ORACLE

Oracle WebLogic Server 12.1.3 Security Target

Version:	**2.1**
Status:	**FINAL**
Last Update:	**2016-12-05**
Classification:	**public**

Trademarks

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

Other company, product, and service names may be trademarks or service marks of their respective owners.

Legal Notice

This document is provided AS IS with no express or implied warranties. Use the information in this document at your own risk.

This document may be reproduced or distributed in any form without prior permission provided the copyright notice is retained on all copies. Modified versions of this document may be freely distributed provided that they are clearly identified as such, and this copyright is included intact.

Revision History

Revision	Date	Author(s)	Changes to Previous Revision
1.30	2016-08-09	Alejandro Masino	First public version
2.1	2016-12-05	Alejandro Masino	Include cryptographic functionality as part of the TOE.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Security Target Identification

Title:	Oracle WebLogic Server 12.1.3 Security Target
Version:	2.1
Status:	FINAL
Date:	2016-12-05
Sponsor:	Oracle Corporation
Developer:	Oracle Corporation
Keywords:	Oracle WebLogic Server, Security Target, Common Criteria

1.2 TOE Identification

The TOE is Oracle WebLogic Server Version 12.1.3.

1.3 TOE Type

The TOE type is a Java Enterprise Edition (Java EE) application server.

1.4 TOE Overview

This Security Target documents the security characteristics of the Oracle WebLogic Server (in the rest of this document the term “WebLogic Server” is used as a synonym for this TOE).

The WebLogic Server is a complete implementation of the Java EE 6 specification which provides a standard set of APIs for creating distributed Java applications that can access a wide variety of services, such as databases, messaging services, and connections to external enterprise systems. End-user clients access these applications using Web browser clients or Java clients.

The TOE comprises the following components:

- Oracle WebLogic Server version 12.1.3
- Oracle WebLogic Server PSU 12.1.3.0.160719
- JDK Java Cryptographic Extension (JCE) provider
- JDK Java Secure Socket Extension (JSSE) provider
- RSA Java Cryptographic Extension (JCE) provider, included in RSA Crypto-J version 6.1.1
- RSA Java Secure Socket Extension (JSSE) provider, included in RSA SSL-J version 6.1.2

The TOE does not include the hardware, firmware, operating system or Java virtual machine used to run the software components.

The TOE can run in as a single WebLogic Server instance in a domain, or as a WebLogic Server instance in a set of distributed nodes that are part of the same domain. In this case, one TOE instance assumes the role of an Administration Server, and one or more instances assume the roles of a Managed Server or a cluster Managed Server.

1.4.1 Required Non-TOE Hardware and Software

The Operational Environment for the TOE allows the use of one of the following operating systems:

- Oracle Linux 6.7
- Oracle Solaris 11.3

The Operational Environment for the TOE allows the use of one of the following Java Runtime Environments:

- Oracle Java Development Kit (JDK) version 7 update 101 or higher
- Oracle Java Development Kit (JDK) version 8 update 91 or higher

The following LDAP servers are allowed for storing TSF data. These external servers are part of the operational environment and therefore not covered with security claims in this Security Target:

- Oracle Internet Directory
- Oracle Virtual Directory
- iPlanet
- Active Directory
- Open LDAP
- Novell LDAP

The following relational databases are allowed to be used with the TOE for both application data access and database-dependent features. These databases are part of the operational environment and therefore not covered with security claims in this Security Target:

- Oracle Database version 12.1.0.1+
- Oracle Database versions 11.1.0.7+ and 11.2.0.3+
- IBM DB2 10.1
- IBM DB2 9.7
- Microsoft SQL Server 2008 R2
- MySQL Database Server 5.5.14+ and 5.6.*
- Sybase Adaptive Server Enterprise 15.7

1.4.2 Intended Method of Use

The TOE is intended to operate in a networked environment, either alone, or with other instances of the TOE, within the context of a single management “domain”. Configuration and security policy for all TOE instances in a domain are managed by a single “Administration server”. Configuration and policy artifacts are automatically distributed by the Administration server to each “managed server” in the domain.

Communication links between individual instances of the TOE can be protected against loss of confidentiality and integrity using separate physical networks or by cryptographic protection mechanisms supported by the TOE.

Data under the control of the TOE is stored in named objects, and the TOE can associate with each named object a description of the access rights to that object.

Instances of the TOE execute as Java processes running on one of the supported Java Virtual Machines and operating system. The TOE does not control or manage the JVM, the operating system or their security policies; instead it does depend for its security on the secure configuration and management of the underlying JVM and operating system.

1.4.3 Major Security Features

The primary security features of the TOE are:

- Identification and Authentication of users supporting several password-based and identity assertion based authentication providers.
- Authorization for subjects to access and perform actions on WebLogic Server resources.
- Web Service Security.
- Audit covering security-related events.
- Security Management for domain administration.
- Protection of User and TSF data through secure channels.
- Cryptographic functionality, including cryptographic algorithms, support of the Transport Layer Security (TLS) protocol, and key management

Cryptographic functionality is provided by the Java Secure Socket Extension (JSSE), the Java Cryptographic Extension (JCE), and the RSA BSAFE® Crypto-J JSAFE and JCE Software Module. These components are part of the TOE and run under the bound Java Development Kit (JDK) package, which is part of the operational environment.

Additionally, correctness of the implementation of the cryptographic functionality has been verified as follows:

- The JSSE and JCE functionality has been tested by the developer as part of the TOE security functionality. The developer has verified that the cryptographic functionality works as expected in the context of the TOE testing.
- RSA BSAFE® Crypto-J JSAFE and JCE Software Module version 6.1.1 has been validated as a FIPS 140-2 cryptographic module.

1.5 TOE Description

Oracle WebLogic Server is an application server that allows users to access applications over various network protocols. WebLogic Server executes Java applications which are registered and are executed by the application server. The figure below shows the main components of the TOE:

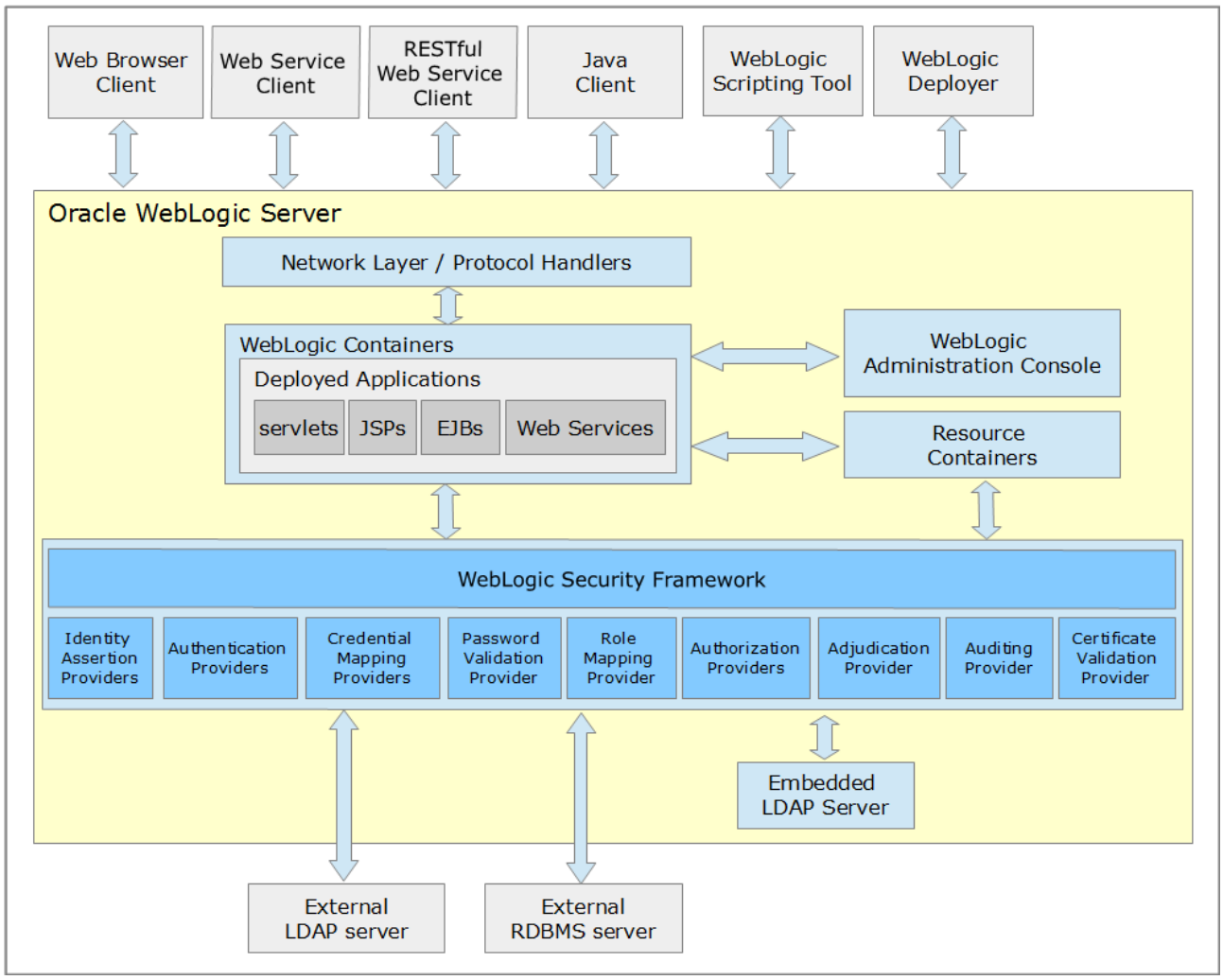

Figure 1: WebLogic Server Architecture

WebLogic Server provides a Java EE-compliant environment which is consistent with the Java EE 6 specification as defined by JSR-316, with additional support of selected Java EE 7 APIs. The applications developed for and served by the TOE are to be written in Java. Developers of the Java application implement the business logic and are free to utilize the supporting functionality of Java EE.

As part of the Java EE framework implemented by WebLogic Server, applications can provide their logic to remote clients through the following network protocols:

- HTTP/HTTPs protocols: Java servlets, Enterprise Java Beans, JMS queues, Web Services and WebSockets provide their functionality based on URLs requested by the client.
- RMI, RMI over IIOP and T3 protocols: Enterprise Java Beans (EJB) and JMS queues can provide services through these protocols.

1.5.1 Administration domains

A WebLogic Server administration domain is a logically related group of WebLogic Server resources that are managed as a unit. A domain includes one or more WebLogic servers and may also include WebLogic Server clusters. Clusters are groups of WebLogic Server instances that work together to provide scalability and high-availability for applications. Applications are deployed and managed as part of a domain.

One instance of WebLogic Server in each domain is configured as an Administration Server. The Administration Server provides a central point for managing a WebLogic Server domain. All other WebLogic Server instances in a domain are called Managed Servers. In a domain with only a single WebLogic Server instance, that server functions both as Administration Server and Managed Server.

The **Administration Server** operates as the central control entity for the configuration of the entire domain. It maintains the domain's configuration documents and distributes changes in the configuration documents to Managed Servers. The Administration Server hosts the WebLogic Server Administration Console, which is a Web application accessible from any supported Web browser with network access to the Administration Server.

Managed Servers host business applications, application components, Web services, and their associated resources. To optimize performance, Managed Servers maintain a read-only copy of the domain's configuration document. When a Managed Server starts up, it connects to the domain's Administration Server to synchronize its configuration document with the document that the Administration Server maintains.

Managed Server Clusters are meant for production environments that require increased application performance, throughput, or high availability. A cluster is a collection of multiple Oracle WebLogic Server instances running simultaneously and working together to provide increased scalability and reliability. In a cluster, most resources and services are deployed identically to each Managed Server (as opposed to a single Managed Server), enabling failover and load balancing. A single domain can contain multiple Oracle WebLogic Server clusters, as well as multiple Managed Servers that are not configured as clusters.

Clustering features provided by the TOE (failover, high availability, etc.) are not considered TOE secure functionality in this evaluation; only protection of the communication between managed cluster servers for confidentiality and integrity is claimed.

1.5.2 Security realms

A security realm comprises mechanisms for protecting WebLogic resources. Each security realm consists of a set of configured security providers, users, groups, security roles, and security policies. A user must be defined in a security realm in order to access any WebLogic resources belonging to that realm. When a user attempts to access a particular WebLogic resource, WebLogic Server tries to authenticate and authorize the user by checking the security role assigned to the user in the relevant security realm and the security policy of the particular WebLogic resource.

A domain may include one or more security realms, but only one security realm is active for the domain.

1.5.3 WebLogic Security Framework

The primary function of the WebLogic Security Framework is to provide a simplified application programming interface (API) that can be used by security and application developers to define security services. Within that context, the WebLogic Security Framework also acts as an intermediary between the WebLogic containers (Web and EJB), the Resource containers, and the security providers.

Security in WebLogic Server is based on a set of Security Service Provider Interfaces (SSPIs). The SSPIs can be used by developers and third-party vendors to develop security providers for the WebLogic Server environment. SSPIs are available for Adjudication, Auditing, Authentication, Authorization, Credential Mapping, Identity Assertion, Role Mapping, and Certificate Lookup and Validation.

1.5.4 Security Providers

Security providers are modular components that handle specific aspects of security, such as authentication and authorization. The WebLogic Security Framework supports the following types of security providers:

Authentication
Authentication is the process whereby the identity of users or system processes are proved or verified. Authentication also involves remembering, transporting, and making identity information available to various components of a system when that information is needed. Authentication providers supported by the WebLogic Security Framework supply the following types of authentication:

- Username and password authentication
- Certificate-based authentication directly with WebLogic Server
- HTTP certificate-based authentication

Identity Assertion
An Authentication provider that establishes a client's identity through the use of client-supplied tokens that may exist outside of the request. Thus, the function of an Identity Assertion provider is to validate and map a token to a username. Once this mapping is complete, an Authentication provider's LoginModule can be used to convert the username to a principal (an authenticated user, group, or system process).

Authorization
Authorization is the process whereby the interactions between users and WebLogic resources are limited to ensure integrity, confidentiality, and availability. In other words, once a user's identity has been established by an authentication provider, authorization is responsible for determining whether access to WebLogic resources should be permitted for that user. An Authorization provider supplies these services.

Role Mapping
One or more roles can be assigned to multiple users and then specify access rights for users who hold particular roles. A Role Mapping provider obtains a computed set of roles granted to a requestor for a given resource. Role Mapping providers supply Authorization providers with this information so that the Authorization provider can determine whether access is allowed for WebLogic resources that use role-based security (for example, Web applications and Enterprise JavaBeans (EJBs)).

Adjudication
When multiple Authorization providers are configured in a security realm, each may return a different answer to the "is access allowed" question for a given resource. Determining what to do if multiple Authorization providers do not agree is the primary function of an Adjudication provider. Adjudication providers resolve authorization conflicts by weighing each Authorization provider's answer and returning a final access decision.

Credential Mapping
A credential map is a mapping of credentials used by WebLogic Server to credentials used in a legacy or remote system, which tell WebLogic Server how to connect to a given resource in that system. In other words, credential maps allow WebLogic Server to log into a remote system on behalf of a subject that has already been authenticated. Credential Mapping providers map credentials in this way.

Certificate Lookup and Validation (CLV)
X.509 certificates need to be located and validated for purposes of identity and trust. CLV providers receive certificates, certificate chains, or certificate references, complete the certificate path (if necessary), and validate all the certificates in the path. There are two types of CLV providers:

- A CertPath Builder looks up and optionally completes the certificate path and validates the certificates.
- A CertPath Validator looks up and optionally completes the certificate path, validates the certificates, and performs extra validation (for example, revocation checking).

Certificate Registry
The Certificate Registry is a mechanism that allows configuring a list of trusted CA certificates per domain. It is both a CertPath Builder and a CertPath Validator. In either case, the Certificate Registry ensures that the chain's end certificate is stored in the registry. The registry is stored in the embedded LDAP server.

Auditing
Auditing is the process whereby information about security requests and the outcome of those security requests is collected, stored, and distributed for the purpose of non-repudiation. In other words, auditing provides an electronic trail of computer activity. An Auditing provider supplies these services.

Table 1 shows the security providers supported in the evaluated configuration.

1.5.5 TOE boundaries

1.5.5.1 Physical

The TOE is comprised by the following components, which are bundled in the same delivery package:

- The Oracle WebLogic Server itself
- The WebLogic Administration Console
- The embedded LDAP server
- Third-party JDBC drivers

The package also includes the following utilities, which are not part of the TOE but can be used in the evaluated configuration:

- The WebLogic Scripting Tool (WLST)
- The WebLogic Deployer utility

The TOE is supplied via the Oracle Software Delivery Cloud allowing a download of electronic copies of the TOE. Patch Set Updates (PSU) are also delivered through the same web site. The integrity and authenticity of the electronic copies are ensured by using cryptographic signatures.

TOE documentation is available at [OWLS_PD]. The following documentation is relevant for this evaluation:

- Guidance Supplement for Oracle Weblogic Server 12.1.3" [CCGUIDE]
- Understanding Oracle WebLogic Server [INTRO]
- Understanding Oracle Fusion Middleware Concepts [ASCON]
- Understanding Domain Configuration for Oracle WebLogic Server [DOMCF]
- Release Notes for Oracle WebLogic Server [WLSRN]
- Planning an Installation of Oracle Fusion Middleware [ASINS]
- Installing and Configuring Oracle WebLogic Server and Coherence [WLSIG]
- Installing Software with the Oracle Universal Installer [OUIRF]
- Creating WebLogic Domains Using the Configuration Wizard [WLDCW]
- Domain Template Reference for Fusion Middleware 12.1.3 [WLDTR]
- Administering Server Environments for Oracle WebLogic Server [CNFGD]
- Administering Server Startup and Shutdown for Oracle WebLogic Server [START]
- Administering JDBC Data Sources for Oracle WebLogic Server [JDBCA]
- Administering JMS Resources for Oracle WebLogic Server [JMSAD]
- Administering the JMS Resource Adapter for Oracle WebLogic Server [JMSRA]
- Administering Clusters for Oracle WebLogic Server [CLUST]
- Administering Node Manager for Oracle WebLogic Server [NODEM]
- Understanding Security for Oracle WebLogic Server [SCOVR]
- Administering Security for Oracle WebLogic Server [SECMG]
- Securing a Production Environment for Oracle WebLogic Server [LOCKD]
- Securing Resources Using Roles and Policies for Oracle WebLogic Server [ROLES]
- Securing WebLogic Web Services for Oracle WebLogic Server [WSSOV]
- Deploying Applications to Oracle WebLogic Server [DEPGD]
- Developing Enterprise JavaBeans for Oracle WebLogic Server [EJBAD]
- Developing Enterprise JavaBeans, Version 2.1, for Oracle WebLogic Server [EJBPG]
- Developing JAX-RPC Web Services for Oracle WebLogic Server [WSRPC]
- Developing JAX-WS Web Services for Oracle WebLogic Server [WSGET]
- Developing and Securing RESTful Web Services for Oracle WebLogic Server [RESTF]
- Developing JDBC Applications for Oracle WebLogic Server [JDBCP]
- Developing Resource Adapters for Oracle WebLogic Server [ADAPT]
- Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server [WBAPP]
- Developing Applications with the WebLogic Security Service [SCPRG]
- Command Reference for Oracle WebLogic Server [ADMRF]
- WLST Command Reference for WebLogic Server [WLSTC]
- The WebLogic Server MBean Reference [WLMBR]

The TOE also includes an Online Help in the WebLogic Server Administration Console [WLACH].

