BEA WebLogic Integration
Security Target

Version 1.0
9/13/2007

Prepared for:
BEA Systems, Inc.

2315 North First Street
San Jose, CA 95131

Prepared By:
Science Applications International Corporation

Common Criteria Testing Laboratory

7125 Columbia Gateway Drive, Suite 300
Columbia, MD 21046

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is WebLogic Integration (version 8.1 SP6 with BEA07-169.00 security advisory patch) running in either a BEA JRockit 1.4.2 or Sun Java 2 1.4.2 environment, provided by BEA Systems, Inc., which is designed to offer security services to protect and be used by (primarily network) applications built in the environment provided by the TOE.

The Security Target contains the following additional sections:

- TOE Description (Section 2)
- Security Environment (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8)

1.1 Security Target, TOE and CC Identification

ST Title – BEA WebLogic Integration Security Target

ST Version – Version 1.0

ST Date – 9/13/2007

TOE Identification – BEA WebLogic Integration, version 8.1 SP6 with BEA07-169.00 security advisory patch (hereafter referred to as WLI)

CC Identification – Common Criteria for Information Technology Security Evaluation, Version 2.2, Revision 256, January 2004.

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Requirements, Version 2.2, Revision 256, January 2004.
    - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Requirements, Version 2.2, Revision 256, January 2004.
    - Part 3 Conformant
    - EAL 2 augmented with ALC_FLR.1

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a letter placed at the end of the component. For example FDP_ACC.1a and FDP_ACC.1b indicate that the ST includes two iterations of the FDP_ACC.1 requirement, a and b.
    - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]).
    - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …").
- Explicit security functional requirements are identified using '_EX' as a suffix to the label of the requirement.
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

2. TOE Description

The Target of Evaluation (TOE) is BEA WebLogic Integration, version 8.1 SP6 with BEA07-169.00 security advisory patch.

The BEA WebLogic Integration (WLI) is an application server that provides a foundation for an enterprise to build and integrate applications and databases. The WLI TOE consists of a WebLogic Integration subsystem and also a single supporting BEA WebLogic Server (WLS) subsystem with the security providers identified in section 2.2.2, below.

2.1 TOE Overview

The BEA WebLogic Integration TOE is an application server platform for building, extending, integrating, deploying, and managing software applications. The TOE consists of the following subsystems that are used in combination to support an end-user developed application:

- WebLogic Server
- WebLogic Integration

WebLogic Server delivers an application infrastructure for building and integrating distributed multi-tier applications. WebLogic Server centralizes application services, such as Web server functionality, business components, and access to back-end enterprise systems. It is based on standards such as J2EE, Web services, and XML, and it provides standards-based integration to enable application integration and investment protection. WebLogic Server includes the WebLogic Workshop® IDE for application development[1], and also provides enterprise-level security and administration facilities. WebLogic Server provides the foundation for WebLogic Platform™. The WebLogic Portal® and WebLogic Integration™ components, and all applications built with these components, utilize the WebLogic Server run-time environment to meet the demands of applications that span one or more enterprises.

[1] WebLogic Workshop is an integrated development environment (IDE) for building enterprise-class J2EE applications on the WebLogic Server. WebLogic Workshop provides a programming model that enables an applications developer to focus on building the business logic of the application rather than on its otherwise complex implementation details. WebLogic Workshop is available in the development environment and not during the run-time operation of the product and as such has been excluded from the scope of this evaluation.

WebLogic Integration is a product built on WebLogic Server that provides the functionality for integrating business systems within an enterprise. WebLogic Integration provides a development and run-time framework that unifies the components of business integration--business process management, data transformation, trading partner integration, connectivity, message brokering, application monitoring, and user interaction--into a single environment.

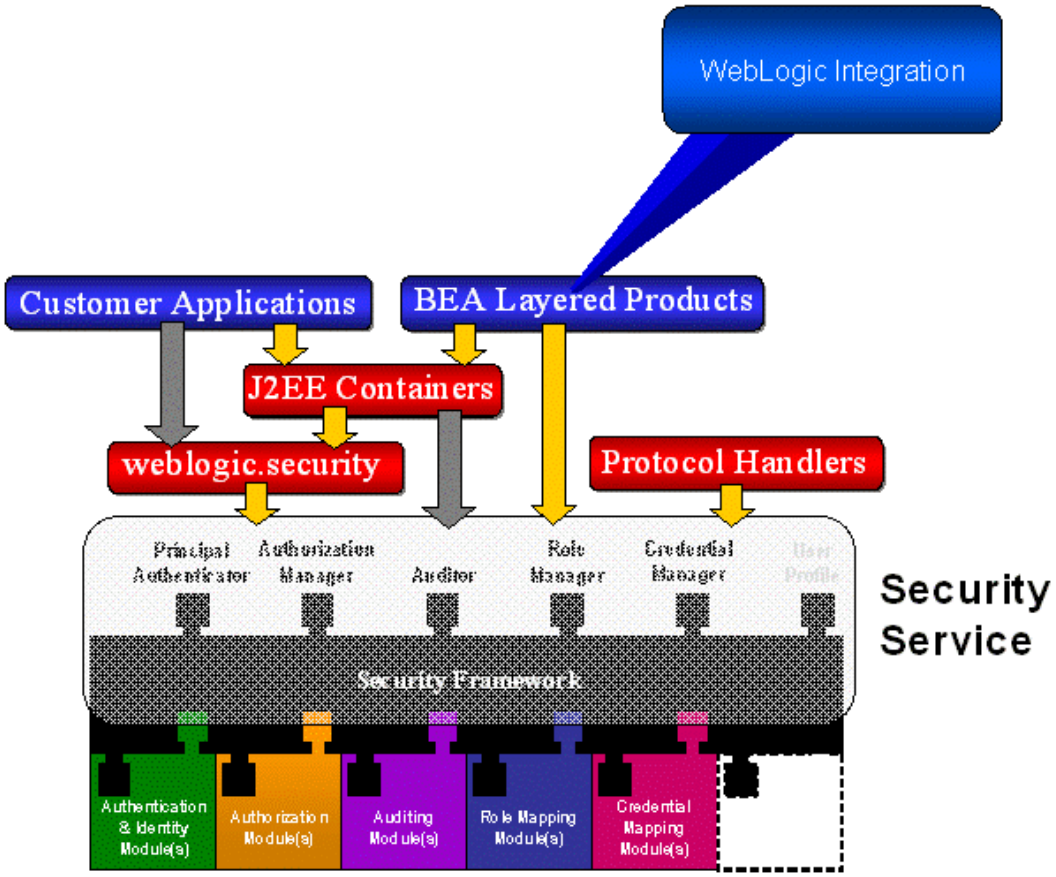
As indicated previously, the scope of this evaluation is the WebLogic Server subsystem, which provides security services for the WebLogic Integration and application programs and the WebLogic Integration subsystem which offers digital encryption for signing and sealing data among Servers and users.

2.2 TOE Security Architecture

As indicated above, WLI consists of two distinct subsystems. The figure below shows a 'Security Service' which includes the basic ‘Security Framework’ of the WebLogic Server and a series of security service provider ‘modules’ (note that the security provider modules in the figure are only examples). The Security Service and the associated modules, identified in section 2.2.2, form the core of the TOE; while the other entities in the figure depicted above the Security Service are examples of applications supported by the TOE. Note that WebLogic Integration is a 'BEA Layered Product' and represents the remainder of the TOE.

