very little practical difference in terms of security when deciding to deploy all software components onto a single machine vs. deploying to multiple machines. One aspect is that communication that happens between components on a single machine can be done without security, but communication between machines should be looked at to see if security should be added (the specifics of how to secure communication between any two components are described in the section Oracle HTTP Server & Tomcat Integration.)

If installing multiple components onto a single machine, the security best practice recommendation would be to install each component separately into their own directories, even if the software allows for it to be a combined installation. Regardless of the number of machines involved, the security best practice recommendation would be to install each component using different user accounts so that if one account is compromised, the other components will not be affected.

Using the Scalability (SCA) Feature

The primary change in security around using the SCA feature in any deployment is whether or not to secure the JMS communications between application tier instances. The JMS traffic primarily contains Object IDs contained in a binary format. In addition, the exposure of the JMS traffic is limited to the network between the application tier instances. Therefore, the risk of not securing JMS traffic is fairly small. However, securing the JMS traffic is a supported configuration should you choose to do so. The potential downside is increased latency of communication between the application tier instances, constantly causing data to become momentarily out of sync. Details of how to secure the JMS communications between application instances can be found in WebLogic documentation. Please consult WebLogic documentation for further detail on securing JMS communication.

Multiple Instances on a Single Machine

Installing multiple instances of OTM, GTM, or FTI on a single machine (in any combination) is a supported configuration. The only recommendation from a security perspective is to install each component of each instance as a separate user, so that if one component of one instance is compromised, all of them are not affected.

Replicated Operational Database

Whether or not to use a Replicated Operational Database (ROD) is a business decision and will be driven by such factors as performance and volume. Always remember to protect the ROD in the same manner as the OLTP is protected, because the data is every bit as sensitive. Special attention should be paid to securing the replication process if there is a concern that data could be intercepted during the replication process itself; please see the documentation of the technology being used to replicate data for more information on how to do that.

Sharing Databases

When more than one application runs inside of a single database instance, care should be taken that rights are not granted between applications. Also, be aware that database performance issues on one application can affect the other applications unless a technology like Oracle Database Resource Manager is used.

Multi-Purpose Hardware

Using hardware for more than one software application is not recommended from both a security perspective as well as a performance perspective (which is in and of itself a security concern). If one application becomes compromised this can pave the way for easier, less restrictive access to other applications, contributing to a cascade of security failures. If hardware must be shared, all components of all applications should be installed into separate directories as different users, to minimize the risk should one component be compromised.

Virtual Machines

Using Virtual Machines (VMs) can help better utilize hardware resources while not opening them up to potential security issues (see *Multi-Purpose Hardware*). The cost is a potential for a performance hit, depending on which VM technology is being used. Always remember to follow the security guidelines of the underlying VM technology.

Production vs. Pre-Production Environments

Test and Development environments often have data in them that is every bit as important to secure as the real Production data. These systems should be secured as if they were Production systems, if appropriate.

Installing Base Components

Installing the Operating System

You should harden the Operating System (OS) as much as possible, according to the OS vendor's own security guidelines. The application's needs in terms of ports used, OS users needed and user privileges needed are fairly light, and all should be easily accommodated.

In keeping with the principal of minimizing your attack surface, unnecessary services should not be installed on the machines being used for the application. Services such as DNS, SMTP, non-application web servers, and the like should not be installed unless absolutely necessary.

Security-Enhanced Linux (SELinux)

SELinux is a set of kernel modifications that runs on many different Linux distributions. OTM and GTM can both be deployed under SELinux without issue. FTI is bound by the same capabilities and limitations as Oracle Business Intelligence, Enterprise Edition; please see the Security Guide for that product for more information and guidance.

Installing Oracle WebLogic Server

Oracle WebLogic Server should be installed and hardened as per the instructions in its own Security Guide. OTM and GTM do not support running inside the same WebLogic instance as other applications. Even apart from this lack of support, it would not be a good idea from a security perspective to run another application inside of the same WebLogic instance, as a security breach in one application could be exploited into breaches of other applications.

Installing Oracle Database 11g

Oracle Database should be installed and hardened as per its Security Guide. OTM, GTM, and FTI can all be run in this configuration.

In addition, there are several security-related technologies that the applications can be used with.

Transparent Data Encryption

Transparent Data Encryption (TDE) encrypts the data in a database in such a way that the application does not need to know that this is happening. The data is encrypted on disk, but is decrypted as soon as it's read into memory. There are currently two types of TDE available, tablespace-level encryption and column-level encryption. All of the products covered in this guide work with tablespace-level encryption, though there is a performance penalty associated with using it. Column-level encryption can also be used, with some limitations (all in the underlying TDE technology). See the Oracle Database documentation on Transparent Data Encryption for more details on limitations and how to enable it. Also see MyOracle Support Document #1214173.1 for more detailed information on specific tests conducted between the application and the technology.

Database Vault

Oracle Database Vault is a security option which provides flexible and highly adaptable security controls that can be transparently applied to existing application environments, protecting against insider threats and enforcing separation of duties. All of the products covered in this guide work with Database Vault; however the application data structures must exist prior to enabling Database Vault. See the Oracle Database documentation on Database Vault for more details on how to enable the feature. Also see MyOracle Support document #1214173.1 for more detailed information on specific tests conducted between the application and the technology.

SecureFiles

SecureFiles delivers substantially improved performance along with optimized storage for unstructured data inside the Oracle database. While not explicitly a security feature, it does provide the option to encrypt LOB data. All of the products covered in this guide have been certified to work with SecureFiles. See the Oracle Database documentation on SecureFiles for more details on how to enable the feature.

Installing Oracle HTTP Server

Oracle HTTP Server should be installed and hardened as per the instructions in its own Security Guide. OTM and GTM can both run in this configuration.

Installing Oracle Business Intelligence, Enterprise Edition

Oracle Business Intelligence, Enterprise Edition should be installed and hardened as per the instructions in its own Security Guide. FTI can be run in this configuration.

Installing Oracle Data Integrator

Oracle Data Integrator should be installed and hardened as per the instructions in its own Security Guide. FTI can be run in this configuration.

Installing the Application

Installation Type

There are three different ways to install OTM and GTM: the silent installer, the console installer and the GUI installer. In terms of security, there is very little difference in which one you pick. The silent installer uses a response file that the installer reads to determine all of the configurable pieces of the install. Using this file gives the silent installer the added benefit that it is more easily reviewed by a team of people for security issues (e.g. not using default values), but has the disadvantage that these pieces of information are all written down in one file that will need to be secured or deleted afterwards. The console and GUI installers have the exact opposite effects; these are harder to review but no data to clean up after. The benefits of the silent installer are probably enough to outweigh the downsides, but it’s such a slight edge that any of the methods can be used with confidence.

Admin Privileges

Linux/Unix

When installing under Linux or any flavor of UNIX, the products may *not* be installed as the root user; the installer will fail.

The products do not inherently require root privileges to run. However, if the product is configured to use a privileged port (a port with a value under 1024), then it must be started as the root user, which grants the process more privileges than is usually desirable. Typically this will be done to run the Web tier instance on port 80 (HTTP) or 443 (HTTPS). There are several ways to use a privileged port and not run as root, such as using Role-Based Access Control (RBAC) under Solaris, or iptables under

Linux; please contact your local Systems Administration group to help evaluate options with your specific operating systems. Also, Oracle HTTP Server allows the process to start as root but immediately switches its userid & groupid to the configured non-root values, thus making it safe to start Oracle HTTP Server as root in order to bind to the privileged ports.

Windows

When installing under Windows, the products must be installed by a user with Administrator privileges if it's desired to run the components as a Windows service. Running as a Windows service allows the applications to start up automatically when the machine itself is started, without requiring a user to log in and manually start the application. It is recommended that all production instances on Windows be installed in this manner.

Network Ports

The following ports are used by OTM and GTM; FTI does not add any ports that are not already used by Oracle Business Intelligence, Enterprise Edition.

Port numbers less than 1024 on Linux/UNIX machines are usually privileged, and therefore require root permissions to run. See *Admin Privileges* for more information. Using non-standard port numbers is recommended to help obfuscate your resources.

Network ports bind to one of the following: a specific IP address, all IP addresses on the machine, or a special address called 'localhost'. The value used for the network port that is bound to a specific IP address must be unique for that IP address, but it can be used again on the same machine if the second use is bound to a different IP address. Network ports bound to all IP addresses or to localhost *must* have a value that is unique on that machine.

The applications do not provide an automated way to change these ports after installation; if they need to be changed, edit the files indicated below and change their values as directed. Filenames include the following strings, which represent specific directories where application components were installed to:

- $APP – directory where the application was installed to
- $OHS – directory where OHS was installed to
    - $INSTANCE_NAME – the name of the OHS instance related to this application
    - $COMP_NAME – the name of the component in this instance

The string "[WEB]" will indicate that this file needs to be changed on all web tier instances in the application installation, and "[APP]" will indicate that this file needs to be changed on all application tier instances in the application installation. The string "[BOTH]" will indicate that this file needs to be changed on all web & application instances.

Port Name	**Binds To**	**Files to be Edited**	**Description**

HTTP Port	IP Address	`[WEB] $OHS/instances/$INSTANCE_NAME/config/OHS/$COMP_NAME/moduleconf/otm.conf`
		Change the following line:
		`<VirtualHost *:80>`
			The port that HTTP traffic comes in on.

Port Name	**Binds To**	**Files to be Edited**	**Description**

		`[WEB] $OHS/config/OHS/$COMP_NAME/httpd.conf`
		Change the following line:
		`Listen 10.143.205.57:80`	

		`[WEB] $APP/tomcat/conf/server.xml`
		Change the following section:
		`<Connector address="localhost" port="8109" protocol="HTTP/1.1"`
		`connectionTimeout="20000"`
		`proxyName="web.company.com" proxyPort="80"`
		`URIEncoding="UTF-8" />`
		Note: The proxyPort attribute may not exist; see *Oracle HTTP Server & Tomcat Integration* for more information.	

		`[BOTH] $APP/glog/config/glog.properties`
		Change the following two lines:
		`webserver.port=80`
		`glog.webserver.URL=http://machine.company.com:80$glog.webserver.urlprefix$`	

HTTPS Port	IP Address	`[WEB] $OHS/config/OHS/$COMP_NAME/ssl.conf`
		Change the following line:
		`Listen 10.143.205.57:443`
			The port on which HTTPS traffic comes.

Port Name	**Binds To**	**Files to be Edited**	**Description**

Tomcat Proxy Port	Localhost	`[WEB]`
		`$OHS/instances/$INSTANCE_NAME/config/OHS/$COMP_NAME/moduleconf/otm.conf`
		Change the following two lines:
		`ProxyPass         /GC3 http://localhost:8109/GC3`
		`ProxyPassReverse  /GC3 http://localhost:8109/GC3`
			The port that Oracle HTTP Server uses to talk to Tomcat.
			NOTE: This port must be unique on each host since it binds to localhost.

		`[WEB]`
		`$APP/tomcat/conf/server.xml`
		Change the following section:
		`<Connector address="localhost" port="8109" protocol="HTTP/1.1"`
		`   connectionTimeout="20000"`
		`   proxyName="web.company.com" proxyPort="80"`
		`   URIEncoding="UTF-8" />`	

Tomcat Shutdown Port	Localhost	`[WEB]`
		`$APP/tomcat/conf/server.xml`
		Change the following line:
		`<Server port="8007" shutdown="SHUTDOWN">`
			The port that is used to tell Tomcat to shut itself down.
			NOTE: This port must be unique on each host since it binds to localhost.

Tomcat Launcher Port	Localhost	`[WEB]`
		`$APP/tomcat/bin/tomcat.conf`
		Change the following line:
		`launcher.port=`*32000*
			Tomcat is started via an Oracle-supplied Java program called Launcher; this is the port over which the Launcher java process and Tomcat java process communicate.
			NOTE: This port must be unique on each host since it binds to localhost.

Port Name	**Binds To**	**Files to be Edited**	**Description**

WebLogic T3 Port	IP Address	`[BOTH]`
		`$APP/glog/config/glog.properties`
		Change the following lines:
		`appserver.port=7001`
		`appserver.port.webservice.weblogic=7001`
		`aa_webserver=http://machine.company.com:7001`
		`glog.scalability.topologyMachineURL=$appserver.protocol$machine1.company.com:7001`
		`glog.scalability.topologyMachineURL=$appserver.protocol$machine2.company.com:7001`
			This is the port that Tomcat talks to WebLogic over.

		`[APP]`
		`$APP/weblogic/weblogic.conf`
		Change the following lines:
		`command.stop.arg=t3://machine.company.com:7001`	

		`[APP]`
		`$APP/weblogic/domains/otm/config/config.xml.fresh`
		Change the following lines:
		`<listen-port>`*7001*`</listen-port>`	

WebLogic Startup Ping Port	IP Address	`[BOTH]`
		`$APP/glog/config/glog.properties`
		Change the following lines:
		`glog.scalability.activatePortOffset=100`
			The WebLogic T3 port is activated very early in the application startup process, and therefore cannot be used to tell whether or not the application tier is ready to process requests. This port is controlled by application code and is used to tell when all of the startup process has been completed.
			Consult the OTM Scalability Guide for more details on configuring this property.

Port Name	Binds To	Files to be Edited	Description

WebLogic Keep Alive Ping Port	IP Address	`[APP] $APP/glog/config/glog.properties`
		Change the following lines:
		`glog.scalability.invokablePortOffset=200`
			This port is used as a keep-alive ping for application servers to account for missed JMS synchronization on startup. This keep-alive relies on a new invocation port ping on each application server. This port is setup after all other startup classes have activated (including JMS messaging).
			This port number is defined as an offset to the WebLogic T3 port.
			For example, using the default values of 7001 and 200, the resulting port number would be 7201.
			At each keep alive ping, the application server invokes all other pingable application servers to:
			1. Make sure the source server is in the scalability map.
			2. Make sure JMS connections exist between the source and destination servers.

WebLogic Launcher Port	Localhost	`[APP] $APP/weblogic/weblogic.conf` Change the following line: `launcher.port=`*32001*	WebLogic is started via an Oracle-supplied Java program called Launcher; this is the port over which the Launcher java process and WebLogic java process communicate. **Note:** This port must be unique on each host since it binds to localhost.

Post Installation Configuration

General/Miscellaneous

Controlling application stack traces

The ability to hide stack traces is controlled via a property (glog.security.stackTrace.hide=[true|false]) and the 'StackTrace – View' Access Control List (ACL). The property is already set correctly by default (true). By default, the 'StackTrace – View' ACL is a child ACL of both the staged 'ADMIN' and 'DEFAULT' ACLs. This means that any user or user role that has the top level 'ADMIN' or 'DEFAULT' ACL will have the ability to view full stack traces. If you do not want to allow the ability to view stack traces, then you should just deny (not grant) the 'StackTrace – View' ACL on the individual user or the user role.

- glog.security.stackTrace.hide
  - defaults to 'true'
  - determines whether or not stack traces are hidden
- ACL: 'StackTrace – View'
  - Provides additional configurability for individual users and user roles when glog.security.stackTrace.hide is true.
  - Is a child Access Control List of the ADMIN and DEFAULT Access Control Lists by default. Any user or user role with either of these top level Access Control Lists will see a stack trace.

Note: The glog.security.stackTrace.hide property and 'StackTrace – View' Access Control List now also control some Stack Traces for integration related activities.

SSL/TLS Certificates

OTM, GTM, and FTI can all be configured to use Secure Sockets Layer (SSL), a security protocol that allows a client program (e.g. web browser) to talk to a server program (e.g. web server) over an encrypted link. Transport Layer Security (TLS) is a more recent version and name of the same protocol. Throughout this section "SSL" will be used to mean both versions unless otherwise explicitly noted.

One very important piece of this protocol is for one program to establish its identity with the other; in this case it's usually the server proving to the client that it really is the machine that it says it is. This is accomplished by having a trusted third party, called a Certificate Authority, vouch for the server trying to prove its identity. The vehicle to accomplish this is the digital certificate, which your network operations or security team can acquire for you.

The terms "client" and "server" in this case refer to the programs or processes initiating the communication (client) and receiving the communication (server). The client usually wants to ensure that the server really is who it claims to be before it starts talking to it. A program can be a client in one transaction and a server in another (e.g. WebLogic sending data to a downstream system would be a client, but when it's receiving data from an upstream system it would be acting as a server).

All of this should work without a problem unless the client program does not recognize the specific Certificate Authority (CA) used by the server to establish its identity. When that happens a human must get involved to make the determination of whether or not to trust the CA; if it should be trusted, it needs to be added to the list of trusted CAs. Alternately, a specific certificate can be evaluated and deemed trustworthy by a human, in which case the certificate itself must be added to a list of trusted certificates.

Certificates purchased from a CA can be expensive and are generally issued for an individual machine; so many implementations either eschew their use for internal, development, and testing machines or use a “self-signed” certificate. Self-signed certificates are certificates that are not issued by a CA and are instead signed by the server itself. These should not be used in production but are perfectly acceptable to be used in a development or test environment. Your network operations team can help in generating the self-signed certificate; they are used in the exact same way as a CA-issued certificate is used.

It is strongly recommended that all traffic on networks outside of your own network be encrypted using SSL wherever and whenever possible.

Performance Implications

There is a slight performance penalty when using SSL, primarily in two places:

1. At the start of each communication a dialog happens where the client and server exchange credentials and negotiate which exact protocol will be used during the life of the communication.
2. All of the content of the data transmitted between the client and server must be encrypted by the sending party and decrypted by the receiving party.

While both of these things happen very quickly, it does still take time and computational power, which can lead to noticeable performance degradations, especially if there is a lot of encrypted network traffic happening.

There are several common ways to mitigate the performance implications including: use of SSL accelerators, use of multiple instances handling the encrypted traffic (e.g. multiple web tier instances), and using SSL/TLS only where it’s needed (e.g. using SSL for network traffic that passes outside of the company firewall but not for traffic that stays completely inside of the firewall). Consult with your network security personnel to determine your exact requirements.

Places Where SSL Can Be Used

General Notes

The applications do not provide an automated way to change the certificate configurations after installation; if they need to be changed, follow the steps below and edit the files indicated; changing their values as directed (the values to be changed are shown in italics). Filenames referenced below include the following strings, which represent specific directories where application components were installed to:

- $APP – directory where the application was installed to
- $OHS – directory where OHS was installed to
    - $INSTANCE_NAME – the name of the OHS instance related to this application
    - $COMP_NAME – the name of the component in this instance
- $CERT – the full path (directory plus filename) of the SSL certificate

Web Tier

SSL can be used on the web tier to encrypt traffic to and from the user’s web browser or for inbound integration using servlets (e.g. WMServlet). This is the most common place where SSL is deployed and doing so changes the web protocol from HTTP to HTTPS. SSL is only used on the web tier as a server, never as a client.

Note: A given web server instance can only be configured to use HTTP or HTTPS, not both at the same time. If both HTTP and HTTPS is desired then two different web server instances must be installed to the same or different machines. See the Oracle Transportation Management Installation Guide for more information on installing more than one Web server instance to the same machine.

Once you have acquired the certificate from your network or security team, you will need to add it to the OHS SSLWallet of each web tier instance. Please refer to the Oracle Database Advanced Security Administrator's Guide for more information on how to do this.

If using a Certificate Authority that is not recognized by the user's web browser or using a self-signed certificate, then you will need to follow the browser vendor's directions on how to get the browser to accept the certificate. Similarly, upstream systems connecting to this one via SSL for inbound integration will need to follow their own directions on how to deal with self-signed certificates or certificates issued by an unknown CA.

In addition to the above, you will also need to make the following change to the `<otm_install_dir>/tomcat/config/server.xml` file if you wish to have a specific Web server instance use HTTPS:

1. Back up the existing server.xml file:

```
$ cd <otm_install_dir>/tomcat/config
$ cp server.xml server.xml_20130427_1542
```

2. Edit the server.xml file and locate the following section:

```
    <Connector address="localhost" port="8009" protocol="HTTP/1.1"
               connectionTimeout="3600000"
               proxyName="slc02jwd.us.oracle.com" proxyPort="7777"
               URIEncoding="UTF-8" />
```

3. Change it to add the line in bold:

```
    <Connector address="localhost" port="8009" protocol="HTTP/1.1"
               connectionTimeout="3600000"
               scheme="https" secure="true"
               proxyName="slc02jwd.us.oracle.com" proxyPort="7777"
               URIEncoding="UTF-8" />
```

4. restart the Web server instance

Failure to follow these steps will result in login requests reverting to HTTP instead of using HTTPS as desired.

Application Tier

SSL can be used on the application tier in a variety of ways, as both a client and a server. Let's look at this from the server side first.