The following components are found in the IT environment:

- The underlying Operating System.
- The underlying Java virtual machine (JVM).

The following components may be found in the IT environment:

- An external LDAP server for storing TSF data.

- An external RDBMS server for storing TSF data, JMS messages.
- An external RDBMS server for user applications.
- An Enterprise Information System providing services to the TOE for user applications or vice versa
- In a Single Sign-On environment when the TOE acts as a SAML v1.1 or v2.0 consumer (destination site or service provider), an external SAML Authority supporting SAML v1.1 or v2.0 (source site or identity provider).
- In a Single Sign-On environment when the TOE acts as the SAML v1.1 or v2.0 authority (source site or identity provider), one or more SAML consumers supporting SAML v1.1 or v2.0 (destination site or service provider).
- In a Single Sign-On Windows Active Directory environment, a Windows domain controller for providing user authentication and generating Kerberos security tokens.

The supported components are shown in section 1.4.1.

1.5.5.2 Logical

The following sections briefly describe the security functionality provided by the TOE. A more detailed explanation can be found in Chapter 7, TOE Summary Specification.

Identification and Authentication

The TOE provides single and multiple identification and authentication using one or more of the following credentials:

- Username and password credential pairs
- X.509 digital certificates
- Identity Assertion tokens

The TOE implements this functionality with the WebLogic Security Framework and the following security providers:

- Authentication providers
- Identity Assertion providers
- Credential Mapping providers

Authorization

The TOE provides a role-based access control policy, applicable to all type of resources, management related or application related. Authorization for performing a certain action on a given resource is defined though security roles, security policies and access decisions.

The TOE implements this functionality with the WebLogic Security Framework and the following security providers:

- Role Mapping provider
- Authorization providers
- Adjudication provider

User Data Protection

The TOE provides protection of user data transmitted between the TOE and IT entities, and between TOE instances within the same application server domain through the use of secure channels with the TLS protocol. Establishment of a secure channel in the different communication paths is optional; protection can be assured by other security measures in the operational environment.

The TOE also supports Web Service Security, which provides integrity and confidentiality of web service payloads.

Auditing

The TOE generates audit records on application and management related events.

The TOE implements this functionality with the WebLogic Security Framework and the following security providers:

- Auditing provider

Security Management

The TOE provides the WebLogic Admin Console for all administrative activities (start/stop of servers, domain configuration, user and group management, role management, policy management, application deployment etc.). The TOE also provides a Java Extension Management (JMX) interface, which allows the use of other JMX clients to perform management activities through the use of Managed Beans (MBeans). The WebLogic Scripting Tool (WLST) is one of these JMX clients.

MBeans are considered JMX resources in the access control policy.

The TOE enforces authentication and authorization for security management actions using the same security framework.

Cryptographic support

The TOE requires cryptography for supporting the following functionality:

- Signature generation and verification for SAML 1.1 and 2.0 assertions.
- Validation of X.509 certificates
- Digest Authentication (only for Web Service Security Username Token Profile 1.0 and 1.1)
- SPNEGO Authentication
- Establishment of secure channels using the TLS protocol for communication between instances of the TOE (admin and managed servers, clustered managed servers) and the TOE with external IT entities (application clients, web browsers, LDAP servers, etc).
- XML signature and encryption for Web Service Security.

The TOE relies on the following components:

- The Java Secure Socket Extension (JSSE) is the Java standard framework for the SSL and TLS protocols, including functionality for data encryption, server authentication, message integrity, and optional client authentication.
- The Java Cryptographic Extension (JCE) provides a framework and a default implementation for encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms.

Correctness of the cryptographic algorithms has been verified as part of the testing process of the TOE, whose test cases cover the following functionality:

- Single Sign-on (SSO) using SAML 1.1, SAML 2.0 and SPNEGO
- Encryption and decryption of sensitive data (e.g. passwords)
- Processing of web services with policies including XML signing and encryption
- Certificate validation
- TLS communication

It is to note that the TOE can be set in FIPS 140-2 mode, and for that purpose, it uses the RSA BSAFE® Crypto-J JSAFE and JCE Software Module version 6.1.1, which is a FIPS-140-2 validated, security level 1 cryptographic module (certificate #2058).

1.5.5.3 Evaluated configuration

The following security providers are allowed in the evaluated configuration:

Security provider	Description
XACML Authorization Provider	Authorization provider based on XACML. This is the default authorization provider.
WebLogic Adjudication Provider	This provider tallies the potentially differing results rendered by multiple Authorization providers' Access Decisions and renders a final verdict on granting access to a resource. This is the default adjudication provider.
XACML Role Mapping Provider	Role Mapping provider based on XACML. This is the default role mapping provider.
WebLogic Auditing Provider	This provider records information from a number of security requests, which are determined internally by the Security Framework. It must be configured in the evaluated configuration.
WebLogic Credential Mapping Provider	This provider maps a user's authentication credentials (username and password) to those required for legacy applications, so that the legacy application gets the necessary credential information.
PKI Credential Mapping Provider	This provider maps a subject (the initiator) and target resource (and an optional credential action) to a key pair or public certificate that can be used by applications when accessing the targeted resource. The PKI Credential Mapping provider uses the subject and resource name to retrieve the corresponding credential from the keystore.
SAML 1.1 Credential Mapping Provider Version 2	This provider generates SAML 1.1 assertions for authenticated subjects based on relying party/destination site configuration.
SAML 2.0 Credential Mapping Provider	This provider generates SAML 2.0 assertions that can be used to assert identity in the following use cases: ● SAML 2.0 Web SSO Profile ● WS-Security SAML Token Profile version 1.1
WebLogic CertPath Provider	This provider completes certificate paths and validates certificates using the trusted CA configured for a particular server instance. This is the default certificate and validation provider.

Security provider	Description
Certificate Registry	This provider allows explicit registration of the list of trusted certificates that are allowed to access the TOE. Only certificates that are registered in the Certificate Registry will be considered valid. The Certificate Registry provides an inexpensive mechanism for performing revocation checking.
WebLogic Authentication Provider	This provider uses the embedded LDAP server to store user and group membership information. This is the default authentication provider.
Oracle Internet Directory Authentication Provider	This provider uses the Oracle Internet Directory server to store user and group membership information
Oracle Virtual Directory Authentication Provider	This provider uses the Oracle Virtual Directory server to store user and group membership information
iPlanet Authentication Provider	This provider uses the iPlanet LDAP server to store user and group membership information
Active Directory Authentication Provider	This provider uses Active Directory server to store user and group membership information
Open LDAP Authentication Provider	This provider uses the Open LDAP server to store user and group membership information
Novell LDAP Authentication Provider	This provider uses the Novell LDAP server to store user and group membership information
SQL Authenticator Provider	This provider uses a SQL database and allows both read and write access.
Read-only SQL Authenticator Provider	This provider uses a SQL database and allows only read access.
SAML Authenticator Provider	This provider may be used in conjunction with the SAML 1.1 or SAML 2.0 Identity Assertion provider to allow virtual users to log in via SAML, or create an authenticated subject using the username and groups retrieved from a SAML assertion.
Password Validation Provider	This provider manages and enforces a set of configurable password composition rules, and is automatically invoked by a supported authentication provider whenever a password is created or updated for a user. This the default password validation provider.
WebLogic Identity Asserter	This provider supports identity assertion with X.509 certificates and CORBA Common Secure Interoperability version 2 (CSI v2). This is the default identity assertion provider.
LDAP X.509 Identity Asserter	This provider receives an X509 certificate, looks up the LDAP object for the user associated with that certificate, ensures that the certificate in the LDAP object matches the presented certificate, and then retrieves the name of the user from the LDAP object.
Negotiate Identity Asserter	This provider enables single sign-on (SSO) with Microsoft clients. It decodes Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) tokens to obtain Kerberos tokens, validates the Kerberos tokens, and maps Kerberos tokens to WebLogic users.

Security provider	Description
SAML Identity Asserter (for SAML 1.1) Version 2	This provider acts as a consumer of SAML 1.1 security assertions, allowing the TOE to act as a destination site for using SAML 1.1 for single sign-on. It validates SAML 1.1 assertions by checking the signature and validating the certificate for trust in the certificate registry maintained by the provider.
SAML 2.0 Identity Asserter	This provider acts as a consumer of SAML 2.0 security assertions, allowing the TOE to act as a Service Provider for: ● Web single sign-on ● WebLogic Web Services Security: accepting SAML tokens for identity through the use of the appropriate WS-SecurityPolicy assertions

Table 1: Supported Security Providers

The following features are not allowed in the evaluated configuration:

- Resources:
  - Common Object Model (COM) resource.
- Security providers:
  - Custom RDBMS Authenticator Authentication provider
  - WebLogic Authorization Provider
  - Windows NT Authentication provider
  - WebLogic Role Mapping Provider
  - Custom Security Providers
- Protocols:
  - Simple Network Management Protocol (SNMP)
  - Distributed Common Object Model (DCOM)
- Java Standards:
  - Java Authentication Service Provider Interface for Containers (JASPIC)

1.5.6 Security Policy Model

The security policy for WebLogic Server is defined by the security functional requirements in chapter 6.1. The following is a list of the elements participating in the policy.

1.5.6.1 Web Services Information Flow Control Policy

Subjects:

- Remote and local applications

Information:

- Web service requests

Operations:

- Receive
- Transmit

1.5.6.2 Role-based Access Control Policy

Subjects:

- User represented by a set of principals: the username and the list of groups where the user belongs.

Objects: all WebLogic resources.

Operations: all operations supported by each WebLogic resource.

1.5.6.3 TSF and user data

TSF data:

- Configuration of the administrative domain
- User accounts, including the security attributes defined by FIA_ATD.1
- Group memberships
- Security Roles
- Security Policies
- Deployment descriptors
- Metadata annotations in EJBs, Servlets and Web Services
- Password policy attributes
- User locking configuration
- Audit records

User data:

- Applications deployed with the TOE and all data controlled by them

2 CC Conformance Claim

This Security Target is CC Part 2 conformant and CC Part 3 conformant, with a claimed Evaluation Assurance Level of EAL2, augmented by ALC_FLR.1.

This Security Target does not claim conformance to any Protection Profile.

Common Criteria [CC] version 3.1 revision 4 is the basis for this conformance claim.

3 Security Problem Definition

3.1 Threat Environment

This section describes the threat model for the TOE and identifies the individual threats that are assumed to exist in the TOE environment.

The **assets** to be protected by the TOE comprise:

- the resources that exist in the TOE to support the TOE security functionality
- the resources that are deployed as part of user applications.
- the information stored, processed or transmitted by the TOE. The term “information” is used here to refer to all data held within the application server domain, including data in transit between TOE instances.

The TOE counters the general threat of unauthorized access to information, where “access” includes disclosure, modification and destruction.

The **threat agents** having an interest in manipulating the data model can be categorized as either:

- Unauthorized users of the TOE, i.e. individuals who have not been granted the right to access the system.
- Authorized users of the TOE, i.e. individuals who have been granted the right to access the system.

The threat agents are assumed to originate from a well managed user community in a non-hostile working environment, and hence the product protects against threats of security vulnerabilities that might be exploited in the intended environment for the TOE with medium level of expertise and effort. The TOE protects against straightforward or intentional breach of TOE security by attackers possessing an enhanced-basic attack potential.

3.1.1 Threats countered by the TOE

T.IMPERSONATE_USER

An unauthenticated individual may impersonate a user of the TOE to gain access to protected TSF data or user data.

T.UNAUTHORIZED_ACCESS

An authenticated user may gain access to resources or perform operations on resources for which no access rights have been granted.

T.DATA_COMPROMISE

An unauthorized user is able to eavesdrop on, or manipulate without detection, the data exchanged between the TOE and a remote trusted IT product, or data in transit between instances of the TOE in the same application server domain.

3.2 Assumptions

3.2.1 Intended usage of the TOE

A.ADMIN

It is assumed that there are one or more competent individuals who are assigned to manage the TOE, the operational environment, and the security of the information it contains. The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the administrator documentation.

A.DEVEL

The developer of user applications executed by the TOE, including web server applications and enterprise beans, is trustworthy and will comply with all instructions set forth by the user guidance and evaluated configuration guidance of the TOE.

3.2.2 Environment of use of the TOE

A.PROTECT

The TOE and the hardware and software executing the TOE will be physically protected from unauthorized modification including unauthorized modifications by potentially hostile outsiders.

A.SYSTEM

The operating system and the Java virtual machine operate according to their specification. These external systems are configured in accordance with the installation guidance and the evaluated configuration guidance of the TOE.

A.CLOCK

The real time clock of the underlying operating system provides reliable time stamps.

A.PKI

It is assumed that digital certificates, CRLs used for certificate validation and private and public keys are generated externally and imported into the TOE, meeting the corresponding standards and providing sufficient security strength through the use of appropriate key lengths and cryptographic algorithms.

A.IDENTITY_PROVIDER

External entities configured in the TOE to provide identity assertions for authentication of users operate according to the specification. Successful verification of the origin of the identity assertion and trust of the external entity are sufficient to authenticate a user in the TOE.

A.DS

External entities providing storage for TSF data in the operational environment like LDAP servers or database servers are trusted, and are protected against unauthorized physical access and modification.

A.COMM_PROT

Communication between the TOE instances that constitute an application server domain, and between the TOE and external entities providing services to the TOE, is protected from eavesdropping and modification.

3.3 Organizational Security Policies

P.ACCOUNTABILITY

The users of the system shall be held accountable for security-relevant actions within the system.

P.CONSISTENCY

Configuration information and TSF data of the administration domain shall be kept consistent in all TOE instances that are part of that domain.

4 Security Objectives

4.1 Objectives for the TOE

O.AUDITING

The TSF must record security relevant actions of users of the TOE. The information recorded with security relevant events must be in sufficient detail to help an administrator of the TOE detect attempted security violations or potential misconfiguration of the TOE security features.

O.IA

The TOE must ensure that only identified and authenticated users gain access to protected resources.

O.AUTHORIZATION

The TSF must control access to resources and operations on resources based on the identity of users. The TSF must allow authorized users to specify which resources and operations can be accessed by which users.

O.SEC_CHANNEL

The TSF shall be able to communicate with remote trusted IT products and other instances of the TOE in the same application server domain using a trusted channel that protects the confidentiality and integrity of user data being transmitted.

O.MSG_PROT

The TSF shall be able to verify the integrity and decrypt web service requests received from IT entities, as well as protect web service requests transmitted to IT entities from tampering and eavesdropping.

O.CONSISTENCY

The TSF must ensure the consistency of configuration information and TSF data between managed servers and the Administration server that belong to the same domain.

4.2 Objectives for the Operational Environment

OE.ADMIN

Those responsible for the administration of the TOE are competent and trustworthy individuals, capable of managing the TOE, the operational environment, and the security of the information it contains.

OE.SYSTEM

Those responsible for the TOE must ensure that the operating system and the Java virtual machine are installed and configured in accordance with the guidance of the TOE and that these mechanisms operate as specified. This also covers that only the Java virtual machines enumerated in this ST are used as underlying platform to ensure that proper date and time information is available to the audit facility.

OE.INSTALL

Those responsible for the TOE must establish and implement procedures to ensure that the software components that comprise the TOE are distributed, installed, configured and administered in a secure manner.

OE.PHYSICAL

Those responsible for the TOE must ensure that the TOE as well as the underlying hardware and software are protected from physical attack, which might compromise IT security objectives.

OE.DEVEL

Those responsible for the TOE shall ensure that the developers of the applications executed by the TOE are trustworthy and implement the applications in accordance with the guidance provided with the TOE.

OE.CLOCK

The real time clock of the underlying operating system shall provide reliable time stamps.

OE.PKI

Digital certificates, CRLs used for certificate validation and private and public keys must be generated externally and imported into the TOE. This material must meet the corresponding standards and provide sufficient strength, through the use of appropriate key lengths and message digest algorithms.

OE.IDENTITY_PROVIDER

External entities used to provide identity assertions to authenticate users must operate according to the specification and must be configured in the TOE to be trusted.

OE.DS

External entities providing storage for TSF data in the operational environment like LDAP servers or database servers must be trusted, and must be protected against unauthorized physical access and modification. Communication between the TOE and those systems must be also protected from eavesdropping and modification.

OE.COMM_PROT

Communication between the TOE instances that constitute an application domain and between the TOE and external entities providing services to the TOE must be protected from eavesdropping and modification through physical or logical means. This objective complements the protection provided by O.SEC_CHANNEL.

4.3 Security Objectives Rationale

4.3.1 Security Objectives Coverage

The following table provides a mapping of TOE objectives to threats and policies, showing that each objective counters or enforces at least one threat or policy, respectively.

Objective	Threats / OSPs
O.AUDITING	P.ACCOUNTABILITY
O.IA	T.IMPERSONATE_USER
O.AUTHORIZATION	T.UNAUTHORIZED_ACCESS
O.SEC_CHANNEL	T.DATA_COMPROMISE
O.MSG_PROT	T.DATA_COMPROMISE
O.CONSISTENCY	P.CONSISTENCY

Table 2: Mapping of security objectives to threats and policies

The following table provides a mapping of the objectives for the Operational Environment to assumptions, threats and policies, showing that each objective holds, counters or enforces at least one assumption, threat or policy, respectively.