Generally, user requests will come in from the network and will be handled by the security framework provided by WebLogic Server. If the user is attempting to access an application associated with the WebLogic Integration subsystem, it will be invoked in addition to the WebLogic Server security framework and hence serves to extend or add security features relative to resources within its control.

Customer applications are acquired and installed by WLI administrators so that the appropriate controls are configured and subsequently enforced before the applications can be accessed.



Notice in the figure above that WebLogic Integration serves as a layered product adding its own security features to those of the underlying WebLogic Server. Note also that while the Security Framework supports plug-in providers not identified as part of the TOE (in section 2.2.2), no additional providers are included within the scope of this evaluation and therefore are not part of the TOE.

2.2.1 Physical Boundaries

The BEA WebLogic Server subsystem (including the WebLogic Server security framework and associated security providers) has a fully J2EE-compliant tiered architecture, and support for tool sets facilitate the separation of presentation, business logic, and data, providing the underlying core functionality necessary for the development and deployment of business-driven applications. Its capabilities support an integrated infrastructure that can connect legacy systems, as well as the Web Services. This subsystem is always invoked when a network resource request is received.

The BEA WebLogic Integration subsystem is a special application operating in the context of WebLogic Server that offers the capability to digitally sign and seal traffic among cooperating WebLogic Integration applications and users. This subsystem is invoked after the WebLogic Server security subsystem for certain types of WebLogic Integration applications.

Note that all of the WebLogic subsystems are Java applications designed to run in a Java 2 (BEA JRockit® 1.4.2_10 SDK or Sun Java 2 SDK 1.4.2_11 with Java HotSpot™ Client VM) environment provided by the hosting operating system. As such, there is reliance on the environment for general operation and protection as well as specific features such as secure data storage and time information.

In addition to the hosting operating system, the BEA WebLogic Integration subsystem depends on the availability of a relational database management system (RDBMS) to store and protect management data associated with trading partners. The following databases are fully supported by the TOE: IBM DB2, Oracle9i RAC, Oracle 10g R1, Oracle 10g R2, Oracle 10g R1 RAC, Oracle 10g R2 RAC, Microsoft SQL Server 2000, Microsoft SQL Server 2005, and Sybase 12.5.03.

2.2.2 Logical Boundaries

The WebLogic Server security framework supports a number of plug-in security providers. Each of the security function summaries below identify (in **bold**) the specific security providers that are included with, and enabled through, the security framework in the evaluated configuration of the TOE. Note that these security providers are default providers developed by BEA and distributed with the product.

2.2.2.1 Security audit

The WebLogic Server security framework audits security relevant events as they occur within the security framework and stores them for later review. The **WebLogic Auditing Provider** supplies these services.

2.2.2.2 Cryptographic support

The WLI ensures that business-to-business traffic supported by the WebLogic Integration subsystem is protected using cryptographic means to digitally sign and encapsulate the traffic while on the network media. Note that the cryptographic mechanisms, while invoked by the TOE, are implemented within the IT environment.

2.2.2.3 User data protection

The following security providers implement access control functionality: **WebLogic Authorization Provider**, **WebLogic Role Mapping Provider, and WebLogic Adjudication Provider**.

Authorization

Authorization is the process whereby the interactions between users and WebLogic resources are limited to ensure appropriate protection of data. In other words, authorization is responsible for controlling access to WebLogic resources based on user identity or other information. The WebLogic Authorization provider supplies these services.

Role Mapping

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 answer the 'is access allowed?' question 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[2], 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 processing each Authorization provider's answer and returning a final decision.

2.2.2.4 Identification and authentication

The following security providers implement identification and authentication functionality: **WebLogic Authentication Provider**, **WebLogic Identity Assertion Provider**, and **WebLogic Credential Mapping Provider**.

[2] A realm is a set of configured security providers, users, groups, security roles, and security policies. Multiple realms can be defined, but only one realm can (and must) be active at any time. The default realm (myrealm) is active when the TOE is installed.

Authentication

Authentication is the process whereby the identity of users or system processes is proved or verified. Authentication also involves making identity information available to various components of a system when that information is needed. The WebLogic Security Service supports Username and password authentication. The WebLogic Authentication provider supplies these services.

Identity Assertion

An Authentication provider that performs perimeter authentication—a special type of authentication using tokens—is called an Identity Assertion provider. Identity assertion involves establishing 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 principals. The WebLogic Identity Assertion provider supplies these services.

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. The WebLogic Credential Mapping provider supplies this service.

2.2.2.5 Security management

The WLI supports a number of roles relevant to one or more of its subsystems, though in the case of this Security Target all of the security relevant roles are considered to be an 'administrator' regardless of any apparent limitations. The WLI uses the WLS (LDAP) database to store data used by the various security providers. In the evaluated configuration, an embedded LDAP server is used for the security provider database, and WLI is designed to ensure that only a user acting in an appropriate role can modify or review WLI configuration data.

2.2.2.6 Protection of the TSF

The WLI encapsulates the applications it protects within the WebLogic Server security framework (and using Integration extensions) to ensure that the security mechanisms are always invoked when resources are requested. WLI operates as a collection of Java applications that operate in their own domains distinct from one another and also from other potentially untrusted entities.

2.3 TOE Documentation

BEA has administration and user guidance documents to help ensure that the evaluated WLI product can be operated securely. These and other documents are further summarized in section 6.2.

3. Security Environment

This section summarizes the threats addressed by the TOE and/or its supporting IT environment (see section 8.1.1 for more information about the association of threats with the TOE and its environment) and assumptions about the intended environment of the TOE. Note that while the identified threats are mitigated by security functions implemented in the TOE and/or its supporting IT environment, the overall assurance level (EAL 2 augmented with ALC_FLR.1) also serves as an indicator of whether the TOE would be suitable for a given environment.

3.1 Threats

T.BYPASS	An attacker may be able to bypass TOE protection mechanisms through unprotected interfaces in order to inappropriately access protected data and services.
T.EXCESS_AUTHORITY	An unauthorized user may be able to exercise administrator authorities to inappropriately manage the TOE.
T.NO_TIME	Those responsible for the TOE may not be able to determine the sequence of audited security relevant events.
T.NOCRYPTO	An attacker may be able to observe authentication data transmitted in the clear due to cryptographic services not being available.
T.STORAGE	An attacker may be able to cause the loss or destruction of Audit and other TSF data.
T.TAMPER	An attacker may be able to inappropriately modify or otherwise tamper with TSF programs and data.
T.TSF_COMPROMISE	A user or process may cause TSF data or executable code to be inappropriately accessed (viewed, modified, or deleted).
T.UNACCOUNTABLE	Users of the TOE may not be held accountable for their security-relevant actions.
T.UNAUTHORIZED_ACCESS	A user may gain access to user data for which they are not authorized according to the TOE security policies.
T.UNDETECTED_ACTIONS	The administrator may not have the ability to detect potential security violations, thus limiting the administrator's ability to identify and take action against a possible security breach.
T.UNIDENTIFIED_USERS	An attacker may gain access to the TOE without being reliably identified allowing them to gain unauthorized access to data or TOE resources.