Application Tier - Inbound Web Services

Application tier Web Services fully support Web Service Security policies. By default all inbound Web Services will require a Username Token in plain text transported over HTTPS. More restrictive policies can be configured e.g. message encryption. See *Web Service Configuration* section for details.

Application Tier - Outbound Web Services

Application tier calling outbound to externally resident Web Services fully supports Web Service Security policies. Please consult the "Configuring Transport-Level Security" section of *Oracle® Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

If using a Certificate Authority that is not recognized by the upstream system's program, or using a self-signed certificate, then further configuration may need to happen on the upstream system; please consult the documentation provided by the upstream system's vendor.

Application Tier - Inbound RMI

RMI (Remote Method Invocation) communication happens between Tomcat on the web tier and WebLogic on the application tier via a protocol called T3; running the T3 protocol over SSL is called T3S. While securing this can be done, the performance degradation is generally not considered worth the price of securing the communication. Unlike a communication such as HTTP, RMI traffic consists mostly of binary Java objects, and it is very difficult to glean any useful information from this traffic should it be intercepted; therefore, encrypting this communication is not recommended. Please consult WebLogic documentation for further detail on securing T3 communication.

If using a Certificate Authority that is not recognized by the Tomcat JVM, or using a self-signed certificate, then further configuration may need to happen within Tomcat's JVM. For further information refer to the section *Installing a Certificate into a JVM's Keystore*.

Application Tier - Inbound / Outbound JMS

JMS (Java Messaging Service) is an API used for communication between 2 or more application tier instances in an OTM or GTM Scalability (SCA) cluster. JMS can also used for GTM AES Customs filing. Please consult WebLogic documentation for further detail on securing JMS communication.

Installing a Certificate into a JVM's Keystore

The following are generic instructions on how to install a certificate in a JVM's keystore. You should be following these directions only when directed to by one of the above sections. The instructions directing you here should have specified the values for `$JAVA_HOME & $KEYSTORE`.

1. Type the following command at a shell prompt:

```
$ $JAVA_HOME/bin/keystore –import –alias <some descriptive name> -file
<certificate file> -keystore $KEYSTORE
```

Where:

- `<some descriptive name>` = any name/alias you want (protect it with quotes if there is a space in the alias, e.g. "app server")
- `<certificate file>` = the full or relative path to the *.cer file being imported

If the certificate is a self-signed certificate the user will be prompted to verify the certificate; ensure that it is indeed the certificate that you wish to import and then follow the prompts to complete the process.

Installing a Trusted SSL Certificates in the OTM Application Server in WebLogic

The following are generic instructions on how to install a trusted SSL certificate in the OTM Application Server in WebLogic via the config.xml.fresh.

1. Backup the config.xml.fresh file.

2. Review the Oracle Fusion Middleware Oracle WebLogic Server Administration documentation for the specific details on creating Keystores, how to load Private Keys, configuring the SSL Certificates within WebLogic.
3. After making the changes required, diff the config.xml.fresh file with the config.xml file after making the changes. Copy the differences from config.xml into config.xml.fresh.

Example steps:

Go to directory:
```
/app/weblogic1036/Oracle_WT1/instances/instance1/config/OHS/otm63/keystores/otm63
```

Run:
```
orapki wallet pkcs12_to_jks -wallet ./ -pwd <PASSWORD> –jksKeyStoreLoc
./server_ewalletK.jks -jksKeyStorepwd <PASSWORD> -jksTrustStoreLoc
./server_ewalletT.jks –jksTrustStorepwd <PASSWORD>
```

1. Configure the SSL in the WebLogic console.
2. Take the entire <server> tag from config.xml and add in the config.xml.fresh file

Example resulting XML from config.xml:

```
<server>
    <name>gc3-OTMDev-WebApp</name>
    <max-message-size>100000000</max-message-size>
    <ssl>
            <enabled>true</enabled>
            <hostname-verifier xsi:nil="true"></hostname-verifier>
            <hostname-verification-ignored>false</hostname-verification-ignored>
            <client-certificate-enforced>false</client-certificate-enforced>
            <listen-port>7002</listen-port>
            <two-way-ssl-enabled>false</two-way-ssl-enabled>
            <server-private-key-alias>orakey</server-private-key-alias>
            <server-private-key-pass-phrase-
            encrypted>{3DES}xxxxxxxxxxxxxxxxxxx==</server-private-key-pass-
            phrase-encrypted>
    </ssl>
    <log>
            <file-name>/app/otm634/logs/weblogic/weblogic.log</file-name>
            <file-count>1</file-count>
            <rotate-log-on-startup>true</rotate-log-on-startup>
    </log>
    <listen-port>7001</listen-port>
    <listen-address>xxx.xxx.xx.xx</listen-address>
    <key-stores>CustomIdentityAndCustomTrust</key-stores>
    <custom-identity-key-store-file-
    name>/app/weblogic1036/Oracle_WT1/instances/instance1/config/OHS/otm63/keyst
    ores/otm63/server_ewalletK.jks</custom-identity-key-store-file-name>
    <custom-identity-key-store-type>JKS</custom-identity-key-store-type>
    <custom-identity-key-store-pass-phrase-
    encrypted>{3DESxxxxxxxxxxxxxxxxxxx==</custom-identity-key-store-pass-phrase-
    encrypted>
    <custom-trust-key-store-file-
    name>/app/weblogic1036/Oracle_WT1/instances/instance1/config/OHS/otm63/keyst
    ores/otm63/server_ewalletT.jks</custom-trust-key-store-file-name>
    <custom-trust-key-store-type>JKS</custom-trust-key-store-type>
    <custom-trust-key-store-pass-phrase-
    encrypted>{3DES}xxxxxxxxxxxxxxx==</custom-trust-key-store-pass-phrase-
    encrypted>
</server>
```

Browser Cookies Used in OTM

The following browser cookies are used in OTM.

Cookie Name	Personally Identifiable Information	Retention Policy	Effect of Refusal	Usage
JSESSIONID	There is no personally identifiable information collected or stored in this cookie.	The cookie only lasts as long as the browser is open; once the browser closes the cookie is discarded.	OTM will not work without this cookie.	See notes below.

JSESSIONID

OTM does not create any cookies for use in the application; however Tomcat creates and sets a cookie for session tracking purpose. By default this cookie is called JSESSIONID and has a value set to random characters. Since HTTP is a stateless protocol Tomcat uses this cookie to maintain session state between requests. Consult the Tomcat documentation for information on customizing this cookie and its behavior.

For maximum security it is recommended that connections to the web server be sent over HTTPS. This will prevent the cookie value from being compromised which can lead to session hijacking attacks.

Default Users

The OTM application has and requires different default users that are utilized on the different tiers of application stack. These users include database users for different components, application server users, and default application users.

Oracle Database Users

The following table lists all of the default Oracle Database users that OTM and associated components create during install and are required for the application to work correctly.

Database User ID	Schema Access Rights	Notes
archive		This user owns the DMP tables used for archiving the data. May not be deleted.

Database User ID	**Schema Access Rights**	**Notes**
glogdba		This user has access to functions and packages owned by glogowner and reportowner, but does not itself own any tables, views, functions, or packages. It must call the vpd.set_user stored procedure to set user context to view data. May not be deleted.
glogowner		This user owns OTM tables, views, functions and packages, can create or alter data structures within the database and can manipulate data. May not be deleted.
glogload		Used for loading data into glogowner and reportowner schemas. May not be deleted.
dir_xml_user		This user should be use for Direct XML integration. This user has the minimum privileges to successfully insert XML transmissions into database objects when using the Direct XML integration feature.
reportowner		This user owns the tables, views, functions and packages required for reporting, and can read the data. May not be deleted.
globalreportuser		This user has read access to all the data in OTM. It is mainly used for reporting. May not be deleted.
hdowner		This user owns FTI tables, views, functions and packages, can create or alter data structures within the database and can manipulate data. May not be deleted.

Database User ID	**Schema Access Rights**	**Notes**
ftiodimaster	FTIMASTER: Full Access GLOGOWNER: No Access HDOWNER: (Uses "FTI Role") Read, Insert, Update & Delete	Used to create Oracle Data Integrator (ODI) Master repository. This user is required by ODI to connect to the ODI Master Repository where all the ODI System & Deployment information are stored. May not be deleted.
ftiodiwork	FTIWORK: Full Access GLOGOWNER: No Access HDOWNER: (Uses "FTI Role") Read, Insert, Update & Delete	Used to create ODI Work repository. This user is required by ODI to connect to the ODI Work Repository where the entire FTI specific ETL project (processes & functions that have the FTI ETL logic) is stored. May not be deleted.
ftiodistage	FTISTAGE: Full Access GLOGOWNER: No Access HDOWNER: (Uses "FTI Role") Read, Insert, Update & Delete	Used to create ODI staging area. This user is required by ODI to connect to the ODI Staging Repository where at run-time based on the ETL project ODI creates temporary staging tables before loading the target database (HDOWNER). May not be deleted.

Note: The above ODI users do not have any access to the OTM GLOGOWNER schema. To extract the data from OTM, the physical connection at deployment time in the ODI Admin tool, you will have to use the 'GLOGDBA' user as part of the configuration as documented in the FTI Deployment steps for ODI.

If you wish to change the passwords for these users, follow these steps:

1. Using SQL*Plus, log into the OTM database as sys or system.
2. Run the following for each user that you wish to change:

```
alter user <user_name> identified by <new password>
```

Some users have additional steps that need to be taken:

glogdba

1. Encode the new password according to instructions in the section "Encoding Values in glog.properties" elsewhere in this document.
2. Edit the `<otm_install_path>/glog/config/glog.properties` file on each OTM web and application server and replace the value of "glog.database.password" with the new password:
    a. `glog.database.password={e<encoded-password>`, where "`<encoded-password>`" is the encoded password value.
3. Restart OTM.

glogload

1. Encode the new password according to instructions in the section "Encoding Values in glog.properties" elsewhere in this document.
2. Edit the `<otm_install_path>/glog/config/glog.properties` file on each OTM web and application server and replace the value of "glog.database.load.password" with the new password:
    a. `glog.database.load.password={e<encoded-password>`, where "`<encoded-password>`" is the encoded password value.
3. Restart OTM.

hdowner

1. Encode the new password according to instructions in the section "Encoding Values in glog.properties" elsewhere in this document.
2. Edit the `<otm_install_path>/glog/config/glog.properties` file on each OTM web and application server and replace the value of "glog.database.fti.password" with the new password:
    a. `glog.database.fti.password={e<encoded-password>`, where "`<encoded-password>`" is the encoded password value.
3. Restart OTM.

Oracle Application Server Default Users

The application server has one user that is utilized in running OTM, and this user has a separate set of instructions to be used if you want to change the password. This user may not be deleted.

WebLogic

The WebLogic configuration that is used for OTM does create a user that can be used to manage the WebLogic console called "weblogic". This user is not required by OTM, and can be deleted. At the very least, the password should be changed after installation. However, OTM does create another weblogic like user that it relies on called system.

User ID	**Default Password**	**Notes**
weblogic	CHANGEME	A true WebLogic user. This user can be modified or deleted according to the WebLogic documentation. This user does not exist in any version of 6.3.1 and newer. However, it will exist in older versions.
system	CHANGEME	Not a true WebLogic user. Used to start and stop the application as well as manage the WebLogic console. May not be deleted.

In order to change the ‘system’ and ‘guest’ user’s password, do the following:

Note: If you are just changing the "guest" user password you can do step 1 and then go to step 16 of these instructions.

1. First, decide on a new password for the user.
2. Log into the WebLogic console using the user system and the current password. Review WebLogic documentation if you are unsure on how to log into the WebLogic console.
3. Click the **Lock and Edit Button**.
4. Click the **Otmv636** link in the Domain structure portion of the screen.
5. Click the **Security** tab in the middle section of the screen.
6. Click **Advanced**.
7. Change the NodeManager Password to the new password that was decided on and re-enter the password in the confirmation box.
8. Click **Save** at the bottom the screen.
9. Click **Release configuration**.
10. Shut down the OTM Webserver(s) and Application Server(s). Ensure that OTM application and web servers are stopped.
11. On the application server navigate to the `<otm_install_path>/weblogic/domain/otm/config` directory.
12. Make a copy of the existing `config.xml.fresh` file and save it as `config.xml.fresh.<Date>`.
13. Open and edit the `config.xml` file and look for the following line:

    ```
    <node-manager-password-encrypted>{3DES}xxxxxxxxxxxxxxxxxxxxxxxx</node-manager-password-encrypted>
    ```

14. Copy the new `<node-manager-password-encrypted>` line from `config.xml`.
15. Now edit the `config.xml.fresh` file and remove the `<node-manager-password-encrypted>` line in the file and replace it with the line you copied from the `config.xml` file.

    The existing line may look similar to this:

    ```
    <node-manager-password-encrypted>{3DES}CLBTJIlA8xlTPPvjOcG06Q==</node-manager-password-encrypted>
    ```

16. Save the changes to the `config.xml.fresh` file.
17. You will need to encode the password for the system or guest user according to the instructions in the section “Encoding Values in glog.properties”. These instructions will refer to both the plain-text password and the encoded password specifically; please be careful to use the correct value.
18. Setup your environment by running `<otm_install_path>/install/gc3env.sh` on UNIX or `<otm_install_path>\install\gc3env.cmd` on Windows.
19. Navigate to the `<otm_install_path>/oracle/script8 directory`.
20. Run `update_password.sh` on UNIX or `update_password.cmd` on Windows. Enter the following parameter values when prompted:
    a. **Enter the glog properties path**: Enter the directory where glog.properties file is located, e.g. `<otm_install_path>/glog/config`.
    b. **Enter the user name for which password needs to be updated**: Enter the name of the user to change, e.g. `system`.
    c. **Enter a new password**: Enter the new plain-text password.
    d. **Enter one or more passwords to match against separated by commas:** Enter the word ‘all’ (without the quotes)

   e. **Enter the connection ID (Press Enter for default):** Accept the default.
   f. There is a log file that can be reviewed to verify that the password was updated in the database. The log file will be in the format of: update_password_<DBSID>_<TIMESTAMP>.log. It could contain errors if a problem occurred. If it was successful then the log file will contain a statement similar to '1 password changed'.

21. On the application server(s) navigate to the `<otm_install_path>/weblogic/` directory and make a backup copy of the `weblogic.conf` file by copying it to `weblogic.conf.org`.
22. Next, edit the file `<otm_install_path>/weblogic/weblogic`.conf and make the following changes depending on which user's password is being changed.
   a. **[system]** search for the string '`var.WL_PW=CHANGEME`' and change the value of this to the clear-text password.
   b. **[system]** add or replace the string '`jvm.arg=-DGC3EncodedPassword=<encoded-password>`', where '`<encoded-password>`' is the encoded password.

      Note: if there not a line of `jvm.arg=-DGC3EncodedPassword` then there is a line of '`jvm.arg=-DGC3Password=<CLEAR_TEXT_PW>`, and then this line needs to be changed to reflect the new password value in clear text.
   c. **[guest]** add or replace the string '`jvm.arg=-DGuestEncodedPassword=<encoded-password>`', where '`<encoded-password>`' is the encoded password.

      Note: if there is not a line of jvm.arg=-DGuestEncodedPassword then there is a line of `jvm.arg=-DGuestPassword=<CLEAR_TEXT_PW>`, and then this line needs to be changed to reflect the new password value in clear text.
23. On the OTM web server(s) navigate to the `<otm_install_path>/tomcat/bin/` directory and make a backup copy of the `tomcat.conf` file by copying it to `tomcat.conf.org`.
24. Next, edit the file `<otm_install_path>/tomcat/bin/tomcat.conf` and make the following changes depending on which user's passwords are being change.
   a. **[system]** search for the string '`jvm.arg=-DGC3EncodedPassword=<encoded-password>`' , where '`<encoded-password>`' is the encoded password.

      Note: if there is not a line of `jvm.arg=-DGC3EncodedPassword` then there is a line of `jvm.arg=-DGC3Password=<CLEAR_TEXT_PW>`, and then this line needs to be changed to reflect the new password value in cleartext.
   b. **[guest]** search for the string '`jvm.arg=-DGuestEncodedPassword=<encoded-password>`' , where '`<encoded-password>`' is the encoded password.

      Note: if there is not a line of `jvm.arg=-DGuestEncodedPassword` then there is a line of '`jvm.arg= DGuestPassword=<CLEAR_TEXT_PW>`, and then this line needs to be changed to reflect the new password value in cleartext.
25. Restart all of the OTM webserver(s) and application server(s). Check the associated console and exception logs to make sure there are no authentication issues.

OTM Application Default Users

User ID	Description	Required	Can be deleted	Notes
system	A special user that is used internally by the application for system level privileges.	Yes	No	Change the password via the instructions provided in the section Oracle Application Server Default Users

User ID	Description	Required	Can be deleted	Notes
guest	A special user that is used internally by the application when user credentials and authentication are needed but the credentials cannot be provided by an actual end user.	Yes	No	Change the password via the instructions provided in the section Oracle Application Server Default Users
glog		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
glogdev		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
blueprint		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
e1		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
ebs		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
DBA.ADMIN	A super user and the Admin user of the DBA domain.	Yes	No	
DBA.OPS	A super user used for diagnostics and configuration	Yes, for cloud implementations	No	This user is optional in non-cloud implementations.
SERVPROV.ADMIN	The Admin user of the Service Provider domain.	Yes	No	

User ID	Description	Required	Can be deleted	Notes
GUEST.ADMIN	The Admin user of the Guest domain.	Yes	No	
EBS.ADMIN	The Admin user of the EBS domain.	Yes if EBS domain is required.	Yes if EBS domain is not needed	
E1.ADMIN	The Admin user of the E1 domain.	Yes if E1 domain is required.	Yes if E1 domain is not needed	
BLUEPRINT.ADMIN	The Admin user of the Blueprint domain.	Yes if BLUEPRINTdomain is required.	Yes if BLUEPRINT domain is not needed	
GLOG.ADMIN		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
STAGE.ADMIN		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
DBA.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
GUEST.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
SERVPROV.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
GLOG.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions

User ID	**Description**	**Required**	**Can be deleted**	**Notes**
STAGE.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
EBS.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
E1.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions
BLUEPRINT.DEFAULT		No	Yes	This user does not exist in new installs in 6.3.x. However, it will exist in older or migrated versions

Unless otherwise noted above, passwords for OTM users can be changed using the OTM User Manager; refer to the online help for details.

Reserved Users

OTM can protect a set of users from modification or deletion. These users are referred to as *reserved users*. For example, all OTM application default users listed above that still get created during installation are staged as reserved users. In general, this protects them from changes that would either destabilize the system or allow for security vulnerability.

OTM supports two modes for handling reserved users:

- Relaxed. In this mode, a reserved user:
  - cannot be deleted[1]
  - cannot be modified to alter their default user role[2]
- Strict. In this mode, a reserved user:
  - cannot be deleted
  - cannot be modified except to change its password or nickname,
  - can only be modified by itself

For backward compatibility, OTM installations are staged with the relaxed mode for handling reserved users. To set the strict mode, set

---

[1] Unless **`glog.realm.allowReservedUserDelete=true`**

[2] Unless **`glog.realm.allowReservedUsersModify-true`**

`glog.realm.strictReservedUsers=true`[3]

OTM Application Default User Roles

User Role ID	Description	Required	Can be deleted	Notes
DBA.ADMIN	Super user access to data and functionality	Yes	No	Assigned to user DBA.ADMIN. Data access crosses domains.
SERVPROV.ADMIN	Access to service provider users	Yes	No	Assigned to user SERVPROV.ADMIN. Data access crosses domains.
ADMIN	Administrative access to data and functionality.	Yes	No	Data is limited to the user's domain. Functionality typically includes diagnostics
OPS	Operations access to functionality	Yes, in a cloud environment	No	Functionality is limited to diagnostics
INTEGRATION	Limited access to external integration related entry points and COMMON entry points	No	No	Not assigned to any user by default
DEFAULT	Default user access to data and functionality	Yes	No	
SERVPROV	Limited access to data and functionality for service provider users	Yes	No	Data and functionality are limited to those needed by a service provider using OTM.
DATAENTRY	Limited access to data	No	Yes	Data is limited to the data entered by the current user. Not assigned to any user by default

---

[3] Note that this is the default mode for cloud installations.

User Role ID	Description	Required	Can be deleted	Notes
EXTERNAL		No	Yes	Not assigned to any user by default
DOCUMENT_REVIEW	Limited access to data and functionality for document reviewers	Yes	No	Data is limited to documents requiring review by the current user.

Reserved User Roles

OTM can protect a set of user roles from modification or deletion. These roles are referred to as *reserved user roles*. All OTM application default user roles listed above are staged as reserved user roles.