Objective	Assumptions / Threats / OSPs
OE.ADMIN	A.ADMIN
OE.SYSTEM	A.SYSTEM
OE.INSTALL	A.ADMIN
OE.PHYSICAL	A.PROTECT
OE.DEVEL	A.DEVEL
OE.CLOCK	A.CLOCK
OE.PKI	A.PKI
OE.IDENTITY_PROVIDER	A.IDENTITY_PROVIDER
OE.DS	A.DS
OE.COMM_PROT	A.COMM_PROT T.DATA_COMPROMISE

Table 3: Mapping of security objectives for the Operational Environment to assumptions, threats and policies

4.3.2 Security Objectives Sufficiency

The following rationale provides justification that the security objectives are suitable to counter each individual threat and that each security objective tracing back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat.

Threat	Rationale for security objectives
T.IMPERSONATE_USER	The threat of unauthenticated access to TSF or user data is removed by the objective O.IA to implement identification and authentication mechanisms in the TOE.
T.UNAUTHORIZED_ACCESS	The threat of an authenticated user of the TOE accessing information resources or performing operations on resources without the permission from the user responsible for the resource is removed by O.AUTHORIZATION requiring access control for resources and the ability for authorized users to specify the access to their resources. This ensures that a user can access a resource only if the requested type of access has been granted by the user responsible for the management of access rights to the resource.
T.DATA_COMPROMISE	The threat of an unauthorized user accessing data exchanged between the TOE and a remote trusted IT product, or between instances of the TOE in the same application server domain is diminished by the functionality provided by O.SEC_CHANNEL requiring the protection of the integrity and confidentiality through the use of a trusted channel, and O.MSG_PROT requiring the protection of the integrity and confidentiality of user data included in web services through the use of Web Service Security. In addition, OE.COMM_PROT complements O.SEC_CHANNEL protecting communication between the TOE instances of the same application server domain and between the TOE and an external entity.

Table 4: Sufficiency of objectives countering threats

The following rationale provides justification that the security objectives for the environment are suitable to cover each individual assumption, that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption, and that if all security objectives for the environment that trace back to an assumption are achieved, the intended usage is supported.

Assumption	Rationale for security objectives
A.ADMIN	The assumption on competent administrators is covered by OE.ADMIN requiring competent and trustworthy administrators and OE.INSTALL requiring procedures for secure distribution, installation and configuration of systems.
A.DEVEL	The assumption on developers of applications executed by the TOE to be trustworthy and to comply with the instructions set forth in the guidance is covered by OE.DEVEL requiring the administrator to ensure that these developers are indeed trustworthy.
A.PROTECT	The assumption on physical protection of the TOE, all hardware and software as well as the network and peripheral cabling is covered by the objectives OE.PHYSICAL requiring physical protection.

Assumption	Rationale for security objectives
A.SYSTEM	The assumption that the environment the TOE relies on to enforce its functionality (the OS and the Java virtual machine) is configured according to the guidance provided by the TOE is covered by OE.SYSTEM requiring the administrator to comply with that guidance.
A.CLOCK	OE.CLOCK assures that the TOE uses a reliable time source to synchronize the real time clock.
A.PKI	OE.PKI assures that the TOE uses valid digital certificates, CRLs and public and private keys, providing sufficient security strength.
A.IDENTITY_PROVIDER	OE.IDENTITY_PROVIDER assures that external entities used to provide identity assertions must operate according to the specification and must be configured in the TOE to be trusted.
A.DS	OE.DS assures that external entities used to provide storage for TSF data are trusted and protected.
A.COMM_PROT	OE.COMM_PROT assures that communication between the TOE instances that constitute an application server domain and between the TOE and external entities is protected.

Table 5: Sufficiency of objectives holding assumptions

The following rationale provides justification that the security objectives are suitable to cover each individual organizational security policy (OSP), that each security objective that traces back to an OSP, when achieved, actually contributes to the implementation of the OSP, and that if all security objectives that trace back to an OSP are achieved, the OSP is implemented.

OSP	Rationale for security objectives
P.ACCOUNTABILITY	The policy to provide accountability for the action of TOE users is implemented by the objective O.AUDITING to provide an auditing mechanism for security-relevant events.
P.CONSISTENCY	The policy to provide consistency of TSF data between TOE instances in the same administrative domain is implemented by the objective O.CONSISTENCY.

Table 6: Sufficiency of objectives enforcing Organizational Security Policies

5 Extended Components Definition

This Security Target does not extend the security components provided by the Common Criteria.

6 Security Requirements

6.1 TOE Security Functional Requirements

The following table shows the SFRs for the TOE, and the operations performed on the components according to CC part 2: iteration (Iter.), refinement (Ref.), assignment (Ass.) and selection (Sel.).

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
FAU - Security audit	FAU_GEN.1 Audit data generation		CC Part 2	No	No	Yes	Yes
	FAU_GEN.2 User identity association		CC Part 2	No	No	No	No
	FAU_SEL.1 Selective audit		CC Part 2	No	No	Yes	Yes
FCS - Cryptographic support	FCS_CKM.1-JCE Cryptographic key generation for the TLS protocol		CC Part 2	No	Yes	Yes	No
	FCS_CKM.2-JCE Cryptographic key distribution for the TLS protocol		CC Part 2	No	Yes	Yes	No
	FCS_COP.1-JCE(TLS) Cryptographic operations for the TLS protocol	FCS_COP.1	CC Part 2	Yes	Yes	Yes	No
	FCS_COP.1-JCE(XMLSIG) Cryptographic operations for XML Signature Syntax and Processing	FCS_COP.1	CC Part 2	Yes	Yes	Yes	No
	FCS_COP.1-JCE(XMLENC) Cryptographic operations for XML Encryption	FCS_COP.1	CC Part 2	Yes	Yes	Yes	No
	FCS_COP.1-JCE(CLV) Cryptographic operations for certificate lookup and validation	FCS_COP.1	CC Part 2	Yes	Yes	Yes	No
	FCS_COP.1-JCE(SPNEGO) Cryptographic operations for SPNEGO	FCS_COP.1	CC Part 2	Yes	Yes	Yes	No
FDP - User data protection	FDP_ACC.2 Complete access control		CC Part 2	No	No	Yes	No
	FDP_ACF.1 Resource Access Control Functions		CC Part 2	No	No	Yes	No
	FDP_IFC.1 Subset information flow control		CC Part 2	No	No	Yes	No
	FDP_IFF.1 Simple security attributes		CC Part 2	No	Yes	Yes	No
	FDP_UCT.1 Basic data exchange confidentiality		CC Part 2	No	No	Yes	Yes
	FDP_UIT.1 Data exchange integrity		CC Part 2	No	No	Yes	Yes

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
FIA - Identification and authentication	FIA_AFL.1 Authentication failure handling		CC Part 2	No	Yes	Yes	Yes
	FIA_ATD.1 User attribute definition		CC Part 2	No	No	Yes	No
	FIA_SOS.1 Verification of secrets		CC Part 2	No	Yes	Yes	No
	FIA_UAU.1 Timing of authentication		CC Part 2	No	No	Yes	No
	FIA_UAU.5 Multiple authentication mechanisms		CC Part 2	No	No	Yes	No
	FIA_UID.1 Timing of identification		CC Part 2	No	No	Yes	No
	FIA_USB.1 User-subject binding		CC Part 2	No	No	Yes	No
FMT - Security management	FMT_MSA.1 Management of security attributes		CC Part 2	No	No	Yes	Yes
	FMT_MSA.3 Static attribute initialisation		CC Part 2	No	No	Yes	Yes
	FMT_MTD.1(APP) Management of TSF data (Applications)	FMT_MTD.1	CC Part 2	Yes	Yes	Yes	Yes
	FMT_MTD.1(RACP) Management of TSF data (Role-based Access Control Policy)	FMT_MTD.1	CC Part 2	Yes	Yes	Yes	Yes
	FMT_SMF.1 Specification of Management Functions		CC Part 2	No	No	Yes	No
	FMT_SMR.2 Restrictions on security roles		CC Part 2	No	No	Yes	No
FPT - Protection of the TSF	FPT_ITT.1 Basic internal TSF data transfer protection		CC Part 2	No	No	No	Yes
	FPT_TRC.1 Internal TSF consistency		CC Part 2	No	No	Yes	No
FTP - Trusted path/channels	FTP_ITC.1 Inter-TSF trusted channel		CC Part 2	No	No	Yes	Yes

Table 7: SFRs for the TOE

6.1.1 Security audit (FAU)

6.1.1.1 Audit data generation (FAU_GEN.1)

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events for the **not specified** level of audit; and

c) **the events listed in Table 8**.

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, **severity level, subsystem, server name, machine name, thread ID**.

Audit event	Description
AUTHENTICATE	A simple authentication (username and password) occurred.
ASSERTIDENTITY	A perimeter authentication (based on tokens) occurred.
USERLOCKED	A user account is locked because of invalid login attempts.
USERUNLOCKED	The lock on a user account is cleared
USERLOCKOUTEXPIRED	The lock on a user account expired.
ISAUTHORIZED	An authorization attempt occurred.
ROLEEVENT	A getRoles event occurred.
ROLEDEPLOY	A deployRole event occurred.
ROLEUNDEPLOY	An undeployRole event occurred.
POLICYDEPLOY	A deployPolicy event occurred.
POLICYUNDEPLOY	An undeployPolicy event occurred.
START_AUDIT	An Auditing provider has been started.
STOP_AUDIT	An Auditing provider has been stopped.

Table 8: Audit Events

6.1.1.2 User identity association (FAU_GEN.2)

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

6.1.1.3 Selective audit (FAU_SEL.1)

FAU_SEL.1.1 The TSF shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes:

a) **event type**

b) **no additional attributes**

Application Note: *The TOE uses the term "severity level" to refer to the event type. The TOE supports the following severity levels: INFORMATION, WARNING, ERROR, SUCCESS, FAILURE. The administrator can enable or disable each level separately, or specify instead the minimum severity level to be audited.*

6.1.2 Cryptographic support (FCS)

6.1.2.1 Cryptographic key generation for the TLS protocol (FCS_CKM.1-JCE)

FCS_CKM.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **defined in Table 9** and specified cryptographic key sizes **defined in Table 9** that meet the following: **generation and exchange of session keys as defined in [RFC5246] and the standards defined in Table 9** .

Key generation algorithm	Key sizes (in bits)	Standards
AES	128 and 256 bits	[FIPS197]
Triple-DES	168 bits	[SP800-67]
HMAC	256 bits	[RFC2104] and [FIPS180-4]

Table 9: Cryptographic key generation for the TLS protocol

Application Note: *This SFR covers the generation of the client and server encryption keys and client and server MAC keys for the TLS protocol (derived from the master secret).*

6.1.2.2 Cryptographic key distribution for the TLS protocol (FCS_CKM.2-JCE)

FCS_CKM.2.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **defined in Table 10** that meets the following: **the standards defined in Table 10**.

Key distribution method	Standards
Key exchange of session and MAC keys	RSAES-PKCS1-v1_5 conformant to [RFC5246] and [RFC3447]
Key agreement of session and MAC keys	Diffie-Hellman conformant [RFC5246]
Exchange of X.509 v3 certificates	conformant to [RFC5246] and [RFC5280]

Table 10: Cryptographic key distribution for the TLS protocol

Application Note: *This SFR covers the key establishment of session keys and exchange of client and server X.509 v3 certificates in the TLS protocol.*

6.1.2.3 Cryptographic operations for the TLS protocol (FCS_COP.1-JCE(TLS))

FCS_COP.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall perform **the operations defined in Table 11** in accordance with a specified cryptographic algorithm **defined in Table 11** and cryptographic key sizes **defined in Table 11** that meet the following: **the standards defined in Table 11**.

Cryptographic operations	Algorithm	Key sizes	Standards
Asymmetric encryption and decryption	RSA	2048 bits	RSA Encryption Scheme with PKCS#1 v1.5 (RSAES-PKCS1-v1_5) conformant to [RFC5246] and [RFC3447]
Symmetric encryption and decryption	AES (CBC mode)	128 and 256 bits	conformant to [FIPS197] and [SP800-38A]
Message authentication code	HMAC-SHA-256	256	conformant to [RFC2104]
Digital signature generation and verification	RSA	2048 bits	RSA Signature Scheme with PKCS#1 v1.5 (RSASSA-PKCS1-v1_5) and SHA-256 conformant to [RFC3447] and [FIPS180-4]

Table 11: Cryptographic operations for the TLS protocol

Application Note: *This SFR covers cryptographic operations used by the TLS protocol.*

6.1.2.4 Cryptographic operations for XML Signature Syntax and Processing (FCS_COP.1-JCE(XMLSIG))

FCS_COP.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall perform **the operations defined in Table 12** in accordance with a specified cryptographic algorithm **defined in Table 12** and cryptographic key sizes **defined in Table 12** that meet the following: **the standards defined in Table 12**.

Cryptographic operations	Algorithm	Key sizes	Standards
Digital signature generation and verification	RSA	2048 bits	RSA Signature Scheme with PKCS#1 v1.5 (RSASSA-PKCS1-v1_5) and SHA-1 or SHA-256 conformant to [RFC3447] and [FIPS180-4]

Table 12: Cryptographic operations for XML Signature

Application Note: *This SFR covers cryptographic operations used for SAML assertions.*

6.1.2.5 Cryptographic operations for XML Encryption (FCS_COP.1-JCE(XMLENC))

FCS_COP.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall perform **the operations defined in Table 13** in accordance with a specified cryptographic algorithm **defined in Table 13** and cryptographic key sizes **defined in Table 13** that meet the following: **the standards defined in Table 13**.

Cryptographic operations	Algorithm	Key sizes	Standards
Symmetric Encryption and Decryption	AES (CBC mode)	128, 192, 256 bits	Conformant to [FIPS197]
Symmetric Encryption and Decryption	Triple-DES (CBC mode)	168 bits	Conformant to [SP800-67]
Key Transport	RSA	2048 bits	RSA Signature Scheme with PKCS#1 v1.5 (RSASSA-PKCS1-v1_5) and SHA-1 or SHA-256 conformant to [RFC3447] and [FIPS180-4]
Key Wrapping	RSA	2048 bits	RSA Encryption Scheme with PKCS#1 v1.5 (RSAES-PKCS1-v1_5) and SHA-1 or SHA-256 conformant to [RFC3447] and [FIPS180-4]
Key Wrapping	RSA	2048 bits	RSA Encryption Scheme with OAEP (RSAES-OAEP) and SHA-1 or SHA-256 conformant to [RFC3447] and [FIPS180-4]

Table 13: Cryptographic operations for XML Encryption

Application Note: *This SFR covers cryptographic operations used for XML encryption and decryption of Web Service messages.*

6.1.2.6 Cryptographic operations for certificate lookup and validation (FCS_COP.1-JCE(CLV))

FCS_COP.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall perform **the operations defined in Table 14** in accordance with a specified cryptographic algorithm **defined in Table 14** and cryptographic key sizes **defined in Table 14** that meet the following: **the standards defined in Table 14**.

Cryptographic operations	Algorithm	Key sizes	Standards
Digital signature verification	RSA	1024, 2048, 3072 bits	RSA Signature Scheme with PKCS#1 v1.5 (RSASSA-PKCS1-v1_5), SHA-1 and SHA-256 conformant to [RFC3447] and [FIPS180-4]

Table 14: Cryptographic operations for certificate validation

Application Note: *This SFR covers cryptographic operations used for certificate validation. Certificate validation is used by the TOE for secure channel establishment (client certificate validation in the TLS protocol), verification of certificate-based authentication, and verification of SAML assertions and SAML protocol request and response messages.*

6.1.2.7 Cryptographic operations for SPNEGO (FCS_COP.1-JCE(SPNEGO))

FCS_COP.1.1 The *Java Cryptographic Extension and the Java Secure Socket Extension in the* TSF shall perform **the operations defined in Table 15** in accordance with a specified cryptographic algorithm **defined in Table 15** and cryptographic key sizes **defined in Table 15** that meet the following: **the standards defined in Table 15**.

Cryptographic operations	Algorithm	Key sizes	Standards
Message Authentication Code	HMAC-SHA1-96	96 bits	HMAC-SHA1-96 Authentication Protocol according to [RFC3962] and [RFC2104].
Encryption and decryption	AES (CBC mode)	128 and 256 bits	AES conformant to [RFC3962], [FIPS197] and [SP800-38A]

Table 15: Cryptographic operations for SPNEGO

Application Note: *This SFR covers cryptographic operations used by for SPNEGO authentication.*

6.1.3 User data protection (FDP)

6.1.3.1 Complete access control (FDP_ACC.2)

FDP_ACC.2.1 The TSF shall enforce the **Role-based Access Control Policy** on

a) Subject: a user represented by a set of principals

b) Objects: the following Weblogic resources:

- **Administrative resources**
- **Application resources**
- **Component Object Model (COM) resources**
- **Enterprise JavaBean (EJB) resources**
- **Enterprise Information System (EIS) resources**
- **Java Database Connectivity (JDBC) resources**

- **Java Messaging Service (JMS) resources**
- **Java Naming and Directory Interface (JNDI) resources**
- **JMX resources**
- **Server resources**
- **Universal Resource Locator (URL) resources**
- **Web Service resources**
- **Work Context resources**

and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

Application Note: *Component Object Model (COM) resources are not allowed in the evaluated configuration, but this resource type is included for completeness.*

6.1.3.2 Resource Access Control Functions (FDP_ACF.1)

FDP_ACF.1.1 The TSF shall enforce the **Role-based Access Control Policy** to objects based on the following:

a) Subject attributes: username, groups, security roles applicable to the subject;

b) Object attributes: resource identity, resource type, resource hierarchy, security policies associated with the resource identity, security policies associated with each element in the resource hierarchy

.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

a) The TSF determines all security roles applicable to the subject as follows:

1. **the basic policy condition of the security role (one of the following) is met:**
   i. **if the basic policy condition assigns a user, then the subject's name matches the user;**
   ii. **if the basic policy condition assigns a group, then the subject is member of that group;**
2. **the date and time policy conditions defined in the applicable Security Role are met;**
3. **the context element policy conditions defined in the applicable Security Role are met.**

b) If the object type is a "JMX resource" and the TSF does not delegate authorization of MBeans to authorization providers:

1. **the TSF determines the security roles authorized to access the JMX resource based on the default security policy for MBeans;**
2. **if access to the JMX resource is not restricted to security roles, then access is granted;**

3. if access to the JMX resource is restricted to security roles, and the subject is assigned to at least one of those security roles, then access is granted;

4. if access to the JMX resource is restricted to security roles, and the subject is not assigned to at least one of those security roles, then access is denied.

c) If the object type is not a "JMX resource", or the object type is a "JMX resource" and the TSF delegates authorization of MBeans to authorization providers, the TSF determines the security policy applicable to the object as follows:

1. it looks up the security policy that matches the resource identity;

2. if the security policy does not exist, it gets the parent resource in the hierarchy defined by the resource type and looks up for a security policy that matches the parent resource. The process continues until a security policy is found or the root node in the hierarchy has been reached;

3. if no security policy is associated with the object, access is denied.