3.2 Assumptions

A.NO_EVIL	Administrators are non-hostile, appropriately trained, and follow all administrator guidance.
A.NO_UNTRUSTED	There are no untrusted user accounts or malicious software on the server platform.

A.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the IT environment.

4. Security Objectives

This section summarizes the security objectives for the TOE and its IT and non-IT environment.

4.1 Security Objectives for the TOE

O.AUDIT_GENERATION	The TOE will provide the capability to detect and create records of security relevant events associated with users.
O.CRYPTOGRAPHY	The TOE shall provide the capability invoke cryptographic services for digital encapsulation of critical user data.
O.ID_AND_AUTH	The TOE will provide identification and authentication mechanisms that control logical access to the TOE.
O.MANAGE	The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.
O.MEDIATE	The TOE will protect user data in accordance with its security policies.
O.ROLES	The TOE will support administrator roles that are differentiated from users not allowed to perform administrative operations.
O.SELF_PROTECTION	The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure.

4.2 Security Objectives for the IT Environment

OE.CRYPTOGRAPHY	The IT environment shall provide cryptographic services for encryption, authentication, and key management services for key generation and key destruction.
OE.JAVA	The Java 2 Security Sandbox will provide for separate domains for security providers and application code within the JVM.
OE.OS	The underlying operating system will protect TSF code and data structures from unauthorized modification and prevent TSF security functions from being bypassed through the OS interfaces. The operating system will provide protected files for the storage of audit records and also tools for review of the audit records. The operating system platform will provide reliable time stamps.

4.3 Security Objectives for the Environment

ON.NO_EVIL	Sites using the TOE shall ensure that administrators are non-hostile, appropriately trained and follow all administrator guidance.
ON.NO_UNTRUSTED	Those responsible for the TOE will ensure that there are no untrusted user accounts or potentially malicious software on the server platform.
ON.PHYSICAL	Physical security will be provided within the domain for the value of the IT assets protected by the operating system and the value of the stored, processed, and transmitted information.

5. IT Security Requirements

This section defines the security functional requirements for the TOE and its IT environment as well as the security assurance requirements against which the TOE has been evaluated. All of the requirements have been copied from version 2.2 of the applicable Common Criteria documents.

5.1 TOE Security Functional Requirements

The following table describes the SFRs that are candidates to be satisfied by WLI.

Requirement Class	Requirement Component
FAU: Security audit	FAU_GEN.1: Audit data generation
	FAU_GEN.2: User identity association
FCS: Cryptographic support	FCS_COP_EX.1: Cryptographic operation
FDP: User data protection	FDP_ACC.1: Subset access control
	FDP_ACF.1: Security attribute based access control
FIA: Identification and authentication	FIA_AFL.1: Authentication failure handling
	FIA_ATD.1: User attribute definition
	FIA_UAU.1: Timing of authentication
	FIA_UAU.5: Multiple authentication mechanisms
	FIA_UID.1: Timing of identification
	FIA_USB.1: User-subject binding
FMT: Security management	FMT_MOF.1: Management of security functions behaviour
	FMT_MSA.1: Management of security attributes
	FMT_MSA.3: Static attribute initialization
	FMT_SMF.1: Specification of Management Functions
	FMT_SMR.1: Security roles
FPT: Protection of the TSF	FPT_RVM.1a: Non-bypassability of the TSP
	FPT_SEP.1a: TSF domain separation

Table 1 TOE Security Functional Components

5.1.1 Security audit (FAU)

5.1.1.1 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 following auditable events: LogonAuditEvent, AccountLockout, AccountLockoutExpiration, IdentityAssertAuditEvent, and AuthorizationAuditEvent]**.

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, 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, **[no additional information]**.

5.1.1.2 User identity association (FAU_GEN.2)

FAU_GEN.2.1 The TSF shall be able to associate each auditable event with the identity of the user that caused the event.

5.1.2 Cryptographic support (FCS)

5.1.2.1 Cryptographic operation (FCS_COP_EX.1)

FCS_COP_EX.1.1 The TSF shall perform digital encapsulation of business-to-business traffic by invoking the DSA (for signing) and Triple-DES (for encryption) cryptographic algorithms supported by the IT environment, using 1024 bit and 168 bit key sizes, respectively.

5.1.3 User data protection (FDP)

5.1.3.1 Subset access control (FDP_ACC.1)

FDP_ACC.1.1 The TSF shall enforce the **[WebLogic Server Access Control SFP]** on **[**

1) Subjects: Threads of control executing on behalf of a caller;

2) Objects:

WebLogic Server Resources (WebLogic Server controls access to the following types of resources:

a) Administrative: Administrative console, weblogic.Admin, and/or MBean APIs,

b) Application: Enterprise JavaBeans,

c) Component Object Model (COM): Classes to be accessed by the COM client application,

d) Enterprise Information System (EIS): Resources that are designed as connectors,

e) Enterprise JavaBean (EJB): EJB JARs, individual EJBs within an EJB JAR, or individual methods on an EJB,

f) Java Database Connectivity (JDBC): Resources that are related to JDBC;

g) Java Message Service (JMS): Resources that are related to JMS,

h) Java Naming and Directory Interface (JNDI): Resources that use the industry-standard JNDI API to enable connectivity,

i) Server: WebLogic Server instances - the allowed operations are Start, Shutdown, Lock, and Unlock,

j) Universal Resource Locator (URL): Resources that are related to Web applications - can be WAR (Web Application aRchive) file or individual components of a Web application (such as servlets and JSPs), and

k) Web Service: Resources that are related to services - can be entire Web Service or individual components of a Web Service; and,

WebLogic Integration Resources: Message Broker Channels, Business Processes, Application Views, Trading Partner Profiles, Trading Partner Services, Service Profiles, Worklists

3) Operations: Access].

5.1.3.2 Security attribute based access control (FDP_ACF.1)

FDP_ACF.1.1 The TSF shall enforce the **[WebLogic Server Access Control SFP]** to objects based on the following: **[**

1) Subject attributes: Username, Group Membership, and Roles;

2) Object attributes (both WebLogic Server and WebLogic Integration Resources): Type of Resource, Resource identity, Security policy; and,

3) Other attributes: Time of Day and Resource default security policy].

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

1) Roles that are applicable to a subject are computed dynamically at the time of call based on username, group membership, and time of day;

2) The Access Decision component of the WebLogic Authorization Provider returns PERMIT, DENY, or ABSTAIN based on the subject's applicable username, group

membership, and roles, the resource's security policy (or the default security policy for the resource type if there is no explicit security policy for the identified resource) and the time of day; and,
3) If multiple authorization providers are configured, the adjudication provider processes the multiple Access Decisions and determines the final PERMIT or DENY decision according to the following rule:
A) If the Require Unanimous Permit attribute is set to TRUE, the WebLogic Adjudication provider acts as follows:
1) If all the Authorization providers' Access Decisions return PERMIT, then return a final verdict of TRUE (that is, permit access to the WebLogic resource);
2) Else return FALSE (that is, deny access to the WebLogic resource since at least one authorization provider has returned ABSTAIN or DENY); or
B) If the Require Unanimous Permit attribute is set to FALSE, the WebLogic Adjudication provider treats ABSTAIN verdicts as PERMIT and acts as follows:
1) If any of the Authorization providers' Access Decisions return DENY, then return a final verdict of FALSE (that is, deny access to the WebLogic resource);
2) Else return TRUE (that is, permit access to the WebLogic resource since all authorization providers have returned PERMIT or ABSTAIN)].