OTM supports two modes for handling reserved user roles:

- Relaxed. In this mode, a reserved user role:
    - cannot be deleted[4]
    - cannot be modified, except to modify grantee users roles, grantee users or grantee users or access control list grants[5]
- Strict. In this mode, a reserved user role:
    - cannot be deleted
    - cannot be modified, except to grant the role to unreserved users or unreserved user roles

In addition, an unreserved user role cannot grant itself to a reserved user or reserved user role.

For backward compatibility, OTM installations are staged with the relaxed mode for handling reserved user roles. To set the strict mode, set:

```
glog.realm.strictReservedUserRoles=true
```
[6]

Custom Restrictions for Users and User Roles

Additional properties are available to further restrict the assignment of user roles and access control roles. These properties are of the form:

```
glog.realm.restrict.<class>=<restricted role>[:<allowed>,<allowed>,...]
```

where

- <class> = the type of restriction

---

[4] Unless `glog.realm.allowReservedUserRoleDelete=true`

[5] Unless `glog.realm.allowReservedUserRoleModify-true`

[6] Note that this is the default mode for cloud installations.

- <restricted role> = the specified role to restrict
- <allowed> = the user or role allowed to access the restricted role

If no users or roles are specified in the allowed list, the role assignment is not allowed.

The following table lists the supported classes

Class	**Restricted**	**Allowed**
userRolesToUsers	User Role	Users that can be granted the restricted user role
acrRolesToAcrRoles	ACR Role	ACR roles that can be granted the restricted ACR role.
acrRolesToUserRoles	ACR Role	User roles that can be granted the restricted ACR Role
acrRolesToUsers	ACR Role	Users that can be granted the restricted ACR Role

As an example, consider the optional **OPS** user role[7]. This user role provides access to the **OPS** ACR role: a set of diagnostic tools and is designed for Operations/IT users. Allowing it to be openly granted to other users and user roles can be a security vulnerability. Adding the following properties limits its use to a reserved user DBA.OPS:

```
glog.realm.restrict.userRolesToUsers=OPS:DBA.OPS
glog.realm.restrict.acrRolesToAcrRoles=OPS
glog.realm.restrict.acrRolesToUserRoles=OPS:OPS
glog.realm.restrict.acrRolesToUsers=OPS
```

Resetting Passwords

Passwords in the OTM database are encrypted and are not presented as clear-text. If you lose a password for a user and need to reset it, you can log in as an Admin user for that domain or DBA.ADMIN and reset their password using the User Manager.

If the Admin user password is lost, or if you want to reset more than one password, you can do the following:

26. Log into the application server machine as the OTM user
27. Setup your environment by running `<otm_install_path>/install/gc3env.sh` on UNIX or `<otm_install_path>/install/gc3env.cmd` on Windows.
28. CD to `<otm_install_path>/oracle/script8`.
29. Run `update_password.sh` on UNIX or `update_password.cmd` on Windows. Enter the following parameter values when prompted:
    a. Enter the glog properties path. Enter the directory where glog.properties is, e.g. `<otm_install_path>/glog/config`.

---

[7] Always staged in a cloud installation

b. Enter the user name for which password needs to be updated. Enter one or more users separated by commas, or 'all', e.g. GUEST.ADMIN,SERVPROV.ADMIN.

c. Enter a new password.

d. Enter one or more passwords to match against separated by commas or "all". The matches must be exact, and unencrypted password values entered will not affect encrypted passwords.

For example:

1. `$./update_password.sh.`
2. Enter the glog properties path: *`/opt/otm/glog/config`*.
3. Enter user name for which password need to be updated: *GUEST.ADMIN*.
4. Enter new password: *foobar*.
5. Entering one or more passwords to match against separated by commas: *all*
   … will change GUEST.ADMIN's password to "foobar" regardless of the current password.
6. `$./update_password.sh`
7. Enter the glog properties path: *`/opt/otm/glog/config`*.
8. Enter user name for which password need to be updated: *all*.
9. Enter new password: *foobar*.
10. Enter one or more passwords to match against separated by commas: *oldfoobar,DEFAULT CHANGEME,DEFAULT*
    … will change ALL passwords to "foobar", but only if they are currently set to "oldfoobar" or "DEFAULT".

    Note: When running this procedure for all users, the application server's 'system' user password is changed as well. If you had previously changed this, you will need to change it back or change it as noted under the section Application Server users below.

OTM Tomcat Servlet Container Default Users

User ID	**Description**	**Required**	**Can be deleted**	**Notes**
OTMApp	A special user that is used internally by Tomcat web container for authentication and authorization. This is used for Oracle WebLogic Application server to communicate with Tomcat.	Yes	No	The actual user ID can be changed. Please see the instructions below to change the password

Changing the Signed Servlet User's Password

1. Choose a new password and encode it using the instructions in the "Encoding Values in glog.properties" section elsewhere in this document. These instructions will refer specifically to the plain-text and encoded password; please be careful to use the correct value as indicated.

2. On every web and application instance edit the file `<otm_install_dir>/glog/config/glog.properties` and change the line containing "glog.signedServlet.password" to be the following:
   - `glog.signedServlet.password={e<encoded-password>`, where "`<encoded-password>`" is the encoded value of the new password.
3. On every web instance edit the file `<otm_install_dir>tomcat/conf/tomcat-users.xml` and change the following line:
   - `<user username="OTMApp" password="password" roles="OTMApp"/>`, where "password" is the clear-text value of the new password.
4. Restart the OTM servers.

Note that the OTM Tomcat Servlet Container User password can also be changed or specified via a JVM system property. The JVM system property is GC3ServletPassword, and its use would be: -`DGC3ServletPassword=<encoded-password>`. This would need to be specified in every `weblogic.conf` and `tomcat.conf` file as a `jvm.arg`.

The OTM Tomcat Servlet Container User ID defaults to OTMApp. However, this can be changed using a JVM system property and a glog property. The JVM system property is GC3ServletUser and its use would be `-DGC3ServletUser=<DIFFERENT_USER_ID>`. The glog property is `glog.signedServlet.user` and its use would be `glog.signedServlet.user=<DIFFERENT_USER_ID>`. The username and roles attributes in the `<otm_install_dir>tomcat/conf/tomcat-users.xml` would also need to be changed to this `<DIFFERENT_USER_ID>` as well.

Encrypting the Signed Servlet Password Stored in tomcat-users.xml

On installation, credentials for the signed servlet user, OTMApp, are provided in `<otm_install_dir>tomcat/conf/tomcat-users.xml` as plain text. To better secure this file and store encrypted credentials,

1. Edit the server.xml file in <otm_install_dir>/tomcat/conf. Look for an XML element of the form

   ```
   <Realm className="org.apache.cataline.realm.UserDatabaseRealm"
   resourceName="UserDatabase"/>
   ```

   and add SHA as a digest attribute:

   ```
   <Realm className="org.apache.cataline.realm.UserDatabaseRealm"
   resourceName="UserDatabase" digest="SHA"/>
   ```

   Save the file.
2. Run the Tomcat utility script `<otm_install_dir>/tomcat/bin/digest.sh –a SHA <password>`, where `<password>` is the text password for OTMApp. The portion of the output after the colon is the encrypted representation of the password.
3. Edit the tomcat-users.xml file in `<otm_install_dir>/tomcat/conf/tomcat-users.xml`. Find the `<user>` XML element with attribute `username="OTMApp"` and replace its `password` attribute value with the encrypted password.
4. Restart the web server.

DBA.OPS User

For many implementations, the personnel responsible for monitoring and configuring the physical infrastructure of the system differ from the personnel responsible for business operations. After installation, the DBA.ADMIN user fulfills both roles. It can monitor diagnostics, configure system resources, schedule planning processes and monitor operations data.

An optional user, DBA.OPS, is available to create a login that monitors and configures the infrastructure without having access to operations data or control. This user is assigned a user role of **OPS**. The OPS user role is limited to DEFAULT VPD access (which effectively constrains DBA.OPS from viewing operational data) and a new **OPS** access control role. The OPS access control role is limited to data source management, scalability management, property management, diagnostic pages, log monitoring and cleanup processes.

To install the DBA.OPS user and associated roles, perform the following steps:

1. Log into the application server machine as the OTM user
2. Setup your environment by running `<otm_install_path>/install/gc3env.sh` on UNIX or `<otm_install_path>/install/gc3env.cmd` on Windows.
3. CD to `<otm_install_path>/oracle/script8`.
4. Run the following command, all on one line:

```
java -Ddbserver=<Database Host> -Dglog.database.sid=<Database Service>
-Dglog.database.port=<Database Port> -Duser.home=../../config
-DtranslateErrors=false -Dglog.propertySets.enabled=false
glog.database.admin.MultiCSV -command ii -connectionId dbaglogowner
-dataDir ./content_ops/ -tableList ./content_ops/csv_ops_tables.txt
```

5. Run the `./update_password.sh` script, specifying `../../config` for the properties path. DBA.OPS for the user, CHANGEME for the password match and dbaglogowner for the connection ID. This will modify the default password for DBA.OPS.
6. Restart all servers

OTM Application Automatic User Creation

In the OTM application there are post-install automatic user creations that can occur during different circumstances. These will occur when a new business domain is created or when a new service provider is created.

The new user that is created during a new business domain creation will have a user ID in the format of `<NEW_DOMAIN_NAME>.ADMIN`. This user will have the ADMIN user role, and this user is required for administration responsibilities in that business domain. There are also other assumptions in the application that there is an `<DOMAIN_NAME>.ADMIN` user. This user will have the default password of 'CHANGEME'. There is no way to disable this user from being created automatically in the application. It is strongly recommended to immediately log into the application as this user and change the password.

Note that the new user, by default, is not reserved. To limit modification of the new ADMIN user, you can mark the user as reserved by setting the following property:

```
glog.realm.adminUsersReserved=true
```

The new user that is created when a new service provider record is created through the application will have a user ID in the format of `SERVPROV.<CURRENT_DOMAIN>-<NEW_SERVPROV_XID>`. If a user already exists with this user ID then an exception will be thrown because there is an incorrect user association record tying the existing user to the service provider. This new service provider user will have the limited SERVPROV user role. This user will have a default password of CHANGEME. It is strongly recommended to immediately log in to the application as this user and change the password.

This automatic user creation during a new service provider being created can be disabled via the following property[8]:

```
glog.servprov.autoCreateUser=[true|false] (defaults to true)
```

Application User Passwords Restrictions

> **Note:** Please note that there are forbidden characters that cannot be used for any application users' passwords. The following '#', '[', ']' characters cannot be used in passwords for all application users.

If there authentication issues even when the forbidden characters are not used, please ensure that the encoded or encrypted value of the password also does contain these characters.

Cleartext Passwords

The following files could have cleartext passwords in them. Proper OS-specific measures should be taken to ensure that only privileged users have read-access to these files:

- `<otm_install_path>/glog/config/glog.properties`
- `[WebLogic] <otm_install_path>/weblogic/weblogic.conf`
- `[Tomcat] <otm_install_path>/tomcat/bin/tomcat.conf`

Encoding Values in glog.properties

Password values in the `<otm_install_path>/glog/glog.properties` file can be encoded. To do so, do the following:

1. Log into the server as the user that the OTM application server runs as.
2. Run the following commands (note the period at the start of the first line):
   - `. <otm_install_path>/install/gc3env.sh`
   - `java glog.util.appclass.Base64Encoding SOME_VALUE`
3. The encoded value will appear on the screen between angle brackets ("<" and ">"), for example the value "CHANGEME" is encoded to "`Q0hBTkdFTUU=`":

   ```
   $ java glog.util.appclass.Base64Encoding CHANGEME
   <Q0hBTkdFTUU=>
   ```

4. Use the encoded value in the glog.properties file by placing a "{e" prefix on it, for example:

   ```
   glog.database.password={eQ0hBTkdFTUU=
   ```

   Note: The "{e" prefix syntax only works in the `glog.properties` file.

Trusted Hosts

In certain parts of the application OTM allows users to input text that will be used to create URLs. These URLs may link to other websites not hosted on OTM. An example would be a package tracking page for a parcel carrier. However linking to external websites presents a potential security threat. Therefore OTM has the concept of "Trusted Hosts". These hosts are defined in a property and if a user enters a URL that is not defined in such a property OTM will not display the URL as a link.

---

[8] This can be set in the `glog.properties` file or the `APP_CUSTOM` property set.

- Trusted Hosts must be specified in `glog.web.security.trustedHost.`

In certain instances, such as invalid redirect URLs, OTM will throw a security exception.

Trusted URLs are used in:

- Text fields where the "`displayAsLink`" attribute is set to true.
- Remarks where the remark qualifier is set to "URL".
- Protecting the URLs OTM re-directs to after a user logs in.

Logging

There is logging capabilities on all of the tiers of the application technology stack. It is recommended to review the correct documentation for that specific component on security concerns around their individual logging capabilities. Please see the OTM Administration Guide for which files need to modified to enable this logging and for links to the individual components' logging documentation.

OTM Application Log Files

The OTM applications have the ability to enable application specific debug logging on the application server and web server tiers. Most of this debug logging is helpful to enable during service request issue diagnosis. However, this logging is bad for performance and could expose important sensitive data to flat log files. In order to obtain optimal performance and prevent information leakage, it is highly recommended to keep all enabled log IDs to a minimum in a production environment.

Default Log files

Log	Filename	Description
SYSTEM	glog.app.log	An application server side log file that contains all of the application default enabled log IDs.
WEB	glog.web.log	The web container side log file that contains the enabled log IDs logging.
EXCEPTION	glog.exception.log	An application server side log file that contains all of the application exceptions and the full associated stack trace.

Specific application log IDs that are enabled could be logging and exposing information about the actual system like URLs, user names, and machine names. These important log IDs and log files should be safe-guarded. However, there are occasions that these log IDs should be enabled and are necessary, though. See the online help for enabling and disabling log IDs.

OTM Server Level Log Files

At the server level there are additional log files that are important and contain important information. These are external to the applications and are generated and logged to by external tools that the application stack relies on.

Log	**Filename**	**Description**
Application Launcher Logs	`startup.log.0` `shutdown.log.0`	Log files used by the external application launcher. The file name and directory is configurable.
Console Logs	`console.log.0`	Log file used for all console output, the JVM garbage collection output, and any application print outs.

Please note that .0 signifies the most current file. Older log files will be rotated and will have a higher number extension.

Oracle HTTP Server Log Files

OTM uses Oracle HTTP Server for web-tier services. Oracle HTTP Server has its own set of log files comprising access logs and error logs, as well as customized logging. These log files can grow very large; please see the Oracle HTTP Server documentation for instructions on how to configure these log files.

Configuration Files

There are different configuration files that exist within the applications as well as for the technology stack components that are utilized by the applications. The configuration files can control application behavior, but can also contain important sensitive data. The sensitive information within these files could consist of URLs of different servers, different user names and passwords, and various security related settings. It is extremely important that at least the following list of files are safeguarded and protected at the server's operating system level. This should be done by utilizing the host operating system's recommendation on role based access security.

The following lists all of the important files containing sensitive configuration data. The individual technology components documentation should also be reviewed for additional security measures that should be taken for securing their additional configuration files.

OTM Configuration/Property Files

File Path/File name	**Description**	**Will/Could contain**
`<OTM_HOME>/glog/config/glog.properties`	The main configuration/properties file for the OTM, GTM and FTI applications. This is the only properties file that should be changed.	Various URLs for the application server and third party servers Various Ports Database Users Database User passwords Application version

File Path/File name	**Description**	**Will/Could contain**
`<OTM_HOME>/weblogic/weblogic.conf`	The configuration file used for the application launcher to start WebLogic.	Environment variables Product Directory information Launcher host and port JVM system properties and arguments WebLogic arguments User system password WebLogic instance name WebLogic URL
`<OTM_HOME>/tomcat/tomcat.conf`	The configuration file used for the application launcher to start tomcat.	Environment variables Product Directory information Launcher host and port JVM system properties and arguments Encoded passwords Tomcat arguments
`Oracle_OTM_v6.3_GA_InstallLog.log`	A log file that is generated from the prompts when installing the OTM application. It is located in the $OTM directory if the install was at least started successfully, and in the user's home directory otherwise.	Directory Paths Environment variables Server names Server URLs Third Party server information

Tomcat

File Path/File name	**Description**	**Will/Could contain**

File Path/File name	**Description**	**Will/Could contain**
`<OTM_HOME>/tomcat/conf/server.xml`	The configuration file for the Tomcat servlet container	IP Address for the web container and associated ports Context Path
`<OTM_HOME>/tomcat/conf/tomcat-users.xml`	A Tomcat configuration file for default users that defines them and their roles	User name Password Role
`<OTM_HOME>/glog/gc3webapp/WEB-INF/web.xml`	A Tomcat configuration file for web applications.	Session Timeout Security Role Mapping Security Filters for Parameter Validation, Cross Site Request Filter, etc. Servlet mapping

WebLogic

File Path/File name	**Description**	**Will/Could contain**

`<OTM_HOME>/weblogic/domains/otm/config/config.xml`
`(config.xml.fresh)`
	The configuration file for the Oracle WebLogic Application Server	Server name
		Listening ports
		IP Address(es)
		User name
		Password
		Key Store Information
		Security Configuration Information

Oracle Database

File Path/File name	**Description**	**Will/Could contain**

File Path/File name	Description	Will/Could contain
tnsnames.ora	The tnsnames.ora file contains database connection information so the database client can communicate with the database server.	Database Server Name Database Port IP Address(es)
init.ora	The init.ora file contains configuration parameters for the Oracle database.	IP Address(es)

Configuring Outbound Connections to use a Proxy Server

OTM can be configured to allow outgoing connections to be made using a non-authenticating HTTP proxy server. Once the proxy server is defined, you must define which connections will not go through the proxy server, as the default will be to use whatever is defined via the properties for all connections.

> **Note**: For performance reasons it is highly recommended that there are no firewalls or proxy servers sitting between the application server and database.

In order to set up to use a proxy server you need to edit the glog.properties file on the OTM application server. This file is in the `<otm_install_path>/glog/config` directory. List the proxy server name and port number in the following two properties:

```
glog.integration.http.proxyHost=otmproxy.company.com
glog.integration.http.proxyPort=8080
```

The value for proxyHost can be a server name or an IP address. The value for proxyPort must be a number. Then, define which servers should not go through the proxy server. Multiple values can be specified, separated by a '|' character:

```
glog.integration.http.client.nonProxyHosts=internal-mail-
host.company.com|*.local.company.com|192.168.101.*
```

As shown in the example above, an '*' may be used as a wildcard in machine names and IP addresses.

This will cover all protocols other than FTP, which has its own set of properties:

```
glog.integration.ftp.proxyHost=otmproxy.company.com
glog.integration.ftp.proxyPort=8080
glog.integration.ftp.nonProxyHosts=integration.company.com|*.local.company.com
```

Here again you may indicate which machines should not go through the proxy server when using FTP.

Automated Processes

There is a Scheduled Process concept that exists within the application that allows certain application processes to run during a specific scheduled time automatically without any user intervention. Please see online help for more information about the Scheduled/Recurring Processes.

The following is a table that describes the Scheduled Processes that are installed and will run with a default installation.

PROCESS_CONTROL_REQUEST	Scheduled to run	Description
ObjectLockCleanup	Once a day, infinitely, at 05 GMT.	This process deletes OBJECT_LOCK records that are older than a defined number of days. These records are used in an OTM Scalability environment to lock business objects. This record deletion is done to prevent table record growth and performance degradation.

See the online OTM product documentation for how to disable Schedule Processes. However, it is not recommended to delete this default Scheduled Recurring Process.

Web Service Configuration

In previous releases, OTM only supported the WS-Security Username Token Profile. Furthermore, for outbound service calls the declaration of this support had to be made in the OTM database via the Web Service Manager User Interface. The inbound security capabilities of the application were not declared at all.

As of version 6.3, the application is now able to access the complete WebLogic support available for Web Service Policy (WS-Policy) assertions for the following WS-Security related features (referred to as WS-SecurityPolicy):

- Username Token Profile
- HTTPS Transport
- Message Encryption

For external clients calling OTM Web Services (e.g. IntXmlService), the WSDL available for that service will now contain the WS-SecurityPolicy assertions configured for the installation.

> **Note:** The default policy for ALL inbound web services is for the Username Token Profile transported over HTTPS but this can be configured to use a different policy (see below).

For OTM calling external web services, it will now be able to consume any embedded policy assertions (i.e. initially limited to those listed above) and ensure that the expected message security is followed.

Modify WS-Security Policy for Inbound Web Services

All JWS Web Services deployed to WebLogic can have policies attached either at build time (via the @Policy annotation), at deployment time (via WebLogic-webservices-policy.xml descriptor) or at runtime (via the Administration console).