4. if a security policy is associated with the object, then the following rules are verified:

i. the basic policy condition defined in the applicable Security Policy (one of the following) is met:

a) if the basic policy condition assigns a user, then the subject's name matches the user;

b) if the basic policy condition assigns a group, then the subject is member of that group;

c) if the basic policy condition assigns a security role, then the subject is entitled to that security role;

ii. the date and time policy conditions defined in the applicable Security Policy are met;

iii. the context element policy conditions defined in the applicable Security Policy are met.

If all conditions are met, then access is allowed. Otherwise, access is denied.

.

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **none**.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none**.

Application Note: *Managed Beans (MBeans) are modeled as JMX resources in the security policy model. By default, the TOE enforces access control on JMX resources using default security policies defined for MBeans and granted to the administrator roles Admin, Deployer, Operator, and Monitor.*

Administrators can change this behavior delegating access control on the Authorization providers and editing security policies on JMX resources through the existing management functions. See section 7.1.2.3 "Security Policies" for more information.

Application Note: *For a detailed description of the basic, date and time and context element conditions applicable to security policies and security roles please refer to sections 7.1.2.3 "Security Policies" and 7.1.2.3 "Security Policies", respectively.*

Application Note: *Security policies and security roles defined in annotations and J2EE deployment descriptors for Web Application, EJB and Web Service resources are considered during application deployment depending on the deployment model chosen ("Deployment Descriptors only", "Customize Roles only", "Customize Roles and Policies", "Advanced"). See section 7.1.2.3 "Security Policies" for more information.*

6.1.3.3 Subset information flow control (FDP_IFC.1)

FDP_IFC.1.1 The TSF shall enforce the **Web Services Information Flow Control Policy** on

a) Subjects: remote and local applications;

b) Information: web service requests;

c) Operations: receive, transmit

.

6.1.3.4 Simple security attributes (FDP_IFF.1)

FDP_IFF.1.1 The TSF shall enforce the **Web Services Information Flow Control Policy** based on the following types of subject and information security attributes:

a) Subject security attributes: WS-Policy assertions associated with the web service;

b) Information security attributes:

- **One of the following WS-Security tokens:**
  - **Username Token conformant to [WSS11-SOAP] and [WSS11-UTP];**
  - **X.509 Certificate Token conformant to [WSS11-SOAP] and [WSS11-X509];**
  - **SAML v1.1 Token conformant to [WSS11-SOAP], [WSS11-SAML] and [SAML11Core];**
  - **SAML v2.0 Token conformant to [WSS11-SOAP], [WSS11-SAML] and [SAML20Core].**
- **XML signatures according to [W3CXMLSIG]**

.

FDP_IFF.1.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

a) if the WS-Policy assertion requires a security token:

- **the WS-Security token meets the requirements stated in the WS-Policy assertion, and**
- **authentication of the WS-Security token succeeds;**

b) if the WS-Policy assertion requires a signed message, the XML signature validation succeeds according to [W3CXMLSIG];

c) if the WS-Policy assertion requires an encrypted message, XML decryption succeeds according to [W3CXMLENC]

.

FDP_IFF.1.3 The TSF shall enforce ~~the~~**no additional information flow control SFP rules**.

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: **the web service request is originated in the TOE**.

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: **none**.

6.1.3.5 Basic data exchange confidentiality (FDP_UCT.1)

FDP_UCT.1.1 The TSF shall enforce the **Web Services Information Flow Control Policy** to **transmit, receive** user data in a manner protected from unauthorised disclosure.

6.1.3.6 Data exchange integrity (FDP_UIT.1)

FDP_UIT.1.1 The TSF shall enforce the **Web Services Information Flow Control Policy** to **transmit, receive** user data in a manner protected from **modification, deletion, insertion** errors.

FDP_UIT.1.2 The TSF shall be able to determine on receipt of user data, whether **modification, deletion, insertion** has occurred.

6.1.4 Identification and authentication (FIA)

6.1.4.1 Authentication failure handling (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when **an administrator configurable positive integer within 5 and 64** unsuccessful authentication attempts occur related to **password-based authentication** .

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been **surpassed** *and occurred within an administrator configurable lockout reset duration*, the TSF shall **lock the user account until one of the following events occurs:**

- **the user account is unlocked by an administrator;**
- **the user account has been locked for a duration that exceeds an administrator configurable lockout duration**

.

Note: *This requirement is met through the configuration of the Lockout Threshold, Lockout Reset Duration and Lockout Duration parameters.*

6.1.4.2 User attribute definition (FIA_ATD.1)

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users:

a) subject identity;

b) groups;

c) security roles;

d) password;

e) X.509 certificate

.

6.1.4.3 Verification of secrets (FIA_SOS.1)

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets *used to authenticate users* meet **the following rules configurable by an authorized administrator:**

- **whether the password may consist of or contain the user's name**
- **whether the password may consist of or contain the reverse of the user's name**
- **minimum length of the password**
- **maximum length of the password**
- **maximum number of repeating consecutive characters**
- **maximum number of instances of any character**
- **minimum number of numeric characters**
- **minimum number of alphabetic characters**
- **minimum number of lowercase characters**
- **minimum number of uppercase characters**
- **minimum number of numeric or special characters**
- **minimum number of non-alphanumeric characters**

.

6.1.4.4 Timing of authentication (FIA_UAU.1)

FIA_UAU.1.1 The TSF shall allow **operations on unprotected resources** on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.4.5 Multiple authentication mechanisms (FIA_UAU.5)

FIA_UAU.5.1 The TSF shall provide **the following authentication mechanisms:**

a) Username/password authentication;

b) X.509 digital certificate identity assertion;

c) CORBA Common Secure Interoperability version 2 identity assertion;

d) SAML 1.1 identity assertion;

e) SAML 2.0 identity assertion;

f) SPNEGO identity assertion

g) Web Service Security Username Token Profile 1.0 and 1.1

h) Web Service Security X.509 Token profile 1.0 and 1.1

i) Web Service Security SAML Token Profile 1.0 and 1.1

to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the **behavior defined by the following possible values of the JAAS control flag that is assigned to each authentication mechanism:**

a) REQUIRED: authentication is required to succeed. The authentication process proceeds down to the next authentication mechanism in the sequence.

b) REQUISITE: authentication is required to succeed. If it fails, the authentication process stops and fails; if it succeeds, the authentication process proceeds down to the next authentication mechanisms in the sequence.

c) SUFFICIENT: authentication is not required to succeed. If it succeeds, the authentication process stops and succeeds; if it files, the authentication process proceeds down to the next authentication mechanism in the sequence.

d) OPTIONAL: authentication is not required to succeed. The authentication process proceeds down to the next authentication mechanism in the sequence.

.

Appilcation Note: *Each authentication mechanism is supported by one of the authentication providers shown in Table 1. Multiple authentication (available authentication mechanisms and authentication sequence) is defined at domain level by the administrator.*

6.1.4.6 Timing of identification (FIA_UID.1)

FIA_UID.1.1 The TSF shall allow **operations on unprotected resources** on behalf of the user to be performed before the user is identified.

FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.4.7 User-subject binding (FIA_USB.1)

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user:

a) subject identity;

b) groups;

c) security roles

.

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users:

a) The subject identity is specified as follows:

1. for password-based authentication, by the username that corresponds to the user that has been identified and authenticated;

2. for identity assertion authentication, by the username that is mapped with the digital certificate or security token.

b) The groups associated with the subject shall be the groups the user is member of, directly and indirectly (through groups and group membership).

c) The security roles associated with the subject shall be the roles that meet:

1. the basic role conditions: user and group membership to roles;

2. date and time role conditions;

3. context element role conditions

.

Application Note: *Security roles defined in annotations and J2EE deployment descriptors for Web Application, EJB and Web Service resources are imported by the XACML Role Mapping provider during application deployment and are applicable to this requirement.*

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: **none**.

6.1.5 Security management (FMT)

6.1.5.1 Management of security attributes (FMT_MSA.1)

FMT_MSA.1.1 The TSF shall enforce the **Role-based Access Control Policy** to restrict the ability to **change_default , modify , delete** the security attributes **corresponding to the access control policies** to **the roles authorized by the Role-based Access Control Policy**.

Application Note: *This activity is restricted to the Admin role by default but can be changed.*

6.1.5.2 Static attribute initialisation (FMT_MSA.3)

FMT_MSA.3.1 The TSF shall enforce the **Role-based Access Control Policy** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **the roles authorized by the Role-based Access Control Policy** to specify alternative initial values to override the default values when an object or information is created.

Application Note: *This activity is restricted to the Admin role by default but can be changed.*

6.1.5.3 Management of TSF data (Applications) (FMT_MTD.1(APP))

FMT_MTD.1.1 The TSF shall restrict the ability to **deploy** ~~the~~ **applications** to **the roles authorized by the Role-based Access Control Policy**.

Application Note: *This activity is restricted to the Admin and Deployer roles by default but can be changed.*

6.1.5.4 Management of TSF data (Role-based Access Control Policy) (FMT_MTD.1(RACP))

FMT_MTD.1.1 The TSF shall restrict the ability to **perform the management functions** ~~the~~ **stated in FMT_SMF.1** to **the roles authorized by the Role-based Access Control Policy** .

6.1.5.5 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

a) Start, suspend, resume and stop servers
b) Management of the domain configuration
c) Configuration of security providers
d) Management of users, groups, and group memberships
e) Management of security roles
f) Management of security policies
g) Management of password policies
h) Management of account locking policies
i) Management of Certificate Validation (level, allowed certificate policies, revocation)
j) Deployment of applications
k) Configuration of Web Services

.

Application Note: *Authorization of management functions are enforced by the Role-based Access Control Policy.*

Application Note: *Management of users, groups, group membership is enforced by the TOE only for certain authentication providers. Table 28 indicates the security authentication providers that can manage user and group information.*

6.1.5.6 Restrictions on security roles (FMT_SMR.2)

FMT_SMR.2.1 The TSF shall maintain the roles:

- **Admin**
- **AdminChannelUser**
- **Deployer**
- **Operator**
- **Monitor**
- **additional security roles**

.

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions

a) basic role conditions,
b) date and time role conditions, and
c) context element role conditions

are satisfied.

Application Note: *Please refer to section 7.1.2.1, "Security Roles" for a detailed description of the basic, date and time and context element policy conditions applicable to security roles.*

Application Note: *Not all security roles defined by default in the TOE are relevant for security management. Please refer to Table 22 for a complete enumeration of the default security roles.*

6.1.6 Protection of the TSF (FPT)

6.1.6.1 Basic internal TSF data transfer protection (FPT_ITT.1)

FPT_ITT.1.1 The TSF shall protect TSF data from **disclosure, modification** when it is transmitted between separate parts of the TOE.

Application Note: *This requirement covers the communication between the Administration server and the managed servers, and between clustered servers. The TOE relies on the JSSE and JCE providers for establishing a secure channel between the server instances.*

6.1.6.2 Internal TSF consistency (FPT_TRC.1)

FPT_TRC.1.1 The TSF shall ensure that TSF data is consistent when replicated between parts of the TOE.

FPT_TRC.1.2 When parts of the TOE containing replicated TSF data are disconnected, the TSF shall ensure the consistency of the replicated TSF data upon reconnection before processing any requests for **further changes in TSF data**.

Application Note: *This SFR covers the synchronization of configuration and TSF data between the Administration server and the managed servers within the same application server domain. Changes in the Administration server remain pending in a queue and are processed in the Managed server when communication is reestablished in the same order they were applied in the Administration server.*

6.1.7 Trusted path/channels (FTP)

6.1.7.1 Inter-TSF trusted channel (FTP_ITC.1)

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2 The TSF shall permit **the TSF , another trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for **communication with application clients and external LDAP servers**.

Application Note: *This requirement covers the communication between the application clients, web browsers, external LDAP servers and external RDBMS with the TOE. The TOE relies on the JSSE and JCE providers for establishing the secure channel.*

6.2 Security Functional Requirements Rationale

6.2.1 Security Requirements Coverage

The following table provides a mapping of SFR to the security objectives, showing that each security functional requirement addresses at least one security objective.

Security functional requirements	Objectives
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SEL.1	O.AUDITING
FCS_CKM.1-JCE	O.SEC_CHANNEL
FCS_CKM.2-JCE	O.SEC_CHANNEL
FCS_COP.1-JCE(TLS)	O.SEC_CHANNEL
FCS_COP.1-JCE(XMLSIG)	O.IA, O.MSG_PROT
FCS_COP.1-JCE(XMLENC)	O.MSG_PROT
FCS_COP.1-JCE(CLV)	O.IA, O.MSG_PROT, O.SEC_CHANNEL
FCS_COP.1-JCE(SPNEGO)	O.IA
FDP_ACC.2	O.AUTHORIZATION
FDP_ACF.1	O.AUTHORIZATION
FDP_IFC.1	O.MSG_PROT
FDP_IFF.1	O.MSG_PROT
FDP_UCT.1	O.MSG_PROT
FDP_UIT.1	O.MSG_PROT
FIA_AFL.1	O.IA
FIA_ATD.1	O.IA
FIA_SOS.1	O.IA
FIA_UAU.1	O.IA
FIA_UAU.5	O.IA
FIA_UID.1	O.IA
FIA_USB.1	O.IA

Security functional requirements	Objectives
FMT_MSA.1	O.AUTHORIZATION
FMT_MSA.3	O.AUTHORIZATION
FMT_MTD.1(APP)	O.AUTHORIZATION
FMT_MTD.1(RACP)	O.AUTHORIZATION
FMT_SMF.1	O.AUTHORIZATION, O.IA
FMT_SMR.2	O.AUTHORIZATION, O.IA
FPT_ITT.1	O.SEC_CHANNEL
FPT_TRC.1	O.CONSISTENCY
FTP_ITC.1	O.SEC_CHANNEL

Table 16: Mapping of security functional requirements to security objectives

6.2.2 Security Requirements Sufficiency

The following rationale provides justification for each security objective for the TOE, showing that the security functional requirements are suitable to meet and achieve the security objectives.

Security objectives	Rationale
O.AUDITING	FAU_GEN.1 defines the events that the TOE is required to audit. Those events are related to other security functional requirements showing which event contributes to make users accountable for their actions with respect to the requirement. FAU_GEN.2 requires that the events are associated with the identity of the user that caused the event. This association can only be established if the user is known, which is not the case for unsuccessful login attempts. FAU_SEL.1 and FMT_MTD.1(RACP) allow an authorized administrator to define the severity level of the events to be audited.
O.IA	The TSF must ensure that only authorized users gain access to the TOE and its resources. Users authorized to access the TOE have to use one or more identification and authentication mechanisms (FIA_UID.1, FIA_UAU.1 and FIA_UAU.5. Password-based authentication is protected against brute force attacks through an account locking mechanism (FIA_AFL.1) and the configuration of a password policy that ensures a quality metric (FIA_SOS.1). Authentication data is protected (FIA_ATD.1) and proper authorization for subjects acting on behalf of users is also ensured (FIA_USB.1). For authenticating identity assertions and web service policies, xml signature validation (FCS_COP.1-JCE(XMLSIG)), and certificate validation (FCS_COP.1-JCE(CLV)) are required. For authentication of SPNEGO identity assertions additional cryptographic functionality is required (FCS_COP.1-JCE(SPNEGO)).

Security objectives	Rationale
	Management of the authorization functionality is specified in FMT_SMF.1 and FMT_SMR.2.
O.AUTHORIZATION	The objective to allow the restriction of access to managed objects is implemented by the Role-based Access Control Policy as specified in FDP_ACC.2 and FDP_ACF.1. The management of the access control settings is specified in FMT_MSA.1, FMT_MSA.3, FMT_SMF.1, FMT_MTD.1(APP), FMT_MTD.1(RACP) and FMT_SMR.2.
O.SEC_CHANNEL	The TOE provides trusted channels (defined by FTP_ITC.1 and FPT_ITT.1) via the TLS protocol. FCS_CKM.1-JCE defines the generation of cryptographic keys, FCS_CKM.2-JCE defines how keys and digital certificates are exchanged and FCS_COP.1-JCE(TLS) cryptographic operations needed for supporting the corresponding cipher suites.
O.MSG_PROT	The TSF ensures that user data included in web service requests received and transmitted are protected from disclosure (FDP_UCT.1) and tampering (FDP_UIT.1). XML signature generation and verification (FCS_COP.1-JCE(XMLSIG)) and XML encryption and decryption (FCS_COP.1-JCE(XMLENC)) are also used for that purpose. Web service requests received by the TOE that do not meet these requirements are rejected (FDP_IFC.1 and (FDP_IFF.1).
O.CONSISTENCY	To ensure the consistency of configuration information and TSF data in multiple instances of the TOE, the TSF implements a replication mechanism between the Administration server and the managed servers (FPT_TRC.1).

Table 17: Security objectives for the TOE rationale

6.2.3 Security requirements dependency analysis

The following table demonstrates the dependencies of SFRs modeled in CC Part 2 and how the SFRs for the TOE resolve those dependencies.

Security functional requirement	Dependencies	Resolution
FAU_GEN.1	FPT_STM.1	The security functional requirement FAU_GEN.1 covering audit generation depends on FPT_STM.1 for gathering the timestamp for the audit records. The TOE relies on the underlying Java virtual machine to provide the appropriate timestamp as defined by OE.SYSTEM.
FAU_GEN.2	FAU_GEN.1	FAU_GEN.1
	FIA_UID.1	FIA_UID.1

Security functional requirement	Dependencies	Resolution
FAU_SEL.1	FAU_GEN.1	FAU_GEN.1
	FMT_MTD.1	FMT_MTD.1(RACP)
FCS_CKM.1-JCE	[FCS_CKM.2 or FCS_COP.1]	FCS_COP.1-JCE(TLS)
	FCS_CKM.4	This dependency is unresolved. The generated keys are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.
FCS_CKM.2-JCE	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1-JCE
	FCS_CKM.4	This dependency is unresolved. The generated keys are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.
FCS_COP.1-JCE(TLS)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1-JCE
	FCS_CKM.4	This dependency is unresolved. The keys used for encryption, decryption, and data authentication are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.
FCS_COP.1-JCE(XMLSIG)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	This dependency is unresolved. The keys used in xml signature generation and verification are generated and imported into the keystore using other tools.
	FCS_CKM.4	This dependency is unresolved. The keys used in xml signature generation and verification are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.
FCS_COP.1-JCE(XMLENC)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	This dependency is unresolved. The keys used in xml encryption/decryption are generated and imported into the keystore using other tools.
	FCS_CKM.4	This dependency is unresolved. The keys used in xml encryption/decryption are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.