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 **[none]**.

5.1.4 Identification and authentication (FIA)

5.1.4.1 Authentication failure handling (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when **[*an administrator configurable positive integer within [greater than or equal to 1]*]** unsuccessful authentication attempts occur related to **[password authentication].**

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **[lock the user's account]**.

5.1.4.2 User attribute definition (FIA_ATD.1)

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: **[Username, Password, Group membership, and Roles]**.

5.1.4.3 Timing of authentication (FIA_UAU.1)

FIA_UAU.1.1 The TSF shall allow **[operations on application services or data explicitly allowed by the administrator]** 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.

5.1.4.4 Multiple authentication mechanisms (FIA_UAU.5)

FIA_UAU.5.1 The TSF shall provide **[the following authentication mechanisms: Password-based authentication by the WebLogic Server Authentication Provider, Token-based authentication by the WebLogic Server Identity Assertion Provider, and Credential mapping to support authentication by legacy systems]** to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the **[following rules:**
1. In the evaluated configuration, users may be authenticated by either the WebLogic Server Authentication Provider, the WebLogic Server Identity Assertion Provider, or both;

2. The WebLogic Server Identity Assertion Provider supports two types of tokens in the evaluated configuration: a) X.509 certificates and b) CORBA Common Secure Interoperability version 2 (CSIv2) identity assertion;
3. If more than one authentication and/or identity assertion provider is configured in a security realm, they can be individually configured as being optional or mandatory for each resources; and
4. An already authenticated user may use the Credential Mapper for obtaining credentials for authentication to legacy applications].

5.1.4.5 Timing of identification (FIA_UID.1)

FIA_UID.1.1 The TSF shall allow **[operations on application services or data explicitly allowed by the administrator]** 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.

5.1.4.6 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: **[user identity, group identities, and roles]**. *(per International Interpretation #137)*

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: **[the user identity, groups, and roles will be assigned to a subjected created to act on behalf of an authenticated user based on the defined user attributes associated with that user]**. *(per International Interpretation #137)*

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: **[user security attributes do not change after being assigned to a newly created subject]**. *(per International Interpretation #137)*

5.1.5 Security management (FMT)

5.1.5.1 Management of security functions behaviour (FMT_MOF.1)

FMT_MOF.1.1 The TSF shall restrict the ability to **[*determine the behaviour of* and *modify the behaviour of*]** the functions **[of the TOE]** to **[the Administrator]**.

5.1.5.2 Management of security attributes (FMT_MSA.1)

FMT_MSA.1.1 The TSF shall enforce the **[Access Control SFP]** to restrict the ability to **[*modify*]** the security attributes **[User name, Password, Groups and Group Membership, Roles, and Security policies]** to **[the Administrator]**.

5.1.5.3 Static attribute initialization (FMT_MSA.3)

FMT_MSA.3.1 The TSF shall enforce the **[Access Control SFP]** to provide **[*[explicitly defined]*]** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **[the Administrator]** to specify alternative initial values to override the default values when an object or information is created.

5.1.5.4 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following security management functions: **[view the WebLogic Integration configuration; create and delete user accounts and modify user security attributes; assign and revoke security roles; and manage the Access Control SFP]**.

5.1.5.5 Security roles (FMT_SMR.1)

FMT_SMR.1.1 The TSF shall maintain the roles **[Administrator]**.

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

5.1.6 Protection of the TSF (FPT)

5.1.6.1 Non-bypassability of the TSP (FPT_RVM.1a)

FPT_RVM.1a.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.1.6.2 TSF domain separation (FPT_SEP.1a)

FPT_SEP.1a.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1a.2 The TSF shall enforce separation between the security domains of subjects in the TSC.

5.2 IT Environment Security Functional Requirements

The following table describes the SFRs that are candidates to be satisfied by the IT environment of WLI.

Requirement Class	Requirement Component
FAU: Security audit	FAU_SAR.1: Audit review
	FAU_STG.1: Protected audit trail storage
FCS: Cryptographic support	FCS_CKM.1a: Cryptographic key generation
	FCS_CKM.1b: Cryptographic key generation
	FCS_CKM.4: Cryptographic key destruction
	FCS_COP.1a: Cryptographic operation
	FCS_COP.1b: Cryptographic operation
	FCS_COP.1c: Cryptographic operation
FMT: Security management	FMT_MSA.2: Secure security attributes
FPT: Protection of the TSF	FPT_RVM.1b: Non-bypassability of the TSP
	FPT_SEP.1b: TSF domain separation
	FPT_STM.1: Reliable time stamps

Table 2 IT Environment Security Functional Components

5.2.1 Security audit (FAU)

5.2.1.1 Audit review (FAU_SAR.1)

FAU_SAR.1.1 The ~~TSF~~ **IT Environment** shall provide **[authorized administrators]** with the capability to read **[all audit information]** from the audit records.

FAU_SAR.1.2 The ~~TSF~~ **IT Environment** shall provide the audit records in a manner suitable for the user to interpret the information.

5.2.1.2 Protected audit trail storage (FAU_STG.1)

FAU_STG.1.1 The ~~TSF~~ **IT Environment** shall protect the stored audit records from unauthorised deletion.

FAU_STG.1.2 The ~~TSF~~ **IT Environment** shall be able to **[*prevent*]** unauthorised modifications to the audit records in the audit trail.

5.2.2 Cryptographic support (FCS)

5.2.2.1 Cryptographic key generation (FCS_CKM.1a)

FCS_CKM.1a.1 The ~~TSF~~ **IT Environment** shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[pseudo-random number generation]** and specified cryptographic key sizes **[40-168 bits]** that meet the following: **[FIPS 140-2]**.

5.2.2.2 Cryptographic key generation (FCS_CKM.1b)

FCS_CKM.1b.1 The ~~TSF~~ **IT Environment** shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **[pseudo-random number generation]** and specified cryptographic key sizes **[512-2048 bits]** that meet the following: **[FIPS 140-2]**.

5.2.2.3 Cryptographic key destruction (FCS_CKM.4)

FCS_CKM.4.1 The ~~TSF~~ **IT Environment** shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method **[zeroization]** that meets the following: **[FIPS 140-2]**.

5.2.2.4 Cryptographic operation (FCS_COP.1a)

FCS_COP.1a.1 The ~~TSF~~ **IT Environment** shall perform **[symmetric key encryption and decryption]** in accordance with a specified cryptographic algorithm **[DES (CBC mode); Triple-DES (EDE CBC mode); and RC4]** and cryptographic key sizes **[40 and 56 bits; 112 and 168 bits; and 40, 56 and 128 bits, respectively]** that meet the following: **[FIPS 140-2]**.