Due to the deployment characteristics of the OTM/GTM application, attaching policies at runtime via the WebLogic console will not be supported. Also, policies attached at build time do not offer the required flexibility.

Every inbound Web Service has its own policy file named to match the service and is present in the `<otm home>/glog/glog_resources/policies` directory, where `<otm_home>` is the installation location of OTM. For example, `<otm home>/glog/glog_resources/policies/IntXmlService-Policy.xml` will contain the WS-SecurityPolicy attached to the IntXmlService.

The OTM WebLogic application server is configured to look for WS-SecurityPolicy files on the classpath. To override the policy delivered as part of the standard installation, a custom WS-SecurityPolicy file must be placed in the `<otm home>/glog/glog_resources/custom/policies/` directory but must use the same file name. For example, to override the standard policy for IntXmlService, the file `<otm home>/glog/glog_resources/custom/policies/IntXmlService-Policy.xml` must exist and contain a valid WS-SecurityPolicy.

See OASIS documentation on WS-SecurityPolicy for details on the XML syntax for policy files.

https://www.oasis-open.org/standards

There are several policy template files delivered with OTM which can be used to construct a suitable policy. These templates are basically copies of standard WebLogic policies delivered as part of the WebLogic application server. The table below matches the custom template to the corresponding WebLogic policy URI.

Custom Template	**WebLogic Policy URI**
otm-Wssp1.2-2007-Https-UsernameToken-Plain.template.xml	Wssp1.2-2007-Https-UsernameToken-Plain
otm-Wssp1.2-2007-UsernameToken-Plain.template.xml	Wssp1.2-2007-UsernameToken-Plain
otm-Wssp1.2-wss10_username_token_with_message_protection_policy.template.xml	Wssp1.2-wss10_username_token_with_message_protection_policy

In addition there is a template called “otm-disable-WSS.template.xml” which can be used to completely disable Web Service Security. This is not recommended for a Production environment but can be useful in a Development environment where there is no need to enable HTTPS.

PKI Configuration for Outbound Service Calls

If any external web service WSDL contains Policy assertions which require the use of PKI message encryption, a new Web Service Security Configuration must be configured in WebLogic. This configuration specifies that credential provider will use X509 certificates to digitally sign and encrypt messages. The default name of the configuration is assumed to be ‘webservices_wss’. If another name is chosen this must be configured as the following property on all the WebLogic application servers.

```
glog.integration.webservice.wssConfig=myCustomName
```

‘*myCustomName*’ is the name declared in the config.xml or Administration console.

To create a Web Service Security configuration in WebLogic console, complete the following:

1. Select Domain e.g. gc3v60.03.
2. Select the **Web Service Security** tab.
3. Lock & Edit.
4. New.
5. Choose unique name.
6. Click **OK**.
7. Select the Configuration name just created.
8. Select Credential Provider.
9. Click **New**.
10. Enter a unique name.
11. Class name = `weblogic.wsee.security.bst.ServerBSTCredentialProvider.`
12. Token type = x509.
13. Click **Finish**.

In addition to the Web Service Security configuration, message encryption requires access to the external services' Public Key Certificate. Therefore the WebLogic System Administrator will need to import the certificate into a trusted keystore configured in WebLogic and configure a PKI Credential Provider to reference the keystore that will contain the imported certificates.

For example, to create Credential Provider in WebLogic console:

1. Select Security Realms
2. otmrealm
3. Providers
4. Credential Mapping
5. Lock & Edit
6. New
7. Give a new name e.g. soapkimapper
8. Provider Specific
    a. Keystore provider = SUN
    b. Keystore type = JKS
    c. Keystore file name = webservices.jks
    d. Keystore Passphrase e.g. changeit
9. Save
10. Restart application server.

Web Service Endpoint Configuration

The alias used to store the certificate in the *webservices.jks* must be associated to every external service endpoint URL supported by the external server. This ensures that the correct certificate is used for encryption/signing of the message.

The OTM/GTM System Administrator must set the following property for each endpoint URL. For example, the property for alias *myalias* and endpoint URL the "https://myserver/services/myEncryptionService" would be:

```
glog.webservice.pki.alias.myalias=https://myserver/services/myEncryptionService
```

SMTP Authentication

If an SMTP server requires authentication this can be specified along with the name of the SMTP server itself. Wherever you specify an SMTP host you can also specify a username and password as follows:

- `username/password@smtp.company.com`: The supplied username and password are used to authenticate with the SMTP host.
- `username/{epassword@smtp.company.com`: Prepending "`{e`" to the password value indicates that the password is Base-64 encoded. The value will be decoded before being sent to the SMTP host. See the "Encoding Values in glog.properties" section elsewhere in this document for more information on encoding passwords.
- `smtp.company.com`: either no authentication is needed or authentication is accomplished via IP address.

Oracle HTTP Server & Tomcat Integration

Oracle HTTP Server is configured to talk to Tomcat (the Java servlet container) via the standard Apache module `mod_proxy`. The `<Connector>` element in the `server.xml` configuration file includes the name and the port number of the web server that is returned to the client, which provides a small amount of additional security against a man-in-the-middle attack. However, if the same web server instance is accessed via two different names (e.g. an internal machine name and an external machine name) then these values will need to be cleared out, otherwise one of the access methods may fail. The risk posed by removing these parameters is very small.

To remove these values:

1. Back up the existing "`server.xml`" file.

```
$ cd <otm_install_dir>/tomcat/conf
$ cp server.xml server.xml_20130417_2156
```

2. edit the "`server.xml`" file
3. locate the <Connector> element and remove the proxyName and proxyPort attributes:

Before

```
<Connector address="localhost" port="8009" protocol="HTTP/1.1"
           connectionTimeout="3600000"
           proxyName="otmweb.company.com" proxyPort="80"
           URIEncoding="UTF-8" />
```

After

```
<Connector address="localhost" port="8009" protocol="HTTP/1.1"
           connectionTimeout="3600000"
           URIEncoding="UTF-8" />
```

4. Restart the web server instance.

Communicating between OTM Components

OTM's architecture requires different components to communication with each other. Most of this communication will be over a network. Certain steps should be taken to protect the communication channels.

Since OTM is a web application users access it using a browser. In this case the browser makes HTTP request to OTM's web server. By default this communication is over HTTP which is a plain text and unsecured protocol. If OTM will be accessed over an un-trusted network it is more secure to access it over HTTPS. By using HTTPS all of the communication between the browser and the web server will be encrypted and therefore secure.

Once the request reaches the web server it will either be handled by the web server or forwarded to Tomcat. Requests for images, CSS, and JavaScript will be handled by the web server and requests for servlets will be forwarded to Tomcat. This forwarding is done using a module called mod_proxy. Mod_proxy forwards the request from the web server to Tomcat. This communication is done over HTTP and is therefore insecure. However in most instances the web server and Tomcat are located on the same physical machine so requests forwarded to the localhost will not travel over the network. In some rare cases the OTM application on Tomcat could make a direct call to the database.

Once the request reaches Tomcat it will either handle the request itself or require information from WebLogic. When Tomcat makes a request to WebLogic it is done through a T3 connection. By default a T3 connection is not secure. Please consult WebLogic documentation on how to create a secure T3 connection if necessary. WebLogic also initiates connections with Tomcat using the OTM signed servlet. In this case an object on WebLogic will be serialized and transferred to Tomcat over HTTP using Basic HTTP Authentication. Tomcat can then de-serialize the object and continue processing. In this case the HTTP connection is unsecure but the payload of the request will be secure due to the Basic HTTP Authentication.

When WebLogic processes a request it will often need to make a connection to the database. In most implementations WebLogic and the database are located on separate machines however they are on a trusted private network. By default the JDBC connections are not secure. Please consult database documentation for details on how to create a secure JDBC connection.

Integrating with Other Components

Oracle Components

Oracle eLocation Server (MapViewer & Spatial)

The OTM and GTM applications support the use of external geomapping services. This communication does not support HTTPS. Not supporting HTTPS is both a limitation of eLocation as well as OTM. This communication also does not require a username or a password.

Embedded Oracle BI Publisher

The OTM and GTM applications embed Oracle Business Intelligence Publisher 10g jar files. This essentially allows a standalone version of BI Publisher to run in the same JVM as these other applications. These jar files for BI Publisher are used for report and document generation from within the application.

The embedded BI Publisher communicates with the Oracle Database by using one of the PRIMARY_JTS connection pools that is created and used by the OTM application. The reporting capability within the applications can essentially tie up a PRIMARY_JTS database connection when running a long report. A large or long running report may actually not be created if the report takes longer than the connection's transaction timeout.

The touch-points between OTM/GTM and embedded BI Publisher are just direct API calls. Since the jars are embedded within the applications, there is no need for external system calls. There is also no special authentication needed between OTM/GTM and the embedded BI Publisher. A user is authentication and authorized within the OTM/GTM for the report functionality. There is no external communication between OTM/GTM and the embedded BI Publisher so there is nothing to be secured.

There are not different users and passwords stored for the embedded BI Publisher, and there is no clear text transmittal of information dictated by the embedded BI Publisher.

External Oracle BI Publisher

To leverage BI Publisher 11g reports, the OTM and GTM applications must interact with an external BI Publisher server or server farm. Each external BI Publisher instance is represented by a Report System record that holds connection, authentication and report location information. Requests for reports are made via web service calls to the report system, including authentication information with each call.

When using an external BI Publisher server, the data source model on the server should respect the user selection passed in the P_DBCONN_TYPE report parameter. If set to OLTP, the report should run against the online, transactional database; if ODS, the report should run against the offline database. To better secure each of these data sources, a report can issue calls to VPD (Oracle's Virtual Private Database) providing the OTM user and user role passed in the P_GL_USER and P_ROLE_ID report parameters, respectively. VPD enforces data security for each OTM user and role. For more information, please consult the Report Designer's Guide.

Oracle OBIEE/FTI

The primary touch-point between the OTM application and FTI application are performed by the Oracle Data Integrator when data is moved from the glogowner schema to the hdowner schema. The other touch-points are URLs that are used to navigate between the separate applications.

Authentication is required for the FTI application. There are FTI only user accounts. These accounts can be created and maintained within the application itself. These users can not be used within the OTM application.

A simple custom secure SSO solution has been implemented for OTM end users to be able to seamlessly navigate from the OTM application to the FTI dashboards. Once users are logged-in to OTM and they use the FTI dashboards they will not be prompted to log-in again. An external or third party SSO solution is not required for customers to be able to seamlessly navigate from OTM to FTI, but it is recommended from a security perspective.

In order to enable the seamless navigation from OTM to FTI without an external SSO solution, this property needs to be set on both the OTM application server and web server (or in the `CUSTOM` property set).

```
glog.security.userSession.enabled= [true|false] with a default of false
```

The GL_USER_AUTH records that are created for the custom SSO solution will be removed when a user logs out, the web server session times out, or an application server is restarted. In the event of a web server or application crash, a recurring process can also be scheduled to delete any orphaned records.

The Role ID field on the OTM User Manager controls the role(s) a user will have within OBIEE/FTI. Since, these user accounts are shared by OTM and OBIEE/FTI any change to the Role ID field in the OTM User Manager will require a restart of the OBIEE server for the changes to be effective within OBIEE/FTI. See the OTM online help to determine which Role ID to use.

EBS (BPEL)

The OTM and GTM applications support integration with the Oracle BPEL engine. The username and password are stored in the EXTERNAL_SYSTEM table. The communication is a web service call so it can be secured using the recommended security for Oracle BPEL.

Third-Party Components

The OTM and GTM applications support integration with third party components for rating and distance calculations. Most of these communications between the application and these third party components are direct java socket connections based on URLs that are configured in the glog.properties files or property sets. There is no authentication in the third party components for these socket connections but these are only accessible by business logic. If the third party vendor has provided java APIs then these are used instead of the direct socket connections.

Third-Party Report Servers

The OTM application can utilize external report servers. The external report server can be Oracle Business Intelligence Publisher, but it doesn’t have to be. The report requests are sent via HTTP or HTTPS in the form of a URL. All information sent to the external report server appears as request parameters on the URL. The URL to the report server may be an external URL that just returns the report content or the report server may begin an interactive session. The URL for this external server and the additional parameters that are required are stored as plain clear text in the REPORT_SYSTEM table in the application database.

Operating System Interactions

Application Diagnostics

OTM has the ability to execute any OS level command through the Application Diagnostic functionality of the Static and Performance Diagnostic captures. OTM has an InvocationCheck on any external calls that are issued. These are controlled via the property “glog.invocation.appdiag”. This property is a multi-value property for a white-list of allowable external commands from APP DIAG. The * is a special value to allow anything. OTM will check the external call against a white-list of properties. If the external call is not in the white-list, then OTM will throw an exception, "This external call is not allowed from OTM". The actual user running the OS level command line from within the application will be the same OS user that started the JVM. Additionally, OTM has a check to make sure the current user running the external call from this Application Diagnostic functionality is and only is 'DBA.ADMIN'.

Web Tool Versions

The WebToolVersions diagnostic collector is a collection of the web server version and the tomcat servlet container information. This default collector does not accept any arguments from an end user, but does spawn out to a java process builder to run OS level commands. The OS level commands for this diagnostic information collector are “apache –v” and “httpd –v”. These OS level commands are not checked at the application level if the invocation is allowed. If the collector is run and the correct OS level privileges are not given to the user that launched the JVM for these commands, then a not authorized message of sorts like Permission denied will be returned from the OS and displayed in the collector output.

Export Domain

The Export Domain command line utility and application functionality will export almost an entire business level domain’s data records. This functionality spawns out to a java process builder to run an OS level command to remove an end user supplied directory on the server. This OS level command is ‘rm –rf’. This will be performed as the OS user that launched the JVM. There is no invocation check done from the application. If the OS level privileges for this directory are not correct for the JVM user, then an OS level error will occur.

SQLLDR from CSVUtil

The applications support the ability to run SQLLDR (Oracle Database Utility) from within the CSVUtil utility. This capability is inherently secure, because by default it is not enabled. Utilizing SQLLDR requires installing the utility on the web server or application server, depending on where it will be used. The OS user that launches the application JVM will also be required to have the correct permissions to the sqlldr command. The CSVUtil utility is available to be run from command line or through glog.integration.servlet.CSVUtilServlet.

Log File Rotation Utilities

In order to rotate log files, the application spawns out to the OS command line to copy log files. This capability relies on the system ‘cp’ command. The use of this command does not accept end user or system administrator arguments. The OS command will be issued as the user that launched the JVM; therefore the JVM user needs authorization for the cp command.

3. Security Features

Security Model Overview

OTM, GTM, and FTI provide built-in support for the primary security components of User Authentication, Authorization/Access Control, and Auditing. The Standard Authentication mechanism for User Authentication is implemented using the JAAS security model on the application tier (WebLogic). In addition to the Standard Authentication mechanism, the application also supports several external authentication paradigms such as IP Authentication, Single Sign-On (SSO), and Lightweight Directory Access Protocol (LDAP). Authorization/Access Control is implemented on the application tier using an extension to the Custom Realm feature of WebLogic. Further Access Control is provided on the database tier using Virtual Private Database (VPD) functionality embedded in the Oracle Database. A custom Audit capability is available for auditing particular activity on the application tier.

Authentication

Standard Authentication

Any attempt to navigate to any web page of the application redirects the user to the default login page. The login page consists of a User Name and a Password field along with a Login button. The username and password values can be included in the URL HTTP POST request parameters. If the user has not yet been authenticated or the corresponding web server session has expired, then the user will be automatically redirected to the login page.

Standard Authentication refers to the default mechanism of authenticating users that is embedded within the application. When a user logs in, the application validates the username and password against the internal user table data. As of release 6.3.2, the application now also has the capability to verify that the user ID being used to log into the application is effective and not expired. The application users are stored in the GL_USER table with the GID/XID paradigm that is standard throughout the application. The format of the user ID is typically DOMAIN_NAME.USER_ID. The GL_USER table also contains other attributes for a particular user like the encrypted password, the default user role, and nickname. Even when using an external authentication mechanism like an external SSO solution, it is necessary to have a record in the GL_USER table for the application user.

After this standard authentication succeeds there are other password policy and authentication security checks that are performed if the applications are configured to do so. These additional verifications depend on the account policies setup for the individual user attempting to login.

The password for Standard Authentication is stored in the GL_PASSWORD column hashed using a one-way encryption algorithm with a SALT and a configurable number of iterations. The SALT is unique per user and is regenerated on user creation or password reset. The iteration count is configured globally, but stored per user. Modification of the iteration value will result in re-hashing of the password with the new count next time a user login is performed in the web UI. See Appendix A: List of Password Encryption Properties for a list of configuration properties.

User Account Policies

The application provides the ability to set up different account policies for each individual user. The Account Policies provide control over password definition/renewal rules and login behavior. Account policies allow configurability of the following password rules:

- Password Rules: validation rules for password strength
- Duration for password expiration
- Warning period for password expiration

- Duplicate password prevention, including configurable number of historical passwords

The account policy allows you to configure the following login behavior:

- Maximum number of failed login attempts before locking the account
- Duration of the account lockout for a failed login
- Login History for auditing purposes

You can also configure how long an account may stay dormant before the account is disabled.

Account policies are stored in the GL_ACCOUNT_POLICY table. The application currently stages two Account policies during installation. These are the NO RESTRICTIONS and the STANDARD account policies. The NO RESTRICTIONS account policy is configured to have no restrictions. The STANDARD account policy is just a standard example for what an account policy may look like within an organization. These are staged data records, but are not assigned to any reserved user that OTM installs with, nor are these automatically assigned to any users that are created. It is a recommended best practice to create your own account policies that are in line with your corporation's security policies, and assign them to the application users are needed.

Password Rules

When creating an account policy, password rules should be created to ensure the strength of passwords chosen by users. The password rules are defined using a regular expression, thus supporting standard rules (i.e. alphanumeric required) as well as providing the ability to create more complex custom rules which adhere to corporate standards.

Login History

All failed attempts to login to the application are automatically logged as exceptions. When Login History is enabled, all attempts (successful or failed) are additionally logged to the GL_LOGIN_HISTORY table. The history can be viewed from within the application using the Login History user interface.

Nickname Capability

The application provides the ability to specify a nickname for users, since the default username must conform to the standard DOMAIN_NAME.USER_XID paradigm. These nicknames must be unique per application installation, and this is enforced by an application check that is performed during user creation and modification. In order to implement Single Sign-On (SSO) it may be necessary to enable the nickname feature since a common practice with SSO is for the username to be the user's email address.

In order to support direct login with this nickname, it may also be necessary to set the following in the glog.properties file on the web tier (or in the `WEB_CUSTOM` property set).

```
glog.webserver.login.allowMixedCaseNickname=true
```

Enabling this property will prevent the automatic uppercasing of lowercase characters entered in the User Name field of the standard Login page.

While these nicknames need to be unique per application instance, case sensitivity is not taken into account. OTM is case sensitive. So there is a potential that two different OTM users could have different case nicknames. While this is fine with-in the OTM / GTM / FTI applications because case matter, other external applications or SSO solutions may not differentiate between the users nicknames because they are case insensitive.

If you use nicknames and concerned about potential case sensitive issues, it is recommended to manually review nicknames being used in the OTM application and fix any to make sure they are case sensitive.

Then set the following in the glog.properties file on the app tier and restart the application.

```
glog.realm.strictNicknameCheck=true
```

Usage: `glog.realm.strictNicknameCheck=(false)[true|false]`

Disabling Application Users

Application users can be deleted from the system when the account is no longer valid. Before deleting the user, any Recurring Processes for that user should be assigned to another user. All records which have a Foreign Key relationship to that user will also have to be reassigned or deleted.

However, as of release 6.3.2, application user IDs can now be set to become effective and expired on specific dates. This is the preferred recommendation for disabling application user accounts. The effective and expiration dates can be set on the User Manager for individual users by a user that has the correct Administration access. These fields are not required. Please also note that the actual date of the effective date is the beginning of the date in the application server time. The actual date of the expiration date is the end of the date in the application server time. If the application user log-in is not effective or expired the user will not be able to log into the application. It is also recommended to check that the application user Id that is set up to expire is not a contact required in the application and also does not have a scheduled recurring process.

Disabling Manual Login for ADMIN Application Users

The OTM and GTM application supports the ability to disallow any <DOMAIN_NAME>.ADMIN user from manually logging in from the login page except for the DBA.ADMIN user. To disallow any <DOMAIN_NAME>.ADMIN user, a read-only property value needs to be set.

```
glog.webserver.login.disallowAdminUsers=[true|false]
```

The default value for this property is false which means <DOMAIN_NAME>.ADMIN users will be able to manually log in. Disabling Manual Login for Application Users

Disabling Manual Login for Any Application User

The OTM and GTM application supports the ability to disallow any user from manually logging in from the login page. To disallow any user, a multiple value read-only property value needs to be set for the particular user gid that should not be allowed to login.

```
glog.webserver.login.disallowedUsers=<GL_USER_GID>
```

The default values for this property include most of the non-required users and non-manual log-in users.