Security functional requirement	Dependencies	Resolution
FCS_COP.1-JCE(CLV)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	This dependency is not applicable. No keys are generated for certificate validation.
	FCS_CKM.4	This dependency is not applicable. Only public keys are used for certificate validation.
FCS_COP.1-JCE(SP NEGO)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	This dependency is not applicable. No keys are generated for this security functionality.
	FCS_CKM.4	This dependency is not applicable. The keys used in these cryptographic operations are not formally destroyed. The object reuse mechanisms in the runtime environment prevent their use except for the intended context.
FDP_ACC.2	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1	FDP_ACC.2
	FMT_MSA.3	FMT_MSA.3
FDP_IFC.1	FDP_IFF.1	FDP_IFF.1
FDP_IFF.1	FDP_IFC.1	FDP_IFC.1
	FMT_MSA.3	FMT_MSA.3
FDP_UCT.1	[FTP_ITC.1 or FTP_TRP.1]	FTP_ITC.1
	[FDP_ACC.1 or FDP_IFC.1]	FDP_IFC.1
FDP_UIT.1	[FDP_ACC.1 or FDP_IFC.1]	FDP_IFC.1
	[FTP_ITC.1 or FTP_TRP.1]	FTP_ITC.1
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_ATD.1	No dependencies.	
FIA_SOS.1	No dependencies.	
FIA_UAU.1	FIA_UID.1	FIA_UID.1
FIA_UAU.5	No dependencies.	
FIA_UID.1	No dependencies.	
FIA_USB.1	FIA_ATD.1	FIA_ATD.1

Security functional requirement	Dependencies	Resolution
FMT_MSA.1	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.2
	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.3	FMT_MSA.1	FMT_MSA.1
	FMT_SMR.1	FMT_SMR.2
FMT_MTD.1(APP)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(RACP)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_SMF.1	No dependencies.	
FMT_SMR.2	FIA_UID.1	FIA_UID.1
FPT_ITT.1	No dependencies.	
FPT_TRC.1	FPT_ITT.1	FPT_ITT.1
FTP_ITC.1	No dependencies.	

Table 18: TOE SFR dependency analysis

6.3 Security Assurance Requirements

The security assurance requirements (SARs) for the TOE are the Evaluation Assurance Level 2 components as specified in [CC] part 3, augmented by ALC_FLR.1.

The following table shows the SARs, and the operations performed on the components according to CC part 3: iteration (Iter.), refinement (Ref.), assignment (Ass.) and selection (Sel.).

Security assurance class	Security assurance requirement	Source	Operations			
			Iter.	Ref.	Ass.	Sel.
ADV Development	ADV_ARC.1 Security architecture description	CC Part 3	No	No	No	No
	ADV_FSP.2 Security-enforcing functional specification	CC Part 3	No	No	No	No
	ADV_TDS.1 Basic design	CC Part 3	No	No	No	No
AGD Guidance documents	AGD_OPE.1 Operational user guidance	CC Part 3	No	No	No	No
	AGD_PRE.1 Preparative procedures	CC Part 3	No	No	No	No

Security assurance class	Security assurance requirement	Source	Operations			
			Iter.	Ref.	Ass.	Sel.
ALC Life-cycle support	ALC_CMC.2 Use of a CM system	CC Part 3	No	No	No	No
	ALC_CMS.2 Parts of the TOE CM coverage	CC Part 3	No	No	No	No
	ALC_DEL.1 Delivery procedures	CC Part 3	No	No	No	No
	ALC_FLR.1 Basic flaw remediation	CC Part 3	No	No	No	No
ASE Security Target evaluation	ASE_INT.1 ST introduction	CC Part 3	No	No	No	No
	ASE_CCL.1 Conformance claims	CC Part 3	No	No	No	No
	ASE_SPD.1 Security problem definition	CC Part 3	No	No	No	No
	ASE_OBJ.2 Security objectives	CC Part 3	No	No	No	No
	ASE_ECD.1 Extended components definition	CC Part 3	No	No	No	No
	ASE_REQ.2 Derived security requirements	CC Part 3	No	No	No	No
	ASE_TSS.1 TOE summary specification	CC Part 3	No	No	No	No
ATE Tests	ATE_COV.1 Evidence of coverage	CC Part 3	No	No	No	No
	ATE_FUN.1 Functional testing	CC Part 3	No	No	No	No
	ATE_IND.2 Independent testing - sample	CC Part 3	No	No	No	No
AVA Vulnerability assessment	AVA_VAN.2 Vulnerability analysis	CC Part 3	No	No	No	No

Table 19: SARs

6.4 Security Assurance Requirements Rationale

The evaluation assurance level has been chosen to match a Basic attack potential, commensurate with the threat environment that is experienced by typical consumers of the TOE. In addition, the evaluation assurance level augmented with ALC_FLR.1, commensurate with the augmented flaw remediation capabilities offered by the developer beyond those required by the evaluation assurance level.

7 TOE Summary Specification

7.1 TOE Security Functionality

The following sections explain how the security functionality is implemented, and how it covers the various SFR classes.

The primary security features of the TOE are:

- Identification and Authentication
- Authorization
- User Data Protection
- Auditing
- Security Management
- Cryptographic functionality

7.1.1 Identification and Authentication

The TOE provides identification and authentication of subjects using the following credentials:

- Username and password credential pairs
- X.509 digital certificates
- Identity Assertion tokens

The security framework implemented in the TOE performs this functionality through the following types of security providers, which can be:

- *Authentication provider*: provides identification and authentication using a username and password credential pair. The TOE supports several authentication providers, which primarily differ in which data store they use (e.g. LDAP server, database).
- *Identity Assertion provider*: provides identification and authentication based on certificates or security tokens.

When an access request is sent to the TOE, the TOE first determines whether authentication is required. If authentication is not required, the request proceeds and the *anonymous* subject is associated with the request. The *anonymous* subject represents the *anonymous* user, which is implicitly a member of the group *everyone*. If access to the resource requires authentication, the TOE requests user authentication using the method associated with the resource.

Once a subject is identified and authenticated, its identity is used to determine a set of principals representing the user and his/her group memberships). The subject and its principals are then associated with the security context.

In case a subject has been identified and authenticated before (for example, because access to a protected resource had been requested) and the new access request uses the same security context, then further identification and authentication are not required. Notice that not all types of requests get a persistent security context (i.e. web services).

7.1.1.1 Authentication in Web Applications

The TOE supports three types of authentication for Web applications:

- *BASIC*: the Web browser pops up a login screen with username and password in response to a WebLogic resource request. Authentication is enforced by a password-based authentication provider.

- *FORM*: the Web browser displays a custom login screen in response to a Web application resource. A password-based authentication provider is also used in this case.
- *CLIENT-CERT*: either a two-way SSL communication is configured, so the Web browser sends the client's digital certificate, or the Web browser sends a security token that is validated by an identity assertion provider. The identity assertion provider verifies the validity of the token or the digital certificate and maps it to a principal in the security realm.

The authentication method and the security roles authorized to access the resources are configured in the `web.xml` deployment descriptor. An ordered list of authentication methods can be declared for a resource, so the container can provide a fall-back mechanism in case the first authentication method fails (e.g. CLIENT-CERT, BASIC).

7.1.1.2 Web Service Authentication

The TOE supports authentication in Web Services according to the Web Service Security standard [WSS11-SOAP] with the following authentication mechanisms:

- Web Service Security Username Token Profile 1.0 and 1.1, according to [WSS11-UTP]
- Web Service Security X.509 Token profile 1.0 and 1.1, according to [WSS11-X509]
- Web Service Security SAML Token Profile 1.0 and 1.1, according to [WSS11-SAML]

Configuration of the authentication mechanisms are defined through Web Service policies according to the WS-Policy 1.2 standard defined in [OASIS-WSSP]. Both Java API for XML Web Services (JAX-WS) and Java API for XML-based RPC (JAX-RPC) are supported.

7.1.1.3 Password-based authentication providers

This type of authentication provider performs identification and authentication based on a username and password credential pair. The authentication provider searches the subject in the data store using the username and compares the provided and the stored password. If both matches (for some providers the comparison is on a hashed password), authentication for that provider succeeds.

The TOE uses the following user attributes from the data store:

- Username
- Password
- Group membership

The TOE can use the embedded LDAP server, an external LDAP server or a RDBMS as the data store. The TOE includes the following password-based authentication providers:

- The WebLogic Authentication provider accesses user and group information stored in the embedded LDAP server.
- The Oracle Internet Directory Authentication provider accesses user and group information stored in Oracle Internet Directory, an LDAP version 3 directory.
- The Oracle Virtual Directory Authentication provider accesses user and group information stored in Oracle Virtual Directory, an LDAP version 3 enabled service.
- LDAP Authentication providers for external stores access user and group information stored in Open LDAP, iPlanet, Microsoft Active Directory, and Novell NDS LDAP servers.
- RDBMS Authentication providers for external relational databases access user and group information stored in: SQL Authenticator, Read-only SQL Authenticator, and Custom RDBMS Authenticator (the latter not supported in the evaluated configuration).

7.1.1.4 Identity Assertion Providers

This type of authentication provider allows users to assert their identities using digital certificates and security tokens, and map the identity assertion to a subject. For example, if the authentication type in a Web application is set to CLIENT-CERT, the TOE performs identity assertion on values from request headers and cookies, using the identity assertion provider associated with the identified security token.

The TOE supports the following Identity Assertion Providers:

- The WebLogic Identity Assertion provider supports certificate authentication using X.509 certificates and CORBA Common Secure Interoperability version 2 (CSIv2) identity assertion. The provider validates the certificate, then obtains the username based on the Subject's distinguished name attribute configured for the provider.
- The LDAP X.509 Identity Assertion provider receives an X.509 certificate as the security token, looks up the LDAP object for the user associated with that certificate, ensures that the certificate in the LDAP object matches the presented certificate, and then retrieves the name of the user from the LDAP object.
- The SAML Identity Assertion provider V2 validates SAML 1.1 assertions and verifies the issuer is trusted. If so, identity is asserted based on the authentication statement contained in the assertion.
- The SAML 2.0 Identity Assertion provider validates SAML 2.0 assertions and verifies that the issuer is trusted. If so, identity is asserted based on the authentication statement contained in the assertion.
- The Negotiate Identity Assertion provider is used for SSO with Microsoft clients that support the SPNEGO protocol. Specifically, it decodes SPNEGO tokens to obtain Kerberos tokens, validates the Kerberos tokens, and maps Kerberos tokens to WebLogic users. The Negotiate Identity Assertion provider utilizes the Java Generic Security Service (GSS) Application Programming Interface (API) to accept the GSS security context via Kerberos.

The TOE also includes the SAML Authentication provider, which can be used in conjunction with the SAML 1.1 or SAML 2.0 Identity Assertion providers to allow the authentication of virtual users. The SAML authentication provider creates a subject using user and group principals extracted from the SAML identity assertion but that do not exist in the TOE.

7.1.1.5 Credential Mapping Providers

This type of provider maps subjects with credentials that can be used for performing authentication in other IT entities or federation services like single sign-on (SSO).

The TOE supports the following Credential Mapping Providers:

- The PKI (Public Key Infrastructure) Credential Mapping provider maps a subject (the initiator) and target resource (and an optional credential action) to a key pair or public certificate that can be used by applications when accessing the targeted resource. The PKI Credential Mapping provider uses the subject and resource name to retrieve the corresponding credential from the keystore.
- The SAML Credential Mapping provider V2 acts as a SAML authority (or SAML 1.1 source site) generating SAML v1.1 identity assertions and allowing single sing-on (SSO).
- The SAML 2.0 Credential Mapping provider acts as a SAML authority (or SAML 2.0 identity provider) and generates SAML 2.0 assertions that can be used to assert identity in the following use cases:
    - SAML 2.0 Web SSO Profile

- WS-Security SAML Token Profile version 1.1

7.1.1.6 Multiple authentication

The TOE supports multiple Authentication providers (and thus multiple LoginModules) for multi part authentication. Authentication providers are called in the order in which they were configured, and their behaviour in the overall authentication process depends on the configuration of the JAAS Control Flag for each Authentication provider. JAAS Control Flag values are:

- *REQUIRED*: authentication is required to succeed. The authentication process proceeds down to the next Authentication provider in the sequence.
- *REQUISITE*: authentication is required to succeed. If it fails, the authentication process stops and fails; if it succeeds, the authentication process proceeds down to the next authentication provider in the sequence.
- *SUFFICIENT*: authentication is not required to succeed. If it succeeds, the authentication process stops and succeeds; if it files, the authentication process proceeds down to the next authentication provider in the sequence.
- *OPTIONAL* authentication is not required to succeed. The authentication process proceeds down to the next Authentication provider in the sequence.

Notice that If authentication is required and all authentication providers are configured as *SUFFICIENT* or *OPTIONAL*, at least one authentication provider must succeed. When authentication providers are created, the JAAS Control Flag attribute is set to *OPTIONAL* by default.

7.1.1.7 User account lockout

The TOE defines a set of attributes to protect user accounts from unsuccessful authentication attempts when using password-based authentication providers. If a user account exceeds the threshold value after an unsuccessful authentication attempt, the user account is locked. The following table shows the lockout attributes and their default values:

Attribute	Description	Default value
Lockout Enabled	Requests the locking of a user account after invalid attempts to log in to that account exceed the specified Lockout Threshold.	Enabled
Lockout Threshold	Number of failed user password entries that can be tried before that user account is locked. Any subsequent attempts to access the account (even if the username/password combination is correct) raise a Security exception; the account remains locked until it is explicitly unlocked by the system administrator or another login attempt is made after the lockout duration period ends. Invalid login attempts must be made within a span defined by the Lockout Reset Duration attribute.	5
Lockout Duration	Number of minutes that a user's account remains inaccessible after being locked in response to several invalid login attempts within the amount of time specified by the Lockout Reset Duration attribute.	30 minutes

Attribute	Description	Default value
Lockout Reset Duration	Number of minutes within which invalid login attempts must occur in order for the user's account to be locked. An account is locked if the number of invalid login attempts defined in the Lockout Threshold attribute happens within the amount of time defined by this attribute.	5 minutes

Table 20: User account lockout attributes

User lockout attributes apply to a security realm and all its security providers.

7.1.1.8 Password validation

The Password Validation provider manages and enforces a set of configurable password composition rules, and is automatically invoked by a supported password-based authentication provider whenever a password is created or updated for a user in the realm. When invoked, the Password Validation provider performs a check to determine whether the password meets the criteria established by the composition rules. The password is then accepted or rejected as appropriate.

Password Validation can be enforced when using the following authentication providers:

- WebLogic Authentication provider
- SQL Authenticator provider
- LDAP Authentication provider
- Oracle Internet Directory Authentication Provider
- Oracle Virtual Directory Authentication Provider
- Active Directory Authentication provider
- iPlanet Authentication provider
- Novell Authentication provider

The Password Validation provider allows the use of the following composition rules to configure the password policy:

- Rules that determine whether the password may consist of or contain the user's name, or the reverse of that name.
- Rules for the minimum and/or maximum number of characters in a password
- Rules about the sequence or inclusion of characters in the password:
  - maximum number of repeating consecutive characters
  - maximum number of instances of any character
  - minimum number of numeric characters
  - minimum number of alphabetic characters
  - minimum number of lowercase characters
  - minimum number of uppercase characters
  - minimum number of numeric or special characters
  - minimum number of non-alphanumeric characters

7.1.1.9 Group membership

A group is a collection of users or groups. A group can contain none, one or more than one user or group, and a user can be a member of one or more groups.

When a user is authenticated, the user name and all groups where the user belongs are mapped to a set of principals. These principals can be associated to security roles and security policies, and are used as part of access authorization to resources.

The table below shows the groups defined by default in the TOE, the users that are member of each group and the security role including a basic role condition for the group (the role mapping mechanism assigns the role to the subject if the user belongs to that group):

Group name	Group membership	Security role
Administrators	By default, this group contains the user information for the initial administrative user entered as part of the installation process.	Admin
Deployers	By default, this group is empty.	Deployer
Operators	By default, this group is empty.	Operator
Monitors	By default, this group is empty.	Monitor
AppTesters	By default, this group is empty.	AppTester
CrossDomainConnectors	By default, this group is empty.	CrossDomainConnector
AdminChannelUsers	By default, this group is empty.	AdminChannelUser
OracleSystemGroup	By default, this group contains the user OracleSystemUser	OracleSystemRole

Table 21: Default groups

In addition, the TOE places dynamically all users in the following groups at run time (these groups are internal and cannot be managed by administrators):

- *users*: this group is implicitly assigned to all authenticated users.
- *everyone*: this group is implicitly assigned to unauthenticated and authenticated users.

7.1.1.10 Certificate Validation

The TOE performs validation of certificates as part of the authentication of identity assertions.

The TOE performs certificate validation as follows based on a set of trusted CAs:

- Verifies that either the root certificate in the chain or an intermediate CA certificate in the chain is trusted.
- Completes the certificate chain with trusted CAs.
- Verifies the signatures in the chain.
- Ensures that none of the certificates in the chain has expired.

By default, the TOE rejects any certificates in a certificate chain that do not have the Basic Constraint extension defined as CA, and accepts both X.509 version 1 and version 3 CA certificates. The following validation levels are also available:

- *strong*: ensures that the Basic Constraints extension on the CA certificate is defined as CA.
- *strong_nov1case*: same as strong, but X.509 version 1 CA certificates are rejected.
- *strict*: ensures the Basic Constraints extension on the CA certificate is defined as CA and set to critical. This option enforces [RFC2459].
- *strict_nov1cas*: same as strict, but X.509 version 1 CA certificates are rejected.
- *off*: no Basic Constraints extension checking. This options is not allowed in the evaluated configuration.

The TOE can also perform certificate revocation checking using the following methods:

- Online Certificate Status Protocol (OCSP) based on [RFC2560]
- Certificate Revocation Lists (CRLs)

Certificate revocation checking is configured on a domain-wide basis for all certificate authorities (CAs), but can be overridden for specific CAs.