5.2.2.5 Cryptographic operation (FCS_COP.1b)

FCS_COP.1b.1 The ~~TSF~~ **IT Environment** shall perform **[authentication with digital signature and verification]** in accordance with a specified cryptographic algorithm **[DSA]** and cryptographic key sizes **[512, 768, 1024 and 2048 bits]** that meet the following: **[FIPS 140-2]**.

5.2.2.6 Cryptographic operation (FCS_COP.1c)

FCS_COP.1c.1 The ~~TSF~~ **IT Environment** shall perform **[data integrity]** in accordance with a specified cryptographic algorithm **[SHA-1 and MD5]** and cryptographic key sizes **[not applicable[3]]** that meet the following: **[FIPS 140-2]**.

5.2.3 Security management (FMT)

5.2.3.1 Secure security attributes (FMT_MSA.2)

FMT_MSA.2.1 The ~~TSF~~ **IT Environment** shall ensure that only secure values are accepted for security attributes.

5.2.4 Protection of the TSF (FPT)

5.2.4.1 Non-bypassability of the TSP (FPT_RVM.1b)

FPT_RVM.1b.1 The ~~TSF~~ **IT Environment** shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.2.4.2 TSF domain separation (FPT_SEP.1b)

FPT_SEP.1b.1 The ~~TSF~~ **IT Environment** shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1b.2 The ~~TSF~~ **IT Environment** shall enforce separation between the security domains of subjects in the TSC.

5.2.4.3 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The ~~TSF~~ **IT Environment** shall be able to provide reliable time stamps for its own use **and for use by the TOE**.

[3] Note that cryptographic hashing algorithms are not keyed.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are the EAL 2 augmented with ALC_FLR.1 components as specified in Part 3 of the Common Criteria. No operations are applied to the assurance components.

Requirement Class	Requirement Component
ACM: Configuration management	ACM_CAP.2: Configuration items
ADO: Delivery and operation	ADO_DEL.1: Delivery procedures
	ADO_IGS.1: Installation, generation, and start-up procedures
ADV: Development	ADV_FSP.1: Informal functional specification
	ADV_HLD.1: Descriptive high-level design
	ADV_RCR.1: Informal correspondence demonstration
AGD: Guidance documents	AGD_ADM.1: Administrator guidance
	AGD_USR.1: User guidance
ALC: Life cycle support	ALC_FLR.1: Basic flaw remediation
ATE: Tests	ATE_COV.1: Evidence of coverage
	ATE_FUN.1: Functional testing
	ATE_IND.2: Independent testing - sample
AVA: Vulnerability assessment	AVA_SOF.1: Strength of TOE security function evaluation
	AVA_VLA.1: Developer vulnerability analysis

Table 3 EAL 2 augmented with ALC_FLR.1 Assurance Components

5.3.1 Configuration management (ACM)

5.3.1.1 Configuration items (ACM_CAP.2)

ACM_CAP.2.1d The developer shall provide a reference for the TOE.
ACM_CAP.2.2d The developer shall use a CM system.
ACM_CAP.2.3d The developer shall provide CM documentation.
ACM_CAP.2.1c The reference for the TOE shall be unique to each version of the TOE.
ACM_CAP.2.2c The TOE shall be labelled with its reference.
ACM_CAP.2.3c The CM documentation shall include a configuration list.
ACM_CAP.2.4c The configuration list shall uniquely identify all configuration items that comprise the TOE.
ACM_CAP.2.5c The configuration list shall describe the configuration items that comprise the TOE.
ACM_CAP.2.6c The CM documentation shall describe the method used to uniquely identify the configuration items that comprise the TOE.
ACM_CAP.2.7c The CM system shall uniquely identify all configuration items that comprise the TOE.
ACM_CAP.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.2 Delivery and operation (ADO)

5.3.2.1 Delivery procedures (ADO_DEL.1)

ADO_DEL.1.1d The developer shall document procedures for delivery of the TOE or parts of it to the user.
ADO_DEL.1.2d The developer shall use the delivery procedures.
ADO_DEL.1.1c The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.
ADO_DEL.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.2.2 Installation, generation, and start-up procedures (ADO_IGS.1)

ADO_IGS.1.1d The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

ADO_IGS.1.1c The installation, generation and start-up documentation shall describe all the steps necessary for secure installation, generation and start-up of the TOE.

ADO_IGS.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2e The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

5.3.3 Development (ADV)

5.3.3.1 Informal functional specification (ADV_FSP.1)

ADV_FSP.1.1d The developer shall provide a functional specification.

ADV_FSP.1.1c The functional specification shall describe the TSF and its external interfaces using an informal style.

ADV_FSP.1.2c The functional specification shall be internally consistent.

ADV_FSP.1.3c The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate.

ADV_FSP.1.4c The functional specification shall completely represent the TSF.

ADV_FSP.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2e The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

5.3.3.2 Descriptive high-level design (ADV_HLD.1)

ADV_HLD.1.1d The developer shall provide the high-level design of the TSF.

ADV_HLD.1.1c The presentation of the high-level design shall be informal.

ADV_HLD.1.2c The high-level design shall be internally consistent.

ADV_HLD.1.3c The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.1.4c The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.1.5c The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.1.6c The high-level design shall identify all interfaces to the subsystems of the TSF.

ADV_HLD.1.7c The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.

ADV_HLD.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.1.2e The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

5.3.3.3 Informal correspondence demonstration (ADV_RCR.1)

ADV_RCR.1.1d The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

ADV_RCR.1.1c For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

ADV_RCR.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.4 Guidance documents (AGD)

5.3.4.1 Administrator guidance (AGD_ADM.1)

AGD_ADM.1.1d The developer shall provide administrator guidance addressed to system administrative personnel.

AGD_ADM.1.1c The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.

AGD_ADM.1.2c The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3c The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4c The administrator guidance shall describe all assumptions regarding user behaviour that are relevant to secure operation of the TOE.

AGD_ADM.1.5c The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6c The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7c The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8c The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

AGD_ADM.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.4.2 User guidance (AGD_USR.1)

AGD_USR.1.1d The developer shall provide user guidance.

AGD_USR.1.1c The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2c The user guidance shall describe the use of user-accessible security functions provided by the TOE.

AGD_USR.1.3c The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

AGD_USR.1.4c The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behaviour found in the statement of TOE security environment.

AGD_USR.1.5c The user guidance shall be consistent with all other documentation supplied for evaluation.

AGD_USR.1.6c The user guidance shall describe all security requirements for the IT environment that are relevant to the user.

AGD_USR.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.5 Life cycle support (ALC)

5.3.5.1 Basic flaw remediation (ALC_FLR.1)

ALC_FLR.1.1d The developer shall provide flaw remediation procedures addressed to TOE developers.

ALC_FLR.1.1c The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.1.2c The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, as well as the status of finding a correction to that flaw.

ALC_FLR.1.3c The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.1.4c The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.

ALC_FLR.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6 Tests (ATE)

5.3.6.1 Evidence of coverage (ATE_COV.1)

ATE_COV.1.1d The developer shall provide evidence of the test coverage.

ATE_COV.1.1c The evidence of the test coverage shall show the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.

ATE_COV.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6.2 Functional testing (ATE_FUN.1)

ATE_FUN.1.1d The developer shall test the TSF and document the results.

ATE_FUN.1.2d The developer shall provide test documentation.