IP Authentication

IP Authentication refers to the ability to authenticate an application user solely based upon an IP address. This capability is disabled by default. The IP authentication logic is checked before the encrypted password values are matched. There are currently two ways that IP authentication could actually authenticate an application user correctly. One of these IP authentications is if the IP address from the remote request matches the IP address of the current application server JVM, or if it matches the IP address of localhost, 127.0.0.1. Any valid application user ID could be used and can get authenticated correctly. The other IP authentication mechanism is if there is an external system that is configured within the application that has a matching IP address to the authentication request. This IP authentication mechanism will try to match the application user authentication request against a client's configured external system's IP address. The current application user name and domain will be used to try to match against an external system that is configured with the same IP address, with or without a user ID, and with a specific domain name or a PUBLIC domain name. The success or failure of this IP authentication for the application user would depend on how the external system is actually configured.

```
glog.realm.allowLocalIPAuthorization=[true|false] (false by default)
```

External Authentication

The application provides a capability to use external authentication mechanisms. These external mechanisms include utilizing a Lightweight Directory Access Protocol (LDAP) and a Single Sign-On (SSO).

LDAP

The applications support integration within an LDAP server for authentication, though it is disabled by default upon installation. When the applications are configured to use LDAP, the login process is still initiated via the OTM login page. The user ID and password credentials that are provided during login are used to authenticate against the LDAP solution that is configured. The correct namespace will also need to be selected from the Login web page for proper authentication to occur. The namespace options that are available to be selected are based on the ldap.namespace properties that are configured within the application. The external authentication would then be done against the namespace's correct LDAP solution via a standard J2EE LDAP API. The standard APIs that are used in the Applications are the `javax.naming.directory.InitialDirContext` constructor and the getAttributes API. The applications user name attributes for "gl_user" are then extracted from the attributes that are passed back from the LDAP solution. These attributes will need to be configured correctly per instructions provided by the LDAP provider. The gl_user attributes that are extracted from the LDAP then must map to the actual GL_USER ID. The GL_USER is then retrieved internally to the application and is used as the application user.

There are many different options for external LDAP solutions. The application is certified with Oracle Internet Directory (OID), a component of the Oracle Access Management ( ) product suite. Consult the OTM OAM Integration Guide for more details.

Advanced Configuration: LDAP

LDAP stands for the Lightweight Directory Access Protocol. It is important to remember that LDAP is in essence a protocol – a common language that various directory products can speak in order to communicate with users and applications -- and other directories. The TCP/IP-based LDAP protocol contains messages allowing an LDAP client (an application or user) to connect to, search, add to, delete from, and modify an LDAP server (the directory).

Overview

LDAP clients connect to an LDAP server as a user in the directory (sometimes called binding to the directory). The LDAP server may choose from a number of authentication protocols (see below) to validate the identity of the connecting user. Once connected, the LDAP user can search or modify the directory (if permission has been granted to perform these operations). In our case, Oracle Transportation Management is the LDAP client. A customer's LDAP directory is the server.

LDAP represents names in a standard format – the Distinguished Name, or DN (see below for more detail on DNs). This format contains name attributes like organization, country, organization unit, etc… Moreover, these attributes are arranged hierarchically. So, there can be multiple organizational units within an organization, and multiple organizations within a country. The directory is searched and organized hierarchically.

Each name is associated with one or more directory objects. These directory objects contain attributes that can be used for authentication, for populating databases, for applications, or any other number of uses.

Because of the focus on clients, the LDAP community also defined standards for the string representation of DNs (RFC 1779), search filters (RFC 1960), and attribute syntaxes (RFC 1778), for a C language-based API (RFC 1823), and for the format of URLs for accessing LDAP services (RFC 1959).

LDAP Schema

A directory schema specifies, among other things, the types of objects that a directory may have and the mandatory and optional attributes that each type of object may have. LDAP (version 3) defines a schema (RFC 2252 and RFC 2256) based on the X.500 standard. The schema includes common objects found in networks, such as countries, localities, organizations, users/persons, groups, and devices. It also defines a way for a client application to access the server's schema so that it can find out the types of objects and attributes that a particular server supports.

The LDAP schema has become one of the basic ways that different LDAP directories can interoperate. Corporations use the schema to store user, profile, organization, contact, and location information. Oracle Transportation Management relies on the username (and for local authentication, password) attributes. Currently, Oracle Transportation Management requires that the username be part of the distinguished name, and requires that the "GLUSER" attribute be added to each user object.

LDAP in Oracle Transportation Management

LDAP is used by Oracle Transportation Management to allow users to log into Oracle Transportation Management using standardized LDAP names, instead of, or in addition to Oracle Transportation Management usernames. Oracle Transportation Management allows authentication to be performed by a remote LDAP server -- a more secure, more centralized approach. Corporate users can securely log onto Oracle Transportation Management with the LDAP login names that they are used to and use everyday.

Oracle Transportation Management allows the LDAP users to be mapped to Oracle Transportation Management users in the LDAP directory itself. This way, Oracle Transportation Management permits a single Oracle Transportation Management user to be mapped to multiple LDAP users. This allows a generic Oracle Transportation Management user such as "GUEST" (GUEST may have primarily read-only access to limited information) to a large group of users, without giving the password to all of these users. LDAP allows Oracle Transportation Management user, security, and policy information to be centralized in one place – the LDAP directory.

In addition, Oracle Transportation Management contains multiple directory support (see NameSpaces below). This allows multiple LDAP directories to be consulted to find names. For instance, a logistics company may wish to authenticate shippers with a local LDAP directory, and service providers with an external LDAP directory. In addition, NameSpaces allow the same directory to be looked up using (for instance) different authentication protocols, or different branches of the same directory tree.

Oracle Transportation Management allows you to choose which directory is consulted upon login. Alternatively, a default search order can be configured, so that multiple directories can be looked up in turn to authenticate a login.

Limitations

Oracle Transportation Management does not support group authentication, group membership testing, or distribution lists. It simply supports username authentication upon login. Oracle Transportation Management does not support the use of user profile attributes from LDAP such as language, time zone, e-mail addresses, or any other user preferences. All user parameters are controlled within Oracle Transportation Management – the GLUSER attribute provides the linkage between an LDAP user and an Oracle Transportation Management user. The LDAP directory itself must be modified to contain the mapping (this in itself could be considered a limitation).

Oracle Transportation Management does not use the LDAP directory to store contact information, e-mail addresses, location information, or other centrally maintained pieces of information vital to large corporations using our product.

LDAP Server

Oracle Transportation Management does not contain an LDAP server. Many customers already have a corporate LDAP Server, and one of the major goals of this feature is to allow corporations to centralize user and security information -- not having it scattered in many different places. Oracle Transportation Management can be configured to talk to an LDAP server by defining a NameSpace.

Single Sign-on Support

With LDAP, Oracle Transportation Management supports the ability to have users login to Oracle Transportation Management using LDAP usernames that they are familiar with. Sometimes, third-party packages allow user to log into the package once. Thereafter, that person will not have to log in to each individual application that may be accessed subsequently. LDAP, as a technology, is often used in the implementation of single sign-on solutions.

Definitions

NameSpace

A NameSpace is where information about an LDAP directory is stored in Oracle Transportation Management. It is an Oracle Transportation Management term, and not an "LDAP term". It contains information such as the branch of the naming hierarchy to search for users, the URL of the LDAP server, the authentication methods to be employed, and the protocol version to use. A user logging in can choose which NameSpace to use for authentication, or use the default. In the default case, Oracle Transportation Management allows multiple directories to be looked up one by one until successful authentication has taken place.

Distinguished Name

A Distinguished Name (or DN) is the standard format for naming within LDAP directories. Quite simply, a distinguished name is an ordered list of naming attributes. These attributes are often syntactically organized into a single string such as "cn=John Doe, ou=Marketing, o=Oracle, c=US" (see RFC 1779). This name consists of Common Name, Organizational Unit, Organization, and Country attributes. The directory uses these attributes to arrange objects in the directory hierarchically. So, there can be multiple organizational units within an organization, and multiple organizations within a country. This way, different branches of the LDAP "tree" can be searched independently. For instance, one might want to search only names within the organization "Oracle".

Oracle Transportation Management requires that the user ID field be part of the Distinguished Name (at least externally to an LDAP client). It also requires that each LDAP user object to be authenticated with Oracle Transportation Management be populated with the GLUSER attribute. The GLUSER attribute should not be part of the Distinguished Name.

A fully qualified DN identifies the name of an object within an LDAP directory. A relative DN identifies a branch of the naming tree, but does not necessarily address a schema object.

LDAP Authentication Protocol

An LDAP Authentication Protocol is used between an LDAP client and LDAP server to authenticate a user within the directory. Oracle Transportation Management supports simple authentication (clear text username and password), and some other authentication protocols (such as CRAM-MD5). Oracle Transportation Management also supports using no authentication at all (although this is somewhat pointless). These protocols can be used in both local and external authentication (see below).

LDAP Encryption Protocol (SSL)

Oracle Transportation Management allows encryption between the Oracle Transportation Management application server and the LDAP server. This insures that password information flowing between the LDAP server and Oracle Transportation Management is not intercepted. Oracle Transportation Management uses SSL to provide this encryption. SSL is a generic transport layer encryption/authentication solution. The LDAP directory server must support SSL in order to use this feature. Although SSL can in theory be used for authentication as well, SSL is used by Oracle Transportation Management to encrypt the communication between Oracle Transportation Management and the LDAP server.

LDAP Authentication Method

Oracle Transportation Management uses two major methods of authentication. Local authentication involves searching for a name object in the directory and extracting some attributes. These attributes are in turn used to authenticate the name. Most commonly, the password attribute is used to validate entries. It's called local authentication because the validation is performed "locally" (by the client). The client logs into the LDAP directory as a sort of super-user (called the Principal). The principal user is used to look up all other users in the directory.

Oracle Transportation Management can also authenticate users by attempting to directly connect to the directory as the user in question, instead of connecting as a principal user and then performing a lookup. This allows the LDAP directory to perform the authentication at the LDAP server. It's called external authentication, because the authentication is performed externally to the LDAP client. This method insures that sensitive authentication information (such as certificates or passwords) never leaves the LDAP server. In the LDAP world, this method of authentication is often called "binding" to the server.

LDAP Protocol Version

LDAP is a TCP/IP based protocol, and this protocol has two major revisions still in use. Version 2 contains most of the basic LDAP functionality. RFC 1777 defines what is now called version 2 of the LDAP (or LDAP v2). The LDAP v3 (RFC 2251) protocol is designed to address some of the limitations of LDAP v2 in the areas of internationalization, authentication, referral, and deployment. It also allows new features to be added to the protocol without also requiring changes to the protocol itself.

LDAP Directories

Organizing LDAP Directories For Oracle Transportation Management

Oracle Transportation Management requires that the user ID field be part of the Distinguished Name (at least externally to an LDAP client). It also requires that each LDAP user object to be authenticated with Oracle Transportation Management be populated with the GLUSER attribute. The GLUSER attribute should not be part of the Distinguished Name.

Each NameSpace contains a User DN field. This contains a relative DN that identifies the branch of the LDAP tree to search for users. Oracle Transportation Management searches this branch, and this branch only – meaning no sub-branches are searched. If you wish to search for sub-branches, you must define each sub-branch as a NameSpace, and use the default search order to search for them one-by-one (see NameSpaces above).

Many directories enforce trueness to the schema defined for a particular object. This means that the object’s schema must be modified to contain the attribute GLUSER in order to have that attribute be successfully added. Sometimes, this schema checking can be turned off in the directory. Another alternative is to use an attribute that already exists in the schema (but is not populated). You can change the NameSpace configuration to define the attribute where the GLUSER information is contained, so that Oracle Transportation Management knows where to get the Oracle Transportation Management user mapping.

LDAP and the Oracle Transportation Management Login

If you have any doubt, choose Default. This most likely will be set up by the Oracle Transportation Management Administrator to serve most needs. The GC3 NameSpace allows logging in via the Oracle Transportation Management username and password (see below). The other choices represent LDAP directories that have been configured to work with Oracle Transportation Management.

The Oracle Transportation Management administrator has the ability to configure which directories are consulted when the Default option is chosen. In fact, when this option is chosen, the administrator has the ability to configure an ordered list of directories to search. Usually, this results in a successful authentication.

The GC3 NameSpace is a special NameSpace that identifies the Oracle Transportation Management realm itself (the usernames and passwords stored directly within Oracle Transportation Management). When you search the GC3 NameSpace, you are performing a search local to the Oracle Transportation Management product.

Configuring LDAP NameSpaces

The glog.ldap.properties file is read by the web server when the first user logs in. It is never read again. You can bounce the Tomcat instance (if you know how to do this) if you need to reload the properties. The application server will automatically adjust.

The following is a sample property file containing one NameSpace:

```
ldap.searchOrder=GC3, CorpDir
ldap.namespace.name=CorpDir
ldqp.namespace.CorpDir.authProtocol=simple
ldap.namespace.CorpDir.ldapUrl=ldap://localhost:389
ldap.namespace.CorpDir.principal=otmdev
ldap.namespace.CorpDir.credential=CHANGEME
ldap.namespace.CorpDir.userDN=o=Oracle, c=US
ldap.namespace.CorpDir.userNameAttribute=uid
ldap.namespace.CorpDir.glUserAttribute=gluser
ldap.namespace.CorpDir.userAuthentication=local
ldap.namespace.CorpDir.credentialAttribute=password
```

The search order parameter is global to all NameSpaces. The namespace.name parameter must precede the other NameSpace parameters. Other NameSpaces can be added below.

NameSpace Attributes

Here is a list of the attributes that comprise a NameSpace.

Attribute	**Description**	**Valid Values**
authProtocol	The authentication protocol to employ.	None simple (default) CRAM-MD5 (v3 external authentication only) DIGEST-MD5 (v3 external authentication only).
Name	Name Of the LDAP namespace – used in user interface display. Required	String (example: CompanyDir) Only alpha-numeric and underscores are allowed.
LdapUrl	URL of the LDAP server. Required	String (example: ldap://somehost:389)
Principal	User to log in as on LDAP server. Required if authentication=local	Distinguished Name (example: Uid=foo)
Credential	LDAP principal password. Required if authentication=local.	String

Attribute	**Description**	**Valid Values**
UserDN	Distinguished name of all users to be searched – the branch of the tree to search. Always specify the most "specific" attributes first – for example, supply Organizational Unit (ou) before Organization (o), which should be supplied before Country (c), etc… Required.	Distinguished Name (example: ou=people, o=acme.com,, c=US)
userNameAttribute	Name of the User ID attribute in the LDAP directory. Required.	String (default: uid)
glUserAttribute	Name of the GLUSER attribute in the LDAP directory. Required.	String (example: gluser)
userAuthentication	Type of authentication employed for this namespace.	Local = authentication based on downloaded attributes. external = an LDAP bind as the user in question (default)
credentialAttribute	Name of the password attribute in the LDAP directory. Required if authentication=local.	String (example: userpassword)
SSL	Connect to directory using SSL (true or false).	true false (default)
version	The version of the LDAP protocol (2 or 3).	2 (default) 3
ctxFactory	Java Naming and Directory Interface (JNDI) service provider to use.	String (default: com.sun.jndi.ldap.LdapCtxFactory)

The Distinguished Name is an ordered list of attributes, and the attributes must be listed by the most specific attribute first. For instance, OrganizationalUnit is more specific than Organization, but less specific than an individual user within that OrganizationalUnit. Similarly, a domain component attribute of Oracle is more specific than dc=com. When you specify a DN, you are really specifying a path from the node you are interested in up to the root of the directory tree.

Diagnosing LDAP Communication

The **LDAP** log ID is available for diagnosing LDAP connections and requests. Through this ID, each LDAP authentication request and response can be written to a system log file. For a request, this includes a list of all properties sent to the LDAP server; for a reply, it includes any error information returned by the server.

Properties sent to the LDAP server include user names and passwords. Though useful for diagnosing a bad LDAP configuration or connection, the echo of these values into the log can be insecure. The following properties control whether to include them:

ldap.debugUserNames=<true|false>
if true, user names are echoed in LDAP request diagnostics

ldap.debugPassword=<true|false>
if true, passwords are echoed in LDAP request diagnostics

Both of these properties default to false.

Common LDAP Questions

Why can't I connect using local authentication, even though the password attribute is present?

There are a number of reasons why this could happen. The password attribute may not be visible to Oracle Transportation Management, even though it may be visible to other clients. The password could be encrypted – you might want to try changing the authenticationProtocol attribute. It is possible that the directory entry is not readable by the Principal being used.

Why does authentication fail for entries not directly below the UserDN in the directory tree?

This is a limitation of the LDAP client – the benefit is rapid lookups. However, many commercial directories allow entries to be indexed and placed in a single branch. In addition, the NameSpaces feature can be configured to allow multiple branches to be searched.

I am using local authentication and my principal user uses an encrypted password, but my users use simple authentication. How do I get authentication to work?

You cannot have the Principal using a different encryption algorithm than the individual users. If you really need this, you can use external authentication and bypass the special authentication for the Principal altogether.

If you have two user groups that use different authentication mechanisms, you can set up two NameSpaces that point to the same directory, but use different authentication methods. Then, set the default search order parameter (see above) to search the two namespaces.

Single Sign-On

OTM supports SSO, where a central application (the SSO provider) authenticates users and then passes the login information to OTM, therefore bypassing the normal OTM login process. *The burden of authentication then falls to the SSO provider; OTM will not provide any.*

For SSO access to OTM, the invoking code needs to pass in the following parameter as part of the HTTP request:

```
appuid=/GUEST.ADMIN/
```

where `/GUEST.ADMIN/` is the user's GID. This can also be passed in as part of the HTTP header (see below to control this behavior). By default, the logic checks the HTTP header first and then the HTTP request.

There are several properties that control SSO. They can be set in the glog.properties file on the web server or the `WEB_CUSTOM` property set:

- `glog.security.sso=true`: is SSO allowed or not? It is false by default
- `glog.security.sso.appUidName=appuid`: to change the name of the UID field
- `glog.webserver.initial_page=url`: used if a redirect is not provided as part of the request. URL can either be an OTM servlet or a fully qualified URL:
    - o `$glog.webserver.urlprefix$$glog.webserver.context$glog.webserver.util.FrameGC3Servlet`
    - o `http://some.domain.com/some_page.html`
- `glog.security.sso.appUidLocation=X`: where X is one of the following:
    a. 1 (default): check HTTP header & then request parameter for the user name
    b. 2: check request parameter only
    c. 3: check HTTP header only
- glog.security.sso.logoutButton=[true|false]
    - o Setting this property to false will enable the logout button to appear with a link to the logout URL specified. The code will default this property to false. The glog.sso.properties file has this property commented out, but will default this property to true.
- glog.security.sso.logoutUrl=logoutUrl: the url to use for single sign-on logout
    - o If this property is not specified in the properties file, then the default OTM logout servlet will be used.
    - o An example would be: ../access/oblix/lang/en-us/logout.html

    Please note that the external SSO logout mechanism should call the OTM logout servlet to log the user out of the OTM application. Failure to do this will allow the user session to remain open until the specified time out.
- glog.security.sso.loginBackdoorName=(gc3backdoor)
    - o defaults to gc3backdoor
    - o A configurable property driven request parameter that when present in the login URL will allow the OTM Single Sign On solution to present the normal OTM Login page so that an ADMIN like user has a way to overwrite their assigned OTM user

Configuring Fusion Transportation Intelligence with OAM (SSO)

Refer to the Configuring FTI with OAM (SSO) in the OAM Integration Guide for details on setting up single-sign on for Fusion Transportation Intelligence with OAM.

Files Accessible without Authentication

The application utilizes the Oracle HTTP Server (OHS) as the web server on the web tier. The web server is used to serve up the static content (images, HTML pages, etc.) and dynamic content (Servlets running in Tomcat) for the application. Although accessing the dynamic content is protected by Standard Authentication, the static content is not. Therefore, any of the files under the Document Root (`<otm_install_dir>/web/htdocs`) are accessible without authentication and can be accessed via a web browser. The static content files consist of static html files, image files, CSS files, JavaScript

(JS) files, and XSL style sheet files. During installation the permissions on these directories are read-only and they should be maintained as such.

Staying Logged into the applications

Maintaining Session State

Since HTTP is a stateless protocol, Tomcat uses a cookie names JSESSIONID to maintain session state between requests. Please refer to the Browser Cookies Used in OTM section.