The functionality above is implemented in the WebLogic CertPath provider. In addition, the Certificate Registry provider also performs certificate lookup and validation by building and validating the certificate chain but ensuring that the chain's end certificate is stored in the certificate registry (the embedded LDAP server). Certificate revocation is performed only removing the certificate from the registry.

For certificate validation, the TOE uses cryptographic functions provided by the JCE provider.

7.1.1.11 SFR coverage

The Identification and authentication functionality is designed to satisfy the following security functional requirements:

- FIA_AFL.1
- FIA_ATD.1
- FIA_UAU.1
- FIA_UAU.5
- FIA_UID.1
- FIA_USB.1
- FIA_SOS.1

7.1.2 Authorization

7.1.2.1 Security Roles

A security role is an identity granted to users or groups based on specific conditions. Multiple users or groups can be granted the same security role and a user or group can be in more than one security role. Security roles are used by policies to determine who can access a WebLogic resource.

Membership to a security role is based on a set of conditions that are dynamically evaluated at run time. The process of computing and granting roles is referred to as role mapping and occurs just before the authorization mechanism renders an access decision for a protected WebLogic resource.

The *XACML Role Mapping provider* determines the security roles applicable to a subject based on one or more conditions set for the role that can be combined using AND/OR logical operators and negations. The following built-in conditions are supported:

- *Basic Role Conditions*: maps the role to a specific user or group, valid values are:

    - *User*: maps the specified user to the role.
    - *Group*: maps the specified group to the role.
    - *Allow access to everyone*: maps all users and groups to the role.
    - *Deny access to everyone*: prevents any user or group from being in the role.
- *Date and Time Role Conditions*: maps the role to principals only during the specified time period
    - *Access occurs between specified hours*: allows access during a specified time period.
    - *Access occurs after*: allows access after a specified date and time.
    - *Access occurs before*: allows access before a specified date and time.
    - *Access occurs on specified days of the week*: allows access on specified days.
    - *Access occurs on the specified day of the month*: allows access on an ordinal day of the month.
    - *Access occurs after the specified day of the month*: allows access after an ordinal day in the month.
    - *Access occurs before the specified day of the month*: allows access before an ordinal day in the month.
- *Context Element Role Conditions*: based on the value of HTTP Servlet Request attributes, HTTP Session attributes, and EJB method parameters
    - *Context element defined*: allows access based on the existence of a specified attribute or parameter.
    - *Context element's value equals a numeric constant*: allows access based on a specified attribute or parameter's number value
    - *Context element's value is greater than a numeric constant*: allows access based on a specified attribute or parameter's number value
    - *Context element's value is less than a numeric constant*: allows access based on a specified attribute or parameter's number value
    - *Context element's value equals a string constant*

The special groups *users* and *everyone* are hard-wired into the authorization logic. A group matching rule always returns true for the group *everyone*, and always returns true for the group *users* if the user has been authenticated (i.e. is not the anonymous subject).

The role mapping process is initiated when a container requests the security framework to determine whether or not the user should be granted access to the resource. The security framework then calls the XACML Role Mapping provider to obtain the list of roles that meet the conditions, and then use the resulting set of roles to call the configured authorization provider or providers to get an access decision.

Role mapping can also occur in the following scenarios:

- when a container needs to obtain the set of roles applicable to a user, or
- when a container needs to know whether a user has a specific role.

There are two types of security roles:

- A ***global role*** are defined at security realm level and can be used in any security policy. Table 22 shows the set of global security roles defined by default in a security realm, the default group defined as the basic condition of the role (e.g. all users belonging to the Administrators group are assigned to the Admin role). The third column indicates whether the role is relevant to security management in the evaluated configuration.
- A ***scoped role*** is particular to an application deployment, and cannot be used in other deployments. Roles derived from deployment descriptors are example of scoped roles.

Role name	Default groups	Security Management
Admin	Administrators	yes
AdminChannelUsers	AdminChannelUsers, Administrators, Deployers, Operators, Monitors, and AppTesters	yes
Anonymous	everyone	no
Deployer	Deployers	yes
Operator	Operators	yes
Monitor	Monitors	yes
AppTester	AppTesters	no
CrossDomainConnector	CrossDomainConnectors	no
OracleSystemRole	OracleSystemGroup	no

Table 22: Default security roles

Note: *Not all roles and groups enumerated in the table above are relevant to security management. All roles provided by default are enumerated for completeness.*

The following examples show how role mapping works together with user and group membership:

- User Joe belongs to the Administrators group. When Joe tries to access the Administrator Console, the TOE prompts for authentication (the web application is a protected resource), and after a successful login, the TOE populates the subject with user *Joe* and group *Administrators*. The role mapping mechanism evaluates the roles dynamically, and assigns the role *Admin* to the subject; further access decisions are based on the security policies assigned to the *Admin* role.
- A role *AdminForEmergencies* has a basic role assignment to user *Mick*, and date and time condition rules that activate the role only on weekends and weekdays between 8 pm. and 8 am. The security role is also assigned in basic conditions of security policies to allow certain security management actions (e.g. start and stop servers, deploy applications). When user *Mick* authenticates, the TOE populates the subject with user *Mick*. During normal hours, the role mapping mechanism does not activate the *AdminForEmergencies* role, so access to security management actions is forbidden. Only during weekends and overnight does the role mapping mechanism activate the role, thus granting access to the user.

7.1.2.2 Resources

A resource is a structured object used to represent an underlying entity or action that can be protected from unauthorized access. Protection is provided through the use of security policies, which define, based on a set of conditions, whether access to the resource is allowed or not.

Resources are arranged in a hierarchical structure starting from the resource type (e.g. EJB type) until the most specific resource (e.g. an EJB method). A security policy can be defined at any level in the resource hierarchy, either at the top (resource type), the bottom (resource instance) or at an intermediate level (e.g. all EJB resources belonging to an application). Lower levels in the hierarchy without a security policy inherit that policy (the policy of a narrower scope overrides policy of a broader scope).

A resource is identified by a set of properties that depends on the resource type. The following sections show the types of resources supported by the TOE, their properties and the resource hierarchy where a security policy can be set.

Admin resources

Policies for administrative resources determine who can complete such tasks as uploading files (used during deployment), viewing the domain and server logs, and unlocking users who have been locked out of their accounts.

This resource type is identified as follows:

```
type=<adm>, category=, realm=, action=
```

where:

type
Resource type. Always "adm" for this resource.

category
Category of the Admin resource

- *UserLockout*
- *Configuration*
- *FileUpload*
- *FileDownload*
- *ViewLog*
- *CrossDomain*
- *AdminChannel*

realm
Administrative resource name (only meaningful to the *UserLockout* category

action
Actions for the following categories:

- UserLockout:
  - unlockuser
- FileDownload:
  - wl_component_request
  - wl_ear_resource_request
  - ear_request
  - wl_xml_entity_request
  - wl_jsp_refresh_request
  - file
  - wl_init_replica_request
  - wl_file_realm_request
  - wl_managed_server_independence_request

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<adm>, category=, realm=, action=
type=<adm>, category=, realm=
type=<adm>, category=
type=<adm>
```

Application resources

An enterprise application, Web application, or other Java EE module that is deployed as a stand-alone application (for example, Web Services and JDBC modules). This type is used to protect all resources that constitute an application.

This resource type is identified as follows:

```
type=<app>, application=
```

where:

type
Resource type. Always "app" for this resource.

application
Application name

This resource type does not have a hierarchy per se. This resource is primarily used in the creation of a security policy for an application that contains multiple, different types of components. In particular, it should be used to define a security policy that is applied equally to all components of an application, instead of to any specific component of the application.

Component Object Model (COM) resources

Represents a package that contains one or more jCOM classes. jCOM is a software bridge that allows bidirectional access between Java/Java EE objects deployed in WebLogic Server and Microsoft ActiveX components. A policy on a COM resource protects access to all jCOM objects in a package.

This resource type is identified as follows:

```
type=<com>, application=, className=
```

where:

type
Resource type. Always "com" for this resource.

application
Application name

className
Class name

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<com>, application=, className=my.package.MyClass
type=<com>, application=, className=my.package
type=<com>, application=, className=my
type=<com>
```

Note: *COM resources are not allowed in the evaluated configuration, but their properties are mentioned here for completeness.*

Enterprise Information System (EIS) resources (resource adapters)

An EIS resource is a system-level software driver used by an application server, such as WebLogic Server, to connect to an Enterprise Information System.

This resource type is identified as follows:

```
type=<eis>, application=, module=, eis=
```

where:

type
Resource type. Always "eis" for this resource.

application
Application name

module
Module name

eis
EIS resource name

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<eis>, application=, module=, eis=
type=<eis>, application=, module=
type=<eis>, application=
type=<app>, application=
type=<eis>
```

Enterprise JavaBean (EJB) resources

An EJB (Enterprise JavaBean) resource is an EJB deployment module (JAR), individual EJB, or individual method in an EJB. EJB resources exist within a hierarchy of resources, and at the top of the hierarchy is an application resource.

This resource type is identified as follows:

```
type=<ejb>, app=, module=, ejb=, method=, methodInterface=, methodParams=
```

where:

type
Resource type. Always "ejb" for this resource.

app
Application name

module
Module name

ejb
EJB name

methodInterface
EJB method interface of the resource:

- Home
- Remote
- LocalHome
- Local

methodParams
Parameter signature of the EJB method

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<ejb>, app=, module=, ejb=, method=, methodInterface=,
methodParams={argumentType1, argumentType2}
type=<ejb>, app=, module=, ejb=, method=, methodInterface=
type=<ejb>, app=, module=, ejb=, method=
type=<ejb>, app=, module=, ejb=
type=<ejb>, app=, module=
type=<ejb>, app=
type=<app>, app=
type=<ejb>
```

Java Database Connectivity (JDBC) resources

A Java DataBase Connectivity (JDBC) resource is a JDBC system resource, JDBC module that is part of an application, JDBC data source, or a specific method within a data source. If a JDBC module is deployed as a stand-alone application, the application is represented by an application resource. JDBC resources exist within a hierarchy of resources, and at the top of the hierarchy is an application resource.

This resource type is identified as follows:

```
type=<jdbc>, application=, module=, resourceType=, resource=, action=
```

where:

type
resource type. Always "jdbc" for this resource.

application
Application name

module
Module name

resourceType
Category of the JDBC resource:

- *ConnectionPool*
- *MultiPool*

resource
Name of the JDBC resource

action
Name of the action being requested on the resource:

- *reserve*
- *admin*
- *shrink*
- *reset*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<jdbc>, application=, module=, resourceType=, resource=, action=
type=<jdbc>, application=, module=, resourceType=, resource=
type=<jdbc>, application=, module=, resourceType=, action=
type=<jdbc>, application=, module=, resourceType=
type=<jdbc>, application=, action=
type=<jdbc>, application=
type=<app>, application=
type=<jdbc>, action=
type=<jdbc>
```

Java Messaging Service (JMS) resources

A JMS resource is a JMS system resource, JMS module that is part of an application, JMS destination, or an operation within a destination. Security policies and roles can be created for all destinations (JMS queues and JMS topics) as a group, or an individual destination (JMS queue or JMS topic) on a JMS server. These resources exist within a hierarchy of resources, and at the top of the hierarchy is an application resource. When a policy is created for a specific destination on a JMS server, the following protection operations on the destination can be used: send, receive, browse, ALL.

This resource type is identified as follows:

```
type=<jms>, application=, destinationType=, resource=, action=
```

where:

type
Resource type. Always "jms" for this resource.

application
Application name

destinationType
Type of destination resource:
- *topic*
- *queue*

resource
Name of the JMS resource, either topic name or queue name

action
Name of the action being requested on the resource:
- *send*
- *receive*
- *browse*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<jms>, application=, destinationType=, resource=, action=
type=<jms>, application=, destinationType=, resource=
type=<jms>, application=, destinationType=, action=
type=<jms>, application=, destinationType=
type=<jms>, application=, action=
type=<jms>, application=
type=<app>, application=
type=<jms>, action=
type=<jms>
```

JMX resources

A JMX resource is an MBean attribute or MBean operation. A policy on a JMX resource controls who can read or write MBean attributes or invoke operations.

This resource type is identified as follows:

```
type=<jmx>, operation=, application=, beanType=, target=
```

where:

type
Resource type. Always "jmx" for this resource.

operation
Name of the operation being requested on the resource:
- *get*
- *getEncrypted*
- *set*
- *setEncrypted*
- *find*
- *invoke*
- *create*
- *unregister*

application
Name of the application or the system resource

beanType
The fully qualified type of the MBean resource on which access is being requested

target
The target of the MBean resource on which access is being requested:

- For an *invoke* or *find* operation, the *method name*.
- For a *get*, *getEncrypted*, *set*, or *setEncrypted* operation, the *property name*.
- For a *create* or *unregister* operation, the target is always null.

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<jmx>, operation=, application=, beanType=, target=
type=<jmx>, operation=, application=, beanType=
type=<jmx>, operation=, application=
type=<jmx>, operation=, application=, beanType=, target=
type=<jmx>, operation=, application=, beanType=
type=<jmx>, operation=, application=
type=<jmx>, operation=
type=<jmx>
```

JMX resources are used by the TOE when MBean authorization is delegated to the authorization providers (security realm attribute *DelegateMBeanAuthorization = True*). Otherwise, access control is enforced using the default security policies for MBeans.

Java Naming and Directory Interface (JNDI) resources

A JNDI resource is a node in a server's JNDI tree. A policy on a JNDI resource determines who can access entities and actions through JNDI. A policy can be created on the root node of the JNDI tree or on individual nodes. One or more of the following operations can be used: modify, lookup and list.

This resource type is identified as follows:

```
type=<jndi>, application=, path=, action=
```

where:

type
Resource type. Always "jndi" for this resource.

application
Application name

path
Elements of the path in the JNDI tree

action
Action being requested on the JNDI resource:

- *lookup*
- *modify*
- *list*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<jndi>, application=, path={pathComponent1,pathComponent2}, action=
type=<jndi>, application=, path={pathComponent1,pathComponent2}
```

```
type=<jndi>, application=, path={pathComponent1}, action=
type=<jndi>, application=, path={pathComponent1}
type=<jndi>, application=, path={}, action=
type=<jndi>, application=, path={}
type=<jndi>, action=
type=<jndi>
```

Server resources

Policies that control the state of a server instance; they can be defined for all server instances in a domain or to individual servers. Operations that can be protected are: boot, shutdown, suspend, resume.

This resource type is identified as follows:

```
type=<svr>, application=, server=, action=
```

where:

type
Resource type. Always "srv" for this resource.

application
Application name

server
Server name

action
Action being requested on the resource:

- *boot*
- *shutdown*
- *suspend*
- *resume*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<svr>, application=, server=, action=
type=<svr>, application=, server=
type=<svr>, action=
type=<svr>
```

Universal Resource Locator (URL) resources

A URL resource is a specific URL or URL pattern in a Web application. Policies for a URL resource can protect all HTTP methods for a specified URL or URL pattern, or only specific HTTP methods. These resources exist within a hierarchy of resources, and at the top of the hierarchy is an application resource.

This resource type is identified as follows:

```
type=<url>, application=, contextPath=, uri=, httpMethod=, transportType=
```

where:

type
Resource type. Always "url" for this resource.

application
Application name

contextPath
Context path for the Web application

uri
The URI of the resource, relative to the context path.

httpMethod
Name of the HTTP methods on the URL resource:

- *OPTIONS*
- *GET*
- *HEAD*
- *POST*
- *PUT*
- *DELETE*
- *TRACE*
- *CONNECT*

transportType
Transport guarantee required to access the URL resource:

- *INTEGRAL*
- *CONFIDENTIAL*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<url>, application=, contextPath="/mywebapp", uri=/foo/bar/my.jsp,
httpMethod=, transportType=
type=<url>, application=, contextPath="/mywebapp", uri=/foo/bar/my.jsp, httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=/foo/bar/my.jsp
type=<url>, application=, contextPath="/mywebapp", uri=/foo/bar/my.jsp/*,
httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=/foo/bar/my.jsp/*
type=<url>, application=, contextPath=/MyWebApp, uri=/foo/Bar/*, httpMethod=
type=<url>, application=, contextPath=/MyWebApp, uri=/foo/Bar/*
type=<url>, application=, contextPath="/mywebapp", uri=/foo/*, httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=/foo/*
type=<url>, application=, contextPath="/mywebapp", uri=/*, httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=/*
type=<url>, application=, contextPath="/mywebapp", uri=*.jsp, httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=*.jsp
type=<url>, application=, contextPath="/mywebapp", uri=/, httpMethod=
type=<url>, application=, contextPath="/mywebapp", uri=/
type=<url>, application=, contextPath="/mywebapp"
type=<url>, application=
type=<app>, application=
type=<url>
```

Web Service resources

A Web Service resource is a Web Service module (WAR or JAR) or an operation within a Web Service module. Web Services are protected by the following hierarchy of resources:

- The application resource for the parent application.
- The Web Service resource for the Web Service module (WAR or JAR). Individual Web Service resources for each Web Service operation.

If the Web Service is implemented with standard Java objects, any of the above resources protect the Java objects. If the Web Service is implemented with an EJB any of the above or any of the following resources protect the EJB implementation:

- The EJB resource for the EJB.
- Individual EJB resources for each EJB method.