ATE_FUN.1.1c The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2c The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.

ATE_FUN.1.3c The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.4c The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.5c The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.

ATE_FUN.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.3.6.3 Independent testing - sample (ATE_IND.2)

ATE_IND.2.1d The developer shall provide the TOE for testing.

ATE_IND.2.1c The TOE shall be suitable for testing.

ATE_IND.2.2c The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

ATE_IND.2.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2e The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.

ATE_IND.2.3e The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

5.3.7 Vulnerability assessment (AVA)

5.3.7.1 Strength of TOE security function evaluation (AVA_SOF.1)

AVA_SOF.1.1d The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.

AVA_SOF.1.1c For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST.

AVA_SOF.1.2c For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric defined in the PP/ST.

AVA_SOF.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_SOF.1.2e The evaluator shall confirm that the strength claims are correct.

5.3.7.2 Developer vulnerability analysis (AVA_VLA.1)

AVA_VLA.1.1d The developer shall perform a vulnerability analysis.

AVA_VLA.1.2d The developer shall provide vulnerability analysis documentation.

AVA_VLA.1.1c The vulnerability analysis documentation shall describe the analysis of the TOE deliverables performed to search for obvious ways in which a user can violate the TSP.

AVA_VLA.1.2c The vulnerability analysis documentation shall describe the disposition of obvious vulnerabilities.

AVA_VLA.1.3c The vulnerability analysis documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.1.1e The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.1.2e The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

6. TOE Summary Specification

This chapter describes the security functions and associated assurance measures.

6.1 TOE Security Functions

6.1.1 Security audit

The WebLogic Auditing Provider offers auditing functions used by the other WebLogic components. Each of the WLI components invokes the Auditing Provider when a security-relevant event occurs, providing all pertinent information, except the time stamp, which the Auditing Provider queries from the hosting operating system when audit data is received.

The Auditing Provider can be configured to filter audit events based on a severity level (INFORMATION, WARNING, ERROR, SUCCESS, FAILURE, and AUDIT_FAILURE). Events that are not filtered are formatted to include a timestamp, severity (indicating success or failure), event type, and event specific information (including the identity of the responsible user when applicable). All recorded audit events are written into a file provided by the hosting operating system and are accessible via operating system functions.

While the set of possible audit events can vary depending on the components plugged into the WebLogic Server security framework, the evaluated set of components generate audit records for at least the following types of security relevant events: logon, account lock, account lock expired (unlocked), assertion of identity, and authorization (i.e., access checks).

The Security audit function is designed to satisfy the following security functional requirements:

- FAU_GEN.1: WLI provides the ability to audit the required audit start-up/shutdown, login, account lock and unlock, identity assertions, and authorization related audit events and each audit event is recorded with the required date/time, event type, subject (when applicable), and success or failure based on severity which is included.
- FAU_GEN.2: As indicated above, the responsible user is associated with audit events that apply to user actions.

6.1.2 Cryptographic support

WLI incorporates the RSA BSAFE Crypto-J library, which has been FIPS 140-2 certified, and the Cert-J library. When the WLI subsystem interacts with another network entity and enforces message-level security via RosettaNet or ebXML protocols, WLI uses the BSAFE library to digitally sign and seal data when sending and alternately to digitally verify the signature of and unseal data when receiving.

The Cryptographic support function is designed to satisfy the following security functional requirements:

- FCS_COP_EX.1: WLI has been designed to invoke the cryptographic finctions of the cryptographic library available in the IT environment in order to sign business traffic using 1024-bit keys for DSA and to encrypt business traffic using 168-bit keys for Triple-DES.

6.1.3 User data protection

6.1.3.1 Roles

Security roles are computed and granted to users or groups dynamically. Users may be placed into groups that are associated with security roles, or be directly associated with security roles. Security roles can be scoped to specific WLI resources within a single application in a WLI domain (unlike groups, which are always scoped to an entire WLI domain.)

The WLI supports the following role conditions:

- User Name of the Caller,
- Caller is a Member of the Group, and
- Hours of Access are Between.

The role mapping process is initiated when a user or system process requests a WLI resource on which it will attempt to perform a given operation. The resource container that handles the type of WLI resource being requested receives the request. The resource container calls the WebLogic Server security framework and passes in the request parameters, including information such as the subject of the request and the WLIresource being requested. The WebLogic Server security framework calls each configured Role Mapping provider to obtain a list of the roles that apply. If a security policy specifies that the requestor is entitled to a particular role, the role is added to the list of roles that are applicable to the subject. This process continues until all security policies that apply to the WLI resource or the resource container have been evaluated. The list of roles is returned to the WebLogic Server security framework, where it can be used as part of other operations, such as access decisions.

The result of the dynamic role association is a set of roles that apply to the principals stored in a subject at a given moment. These roles can then be used to make authorization decisions for protected WLI resources, as well as for resource container and application code.

6.1.3.2 Resources

A WLI resource is a structured object used to represent an underlying WebLogic Server entity, which can be protected from unauthorized access using security roles and security policies. The WebLogic Server protects the eleven types of resources listed below.

Type of Resource	Description and how protected	Initial Default Security Policy
Administrative	Access can be granted to resources that allow users to perform administrative tasks (e.g., weblogic.Admin, and MBean APIs).	Default global roles: Admin
Application	Resources that represent enterprise applications, packaged as EAR (Enterprise Application aRchive) files. Use this type of WebLogic resource to protect all EJBs (Enterprise JavaBeans) within an entire application.	None
Component Object Model (COM)	Resources that are designed as program component objects according to Microsoft's framework. In the left hand pane of the WebLogic Server Administration Console, click the Services node, and then click the JCOM node underneath it. Grant the COM client user access to the classes that the COM client application needs to access.	Default group: none
Enterprise Information System (EIS)	Resources that are designed as connectors, which allow for the integration of Java applications with existing enterprise information systems. If the resource adapter has not defined specific security policies, WebLogic Server overrides the runtime environment for the resource adapter with the default security policies specified in the J2EE Connector Architecture Specification. If the resource adapter has defined specific security policies, WebLogic Server first overrides the runtime environment for the resource adapter first with a combination of the default security policies for resource adaptors and the specific policies defined for the resource adapter. Resource adapters define specific security policies using the security-permission-spec element in the ra.xml deployment descriptor file.	Default group: everyone

Type of Resource	Description and how protected	Initial Default Security Policy
Enterprise JavaBean (EJB)	Resources that are related to EJBs. Use this type of WebLogic resource when you want to protect EJB JARs, individual EJBs within an EJB JAR, or individual methods on an EJB.	Default group: everyone
Java Database Connectivity (JDBC)	Resources that are related to JDBC. This type of WebLogic resource includes groups of connection pools, individual connection pools, and multipools. Connection pools are unprotected unless you define security policies for connection pools (as a resource type) or for individual connection pools. If you define a security policy for connection pools, access is restricted to *exactly* what is defined in the security policy. Security policies in fileRealm.properties can be used to secure connection pools	Default group: everyone
Java Messaging Service (JMS)	Resources that are related to JMS.	Default group: everyone
Java Naming and Directory Interface (JNDI)	Resources that use the industry-standard JNDI API to enable connectivity to heterogeneous enterprise naming and directory services	Default group: everyone
Server	WebLogic Server instances. The allowed operations are Start, Shut down, Lock, and Unlock	Default global roles: Admin Operator
Universal Resource Locator (URL)	Resources that are related to Web applications. This type of WebLogic resource can be a WAR (Web Application aRchive) file or individual components of a Web application (such as servlets and JSPs).	Default group: everyone
Web Service	Resources that are related to services, which can be shared by and used as components of distributed, Web-based applications. This type of WebLogic resource can be an entire Web service or individual components of a Web service. WebLogic Web services are packaged as standard J2EE Enterprise applications. Consequently, access to the Web service is secured by securing access to some or all of the J2EE components that make up the Web service: the Web service, the Web service URL, the stateless session EJB that implements the Web service, and a subset of the methods of the stateless session EJB.	Default group: everyone

WebLogic Integration adds protection for WebLogic Integration resources where access is explicitly defined when the Message Broker Channels, Business Processes, Application Views, Trading Partner Profiles, Trading Partner Services, Service Profiles, and Worklists are created.