Session Timeout

The application relies on the Tomcat web container capability for Http Session Timeouts for invalidating inactive user sessions on the web tier.

This means that any user after a configurable amount of time will have to re-authenticate if the web container session gets timed out, and then they try to reach an application resource. This is configured in the web.xml file under the webserver instance installation directory `(<OTM_HOME>/glog/gc3webapp/WEB-INF/)`. The staged default of the application is 30 minutes, and it is strongly recommended to change this depending on the corporation's policy for idle timeouts.

Change the session-timeout value in the web.xml file to your desired session timeout value.

```
<session-config>
    <session-timeout>30</session-timeout>
</session-config>
```

Data Integration (Inbound) Authentication

The applications have numerous mechanisms for inbound data integrations to occur. These include web services, Oracle Advanced Queuing, and HTTP posts. These integration mechanisms vary on which tier they are deployed, but they at the very least still utilize the same standard authentication mechanisms as the rest of the application.

Web Services

There are inbound Web Service call capabilities for the applications. These web services are deployed on the Oracle WebLogic application server. In order to use these web services, calls will need to be made to the URL for the application server. It may be necessary to open firewall access to the application server to allow this capability. Please see the Integration Guide for more details on the configuring web service capabilities.

The web service capabilities do adhere to the Web Service Security Specification. This specification is an OASIS standard for defining security related information as part of a SOAP message. See http://www.oasis-open.org/. This security model is designed to support a number of Profiles where each Profile is a different representation of security claims. The Username Token Profile is, as the name suggests, a standard way of specifying user credentials i.e. username and password.

User credentials can be specified in a Username Token when calling OTM web services or included in a message when calling out to a web service from OTM.

The WS-Security token is passed in the SOAP envelope header. Following is an example of a username token with plain text.

```
<?xml version=”1.0” encoding=”UTF-8”?>
<env:Envelope xmlns:env=”http://schemas.xmlsoap.org/soap/envelope/”>
```

```
   <env:Header xmlns:env=”http://schemas.xmlsoap.org/soap/envelope/”>
      <wsse:Security xmlns=”http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wsswssecurity-secext-1.0.xsd”>
         <wsse:UsernameToken>
            <wsse:Username>GUEST.ADMIN</wsse:Username>
            <wsse:Password Type=”#PasswordText”>CHANGEME</wsse:Password>
         </wsse:UsernameToken>
      </wsse:Security>
   </env:Header>
   <env:Body>
   ...etc..
   </env:Body>
</env:Envelope>””””””””””
```

Ultimately, this will authenticate using the same authentication mechanism as the standard login. This will first validate the user name and password provided against the GL_USER table. If that authentication succeeds then the JAAS authentication mechanism will also be initiated. IP authentication can also be used in web services, if it configured.

SSL should be used to make the connection more secure. Please see the SSL/TLS Certificates Section for details on how to configure SSL.

Oracle Advanced Queuing (OAQ)

The OTM application has the ability to process data integrations utilizing the Oracle Advanced Queuing capabilities. Most of this functionality resides on the Oracle Database tier. Please review the OAQ documentation and the OTM Data Management Guide for more specific details.

In order for the inbound data integrations to work correctly, two different authentications are required. First in order for the integration data to reach the inbound database queue, a valid database user and password is required to log into the database. There are many different applications that could be used to populate these database queues, and they will require a database authentication. The applications do have a java utility class that can be used to populate the data in the queues, it is GlogOaqUtilClient. The database connection information and a database username and password are required at the command line. The password at the command line will need to be plain text. It could be run from a command line on the server that has the application server installed. For full usage details run it by issuing the following command:

```
java glog.integration.oaq.GlogOaqUtilClient
```

After the data is inserted into the associated queue tables, a database listener will attempt to populate the OTM tables of I_TRANSMISSION and I_TRANSACTION. Before the data is actually processed for these tables, the same standard authentication mechanisms will be invoked. The user name and password will first be authenticated against the GL_USER table, and then a subsequent JAAS authentication will occur. The password that is specified in the integration XML can be plain text or an encrypted password value.

The OAQ functionality is not enabled by default and requires database and application server configuration to work correctly. For more details on configuring and using OAQ, consult the Integration Guide.

HTTP Post (Integration servlets)

The OTM application has the ability to receive data integrations utilizing HTTP Post requests. There is a list of servlets that can be utilized for external inbound integration.

Servlet	Description
glog.integration.servlet.WMServlet	WMServlet is the default servlet to be used when sending the Transmission or Message XML.
glog.integration.servlet.LargeTransmissionServlet	LargeTransmissionServlet can be used for sending exceptionally large Transmission(s) into OTM. The difference with WMServlet is that parsing of the XML is handled in the servlet, and there is suppression of storing the complete transmission in the database. The individual transactions are stored in the database. **Note**: this servlet is deprecated and will be removed in a future release.
glog.integration.servlet.TransformerServlet	TransformerServlet is used to apply an XSL transformation to an XML to convert it into a valid Transmission XML. Refer to the Transform Inbound XML with XSL section in the Integration Guide for additional details.
glog.integration.servlet.DirLoadServlet	DirLoadServlet provides a faster option than WMServlet for loading data into OTM by bypassing the application server. It can be used for inserting/creating data. This is not a preferred integration method from a security perspective because it requires direct database access from the web tier. Refer to the Integration Guide for additional details.

These servlets run on the web tier, specifically within the Tomcat web container. For more specific information about the usage and details about this please refer to the Integration Guide.

The same standard authentication and authorization mechanism is used for these external Integration servlets. The user name and password will first be authenticated against the GL_USER table, and then a subsequent JAAS authentication will occur. The IP Authentication capability can also be used for these Integration servlets.

Command-line Utilities

The OTM and GTM application have command line utilities that are included in the product. Some of these are invoked by scripts that are installed with the product. Others are provided as a convenience for administrators. Access to these utilities should be protected at the operating system level.

> **Note**: These utilities are provided as-is and may be removed at any time.

CSV Utility (glog.database.admin.CSVUtil, glog.database.admin.MultiCSV)

The CSV utility class provides data content import and export functionality. This utility has command line capabilities with many different arguments and different modes for data import and data export functionality. CSVUtil does not require authentication or authorization from the command line for the data content to be imported or exported. The database connection information and authentication information is configured through properties. This utility is required by the OTM installation for importing database content during installation and patching.

DB Sid Property Display (glog.database.admin.PropertyDisplayDbsid)

The database SID property display utility displays the database sid for the connection ID that is passed in as an argument. This utility is used during install and updates for the name of the update_content log. There is no authentication or authorization for this command line utility. There are install and installed scripts that use this command line utility. The database connection information and authentication information is configured through properties.

Set User Password (glog.database.admin.SetUserPassword)

The Set User Password utility allows application users' passwords to be set or reset to CHANGEME. This utility allows for an individual user's password to be set, as well as all of the application users' passwords. This utility also allows passwords to be set or reset to CHANGEME if their current password matches the "if password" argument. There is no authentication or authorization for this command line utility. The database connection information is pulled out of glog.properties files. There are install and installed scripts that use this command line utility.

```
SetUserPassword -connectionId <data source> -users <GL_USER_GID|all> [-password
<text password>] [-ifPasswords <passwords>] [-test]
```

X Lane Session Client (glog.database.admin.XlaneSessionClient)

This command line utility allows the business API to be called to formulate regions for the application logic. This utility does require that an application user ID is specified as an argument, as well as the particular application server URL that the API will be invoked on. A password does not need to be specified for the utility if it is run from the same server. IP authentication will need to be enabled for no password to be specified.

```
-glUserGid <GUEST.ADMIN> -glPassword (if password is null, then local ip
address is used, so if ip auth is enabled, then any user will be authenticated
if run on the same server) -serverURL
```

Functional Security Migration (glog.database.security.migration.Migrator)

This command line utility is used for migrating a customer's custom functional security settings from the groupmembers table to the ACR_* tables. This was required when upgrading from older versions of OTM. The ACR_* tables were added in 6.1. This migration is invoked from installed scripts and occurs during database migration. There is no authentication or authorization for this command line utility. The database connection information is pulled from glog.properties files.

Integration Saved Query Finder (glog.integration.gidfinder.GidFinder, glog.integration.gidfinder.MultiArgsGidFinder)

These command line utilities allow integration saved queries to be executed from a command line. These utilities do require arguments for the database connection information including the user name and password. Usage:

```
java glog.integration.gidfinder.GidFinder dbHost dbPort dbSid dbUser dbPW
savedQueryGid [argName argValue]*
```

XPath Handler (glog.integration.gidfinder.XPathHandler)

This command line utility takes a filename and an XPath expression. It reads in the file, parses it, and then executes the XPath expression against the document. This utility does not do anything for the application so therefore there is no authentication. Usage:

```
java XPathHandler [filename] [xpath_expression]
```

OAQ Utility Client(glog.integration.oaq.GlogOaqUtilClient)

This command line utility provides Oracle Application Queue functionality from a command line for the OTM application. This utility allows enqueuing and dequeuing of application integrations. There is database and application level authentication. Usage:

```
java glog.integration.oaq.GlogOaqUtilClient -dbUrl <dbUrl> | -dbHost <dbHost> -
dbPort <dbPort> -dbSid <dbSid> -dbUser <dbUser> -dbPW <dbPW> -queueName
<queueName> -cmd {enq|enqueue} -xmlFile <xmlFile> | -xmlDir <xmlDir> -userName
<userName> -password <password> -refnum <refnum> -subject <subject> -extSysGid
<extSysGid> -encoding <encoding>
```

Direct Mail (glog.integration.tools.DirectMail)

This utility sends a simple text/plain message to the "to" address, from the "from" address, using the smtphost as the machine with the SMTP server running. If multipart is "true" then send a multipart message else if multipart is "false" send a text/plain message. This utility class also cleans up all event queues as well. There is no authentication or authorization for this utility class. Usage:

```
sendmessage to from smtphost multipart
sendmessage -to Recipient -from Sender -subject Subject -text Text -cc
Recipient -smtphost Host -attach Filename
```

Note: -to, -from, -text and -smtphost are required

Mail Message (glog.integration.tools.MailMessage)

Send a simple text/plain message to the "to" address, from the "from" address, using the SMTP host specified in the "smtphost" parameter. If multipart is "true" then send a multipart message else if multipart is "false" send a text/plain message. Usage:

```
sendmessage to from smtphost multipart
sendmessage -to Recipient -from Sender -subject Subject -text Text -cc
Recipient -smtphost Host -attach Filename
```

Note: -to, -from, -text and -smtphost are required.

Connection Tester (glog.integration.tools.SimpleConnection)

This command line utility establishes a simple JDBC connection based on the arguments passed into it. Syntax:

```
java SimpleConnection DRIVER URL UID PASSWORD
```

Transmission Purge (glog.integration.tools.TransmissionPurge)

This command line utility allows the deletion of i_transmission database records. If all arguments are supplied, the utility deletes records in the i_transmission, i_transaction, i_log, i_log_detail tables for transmissions where the contained transactions are older than maxTransmissionLife and fully processed (optional). There is authentication or authorization for this utility. The database connection information is retrieved from glog properties files. Usage:

```
java TransmissionPurge -connectionId <connectionId> -maxTransmissionLife
<daysOfTransmissionLife) -fullyProcessed <Y/N>
```

Database Sequence Creator (glog.oracle.java.SequenceCreator)

This utility will drop and then create sequences for every table having a single value numeric primary key. Before the dropping of the sequence number table, the utility determines the sequence creation start value from the maximum primary key value. There is no authentication or authorization for this command line utility. The database connection information is retrieved from glog properties files. This utility is deprecated and will be removed in a future release. Usage:

```
SequenceCreator <connectionId>
```

Cache Diagnostic (glog.server.cache.CacheDiagCommandLine)

This command line utility allows the application business caches to be printed out to a file or standard output. This utility also allows for the application business cache statistics to be reset. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.cache.CacheDiagCommandLine [-user <gc3 user (DBA.ADMIN)>] [-
password <gc3 password (CHANGEME)>] [-encodedPassword <encoded password>] [-
server <server>] [-file <output file or !BYDATE (STDOUT)>] [-dir
<outputDirectory>] [-loop <# of iterations (1)>] [-sleep <iteration delay in
sec (60)>] [-reset] [-cache <cache name>] [-zone <zone name>] [-maxItems
<maximum # of cache items (300)>] [-items <Cache Name | all>] [-help]
```

> **Note**: If the user name is not provided, then the utility defaults the user to DBA.ADMIN. If the password is not provided then the password will default to CHANGEME.

Connection Pool Diagnostic (glog.server.cpDiag.CPDiagCommandLine)

This command line utility allows the application database connection pools' configuration and information to be printed out to a file or standard output. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.cpDiag.CPDiagCommandLine [-user <gc3 user (DBA.ADMIN)>] [-password
<gc3 password (CHANGEME)>] [-encodedPassword <encoded password>] [-server
<server>] [-file <output file or !BYDATE (STDOUT)>] [-dir <outputDirectory>] [-
loop <# of iterations (1)>] [-sleep <iteration delay in sec (60)>] [-xml] [-
help]
```

Event Diagnostics Command Line Utility (glog.server.event.EventDiagCommandLine)

This utility allows application in-memory event queue, data queue, and Oracle Application Queue information to be printed out to a file or standard output from a command line. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.event.EventDiagCommandLine [-user <gc3 user (DBA.ADMIN)>] [-
password <gc3 password (CHANGEME)>] [-encodedPassword <encoded password>] [-
server <server>] [-file <output file or !BYDATE (STDOUT)>] [-dir
<outputDirectory>] [-loop <# of iterations (1)>] [-sleep <iteration delay in
sec (60)>] [-reset] [-maxEvents <maximum # of queued events (300)>] [-threads]
[-events] [-xml] [-memory] [-data] [-oracle] [-help]
```

Message Diagnostics (glog.server.message.MessageDiagCommandLine)

This command line utility allows application JMS message information to be printed out to a file or standard output from a command line. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.message.MessageDiagCommandLine [-user <gc3 user (DBA.ADMIN)>] [-
password <gc3 password (CHANGEME)>] [-encodedPassword <encoded password>] [-
server <server>] [-mode (topic|bean|query|cache)] [-file <output file or
!BYDATE (STDOUT)>] [-dir <outputDirectory>] [-loop <# of iterations (1)>] [-
sleep <iteration delay in sec (60)>] [-reset] [-xml] [-help]
```

Note: If the user and password is not specified then these will default to DBA.ADMIN and CHANGEME respectively.

Session Bean Performance Diagnostic (glog.server.sessionperf.SessionPerfCommandLine)

This command line utility allows application EJB Session Bean performance information to be printed out to a file or standard output from a command line. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.sessionperf.SessionPerfCommandLine [-user <gc3 user (DBA.ADMIN)>]
[-password <gc3 password (CHANGEME)>] [-encodedPassword <encoded password>] [-
server <server>] [-file <output file or !BYDATE (STDOUT)>] [-dir
<outputDirectory>] [-loop <# of iterations (1)>] [-sleep <iteration delay in
sec (60)>] [-reset] [-help]
```

Object Lock Diagnostic (glog.server.synch.object.ObjectLockDiagCommandLine)

This command line utility allows application object lock diagnostics information to be printed out to a file or standard output from a command line. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.synch.object.ObjectLockDiagCommandLine [-user <gc3 user
(DBA.ADMIN)>] [-password <gc3 password (CHANGEME)>][-encodedPassword <encoded
password>] [-server <server>] [-file <output file or !BYDATE (STDOUT)>] [-dir
<outputDirectory>] [-loop <# of iterations (1)>] [-sleep <iteration delay in
sec (60)>] [-reset] [-help]
```

Mediator Diagnostic (glog.server.workflow.mediator.MediatorDiagCommandLine)

This command line utility class provides application mediator diagnostics information to be printed out to a file or standard output from a command line. Usage:

```
java -Duser.home=<config dir> –Dmode=client
-Djava.security.auth.login.config=<config dir>/otm_jaas.config
-DGuestPassword=<guest password>
glog.server.workflow.mediator.MediatorDiagCommandLine [-user <gc3 user
(DBA.ADMIN)>] [-password <gc3 password (CHANGEME)>] [-encodedPassword <encoded
password>] [-server <server>] [-file <output file or !BYDATE (STDOUT)>] [-dir
<outputDirectory>] [-loop <# of iterations (1)>] [-sleep <iteration delay in
sec (60)>] [-show_items] [-show_topics] [-max_groups <maximum # of groups
(300)>] [-max_items_per_group <maximum # of items per group (60)>] [-groupGrep
<group regexp>] [-processGrep <process ID regexp>] [-xml] [-help]
```

> **Note**: If the user and password is not specified then these will default to DBA.ADMIN and CHANGEME respectively.

Wait for Server (glog.util.admin.WaitForServer)

This command line utility will wait for server to respond. It receives two parameters: a server name and a port. The utility will try to connect to the specified server on the specified port. If the port is open, then it exits immediately; if not, then it waits 10 seconds and tries again until the port responds. There is no authentication or authorization. Usage:

```
glog.util.admin.WaitForServer <servername> <port>
```

Encode into Base 64 Utility (glog.util.appclass.Base64Encoding)

This command line utility will encode the value that is received on the command line, and display the encoded value. The utility will also decode an encoded value that is specified on the command line. There is no authentication or authorization for this utility. Usage:

```
java glog.util.appclass.Base64Encoding <originalValue> [-decode]
```

Oracle Access Manager Utilities (glog.util.oam.event.OnOTMUserAdd, glog.util.oam.event.OnOTMUserChange, glog.util.oam.event.OnOTMUserDelete)

These command line utilities allow OTM users to be created in the application when they are invoked from an Oracle Access Manager instance. Usage:

```
-server -user -password -email -oamServer -oamUser -oamPwd -log -
manyToOneUserModel –retainOTMUsers -debug
```

GLProperties Diagnostic (glog.util.GlProperties)

This command line utility displays all application properties on the command line.

XML Helper (glog.webserver.util.XMLHelper)

This command line utility will transform the specified XML file using the specified XSL style sheet and will write the results to the specified file. Usage:

```
java glog.webserver.util.XMLHelper sourceXML stylesheet result
```

Securing Command Line Tools

Of the command line tools described above, the following allow for two modes of security when making database connections:

- CSV Utility
- Set User Password
- Transmission Purge

By default (and during installation and upgrades), these tools rely on properties to make their connections to the database. The **–connectionId** parameter refers to a *Data Source*, a reference to a set of properties defining the database host, port, service, user and password for the connection, along with an Oracle Transportation Management user for VPD data rights.

Each of these tools can be run with specific parameters to define the database connection:

- -dbHost <host name>
- -dbPort <port #>
- -dbService < service name>
- -dbURL <full URL> (this takes the place of host, port and service for complex connections)
- -dbUser <database user>
- -dbPassword <database password>
- -otmUser <Oracle Transportation Management business user>
- -otmPassword <business user password>

The use of **–connectionId** can be disabled by setting the property:

```
glog.commandline.allowDataSourceConnections=false
```

This forces a command line user to specify full URL and credential information for any database connections.

By default, the business user password is optional. If not supplied, the system will look it up. To require it, set:

```
glog.commandline.allowUnauthenticatedOTMUsers=false
```

Note: Both of these properties must be set to true during installation and upgrades. The installation and upgrade scripts use the data source paradigm and will not function properly in a more secure environment.

JMX Authentication

The Java Virtual Machine (JVM) has Java Management Extension (JMX) capabilities which allow management and diagnostics to be exposed to system administrators and developers. The application runs within a server container on a Java Virtual Machine. Therefore the JVM that runs the application can be locally or remotely accessed if configured to do so. The Application does not configure anything out of the box for these JMX capabilities. Please consult the specific JVM vendor's documentation for more details about JMX.

Authorization/Access Control

Overview

The application has a custom authorization and access control mechanism. This includes user Role Based Access Control (RBAC) capabilities. The Access Control functionality allows an application administrator to configure and maintain application level privileges. The User Role functionality allows the ability to control different application feature accesses at a higher group level as opposed to just an individual user level.

The User Roles, Access Control Lists (Functional Security), and User Access capabilities tie the authorization together to the individual users.

User Roles

The applications have a User Role concept. User roles are a way to configure and group users with similar characteristics together. Being able to define similar user characteristics at a group level instead of at an individual level provides easier security configuration capabilities and easier maintenance. The user roles within the applications are where most data visibility and authorization capabilities are defined and configured.

The user roles in the application are non-hierarchical, meaning they cannot build on top of each other and cannot inherit attributes from each other. User roles cannot union attributes from one user role into another. However, individual user roles can be configured so that they are granted to a specific user and other user roles, which allow users to switch to the other role. This switching does not allow the union of user role attributes from one user role into another. User role definitions can be changed while the application is running and these changes should be reflected without an application server restart. However, there is some overhead associated with this and performance issues could occur. It is not recommended to change the user role definitions in production during peak volumes.