This resource type is identified as follows:

```
type=<webservices>, application=, method=, signature=
```

where:

type
Resource type. Always "webservices" for this resource.

application
Application name

method
Web Service name

signature
Web Service parameters

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<webservices>, application=, method=, signature={argumentType1, argumentType2}
type=<webservices>, application=, method=
type=<webservices>, application=
type=<app>, application=
type=<webservices>
```

Work Context resources

Work Contexts enable Java EE developers to define and pass properties without including them in a remote call. A Work Context resource represents the operations that create, delete, read, or modify a property. A policy can be created for all operations of a given property, or one policy for each operation.

This resource type is identified as follows:

```
type=<workcontext>, path=, actionName=
```

where:

type
Resource type. Always "workcontext" for this resource.

path
Path of the work context

actionName
Action requested on the work context resource:

- *create*
- *delete*
- *modify*
- *read*

The following sequence shows the hierarchy for this resource type from most to least specific:

```
type=<workcontext>, path={pathComponent1,pathComponent2}, actionName=
type=<workcontext>, path={pathComponent1,pathComponent2}, actionName=
type=<workcontext>, path={pathComponent1,pathComponent2}
type=<workcontext>, path={pathComponent1}, actionName=
type=<workcontext>, path={pathComponent1}
type=<workcontext>, path={}, actionName=
type=<workcontext>, path={}
type=<workcontext>
```

7.1.2.3 Security Policies

A security policy specifies which users, groups, or roles can access a resource under a set of conditions. Security policies can be assigned to any of the defined resources, or to attributes or operations of a particular instance of a resource (e.g. an EJB method).

The TOE provides a default security policy (known as the root-level policy) for each resource type (see table below). This security policy protects all resources of a specific type, but can be overridden by defining security policies at lower levels of the resource hierarchy.

Resource type	Default security policy
Administrative	Default global role: Admin
Application	None
Component Object Model (COM)	None
Enterprise JavaBean (EJB)	Default group: everyone
Enterprise Information System (EIS)	Default group: everyone
Java Database Connectivity (JDBC)	Default group: everyone
Java Messaging Service (JMS)	Default group: everyone
Java Naming and Directory Interface (JNDI)	Default group: everyone
JMX	None
Server	Default global roles: Admin, Operator
Universal Resource Locator (URL)	Default group: everyone
Web Service	Default group: everyone

Resource type	Default security policy
Work Context	Default group: everyone

Table 23: Default Security Policy for root resource types

The XACML Authorization provider determines dynamically whether the policy is applicable to a subject and resource based on the hierarchy of resources and the set of conditions defined in the policy.

Conditions defined for a security policy can be a combination of AND, OR and negation logical operators. The following conditions are supported:

- *Basic policy conditions*:
  - *User*: allows a specific user to access the resource.
  - *Group*: allows all users or groups in the specified group to access the resource unless a User or Role condition contradicts the Group condition.
  - *Role*: allows all users or groups in the specified role to access the resource unless a User or Group condition contradicts the Role condition.
  - *Allow access to everyone*: allows access for all users, groups, and roles.
  - *Deny access to everyone*: prohibits access for all users, groups, and roles.
  - *Element requires signature by*: creates a condition for a security policy based on who has digitally signed an element in the SOAP request message that invokes a Web Service operation (used only when securing Web Services resources).
- Date and time policy conditions:
  - *Access occurs between specified hours*: allows access during a specified time period.
  - *Access occurs after*: allows access after a specified date and time.
  - *Access occurs before*: allows access before a specified date and time.
  - *Access occurs on specified days of the week*: allows access on specified days.
  - *Access occurs on the specified day of the month*: allows access on an ordinal day of the month.
  - *Access occurs after the specified day of the month*: allows access after an ordinal day in the month.
  - *Access occurs before the specified day of the month*: allows access before an ordinal day in the month.
- Context element policy conditions: based on the value of HTTP Servlet Request attributes, HTTP Session attributes, and EJB method parameters
  - *Context element defined*: allows access based on the existence of a specified attribute or parameter.
  - *Context element's value equals a numeric constant*: allows access based on a specified attribute or parameter's number value
  - *Context element's value is greater than a numeric constant*: allows access based on a specified attribute or parameter's number value
  - *Context element's value is less than a numeric constant*: allows access based on a specified attribute or parameter's number value

- *Context element's value equals a string constant*

The special groups *users* and *everyone* are hard-wired into the authorization logic. A group matching rule always returns true for the group *everyone*, and always returns true for the group *users* if the user has been authenticated (i.e. is not the anonymous subject).

The TOE also supports the declarative and programmatic authorization mechanisms for EJBs, URLs and Web Services, as defined in the J2EE standard. When an application is deployed in the TOE, the security information contained in the deployment descriptors (J2EE roles and policies) and annotations included in the application are imported by the XACML Role Mapping and XACML Authorization providers. How these mechanism are imported in the security model depends on the option chosen for the application at the time it is deployed or the default established for the security realm if the option is not specified:

- *Deployment Descriptors only*: only the policies in the J2EE deployment descriptors and annotations are considered during application deployment.
- *Customize Roles only*: uses J2EE policies defined in the deployment descriptors and annotations but ignores the J2EE roles. It uses the standard role mapping mechanism implemented in the TOE.
- *Customize Roles and Policies* : uses the standard role mapping and authorization mechanisms implemented in the TOE. J2EE roles and policies defined in the deployment descriptors or annotations are ignored.
- *Advanced*: allow customization of the behavior during deployment (initialize or ignore J2EE roles and policies from deployment descriptors), and authorization (check roles and policies in all Web applications and EJBs, or only in those protected by deployment descriptors).

Resources other than Web applications, EJBs and Web Services can be protected through security roles and policies, regardless of the security model chosen.

In addition, programmatic authorization is also supported. The TOE only provides security related information (username, verification of whether the user belongs to a given role, etc.) so the application can take the proper access decision. This is independent of the application deployment behavior chosen.

The following sections explains in more detail the J2EE authorization mechanisms supported by the TOE.

Authorization in Enterprise JavaBeans (EJB)

The TOE supports the following authorization mechanisms for EJBs in accordance to the J2EE standard:

- Declarative Security Via Metadata Annotations
- Declarative Security Via Deployment Descriptors
- Programmatic authorization using EJB methods

Metadata Annotations

Metadata annotations in EJBs specify the roles that are allowed to invoke all, or a subset, of the EJB’s methods (see section "Enterprise JavaBean (EJB) resources" to see the hierarchy in EJB resources). This model gives the application developer more control without having to implement programmatic authorization in EJBs.

The following security-related annotations are available:

Annotation	Description
javax.annotation.security.DeclareRoles	Explicitly lists the security roles that will be used to secure the EJB.
javax.annotation.security.RolesAllowed	Specifies the security roles that are allowed to invoke all the methods of the EJB (when specified at the class-level) or a particular method (when specified at the method-level.)
javax.annotation.security.DenyAll	Specifies that the annotated method can not be invoked by any role.
javax.annotation.security.PermitAll	Specifies that the annotated method can be invoked by all roles.
javax.annotation.security.RunAs	Specifies the role which runs the EJB. By default, the EJB runs as the user who actually invokes it.

Table 24: EJB annotations

Authorization is enforced only when clients request EJB methods that are protected by a role specified in an annotation.

If deployments descriptors are also used to enforce security, deployment descriptor elements always override their annotation counterparts in case of conflict.

Deployment Descriptors

This method provides declarative security to EJBs. It uses only roles and policies defined by a developer in the following deployment descriptors:

- The `ejb-jar.xml` file maps the EJB name and optionally the method name with the security roles authorized to access. This information is imported by the XACML Authorization provider.
- The `weblogic-ejb-jar.xml` file maps a security role to one or more principals (users or groups). This information is imported by the XACML Role Mapping provider.

Authorization is enforced only when clients request EJB methods that are protected by a policy in the deployment descriptor

If deployments descriptors are also used to enforce security, deployment descriptor elements always override their annotation counterparts in case of conflict.

Programmatic methods

The TOE provides the following methods to allow deployed applications the implementation of programmatic authorization in EJBs:

- EJBContext.isCallerInRole
- EJBContext.getCallerPrincipal

In this case, the TOE only provides this information to the deployed application and does not make any access decision.

Authorization in Web applications

The TOE supports additional authorization mechanism for Web applications in accordance to the J2EE standard:

- Declarative Security Via Metadata Annotations
- Declarative Security Via Deployment Descriptors
- Programmatic authorization in servlets

Metadata Annotations

Metadata annotations in servlets specify the roles that are allowed to invoke all, or a subset, of the HTTP protocol methods.

The following security-related annotations are available:

Annotation	Description
`javax.servlet.annotation.ServletSecurity`	Encloses security constraints to be enforced by a Servlet container on HTTP protocol messages.
`javax.servlet.annotation.HttpConstraint`	Specifies the security constraints (roles allowed, deny all, permit all) to be applied to all HTTP protocol methods not considered in the more specific HttpMethodConstraint annotation.
`javax.servlet.annotation.HttpMethodConstraint`	Specifies the security constraints (roles allowed, deny all, permit all) to be applied to an HTTP protocol method.

Table 25: Servlet annotations

If deployments descriptors are also used to enforce security, deployment descriptor elements always override their annotation counterparts in case of conflict.

Deployment Descriptors

This method provides declarative security. It uses J2EE roles and policies defined by a developer in the following deployment descriptors:

- The `web.xml` file deployment descriptor maps the URL and optionally the method name with the security roles authorized to access. This information is imported by the XACML Authorization provider.
- The `weblogic.xml` file contains the mapping between security roles and principals (users or groups). This information is imported by the XACML Role Mapping provider.

Authorization is enforced only when clients request URLs are protected by a policy in the deployment descriptor,

Programmatic methods

The TOE provides the following methods to allow deployed applications the implementation of programmatic authorization in the servlet code:

- javax.servlet.http.HttpServletRequest.isUserInRole
- javax.servlet.http.HttpServletRequest.getUserPrincipal

In this case, the TOE only provides this information to the deployed application and does not make any access decision.

Authorization in Web Services

The TOE supports additional authorization mechanism for Web services in accordance to the J2EE standard:

- Declarative Security Via Metadata Annotations
- Declarative Security Via Deployment Descriptors

Metadata Annotations (JAX-RPC only)

Metadata annotations in Java Web Service (JWS) files allows access control security for web services. The following security-related annotations are available:

Annotation	Description
`weblogic.jws.security.RolesAllowed`	Specifies whether to enable basic authentication for a Web Service
`weblogic.jws.security.SecurityRole`	Specifies the name of a role that is allowed to invoke the Web Service. If the `mapToPrincipals` attribute is set, the mapping to the list of principals is applicable within the context of the web service; if the attribute is not declared, the standard role mapping mechanism is enforced.
`weblogic.jws.security.RolesReferenced`	Specifies the list of role names that reference actual roles that are allowed to invoke the Web Service.
`weblogic.jws.security.SecurityRoleRef`	Specifies a role name reference that links to an already-specified role that is allowed to invoke the Web Service.
`weblogic.jws.security.RunAs`	Specifies the role and user identity which actually runs the Web Service in WebLogic Server.

Table 26: Web Service annotations

Authorization is enforced only when clients request web services that are protected by a role specified in an annotation.

Deployment Descriptors

This method provides declarative security. It uses only roles and policies defined by a developer in the following deployment descriptors:

- The `web.xml` file deployment descriptor maps the Web Service and optionally the method name with the security roles authorized to access. This information is imported by the XACML Authorization provider.

7.1.2.4 Access Decisions

An Access Decision is the component of an authorization provider that determines whether or not a subject has permission to perform a given operation on a resource with specific parameters in an application. Given this information, the Access Decision responds with a result of *PERMIT*, *DENY*, or *ABSTAIN*.

If there are multiple authorization providers configured, the WebLogic Adjudication Provider is required to tally the multiple Access Decisions and render a verdict, which can be a *PERMIT* or *DENY* decision. The WebLogic Adjudication provider has an attribute called Require Unanimous Permit that governs its behavior.

By default, the Require Unanimous Permit attribute is set to *TRUE*, which causes the WebLogic Adjudication Provider to act as follows:

- If all the Authorization providers' Access Decisions return *PERMIT*, then return a final verdict of *TRUE* (that is, permit access to the resource).
- If some Authorization providers' Access Decisions return *PERMIT* and others return *ABSTAIN*, then return a final verdict of *FALSE* (that is, deny access to the resource).
- If any of the Authorization providers' Access Decisions return *ABSTAIN* or *DENY*, then return a final verdict of *FALSE* (that is, deny access to the resource).

If the Require Unanimous Permit attribute is set to *FALSE*, the WebLogic Adjudication provider acts as follows:

- If all the Authorization providers' Access Decisions return *PERMIT*, then return a final verdict of *TRUE* (that is, permit access to the resource).
- If some Authorization providers' Access Decisions return *PERMIT* and others return *ABSTAIN*, then return a final verdict of *TRUE* (that is, permit access to the resource).
- If any of the Authorization providers' Access Decisions return *DENY*, then return a final verdict of *FALSE* (that is, deny access to the WLS resource).

For the evaluated configuration, only the XACML Authorization provider is allowed, therefore the attribute has no effect in the authorization process.

7.1.2.5 SFR coverage

Authorization functionality is designed to satisfy the following security functional requirements:

- FDP_ACC.2
- FDP_ACF.1

7.1.3 User Data Protection

7.1.3.1 Web Services

The TOE provides integrity and confidentiality of SOAP messages in web services according to the OASIS Web Services Security standard [OASIS-WSS], specifically [WSS11-SOAP]. Message-level security in web services is defined through security policy assertions that follow the Web Services Security Policy 1.2 specification [OASIS-WSSP]. XML Signature Syntax and Processing [W3CXMLSIG] and XML Encryption Syntax and Processing [W3CXMLENC] are used to provide message integrity and confidentiality, respectively.

Policy assertions describe whether and how the SOAP messages resulting from an invoke of an operation should be digitally signed or encrypted. They can also specify that a client application authenticate itself using a username, SAML, or X.509 token (this is handled by the corresponding identity assertion provider, see section 7.1.1.

The TOE includes a set of predefined policy assertion files that can be used to protect user data in web services. If the TOE detects that a web service request includes an invalid signature or authentication of its security token fails, the TOE enforces the information flow control policy and rejects the request.

The TOE performs XML signature generation and validation, and XML encryption and decryption using the cryptographic functionality provided by the TOE.

7.1.3.2 Secure communication

The TOE supports the HTTP, IIOP and T3 protocols for communication. User data can be also protected by establishing a secure channel over any of those protocols: HTTPS, IIOPS and T3S, respectively. The TLS protocol and supported cipher suites are implemented by the JSSE and JCE providers .

Secure communication is provided for communication between:

- the TOE and remote client applications (web browsers, client applications using RMI, web services or http requests);
- the TOE and IT external entities (e.g. LDAP servers);
- clustered managed servers within the domain; and
- Administration server and managed servers within the domain.

See section 7.1.6 for more information.

7.1.3.3 SFR coverage

User Data Protection functionality is designed to satisfy the following security functional requirements:

- FDP_IFC.1
- FDP_IFF.1
- FDP_UCT.1
- FDP_UIT.1

7.1.4 Auditing

The TOE provides auditing services through the WebLogic Auditing provider. Each TOE component (authentication providers, authorization providers, etc.) invokes the Auditing Provider when a security-relevant event occurs, providing all pertinent information, except the timestamp, which the Auditing Provider queries from the underlying operating system.

Audit records include a timestamp, severity level, event type, and event specific information (including the identity of the responsible user when applicable, the applicable server, and the component logging the audit event). All recorded audit events are written into a file provided by the hosting operating system and are accessible via operating system functions.

Table 8 shows the audit events that are generated by the security providers bundled into the WebLogic Server security framework.

The WebLogic Auditing Provider can be configured to generate audit records based on the following severity levels:

1. INFORMATION
2. WARNING
3. ERROR
4. SUCCESS
5. FAILURE

The administrator can specify the level of audit in two ways:

- For each severity level. The audit events will be generated depending on whether the corresponding severity level is enabled or disabled.
- Specifying the minimum severity level that will be audited in the order shown above (e.g. if the minimum severity level is set to WARNING, audit will be generated for WARNING, ERROR, SUCCESS and FAILURE events).

7.1.4.1 SFR coverage

The Security Audit functionality is designed to satisfy the following security functional requirements:

- FAU_GEN.1
- FAU_GEN.2
- FAU_SEL.1

7.1.5 Security Management

The TOE implements the concept of an application server domain: one or more TOE instances provide services within a single unit of administration, sharing configuration information (server configuration, deployed applications, etc.) and the same security realm data (security providers, users, groups, security roles, security policies, etc.). Security management is always performed through one of the TOE instances that assumes the role of the domain's Administration server.

Security management is performed by administrators through the use of one of the following tools:

- The WebLogic Administration Console
- Any application using the Java Management Extension (JMX) application program interface (API). The WebLogic Language Scripting Tool (WLST) in an utility bundled in the TOE package.

Administrators always connect to the Administration server (one of the TOE instances in the domain) through a secure channel provided by the TOE .

When a security domain is created, the TOE also creates a default security realm and assigns default values for TSF data (e.g. security providers, users, security roles, root-level security policies, password policy, etc.). The Administrator can change these default values through the interfaces mentioned above.

By default, the TOE enforces access control on security management actions through a set of default security policies for MBeans defined in [WLMBR] (see section "Default Security Policies for MBeans"). These default security policies establish which security roles can access to the MBean model at object, attribute, or operation level. The following table describes the management roles for system administration operations, and a general description of the management actions granted to each role through the default security policies:

Global role	Access of default policies
Admin	• Start, resume, and stop servers.

Global role	Access of default policies
	• Modify the domain configuration (security providers, users, groups, security roles, security policies, password policies, account locking policies, certificate validation). • View the server configuration, including the encrypted value of some encrypted attributes. • Deploy Enterprise Applications and Web application, EJB, Java EE Connector, and Web Service modules.