Access to EJB and URL (Web) Resources can be controlled using either the Administration Console or Deployment Descriptors or a combination. This is controlled using the *fullyDelegateAuthorization Flag* and *Ignore Security Data in Deployment Descriptors Check Box*. Note that deployment descriptors are stored as XML files.

When the value of the fullyDelegateAuthorization flag is false, the WebLogic Security Service *only* performs security checks on URL and EJB resources that have security specified in their associated deployment descriptors (DDs). This is the default. Alternately, when the value of the fullyDelegateAuthorization flag is true, the WebLogic Security Service performs security checks on all URL (Web) and EJB resources, regardless of whether there are any security settings in the deployment descriptors (DDs) for these WebLogic resources.

If the Ignore Security Data in Deployment Descriptors check box is checked, the security policy for URL and EJB resources is determined by the WebLogic Server Administration Console. Alternately, if the Ignore Security Data in

Deployment Descriptors check box is not checked, the security policy for URL and EJB resources is determined by the deployment descriptors (that is, the ejb-jar.xml, weblogic-ejb-jar.xml, web.xml, and weblogic.xml files).

Note that Deployment Descriptors are used when deploying an object and, depending on the conditions above, result in a security policy for the associated object that is used subsequently to determine access to the object. If no security policy is specifically assigned to an object, then the default security policy is used.

6.1.3.3 Security Policies

A security policy is created when an association is defined between a WLI resource and one or more users, groups, or security roles and is stored in the Authorization provider's database. Security policies can be assigned to any of the defined WLI resources or to attributes or operations of a particular instance of a WLI resource. If a security policy is assigned to a type of WLI resource, all new instances of that resource inherit that security policy. Security policies assigned to individual resources or attributes override security policies assigned to a type of WLI resource. Furthermore, a time constraint can be defined for a security policy.

The WebLogic Authorization Provider is configured for WLI resources and security policies are stored in the embedded LDAP server. These security policies are based on security roles and default global groups.

6.1.3.4 Access Decisions

An Access Decision is the component of an Authorization provider that determines whether or not a subject (i.e., a thread acting on behalf of users) has permission to perform a given operation on a WLI 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, an Adjudication provider is required to process the multiple Access Decisions and render a verdict. In WLI, the WebLogic Adjudication Provider is used to process the results that multiple Access Decisions return, and determines the final 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 WLI resource).
- Otherwise return a final verdict of FALSE (that is, deny access to the WLI resource).

If the Require Unanimous Permit attribute is set to FALSE, the WebLogic Adjudication provider treats ABSTAIN verdicts as PERMIT and acts as follows:

- If any of the Authorization providers' Access Decisions return DENY, then return a final verdict of FALSE (that is, deny access to the WLI resource).
- Otherwise return a final verdict of TRUE (that is, permit access to the WLI resource).

The User data protection function is designed to satisfy the following security functional requirements:

- FDP_ACC.1: As indicated above, WLI enforces an access control policy between users and a broad range of objects, as required.
- FDP_ACF.1: As indicated above, WLI enforces a fairly complex set of access control rules, as required.

6.1.4 Identification and authentication

The WLI provides the following types of authentication providers:

- WebLogic Authentication Provider

- WebLogic Identity Assertion Provider

In addition, the WLI provides the WebLogic Credential Mapping Provider that maps a user's authentication identity to those required for legacy applications, so that the legacy application gets the necessary credential information, when necessary, that is associated with a user of the TOE that has already been identified and authenticated by the TOE.

Regardless of provider, WLI maintains at least the following user attributes:

- Username
- Group memberships
- Password

WLI can also associate users with roles either directly or indirectly via roles assigned to groups.

The WebLogic Authentication Provider performs authentication based on a username and password. The minimum password length required by this provider is eight (8) characters and the hashed passwords are stored in the Embedded LDAP Server.

When required (see below), a username and password are requested from the user and sent to WebLogic Server. When WebLogic Server receives the information, the password presented is hashed and the WebLogic Authentication Provider compares it to the stored hashed password to determine whether it matches and, hence, whether authentication is successful.

The WebLogic Identity Assertion Provider is a specific form of Authentication provider that allows users or system processes to assert their identities using tokens. 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 principals.

The WebLogic Identity Assertion Provider supports certificate authentication using:

- X509 certificates, and
- CORBA Common Secure Interoperability version 2 (CSIv2) identity assertion.

Regardless of authentication provider, the end result is that the users authenticated identity is used to determine a set of principals (internal identities) that serve to represent the specific users, associated groups, as well as roles. These principals are then associated with a thread that will act on the authenticated user's behalf (i.e., subject), and those principals will not change for the lifetime of the thread.

Each Authentication Provider can be configured independently to determine whether or how authentication must occur. The following attributes can be assigned to each authentication provider for each resource:

- REQUIRED - requires this Authentication provider to succeed. Regardless of whether it succeeds, authentication proceeds to other Authentication providers that have been configured as part of the login sequence.
- REQUISITE - requires this Authentication provider to succeed. If it succeeds, authentication proceeds to other Authentication providers. If it fails, control immediately returns to the application (authentication does not proceed).
- SUFFICIENT - does not require this Authentication provider to succeed. If it succeeds, control immediately returns to the application (authentication does not proceed to other Authentication providers). If it fails, authentication proceeds to other Authentication providers that have been configured as part of the login sequence.
- OPTIONAL - does not require this Authentication provider to succeed. Regardless of whether it succeeds, authentication proceeds to other Authentication providers that have been configured as part of the login sequence.

The caller into the TSF (e.g., a resource container) determines if a user is to be authenticated. If the container does not require authentication, the user is assigned the identity "<Anonymous>". All users (including the "<Anonymous>" user) are members of the "everyone" group. All identified/ authenticated users are members of the "users" group (but "<Anonymous"> is not). The "users" group is also a member of "everyone". The WebLogic Server administrator can set policies on resources to prevent anonymous users from accessing protected resources, but by default the following resources are accessible by anonymous users (because the default policy grants access to the "everyone" group): EIS; EJB; JDBC; JNDI; JMS; URL; Web Services. If multiple Authentication Providers were configured (not allowed in the evaluated configuration), then the REQUIRED, REQUISITE, SUFFICIENT, OPTIONAL flags control how the Authentication Providers are used in the login sequence. If additional Authentication providers were added, by default the Control Flag attribute would be set to OPTIONAL.