User roles allow for the configuration of the Virtual Private Database (VPD) settings including the VPD Context ID, the VPD Profile ID, and the VPD Domain Name that would apply to the users with this user role assigned. The VPD settings will be discussed in the section Virtual Private Database Overview. The user roles also provide a capability to configure the Security Level that is applicable to the user access concepts that will be discussed in the section User Access. The user roles also provide a capability to specify the user role grants and the user grants. The Grantee user role specifies other user roles that will have the ability to use the current user role. The Grantee user specifies individual users that will have the ability to use the current user role. In addition, user roles provide the capability of specifying Access Control Lists (ACL) that will apply to all of the users that are assigned the user role. The following menu item provides the ability to create, view, and modify user roles.

- **Configuration and Administration > User Management > User Role**

Application administrators can set the default user role for any individual user during creation or modification of the user. In addition, there is a way to have the default user role for new users specified on the domain itself. This allows any user that is created in that domain to have the default user role specified for that domain automatically. This can be configured through the following Add Domain and Manage Domain menu items.

- **Configuration and Administration > Domain Management > Add Domain**
- **Configuration and Administration > Domain Management > Manage Domain**

The application provides a capability for individual users who are already logged into the application to temporarily change their current user role to another one, that they were previously granted. This capability is possible by using the Role hyperlink that exists on the standard header user interface, seen after successful login. The ability to Switch User Roles adds a layer of complexity and should be avoided unless necessary.

Default User Roles

The application installs several user roles by default. Please see the below table and the information following it for the details.

User Role ID	Description	Required
ADMIN	Intended for use by an application Domain administrator.	

User Role ID	Description	Required
DBA.ADMIN	Intended for the use by application super administrator(s).	Yes
SERVPROV.ADMIN	Intended for the administrator(s) of the Service Provider domain.	Yes
DEFAULT	A default User Role	No
SERVPROV	Intended for Service Provider users	
DATAENTRY	Example	No
EXTERNAL	Example	No
INTEGRATION	Intended for use by an external integration user	No

The DBA.ADMIN and SERVPROV.ADMIN user roles are special roles in that they are primarily intended for the DBA.ADMIN and SERVPROV.ADMIN users respectively. The ADMIN user role is intended for the Domain Admin user. The ADMIN user role will automatically be assign to the ADMIN user that is created when creating a new domain in the application. The DEFAULT user role is used for the DEFAULT users that are automatically created in each new domain. The SERVPROV user role is intended for use with service provider users. The DATAENTRY and EXTERNAL user roles are not needed by default, but are provided as example user roles that could be modified and used in an implementation. The INTEGRATION user role is not needed and is not assigned to any user by default, but is provided to easily assign external integration user(s) the access control entry points required for inbound external integration.

Access Control Lists/Functional Security

Access control is a general security term for grouping application permissions together for the purpose of being granted or denied to a user. This capability is referred to as Authorization, which is the definition of what functions the user is permitted to perform, after they have been authenticated. The application has a concept of Access Control Lists (ACL) for authorization. This functionality is also often referred to as Functional Security. Within the applications, an ACL is a grouping of application specific entry points. These application specific entry points include servlets, user interface actions, user interface queries, application workflow topics, enterprise Java session beans and the session bean’s public methods, Mbeans and Mbean methods, application Log IDs, and a miscellaneous group of Other. Please note that Other is an empty list which is not currently used by the application, but it is reserved for future use. The most important entry points to be concerned about are the servlets, user interface actions, user interface queries and the application log IDs. The Mbeans and Mbean methods entry points are generated, but not currently checked within the application. Due to the nature of a web application it is necessary to manage access to functions at a more granular level than the menu (i.e. servlet level), otherwise users would be able to access functions by entering the corresponding URL directly into the web browser. All of the access control entry points that are required for the applications are installed with the OTM application and are grouped by default into numerous ACLs.

The Access Control Lists within the applications are a very configurable way to control security. The Access Control Lists (ACLs) are hierarchical so an ACL can contain specific entry points and/or other ACLs.

> **Note**: An ACL cannot contain itself, and the same ACL cannot be added more than once to the same ACL hierarchy.

Within the applications, the access control lists are generally user role based, meaning that they are granted or denied at the user role level. This means the permissions that are established or taken away by the ACL would apply to all users that have been assigned the user role. However, individual users can additionally have an ACL granted or denied at the User level. For example, a user called GUEST.SCOTTTIGER with a DEFAULT user role that has the DEFAULT ACL assigned to it could get an additional ACL assigned to them for access to an additional log ID that the rest of the users with the same user role would not have. In addition, the same user with the same user role and ACL could also be denied access to a certain user interface action for shipment manipulation.

Listed below are some important ACLs that should be understood.

Access Control List ID	**Brief Description**
COMMON	List common entry points that every user will need
everyone	List that contains basic entry points that are required for all users
Administration	List of administrator-like entry points
Diagnostics	List of application diagnostic entry points
ADMIN	Parent ACL for all administrators
DEFAULT	Parent ACL for a default list
Power Data - Update	Child ACL for generic Power Data entry points for record updating
INTEGRATION	Parent ACL for external Integration entry points
External Integration	Child ACL for external Integration entry points

The COMMON ACL is the default grouping of application entry points. Most of the application entry points exist in the COMMON ACL, and are needed for just the basic navigation capabilities of the applications. When creating a new ACL, make sure to include the COMMON ACL as a child ACL.

The ‘everyone’ ACL is another special ACL that needs to be discussed. The ‘everyone’ ACL is the ACL which contains specific application entry points that every user has access to. The ‘everyone’ ACL does not need to be included as a child ACL in any custom ACL that is created, and may cause exceptions if it is.

The diagnostics ACL is a grouping of application diagnostic and performance monitoring related entry points. For example, the Diagnostics ACL entry points include the Cache and Event Diagnostic servlets.

The Administration ACL contains administration user interfaces and actions. For example, the Administration ACL includes entry points for the properties, scalability topology, and account-related user interfaces. This Administration ACL by default is granted to the ADMIN ACL. The ADMIN ACL is a top level parent ACL that is staged by the application and by default is granted to the DBA.ADMIN and ADMIN user roles that are installed with the application.

The DEFAULT ACL is another top level parent ACL and has the same child ACLs as the ADMIN ACL, except for the child ACLs of Administration and Diagnostics. The DEFAULT ACL is granted to the

DEFAULT user role that is installed with the product. Both the ADMIN and DEFAULT ACLs contain numerous other child ACLs that are groupings of similar functional areas of the application.

The access control lists are stored in the ACR_ROLE table. The application entry points are stored in the ACR_ENTRY_POINT table. The ACR_ROLE_ROLE table stores the parent ACL to child ACL mapping. The mappings of the entry points to the list are stored in ACR_ROLE_ENTRY_POINT table. The USER_ROLE_ACR_ROLE and GL_USER_ACR_ROLE tables define the mapping of the ACLs to the user role and user.

OTM Web Services Control

The OTM application extended functional security and now provides a security entry point for each individual OTM inbound Integration JAX-WS WebService. The new Web service Entry Points will be based on their service name attribute.

OTM Web Service Access Control Entry Points

- CommandService
- EchoXmlService
- AgentService
- DriverService
- GtmRestrictedPartyService
- OrderMovementService
- OrderReleaseService
- SellSideShipmentService
- ShipmentService
- GtmSanctionedTerritoryService
- IntXmlService
- MessageService
- IntGtmXmlService

All of the OTM Web Service Access Control Entry Points are grouped into the “Integration Actions” and “External Integration” Access Control Lists by default.

Note: In future releases these Web Service entry points will be removed from the legacy grouping into “Integration Actions”. These are only grouped in these access control lists for backward compatibility.

Additional consideration: All of the OTM JAX-WS WebServices are deployed with “/GC3Services/” as part of the path. If you have a custom JAX-WS WebService do not deploy it with “/GC3Services/” as path of the path unless you want OTM to handle the security for it. It will then require you to manually add a security entry point for the web service.

Custom Action Control

You can control security for custom actions without manually adding records to the database and bouncing the application server(s). The custom actions capabilities include a custom action for a report, a custom action that runs an agent action, a custom action that is set up for a custom RIQ screen and any other customizable action. A new UI action called "Secure Action" is available from the Action Manager. This action has been created in order to be used to create the security entry point and to ensure proper Access Control List Role assignment. The ACR_ENTRY_POINT and ACR_ROLE_ENTRY_POINT records will be created after the "Secure Action" action is run.

Please note: that this "Secure Action" action can only be ran once per custom action created. Any subsequence Access Control List Role assignments will need to be done using the Access Control List Manager.

The new Custom Action Entry Point will be in this format:

```
glog.webserver.util.QueryResponseServlet.action.<custom_action_gid>
```

Individual Generic Power Data Control

The OTM application extended functional security and provides a security entry point for each individual power data edit screen. Currently, there are two access control lists that control all of access to the generic power data screens. These are "Power Data – View" and "Power Data – Update". These ACLs control use of all of the generic power data at a very high level and provide access to all power data screens or no power data screens. This high level is at a generic servlet level that provides the generic capability of power data screens. These generic servlets used for power data are the glog.webserver.powerdata.GenericManagementServlet and glog.webserver.powerdata.GenericSaveServlet. In previous versions, in addition to the all or nothing option, you could also restrict generic power data via the corresponding UI query security entry point to prevent new records and editing of records. However, there is certain data that end users need to be able to query from the UI in a picklist or dropList and denying access to the query would not have allowed this. This is why OTM extended functional security for each individual Power data screen to provide an extra granularity of control.

The new Power Data Entry Points will be in this format:

```
glog.webserver.powerdata.GenericSaveServlet.powerdata.<(typically)lower_case_ta
ble_name>
```

List of a few examples:

- `glog.webserver.powerdata.GenericSaveServlet.powerdata.equipment_refnum_qual`
- `glog.webserver.powerdata.GenericSaveServlet.powerdata.java_plugin`
- `glog.webserver.powerdata.GenericSaveServlet.powerdata.s_ship_unit_refnum_qual`
- `glog.webserver.powerdata.GenericSaveServlet.powerdata.sku_cost_type`

All of the new individual related Power Data entry points have been grouped into the already existing "Power Data – Update" access control list for backward capability. This feature is not enabled by default, so if you are interested in this capability, please set the property correctly in the webserver glog.properties file.

This property controls the ability to control Functional Security at the individual Power Data level.

```
glog.webserver.powerdata.authorization=(false) [true|false]
```

- true - enables the individual Power Data check
- false- disable the individual Power Data check and is the code default.

  Note: Not every menu item under the default power data menus is a true generic power data UI. True generic power data UIs use the `glog.webserver.powerdata.GenericManagementServlet` and `glog.webserver.powerdata.GenericSaveServlet` servlets.

User Access

The applications have another security feature referred to as user access. This functionality allows different access configurations of user interface components for end users. The user interface components which can be controlled with this functionality consists of action checks, action executions, action reasons, the Ask OTM saved query, field screen sets, power actions, report workspaces, saved queries, screen sets, status type filters, user menus, and user preferences.

The user access security mechanism can tie into user roles by being able to define the user access at the user role level. This would affect all individuals that have that user role assigned. The user access security can also be assigned directly to individual users.

User access and access control lists are complementary to each other. While the Functional Security manages access to code entry points, access control manages access to the user interface components that are directly exposed to the end user.

The user access configurations inherit access to objects based on a hierarchy. The hierarchy is ranked from the more general setting of domain down to a specific setting of an individual user, level, role, and domain. The following list is the hierarchy from general to most specific:

- Domain
- User Role + Domain
- User Level + Domain
- User Level + User Role + Domain
- User + User Level + User Role + Domain

If there are access conflicts because of different configurations between the hierarchy levels, then the user access specified in the lowest and most specific hierarchy level is used. For example, if user access configurations are made at the User Role level and Domain level, then the user access defined at the user role takes precedence.

There are also Include and Exclude options for certain user access configuration capabilities. The Include and Exclude functionality provides the ability to grant or deny access. The Include and Exclude functionality are only available for the Ask OTM saved query, saved query, screen set, and user menu user access configuration types.

The user access configuration changes take effect immediately. However, the users that would be affected by the changes would need to log out of the application or have their HTTP session timeout and then log back in for the changes to take effect.

An additional capability that user access functionality provides is the ability to prevent user access changes. The administrator could set up user access at a determined level, mark it as final, and then prevent other users from changing it. By enabling the Prevent Access Changes check box as part of defining the user access records, the administrator prevents other users from having the ability to alter the user access configuration.

Data Access Control

Virtual Private Database Overview

The OTM and GTM applications utilize the Oracle database functionality of VPD. VPD stands for Virtual Private Database. VPD allows for fine grained row level data security at the database tier layer. In brief, VPD works by dynamically adding a SQL WHERE clause to the SQL statement to provide data security. The dynamic SQL WHERE can be different for all of the SQL statements like SELECT, INSERT, UPDATE, and DELETE. There are many advantages of using the Oracle Database VPD capabilities. The biggest advantage is that the data in the database is secured no matter how the data is accessed by different applications.

Within the Oracle database there are different VPD Policy Types. These policy types control how the Oracle database actually caches the Policy Predicates. They are described here:

VPD Policy Types	**Description**
STATIC	Executes policy function once, then caches it's predicate in SGA, mainly used for view replacement.
SHARED_STATIC	Basically the same as static except it shares policy among multiple tables.
DYNAMIC	Policy function re-executes every time a table is accessed, mainly used for time dependent security.
CONTEXT_SENSITIVE	Executes policy function at parse time or whenever local context change. Mainly used for three tier, session pooling applications.
SHARED_CONTEXT_SENSITIVE	Executes policy function when the table is first referenced in a database session or whenever local context changes.

The applications rely on all of the VPD Policy Types to be set to SHARED_CONTEXT_SENSITIVE in the Oracle database. Changing the VPD Policy Type to something other than SHARED_CONTEXT_SENSITIVE, could introduce a data security issue.

The applications utilize VPD policies to provide two capabilities: Domain Grants and External Predicates. The domain grants capability allows the ability to give access to data in a different domain. These are stored in the DOMAIN_GRANTS_MADE table and can be managed through the Domain Grants user interface. By default, the applications attach domain grant predicates to every table in a query. This can have an adverse affect on query performance. To help negate the performance impact of the domain criteria, the applications utilize a capability referred to as "Active Table". Active table provides the ability to specify which child table policies to use for a given SQL statement. This topic is covered in more detail in the Active Table section of the Administration Guide.

External predicates provide the ability to attach custom predicates to individual tables. The predicates will automatically be appended to any SQL statement, in addition to domain grant predicates. External predicates are defined in a VPD Profile, which is assigned to a user role. External predicates provide the ability to define custom data security rules (i.e. users can only see orders with a source location associated with them).

A VPD Profile is basically a set of VPD rules. A VPD rule provides the custom configuration capability to specific fine-grained data access control to a group of users or individual users. This data access control can be across or within Domains. The VPD Profile is then assigned to a user role. The VPD Profile allows for the use of external predicate rules, the use of the insert user rule, and the use of the service provider rule. The Use External Predicate rule enables and disables the external predicates specified in the rules. The Use Insert User rule limits the data access to only the data that the current user entered. This is ideal for a data entry employee who enters data across different domains, and should only be able to view and modify data that was entered by them. The Use of the Service Provider rule limits the data access to only the data where the user is associated to the service provider.

Please be aware that there are application and database performance implications when utilizing the VPD capabilities of the Oracle database. There is slight overhead in the application when calling the needed PL/SQL procedure to set up the user context when using any database connection. The dynamic SQL WHERE clauses that get appended to SQL statement could also cause additional overhead, and could completely change the execution plan used by the database. Also, depending on the custom predicates and domain grants that are configured within the application there could be additional performance concerns that a DBA would have to tune.

The applications provide the ability to define an application context, allowing context variables to be embedded within external predicates. The default application context that is used in the VPD functionality is gl_user_ctx. The default application context of gl_user_ctx has attributes that are used to build the standard VPD predicates within the application. A few examples of these attributes are DOMAIN_NAME, FROM_DOMAIN, GL_USER_GID, and VPD_DOMAIN_NAME. Custom application contexts can be implemented by database administrators for these applications. The VPD.SET_USER_R PL/SQL stored procedure is called by the application to set up the local application contexts used.

Default VPD Profiles

There are pre-existing and default VPD Profiles that are installed with the applications.

Default VPD Profiles	**Description**
DATAENTRY	Limits data access to only those records in which a user has entered the data. Thus, in effect, users create personalized databases limited to the records they have created.
DBA	Provides data access for the DBA.ADMIN user. All data is visible with this profile, so it should not be used with any other user.
DEFAULT	Provides data access to the entire domain, PUBLIC data, and any other data to which they have been granted access.
INVOLVED_PARTY_DOMAIN_VIS	Limits data access to only records for business objects like order base, order release, and shipments in a particular domain in which the user is an involved party in that domain.
INVOLVED_PARTY_USER_VIS	Limits data access to only records for business objects like order base, order release, and shipments in which the user is an involved party.
SERVPROV	This VPD profile should be set for all Oracle FTI users who are service providers. It contains the applicable Oracle FTI specific external predicates for HD tables that limits data access to only those shipments, rates and capacity usages in which the user is associated with the Service Provider.
FTI_DEFAULT	This VPD Profile is applicable for all Oracle FTI users who are not service providers in OTM. This includes all the external predicates available in the existing DEFAULT OTM VPD profile and the new external predicates specific to the Oracle FTI solution's historical database tables. There are some standard external predicates (specific to FTI) applied to each FTI table/materialized views that are associated to this VPD profile. Every FTI user should be associated to either the FTI_DEFAULT or SERVPROV VPD profile.

VPD is necessary for any custom Oracle Database users that would use any external database tools like the Oracle SQL*Plus utility, to connect to the database and try to use anything in the glogowner schema. The reason for this is because non OTM database users are not granted the required data access. These database users will need to execute the VPD.SET_USER_R procedure with the correct parameters. An example would look like:

```
exec vpd.set_user_r('GUEST.ADMIN','ADMIN', null,);
```

Auditing

Login

The applications have a capability to enable user login auditing functionality. There are two ways that this can be done. The first way to accomplish the user login audit capability is to set the `glog.login.suppressAccountPolicy` property to false and the `glog.login.recordInvalidLogin` property to true. This will just enable a basic login history to be recorded. If the user was successfully authenticated then, the Last Login Date on the GL_USER table/GL User interface will be updated. If the user was not successfully authenticated or the login had a different status like the user's password expired, then a record will be inserted into the GL_LOGIN_HISTORY table.

The other way to accomplish the user login audit capability is set the `glog.login.suppressAccountPolicy` property to false, and then set up an account policy for the individual users. The account policy could then specify to Keep Login history. This option of user login auditing will also write to the same tables and be reflected on the same user interface as the original option discussed above. The account policy configured could specify to Keep Login History or the same property of `glog.login.recordInvalidLogin` can be set to true. If an account policy is configured and the `glog.login.statusInvalidLogin` property is set to true, then this could display locked-out or password expiration messages even on a failed login. This is provided as an option that is disabled by default for clients that wanted to discourage password pumps.

The following properties are used to control this functionality.

- `glog.login.suppressAccountPolicy=[true|false]`: defaults to true
- `glog.login.recordInvalidLogin=[true|false]`: defaults to true
- `glog.login.statusInvalidLogin=[true|false]`: defaults to false

The Login History user interface can then be used to audit user login attempts. This user interface will be able to show an administrator a user's login attempt, login status, login time, and domain.

- **Configuration and Administration > User Management > Login History**

Login History - IP Address

In order for the correct end user's IP Address to be captured correctly as part of the Login History, there could be additional properties changes and other configuration file changes required. These charges are due to OTM using the OHS web server and the additional and dynamic proxy and load balancing servers a client could configure in front of the OTM OHS web server. For any additional load balancer or proxy server a client has, the client will need to ensure to pass the correct IP Address through each layer. You first need to determine and then configure where the end user's IP Address will be.