AdminChannelUser	• Access the administrative channel.
Deployer	• View the server configuration, including some encrypted attributes related to deployment activities. • Change startup and shutdown classes, Web applications, JDBC data pool connections, EJB, Java EE Connector, Web Service, and WebLogic Tuxedo Connector components. If applicable, edit deployment descriptors. • Access deployment operations in the Java EE Deployment Implementation (JSR-88).
Operator	• Start, resume, and stop servers. • View the server configuration, except for encrypted attributes.
Monitor	• View the server configuration, except for encrypted attributes. • This security role effectively provides read-only access to the WebLogic Server Administration Console, WLST, and MBean APIs.

Table 27: Administration roles

Alternatively, the TOE can enforce access control on security management actions by delegating the authorization of MBeans to authorization providers. If the TOE is configured to do so (security realm attribute *DelegateMBeanAuthorization = True*, security policies assigned to JMX resources are used. When turning on this configuration option, the TOE creates all necessary security policies on JMX resources to start with a baseline similar to the default security policies for MBeans. These policies can then be updated using the JMX Policy Editor in the WebAdmin Console or through WLST commands.

Regardless of whether the TOE delegates authorization of MBeans to authorization providers or keeps the default mechanism, the TOE enforces additional access control rules on security management actions through the Admin and Server resources (see section 7.1.2.2 "Resources" .

The assignment of roles to users is accomplished either directly or via the assignment of groups associated with roles. Additionally, roles are associated with security policies that serve to grant access to applicable resources. See section 7.1.2 for more information.

The TOE provides security management of TSF data using the following data storage:

- An *embedded LDAP server* is used as the default storage when an application server domain is created. The embedded LDAP server is located in each Administration server and managed servers of the domain.

- An *RDBMS security store* is an external RDBMS used as an alternative storage to the embedded LDAP server. It is required by certain domain configurations (e.g. when SAML 2.0 services are provided by more than one TOE instances in the domain). In this case the same RDBMS security store is shared by all TOE instances in the domain
- *External LDAP servers and RDBMS* can be used to access users, groups and group membership information. In this case the same external server is shared by all TOE instances in the domain. Security management can be performed through the TOE interfaces if the associated authentication provider supports the optional user/group management methods (see table 28 for the security providers allowing security management). Otherwise, management must be performed using the native management interfaces or, in some cases, third-party tools.

The TOE provides security management of TSF data for two distinct groups:

- Users, groups and group membership information can be stored in the embedded LDAP server, in one or more external LDAP servers, in one or more external RDBMSs, or in any combination of the three. Where these TSF data are stored depends on the authentication providers configured in the application server domain (i.e. in a multiple authentication scenario, the WebLogic authentication provider accesses the embedded LDAP server but the SQL authentication provider accesses information from an external RDBMS).
- The rest of the TSF data (security roles, security policies, SAML information, credential information, etc.) can be stored either in the embedded LDAP server or in the RDBMS security store mentioned earlier. This is specified when the application server domain is created.

The following table lists the security attributes and TSF data for each type of security provider and the storage supported by each of the security providers allowed in the evaluated configuration.

Security provider type	**TSF data**	**Security provider**	**Storage**	**Managed by the TOE**
Authentication	User and group information.	WebLogic Authentication Provider	Embedded LDAP	yes
		Oracle Internet Directory Authentication Provider	External LDAP	no
		Oracle Virtual Directory Authentication Provider	External LDAP	no
		iPlanet Authentication Provider	External LDAP	no
		Active Directory Authentication Provider	External LDAP	no
		Open LDAP Authentication Provider	External LDAP	no
		Novell LDAP Authentication Provider	External LDAP	no
		SQL Authenticator Provider	External RDBMS	yes

Security provider type	TSF data	Security provider	Storage	Managed by the TOE
		Read-only SQL Authenticator Provider	External RDBMS	no
		SAML Authenticator Provider	None	
Role Mapping	Security roles and association rules to users and groups.	XACML Role Mapping Provider	Embedded LDAP or RDBMS Security Store	yes
Authorization	Security policies and association rules to security roles and resources.	XACML Authorization Provider	Embedded LDAP or RDBMS Security Store	yes
Adjudication	None	WebLogic Adjudication Provider	None	
Auditing	None.	WebLogic Auditing Provider	None	
Credential Mapping	Username-Password credential mapping (WebLogic Credential Mapping Provider). Partner information (SAML 1.0 and 2.0 Credential Mapping providers)	WebLogic Credential Mapping Provider	Embedded LDAP or RDBMS Security Store	yes
		PKI Credential Mapping Provider	Embedded LDAP or RDBMS Security Store	yes
		SAML 1.1 Credential Mapping Provider Version 2	Embedded LDAP or RDBMS Security Store	yes
		SAML 2.0 Credential Mapping Provider	Embedded LDAP or RDBMS Security Store	yes
Identity Assertion	User and X.509 certificate (LDAP X509 Identity Assertion provider). Partner information (SAML 1.0 and 2.0 Identity Assertion providers)	WebLogic Identity Asserter	Embedded LDAP	yes
		LDAP X.509 Identity Asserter	External LDAP	no
		Negotiate Identity Asserter	None	
		SAML Identity Asserter (for SAML 1.1) Version 2	Embedded LDAP or RDBMS Security Store	yes

Security provider type	TSF data	Security provider	Storage	Managed by the TOE
		SAML 2.0 Identity Asserter	Embedded LDAP or RDBMS Security Store	yes
Certificate Registry	Registered end certificates	Certificate Registry	Embedded LDAP or RDBMS Security Store	yes
CertPath	None	WebLogic CertPath Provider	None	
Password Validation	None	Password Validation Provider	None	

Table 28: TSF data and storage by security provider

The TOE ensures that the TSF data remain consistent in the application server domain. When more than one server is defined, domain configuration, and security realm information stored in the embedded LDAP server (if used) are replicated as follows:

- When a Managed server starts, it polls configuration information from the Administration server. The Managed server can also refresh the security realm information stored in the local embedded LDAP server grabbing the information from the embedded LDAP located at the Administration server.
- The Administration server keeps track of the changes in the domain configuration and the security realm that occur in the Administration server. It also maintains a list of the managed servers and their level of update. When a change occurs, the Administration server sends the necessary updates to the managed servers until changes are applied and TSF data is synchronized.
- Communication between the TOE instance servers can be protected through a secure channel provided by the TOE .

The User Data Protection security function effectively enforces restrictions related to administration functions. In particular, access to view or modify TSF data from the TOE is restricted to one or more of the administrative roles identified above. In particular, user definitions (users, credentials, groups and group memberships), role definitions, and security policy settings for audit, identification and authentication, and user data protection are all restricted to one or more of the identified administrator roles. Notice that this functionality is enforced only to the embedded LDAP server: user and group data in an external store may be accessible and/or manageable through the external stores's own interfaces by third party management tools or applications.

Network channels supporting one or more protocols can be created for different communication purposes and can be also protected using one-way or two-way TLS for:

- communication between managed cluster servers;
- communication between the Administration server and the managed servers;
- communication between the TOE and JMX clients (e.g. WLST);
- communication between the TOE and client applications or web browsers.

The TLS protocol and its corresponding cipher suites are implemented by the JCE and JSSE providers, which support TLS v1.0, v1.1 and v1.2.

7.1.5.1 SFR coverage

Security management functionality is designed to satisfy the following security functional requirements:

- FMT_MSA.1
- FMT_MSA.3
- FMT_MTD.1(APP)
- FMT_MTD.1(RACP)
- FMT_SMF.1
- FMT_SMR.2
- FPT_TRC.1
- FPT_ITT.1
- FTP_ITC.1

7.1.6 Cryptographic functionality

The TOE relies on the cryptographic functionality provided by the Java Cryptographic Extension (JCE) and the Java Secure Socket Extension (JSSE) service providers. For operating in FIPS mode, the TOE can use instead the RSA BSAFE® Crypto-J JSAFE and JCE Software Module version 6.1.1. Both modules are FIPS 140-2 validated.

The TOE uses cryptography as follows (for the specific list of algorithms, key sizes and standards, please refer to tables included in section 6.1.2 - Cryptographic support (FCS)):

- Generation and verification of SAML 1.1 and 2.0 identity assertions: the TOE perform XML signature generation and verification according to [SAML11], [SAML20], and [W3CXMLSIG] using the algorithms implemented in the JCE.
- Digest authentication: the WebLogic authentication provider perform this function according to [RFC2617] using the MD5 algorithm implemented in the JCE.
- TLS protocol versions 1.0, 1.1 and 1.2: the TOE uses completely the TLS protocol implemented in the JSSE. Verification of client and server certificates is performed by the TOE using the appropriate security provider.
- XML signature and encryption for Web Service Security: the TOE uses XML signature generation and verification according to [W3CXMLSIG], and [W3CXMLENC].
- Validation of X.509 certificates: the TOE relies on the RSA signature verification algorithm implemented in the JCE.
- Verification of SPNEGO tokens through the GSS-API protocol.

The TOE functionality does not include certificate management, asymmetric key generation and keystore management, these tasks must be performed using a tool compatible with the KS keystore (i.e. the Keytool utility included in the JDK).

7.1.6.1 SFR coverage

Cryptographic functionality is designed to satisfy the following security functional requirements:

- FCS_CKM.1-JCE
- FCS_CKM.2-JCE

- FCS_COP.1-JCE(TLS)
- FCS_COP.1-JCE(XMLSIG)
- FCS_COP.1-JCE(XMLENC)
- FCS_COP.1-JCE(CLV)
- FCS_COP.1-JCE(SPNEGO)

8 Abbreviations, Terminology and References

8.1 Abbreviations

CLV
Certificate Lookup and Validation

COM
Component Object Model

EIS
Enterprise Information System

EJB
Enterprise JavaBean

JAAS
Java Authentication and Authorization Service

JACC
Java Authorization Contract for Containers

JCE
Java Cryptography Extension

JDBC
Java Database Connectivity

JMS
Java Messaging Service

JMX
Java Management Extensions

JNDI
Java Naming and Directory Interface

JSSE
Java Secure Socket Extension

JWS
Java Web Service

OAEP
Optimal Asymmetric Encryption Padding

PSU
Patch Set Update

SAML
Security Assertion Markup Language

SNMP
Simple Network Management Protocol

SPNEGO
Simple and Protected GSS-API Negotiation Mechanism

SSO
Single Sign-On

URL
Universal Resource Locator

XACML
eXtensible Access Control Markup Language

8.2 Terminology

This section contains definitions of technical terms that are used with a meaning specific to this document. Terms defined in the [CC] are not reiterated here, unless stated otherwise.

Identity Assertion
Special type of authentication whereby a client's identity is established through the use of client-supplied tokens that are generated from an outside source. Identity is asserted when these tokens are mapped to usernames.

Security realm
A scoping mechanism to define a set of configured security providers, users, groups, roles, and security policies. Multiple security realms can exist in a domain; however, only one can be the default (active) security realm.

User
Humans or machines interacting with the TOE via the provided user and programmatic interfaces. The term user in this document includes administrators of the TOE unless a specific distinction is made in the text.

8.3 References

ADAPT	**Developing Resource Adapters for Oracle WebLogic Server**	
	Version	E41877-02
	Date	August 2015
	Location	https://docs.oracle.com/middleware/1213/wls/ADAPT.pdf
ADMRF	**Command Reference for Oracle WebLogic Server**	
	Version	E42026-03
	Date	August 2015
	Location	https://docs.oracle.com/middleware/1213/wls/ADMRF.pdf
ASCON	**Understanding Oracle Fusion Middleware Concepts**	
	Version	E48202-01
	Date	May 2014
	Location	https://docs.oracle.com/middleware/1213/core/ASCON.pdf
ASINS	**Planning an Installation of Oracle Fusion Middleware**	
	Version	E48353-01
	Date	May 2014
	Location	https://docs.oracle.com/middleware/1213/core/ASINS.pdf

CC **Common Criteria for Information Technology Security Evaluation**
Version 3.1R4
Date September 2012
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R4.pdf
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R4.pdf
Location http://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R4.pdf

CCGUIDE **Guidance Supplement for Oracle® Weblogic Server 12.1.3**
Version 1.3
Date December 2016

CLUST **Administering Clusters for Oracle WebLogic Server**
Version E41944-06
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/CLUST.pdf

CNFGD **Administering Server Environments for Oracle WebLogic Server**
Version E41942-07
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/CNFGD.pdf

DEPGD **Deploying Applications to Oracle WebLogic Server**
Version E41940-03
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/DEPGD.pdf

DOMCF **Understanding Domain Configuration for Oracle WebLogic Server**
Version E41943-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/DOMCF.pdf

EJBAD **Developing Enterprise JavaBeans for Oracle WebLogic Server**
Version E47839-05
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/EJBAD.pdf

EJBPG **Developing Enterprise JavaBeans, Version 2.1, for Oracle WebLogic Server**
Version E47840-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/EJBPG.pdf

FIPS180-4 **SECURE HASH STANDARD (SHS)**
Version FIPS 180-4
Date March 2012
Location http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

FIPS197 **Specification for the ADVANCED ENCRYPTION STANDARD (AES)**
Version FIPS PUB 197
Date November 26, 2001
Location http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

INTRO **Understanding Oracle WebLogic Server**
Version E41937-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/INTRO.pdf

JDBCA **Administering JDBC Data Sources for Oracle WebLogic Server**
Version E41864-09
Date May 2016
Location https://docs.oracle.com/middleware/1213/wls/JDBCA.pdf

JDBCP **Developing JDBC Applications for Oracle WebLogic Server**
Version E41865-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/JDBCP.pdf

JMSAD **Administering JMS Resources for Oracle WebLogic Server**
Version E41859-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/JMSAD.pdf

JMSRA **Administering the JMS Resource Adapter for Oracle WebLogic Server**
Version E41853-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/JMSRA.pdf

LOCKD **Securing a Production Environment for Oracle WebLogic Server**
Version E41900-06
Date March 2016
Location https://docs.oracle.com/middleware/1213/wls/LOCKD.pdf

NODEM **Administering Node Manager for Oracle WebLogic Server**
Version E41941-05
Date May 2016
Location https://docs.oracle.com/middleware/1213/wls/NODEM.pdf

OASIS-WSS **OASIS Web Services Security (WSS)**
Date February 2006
Location https://www.oasis-open.org/standards#wssv1.1

OASIS-WSSP **OASIS WS-Security Policy 1.2**
Date July 2007
Location http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-os.pdf

OUIRF **Installing Software with the Oracle Universal Installer**
Version E48351-01
Date May 2014
Location https://docs.oracle.com/middleware/1213/core/OUIRF.pdf

OWLS_PD **Oracle Fusion Middleware 12c 12.1.3 - Oracle WebLogic Server - Books**
Date received 2014-12-15
Location http://docs.oracle.com/middleware/1213/wls/docs.htm

RESTF **Developing and Securing RESTful Web Services for Oracle WebLogic Server**
Version E47709-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/RESTF.pdf

RFC2104 **HMAC: Keyed-Hashing for Message Authentication**
Author(s) H. Krawczyk, M. Bellare, R. Canetti
Date 1997-02-01
Location http://www.ietf.org/rfc/rfc2104.txt

RFC2459 **Internet X.509 Public Key Infrastructure Certificate and CRL Profile**
Author(s) R. Housley, W. Ford, W. Polk, D. Solo
Date 1999-01-01
Location http://www.ietf.org/rfc/rfc2459.txt

RFC2560 **X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP**
Author(s) M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams
Date 1999-06-01
Location http://www.ietf.org/rfc/rfc2560.txt

RFC2617 **HTTP Authentication: Basic and Digest Access Authentication**
Author(s) J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart
Date 1999-06-01
Location http://www.ietf.org/rfc/rfc2617.txt

RFC3447 **Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1**
Author(s) J. Jonsson, B. Kaliski
Date 2003-02-01
Location http://www.ietf.org/rfc/rfc3447.txt

RFC3962 **Advanced Encryption Standard (AES) Encryption for Kerberos 5**
Author(s) K. Raeburn
Date 2005-02-01
Location http://www.ietf.org/rfc/rfc3962.txt

RFC5246 **The Transport Layer Security (TLS) Protocol Version 1.2**
Author(s) T. Dierks, E. Rescorla
Date 2008-08-01
Location http://www.ietf.org/rfc/rfc5246.txt

RFC5280 **Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile**
Author(s) D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk
Date 2008-05-01
Location http://www.ietf.org/rfc/rfc5280.txt

ROLES **Securing Resources Using Roles and Policies for Oracle WebLogic Server**
Version E41904-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/ROLES.pdf

SAML11 **Security Assertion Markup Language (SAML) v1.1**
Date September 2003
Location https://www.oasis-open.org/standards#samlv1.1

SAML11Core **Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1**
Date September 2003
Location http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf

SAML20 **Security Assertion Markup Language (SAML) v2.0**
Date March 2005
Location https://www.oasis-open.org/standards#samlv2.0

SAML20Core **Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0**
Date March 2005
Location http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf

SCOVR **Understanding Security for Oracle WebLogic Server**
Version E42028-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/SCOVR.pdf

SCPRG **Developing Applications with the WebLogic Security Service**
Version E42029-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/SCPRG.pdf

SECMG **Administering Security for Oracle WebLogic Server**
Version E41905-08
Date April 2016
Location https://docs.oracle.com/middleware/1213/wls/SECMG.pdf

SP800-38A **Recommendation for Block Cipher Modes of Operation: Methods and Techniques**
Version NIST Special Publication 800-38A 2001 Edition
Date December 2001
Location http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

SP800-67 **Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
Version Revision 1
Date January 2012
Location http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf

START **Administering Server Startup and Shutdown for Oracle WebLogic Server**
Version E41938-05
Date May 2016
Location https://docs.oracle.com/middleware/1213/wls/START.pdf

W3CXMLENC **XML Encryption Syntax and Processing**
Date December 10, 2002
Location http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

W3CXMLSIG **XML Signature Syntax and Processing (Second Edition)**
Date 10 June 2008
Location http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/

WBAPP **Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server**
Version E41936-07
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WBAPP.pdf

WLACH **WebLogic Server Administration Console Online Help**
Version 12.1.3
Date received May 2015
Location Included in WebLogic Server Administration Console

WLDCW **Creating WebLogic Domains Using the Configuration Wizard**
Version E41890-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WLDCW.pdf

WLDTR **Domain Template Reference**
Version E41892-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WLDTR.pdf

WLMBR **MBean Reference for Oracle WebLogic Server**
Version E41843-02
Date February 2015
Location http://docs.oracle.com/middleware/1213/wls/WLMBR/toc.htm

WLSIG **Installing and Configuring Oracle WebLogic Server and Coherence**
Version E48355-02
Date July 2014
Location https://docs.oracle.com/middleware/1213/core/WLSIG.pdf

WLSRN **Release Notes for Oracle WebLogic Server**
Version E41931-13
Date April 2016
Location https://docs.oracle.com/middleware/1213/wls/WLSRN.pdf

WLSTC **WLST Command Reference for WebLogic Server**
Version E35669-03
Date February 2016
Location https://docs.oracle.com/middleware/1213/wls/WLSTC.pdf

WSGET **Developing JAX-WS Web Services for Oracle WebLogic Server**
Version E47706-04
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WSGET.pdf

WSRPC **Developing JAX-RPC Web Services for Oracle WebLogic Server**
Version E47707-03
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WSRPC.pdf

WSS11-SAML **OASIS Web Services Security SAML Token Profile 1.1**
Date February 2006
Location https://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf

WSS11-SOAP **OASIS Web Services Security SOAP Message Security 1.1 (WS-Security 2004)**
Date February 2006
Location https://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

WSS11-UTP **OASIS Web Services Security Username Token Profile 1.1**
Date February 2006
Location https://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf

WSS11-X509 **OASIS Web Services Security X.509 Certificate Token Profile 1.1**
Date February 2006
Location https://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf

WSSOV **Securing WebLogic Web Services for Oracle WebLogic Server**
Version E42030-02
Date August 2015
Location https://docs.oracle.com/middleware/1213/wls/WSSOV.pdf