Property	Description	Default

Property	**Description**	**Default**
glog.webserver.login.ipAddressLocation	A property to configure where to retrieve the end user's IP address from. 1: Look at the configured header name first and then at the configured request parameter name. 2: Look at the configured request parameter name only. 3: Look at the configured header name only. 4: Look at the configured cookie name.	3
glog.webserver.login.header.ipAddressHeaderName	A property to configure the name of the header that will contain the end user's IP address.	X-Forwarded-For
glog.webserver.login.header.ipAddressPosition	When using the X-Forwarded-For Header there could be a comma separated list of IP Addresses for each proxy server that the request passes through. This property allows for the configuration of which IP address will be the end user's actual IP address. Note that the number of proxies that it takes to get to the OTM server could be dynamic. 0: Use the entire comma separated list of IP addresses value that is retrieved. A positive number greater than 0: an index into the comma separated list of IP addresses (if the position is greater than the number of addresses in the list, then the last address is used). A negative number less than 0: an index into the comma separated list of IP Addresses from the last address listed. For example, if the position was -2 then it would be the next to last address. (e.g. for the list 10.1.1.2,10.12.1.20,10.130.1.4,10.120.2.4 a -2 position would use 10.130.1.4) (if the position is less than the number of addresses in the list, then the first address is used)	1

Property	**Description**	**Default**
glog.webserver.login.cookie.ipAddressCookieName	A property to configure the cookie name that will contain the end user's IP address. In order to use this property an OHS configuration file must be changed to properly set this cookie value. See the section below this table for details on what needs to be changed.	IP_ADDR
glog.webserver.login.request.ipAddressRequestParameterName	A property to configure the request parameter name that will contain the end user's IP address.	IP_ADDR

Setting the Cookie Name/Value

In order to get the client IP address stored into a cookie there is a file that needs to be edited on *each* Web server. The following abbreviations are used:

- $APP – directory where the application was installed to
- $OHS – directory where OHS was installed to
    - $INSTANCE_NAME – the name of the OHS instance related to this application
    - $COMP_NAME – the name of the component in this instance

Edit the following file:

```
$OHS/instances/$INSTANCE_NAME/config/OHS/$COMP_NAME/moduleconf/otm.conf
```

And locate the lines that look like this:

```
   <IfModule mod_rewrite.c>
           RewriteEngine On
#          RewriteRule .* - [CO=IP_ADDR:%{REMOTE_ADDR}::0:/]
   </IfModule>The string “IO_ADDR”
```

The string “`IO_ADDR`” is the cookie name; replace it with whatever name you would like to use and be sure to specify the same name as the value of the `ipAddressCookieName` property. You will also need to uncomment the RewriteRule by removing the pound-sign (‘#’) from the beginning of the line.

Data Auditing

The applications have a data auditing capability. Many of the application functions can be audited. Ultimately, all business functions act upon business objects that store data in a database table. Within the applications you can audit user interface business actions, agent actions, application events related to data change, and get actual data change information. The data change information is available for all database columns within the tables that make up a business object. The business object is what is referred to as a data query type. All of the major entities within the application are data query types. A client can configure by a domain or globally, what data query types need to have the audit trail capability. A client can also enforce their users to provide event reasons for running particular user interface actions, which would give the audit information context as to why to the data needed to be changed. There is also the capability to capture the pre-changed and post-change data information.

There is a lot of information that is recorded in an audit record. This information includes the user interface action or agent action, the event reason and associated comments, the time the data modification took place, the business object ID, the table name, the database action taken, and the pre and post data changes.

Although there is some application performance overhead associated with utilizing the data auditing capabilities, it should be minimal.

You can control what users have access to audit log data. Typically an application Administrator or domain level Administrator should be responsible for reviewing and/or purging the audit logs. Some audit data is available with just the DEFAULT user role and Audit ACL, but in order to see all audit details, a user would need to also have the Administration ACL. Be aware that the Audit – Edit ACL can be granted to other parent ACL. You can review the audit trail records by using the Audit Trail Management user interfaces. These include the Audit Trail Manager, the Audit Trail By User, Audit Trail By Event, and the Audit Trail By Notification managers. There are also application SmartLink user interfaces available on the business entities managers for the Data Query Types that support the audit capability. These SmartLinks are links to the Audit Trail Data just for that particular business object.

The applications provide a capability for purging audit trail records. The purging can be done by the data query type or business object. The purging of the audit trail records is not tied to the actually business object’s purge. This means that when there is a purge of the business objects, it will not automatically purge the audit trail records. In order to purge the audit trail records an additional purge will need to be scheduled to clean up these related records.

The following property enables additional behavior to compare the before and after values.

```
glog.audit.beforeafter=[on|off]
```

System Change Control Auditing

The applications do not provide any capability for auditing system level changes that are made outside the system. Access to configuration files should be restricted by operating system level capabilities and a formal change control process should be implemented.

Please note that the PropertiesServlet and WebPropertiesServlet do provide a convenient way through the web browser to change application specific property settings for the duration of the server being up. There is a capability to review the history of property changes made by these properties servlets for the lifetime of the application’s uptime. However, during a restart of the entire application these changed values and history will be lost.

4. Security Considerations for Implementers

Creating Users

The DBA.ADMIN super user can create and edit users in any business domain.Content Links Embedded via Email

When sending out large e-mails, typically from a large report attachment, it's possible to exceed the maximum mail size on the SMTP server. OTM has the following options for handling these e-mails:

- For large reports, the normal security level of 2 (send report via attachment) can be replaced with a security level of 1 or 3. In each of these cases, OTM writes the report to a temporary file on the application server. The email recipient receives a link to a page to retrieve the report contents via HTTP. With a security level of 1, the recipient must have a valid OTM login to view the report. Security level 3 is deprecated. It was used to allow on-OTM users to view reports stored on the application server, circumventing the normal login process.
- For other large e-mails, OTM checks the email size against a maximum[9], If the e-mail is too large, OTM creates a temporary file on the server and provides a link to the file in the e-mail. When a user receives the e-mail, he is redirected to a page that retrieves the file from the server. This page requires user login.

In older versions of OTM, non-OTM users could gain access to large e-mail content stored on the application server. This has been deprecated for security reasons. If large e-mails are needed for non-OTM users, the old behavior can be restored with the following properties:

`glog.webserver.login.suppress.ViewReportRedirectServlet=true`

to allow circumventing of user login when report Security Level of 3 is specified

`glog.webserver.login.suppress.MailFileServlet=true`

to allow circumventing of user login when a non-OTM user tries to access an e-mail link due to #2 above

Configuration & Diagnostic Information via FTP/Email

The applications allow you to email or FTP configuration and diagnostic information to any contact in the application. The application contact will need to be properly configured for email or FTP, in order this functionality to work. The email and FTP functionality will also have to be configured correctly for this to work.

Access to the user interface for this capability is controlled via the Diagnostic Access Control List. By default this Diagnostic Access Control List is only granted to the ADMIN Access Control List.

There is no property or additional configuration to turn this functionality on or off. An application user will have to explicitly navigate to the correct user interfaces and initialize these collections to occur. These collections can be done ad-hoc or scheduled, allowing collections to occur on a repeating time interval.

> **Note**: The contents of the configuration or diagnostic information files that are emailed or FTP are not encrypted. Make sure to only send to trusted internal contacts.

---

[9] Given by the property `glog.mail.maxContentSize`

The email that is sent will have the subject of ‘Support Diagnostics’. The body of the email will contain a list of the application diagnostics that were selected and what the diagnostic attachment will contain. The actual diagnostics will be an attachment in the email as a zip file, but will have the file extension of .zop.

External Content Virus Checking

There are many use cases in OTM for a user to upload arbitrary data content to an OTM server. E.g., scanned images can be attached to shipments or structured data can be uploaded via CSV. In each of these cases, OTM supports virus checking the content against an ICAP-compliant virus scanning server.

ICAP is a common protocol to handle content checking and transformation. Details regarding the protocol can be found at http://www.icap-forum.org/icap?do=resources. When sent a request, the ICAP server is responsible for scanning the supplied content for viruses and responding:

- the content is clean
- the content is infected with details on the likely infection

It's also possible for OTM to be unable to connect to the ICAP server within a given time limit.

With virus checking enabled, OTM only stores content (either locally in the OTM schema or a specified content management system), if the ICAP server reports the content is clean. Any infection or failure to communicate with the ICAP server forces OTM to fail the content upload.

The following OTM properties are used to specify the ICAP host:

Property	**Use**	**Default**
glog.icap.antivirus.host	ICAP host	
glog.icap.antivirus.port	ICAP host	1344
glog.icap.antivirus.timeout	ICAP connection timeout, in milliseconds	3000
glog.icap.antivirus.debug	ICAP debugging flag	false

Content Use Cases in OTM

There are a number of use cases in OTM where content is brought into the system. To fully secure the system, any external content should be virus checked before being persisted, displayed or processed by OTM. There are, however, cases where the safety of content may be assumed by a particular customer. Examples of this could include:

- users who are trusted to upload document content based on AV software installed on their workstations
- upstream systems that are guaranteed to send safe content to the OTM integration layer
- external reporting engines like BI Publisher that perform their own AV check before sending a generated report to OTM

To handle situations like these, content checking in OTM can be configured on a case-by-case basis. The following table describes the content use cases in OTM:

Use Case	Description	Enabled by Default
BrandingThemeUpdate	Updating a branding theme	yes
BrandingImagesUpload	Uploading branding images	yes
ContainerOptimization	A container optimization problem staged as XML	yes
CSVContent	CSV content sent in a <CSVDataLoad> or <CSVFileContent> integration transaction	yes
DBXML	DB XML Integration	yes
DiagnosticsLog	Planning diagnostic load import	yes
DocumentIntegration	Binary or text content sent in a <Document> integration transaction	yes
DocumentStorage	Retrieving document content from a CMS (e.g. WCC or Sharepoint)	no
DocumentUpload	Uploading document content via the document finder or manager	yes
MessageIntegration	Mobilcomm message content sent in via a web service	no
MigrationProjectUpload	Uploading a migration project zip file	yes
OutXmlTemplate	Uploading an Out XML Template	yes
ProcurementAttachment	Attachments uploaded as part of procurement bidding	yes
ProcurementBid	Carrier response data for procurement	yes
ReportExternal	Retrieving report content from an external server other than BI Publisher	yes
ReportExternalBIPublisher	Retrieving report content from a BI Publisher server	no

Use Case	Description	Enabled by Default
ReportTemplate	Uploading template files for embedded reporting, including: Format template Query template Translation template Watermark template	yes
StylesheetContent	Uploading stylesheet content for notification	yes
TransmissionIntegration	Transmission content sent in through the following integration methods: OAQ HTTP POST (i.e. DirLoadServlet, LargeTransmissionServlet, WMServlet, TransmissionReceiver) Web Service (i.e. IntXmlService, WMService)	no

Disabling Virus Checking

Virus checking can be disabled globally or for one or more use cases. To disable it globally, simply remove the **glog.icap.antivirus.host** property or leave it blank.

To disable a use case, set the property:

```
glog.icap.antivirus.useCase.<Use Case>=false
```

where **<Use Case>** matches the Use Case column listed in the table above.

Note that there is overhead when enabling virus checking:

- additional memory may be needed to create an input stream for the virus checker that can be reread when passed
- additional time is needed to connect to the AV host, send the bytes to check and await the response

For certain use cases like **DocumentStorage**, **DocumentUpload**, **ReportExternal**, **ReportExternalBIPublisher** and **TransmissionIntegration**, this overhead may have a significant impact on performance. A document-centric application like GTM, for example, is likely calling out to BI Publisher to generate AES reports in background workflow. If the AV host communication is slow, overall transaction throughput is reduced. Similarly, a customer with large integration volumes and large orders may see significant transactional slowdown when every transmission is sent through a virus checker. The tradeoff of security vs. performance can be balanced by carefully disabling use cases.

Secure the Inbound External Integration Servlets

It is highly recommended for clients to check and ensure that their external integration users have the correct permissions to use these non-UI external integration servlets listed below. If proper access is not given to the user then failures will occur.

Most of these non-UI external integration servlets were already grouped into the child access control lists of "Integration – View" or "Integration Actions" in 6.3. All of these external integration servlets except for the ValidateOAMUserServlet (Administration) are also grouped into a new child access control list of "External Integration" as well. Also a new User Role of "INTEGRATION", a parent level access control list of "INTEGRATION" and a new child level "External Integration" access control list have been created by default so you can easily assign or extend just the external integration related access control entry point to your external integration users depending on the access needs to other parts of the application.

It is highly recommended to identify and correct the external integration users' access now so you will not experience any issues in future versions. Depending on your current user role and access control list assignment strategy and business needs, you need to determine if you should and can just use the new "INTEGRATION" user role, the new parent level "INTEGRATION" access control list or the new child level "External Integration" access control list. It is best to isolate the true external integration only users from other parts of the application by using one of these.

List of non-UI External Integration related servlets:

- `glog.integration.servlet.LargeTransmissionServlet`
- `glog.integration.servlet.DBXMLServlet`
- `glog.integration.servlet.ExternalSystemServlet`
- `glog.integration.servlet.PythonTransformerServlet`
- `glog.integration.servlet.RateMgmtServlet`
- `glog.integration.servlet.TransformerServlet`
- `glog.integration.servlet.BatchPythonServlet`
- `glog.integration.servlet.WMServlet`
- `glog.integration.servlet.BatchCSVUtilServlet`
- `glog.integration.servlet.DirLoadServlet`
- `glog.integration.servlet.ValidateOAMUserServlet`
- `gtm.integration.itm.servlet.ITMIntegrationServlet`

  Note: In future releases these external integration entry points will be removed from the legacy groupings of "Integration – View" or "Integration Actions". These are only still grouped in these access control lists for backward compatibility.

Please additionally note, that some of these not well known servlets like RateMgmtServlet and the other already deprecated non-UI external integration servlets like the LargeTransmissionServlet and Python related servlets still exist only for backward compatibility and will be removed in future versions of the application.

Agents

The applications have a major important capability of running automated agents to perform business procedures on the desired business objects, or perform various other tasks. The automated agent capability is a pseudo-application specific language that allows complete custom functionality based on the needs of the client's business practices. These agents and the associated logic are automatically run on the application tier when configured to do so and are triggered via application generated lifetime events that are raised based on various actions that took place within the application. Consult Online Help for more details about automation agents and the specific options available for the different business entities that exist within the application. By default, these automation agent capabilities can be configured by any users that have the DEFAULT and ADMIN ACLs. However, agent activity can have a significant impact on performance and all agent creation/modification should be handled by a formal change control process.

Agent Run As

By default, agent actions run as the Admin user for the domain of the business object. There is a capability of having the agent and its associated business object procedures and tasks run as a different user or a different user role than the user related to the application event that triggered the agent to run. The different user or different user role should be related to the business object itself.

In order to change this configuration, an administrator would first need to check the value of the `glog.process.execAsEnabled.agent` property. Then on the automation agent itself on the main tab there is a Run As Type dropdown list field, as well as fields for either the user or user role to use when the agent runs.

`glog.process.execAsEnabled.agent =[false | login (LOGIN_USER) | business,true,y,yes (Business Object)]`. The value of true is the default value configured with installation of the applications. The value of login indicates that the logged in user should be used.

Direct SQL Update

Within an automation agent, there is a capability of being able to configure any number of business actions that will directly perform any SQL statement or call any PL/SQL stored procedure.

The applications have no capability of being able to audit the data changes that are performed by this direct SQL or stored procedures. There can be application tier logging enabled but this should not be used as an audit feature due to the significant performance impact.

The Direct SQL Update agent action, although powerful, can have a significant impact on performance. Currently this capability can only be "disabled" in an implementation by deleting the agent actions record of DIRECT SQL UPDATE.

Agent Gates/Status Functor Plug-ins

The applications have the obscure and undocumented capability of being able to write custom java code for agent gates/status functors. Oracle does not provide the documentation to accomplish this custom capability so therefore this unlikely to be exploited easily.

This capability could be exploited because it opens the applications to arbitrary custom java code to run within the same JVM as the applications.

Custom Programs

Do not install, deploy, or run any custom application with-in any of the of the same JVM instances that are used for the applications. The installation, deployment and running of any custom program inside of the JVM instances for the applications is not supported and could expose serious security risks.

In order to guard against the installation, deployment, and running of custom applications within any of the same JVM instances, it is recommended to enforce strong OS level file system privileges.

Application Extension and Plug-in Capabilities

The applications have certain pre-determined extension and plug-in capabilities. Please note that while these capabilities exist, they are not required to be used for implementations and any problems that would arise are not supported.

External Third-party Engines

The applications have the capability of utilizing external engines with which Oracle does not necessarily have a certified integration. The applications have the ability to use external distance and rating engines by configuring the application and implementing an externally exposed java application program interface. Although this capability must be configured, it is important from a security perspective to be aware of its existence. Please see the External Programming Interface Guide for the exact details and steps required for this functionality to work.

Application Diagnostic Plug-ins

The applications have the ability to execute any OS level command through the application diagnostic functionality of the Static and Performance Diagnostic captures. The applications do not validate that the command is a legitimate executable, it is actually a valid command, nor if any of the options passed to the command are malicious. The user running the command line would be the same OS user that started the JVM. The commands that might be spawned out to a command line from the application server JVM should be secured at the OS level by limiting the privileges of the OS user that started the JVM. The applications have a hard-coded check to make sure the current user running the external call from this application diagnostic functionality is and only is the reserved user of 'DBA.ADMIN'. The applications have an Invocation Check on the external calls that are issued to the OS. These are controlled via the property (`glog.invocation.appdiag`). The `glog.invocation.appdiag` property is a multi-value property for a 'white-list' of allowable external commands from the Application Diagnostic functionality. The * is a special value to allow anything. The application will check the external call against a white list of properties. If the external call is not in the white list, then the application will throw an exception similar to, "This external call is not allowed". The ability to spawn to any command line executable from the Static and Performance Diagnostic captures is basically off by default, because we do not ship with any defaults for the `glog.invocation.appdiag` property. The white list has no defaults.

By default the related application entry points for the application diagnostics are in the Diagnostics ACL. By default, these entry points will prevent user interface access to the Static and Performance Diagnostic capture functionality. However the Diagnostic ACL will not be used in controlling the underlying command line spawn utility.

The application diagnostic functionality can be exploited very easily if proper OS level permissions are not enforced. The possible OS commands that could be executed are endless.

Password Validation Regular Expressions

The applications have the capability of configuring Account Policy Password rules that must be adhered to when entering and changing an application user's password. Administrators can specific regular expressions that the password must match. Please see the Password Rules section of online help for more information.

5. Appendices

Appendix A: List of Password Encryption Properties

Property	**Description**	**Default**
`glog.crypto.salt.iteration`	The value is used to repeatedly calculate a digest value from a salted password using the resulting hash as the input to next iteration.	1000
`glog.crypto.password.encrypter`	This defines the unique ID number of encryption algorithm used for GL_USER passwords. Number '1' refers to the default algorithm used in pre-6.3 versions. Number "2" is a new algorithm in 6.3. The number is used to identify the relevant glog.encrypter property which identifies the scheme to be used. The base value for new installs of the application will default to the latest most secure scheme. For migrations, the value will default to the previous value. It is recommended that after an upgrade and migration to 6.3 that you change the value to 2.	1
`glog.crypto.encrypter.<n>.scheme`	Identifies the hashing algorithm scheme name for each algorithm ID number. For example, `glog.crypto.encrypter.2.scheme=MD_SHA2`	NA

`glog.crypto.<scheme>.class`
`glog.crypto<scheme>.algorithm`
	Identifies the Java Cryptography Architecture standard names for scheme names in OTM/GTM and the implementation class. For example:
	`glog.crypto.MD_SHA2.algorithm=SHA-256`
	`glog.crypto.MD_SHA2.class=com.my.pkg.MyClass`
		NA

`glog.crypto.cipher.deprecated`	Lists all known algorithms (Java Cryptography Architecture standard names) which are deprecated for use with OTM/GTM.	
`Glog.realm.crypto.warnStartupDeprecatedAlgorithm`	If 'true' outputs log message to system console if application is currently configured to use a deprecated algorithm.	true
`glog.realm.crypto.warnDeprecatedLogin`	If 'true' outputs log message to system console if user login is using a deprecated algorithm i.e. system has been updated to use new algorithm but user has not reset their password.	true
`glog.webserver.login.updatePasswordEncryptionOnAuthentication`	If 'true' will re-hash password based on new iteration count, salt and algorithm if any changes have occurred since last login.	true

Appendix B: Secure Deployment Checklist

The following security checklist includes guidelines that help secure the applications.

1. Install only what is required.
2. Remove unnecessary default user accounts.
3. Enable data dictionary protection.
4. Practice the principle of least privilege.
    a. Grant necessary privileges only.
    b. Revoke unnecessary privileges from the PUBLIC user group.
    c. Restrict permissions on run-time facilities.
5. Enforce access controls effectively and authenticate clients stringently.
6. Restrict network access.
    a. Use a firewall.
    b. Never poke a hole through a firewall.
    c. Monitor listener activity.
    d. Monitor who accesses your systems.
7. Check network IP addresses.
8. Encrypt network traffic.
9. Harden the operating system.
10. Apply all security patches and workarounds.
11. Quarterly, always apply all Oracle CPU security patches released.