Note that access to the administrator console always requires authentication and is not subject to the resource authentication configuration settings explained above. Note also that of the four authentication provider attributes defined above: REQUIRED indicates a mandatory authentication provider (and allows further authentication processing though it ultimately will not succeed); REQUISITE also indicates a mandatory authentication provider (but will stop authentication processing upon failure); SUFFICIENT effectively indicates an optional authentication provider in that if it fails authentication can still be successful per other authentication providers – it also has the effect of rendering all subsequent authentication providers optional when it succeeds; and, OPTIONAL is also optional and has no effect on the other authentication providers.

WLI defines a set of attributes to protect user accounts as defined below. If a user account exceeds the values set for the attributes on the User Tab, the user account becomes locked. The User Lockout attributes apply to the security realm and all its security providers.

Attribute	Description	Default
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. [4] 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
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

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

- FIA_AFL.1: As indicated in the table above, the administrator can configure a number of constraints dealing with authentication failures, including locking accounts after a predefined number of failures.
- FIA_ATD.1: As indicated above, users are associated with usernames, credentials (such as passwords), and groups, and roles.
- FIA_UAU.1: Each resource offered by the TOE can be configured so that users must be successfully identified and authenticated prior to access.

[4] Note that the TOE will not allow a Lockout Threshold of '0' to be assigned.

- FIA_UAU.5: As indicated above, multiple authentication mechanisms are available and they can be assigned individually to resources in a flexible manner ranging from 'optional' to 'required'.
- FIA_UID.1: Each resource offered by the TOE can be configured so that users must be successfully identified and authenticated prior to access.
- FIA_USB.1: Upon successful logon, a thread is created to act on behalf of the authenticated user and is assigned principals (identities) representing the user, groups, and roles.

6.1.5 Security management

The embedded LDAP server is used as the database that stores user, group, security roles, and security policies for the WLI security providers. The embedded LDAP server is a complete LDAP server. It supports the following access and storage functions:

- Access and modification of entries in the LDAP server
- Use of an LDAP browser to import and export security data into and from the LDAP server.
- Read and write access by the WLI security providers.

The following table lists the security attributes and TSF data stored in the LDAP server for each type of security provider.

Security Provider	LDAP Information
Authentication	Stores user and group information.
Authorization	Stores security roles, and security policies.
Role Mapping	Supports dynamic role associations by obtaining a computed set of roles granted to a requestor for a given WebLogic resource.
Auditing	None.
Credential Mapping	Stores Username-Password credential mapping information.
Identity Assertion	Stores user and group information.

WebLogic Server defines the following roles for system administration operations, and the permissions granted to each role.

Global Role	Global Role Permissions
Administrator	View the server configuration, including the encrypted/hashed value of encoded attributes. [5] Modify the entire server configuration. Deploy applications, EJBs, startup and shutdown classes, J2EE Connectors, and Web Service components, and edit deployment descriptors. Start, resume, and stop servers by default.
Deployer	View the server configuration, except for encoded attributes (encrypted/hashed or not). Deploy applications, EJBs, startup and shutdown classes, J2EE Connectors, and Web Service components, and edit deployment descriptors.
Operator	View the server configuration, except for encoded attributes (encrypted/hashed or not). Start, resume, and stop servers by default.
Monitor	View the server configuration, except for encoded attributes (encrypted/hashed or not).

[5] As indicated in this table, there are some attributes that are normally encoded by being encrypted or hashed. The Administrator role is the only role that can view the encoded form of these attributes and no role can view the unencoded (i.e., plain text) form of these attributes.

In addition, the WebLogic Integration subsystem defines the following additional roles: IntegrationAdmin, IntegrationOperator, and IntegrationMonitor.

While any number of additional roles can be created for use by applications, only these roles have permission to view or change the configuration of a WLI. Furthermore, for the purposes of this Security Target, all of these roles are considered to be instances of the "Administrator" defined in FMT_SMR.1, regardless of the fact that some of them are limited in their overall functions.

The assignment of roles to users is accomplished either directly or via the assignment of groups associated with roles. Conversely, each role is associated with groups that serve to grant access to applicable WLI resources.

The User Data Protection security function effectively enforces restrictions related to administration functions. In particular, access to view or modify TSF data, including that used to define the operation of the TOE, is restricted to one or more of the administrative roles identified above. Of particular note, 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.

Furthermore, WLI offers interfaces that allow an administrator to effectively manage the TOE; including, viewing configuration data, managing user accounts and their attributes; managing roles; and, management of the access control settings.

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

- FMT_MOF.1: The ability to determine and modify the behavior of the functions of the TOE is restricted to an administrator using the user data protection function and appropriate access control settings on the applicable resources.
- FMT_MSA.1: The ability to modify User names, Passwords, Groups and Group Membership, Roles, and Security policies is restricted to an administrator using the user data protection function and appropriate access control settings on the applicable resources.
- FMT_MSA.3: The table in section 6.1.3.2 identifies the default authorizations (i.e., initial default policies) for the various types of objects. Just like access settings themselves, these default policies are protected via the user data protection security functions so that only an administrator can change them.
- FMT_SMF.1: As indicated above, WLI provides at least the administrative functions to view the WLI configuration; create and delete user accounts and modify user security attributes; assign and revoke security roles; and manage the Access Control SFP.
- FMT_SMR.1: WLI supports the definition of numerous roles, a number of which correspond to the "administrator" as indicated above.

6.1.6 Protection of the TSF

The WLI is designed to operate in domains provided by the underlying Java runtime environment. WLI maintains its domains in a manner that separates threads acting on behalf of users separate from its own threads. Furthermore, it manages user threads so that they are kept distinct and separate from one another.

The interfaces, primarily from a network, offered by WLI have all been carefully designed, implemented, and tested to ensure that they do not offer opportunities to tamper with or interfere with the operation of the security functions and also to ensure that they do not offer any access to protected resources that is not subject to the various security policies.

The Protection of the TSF function is designed to satisfy the following security functional requirements:

- FPT_RVM.1a: WLI is designed to encapsulate its protected resources and offer access only through well defined interfaces that ensure that the applicable security policies are enforced as configured by an administrator.

- FPT_SEP.1a: WLI is designed to keep its own functions distinct and separate from those of the untrusted subjects it instantiates and also to keep all of its untrusted subjects distinct and separate from one another.

6.2 TOE Security Assurance Measures

6.2.1 Configuration management

The configuration management measures applied by BEA ensure that configuration items are uniquely identified, and that documented procedures are used to control and track changes that are made to the TOE. BEA performs configuration management on a defined list of configuration items including, but not limited to, the TOE implementation representation, design, tests, user and administrator guidance, and the CM documentation.

These activities are documented in:

- BEA WebLogic Platform Version 8.1 Configuration Management, version 1.5, 16 November 2006

The Configuration management assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- ACM_CAP.2

6.2.2 Delivery and operation

BEA provides delivery documentation and procedures to identify the TOE, secure the TOE during delivery, and provide necessary installation and generation instructions. BEA's delivery procedures describe all applicable procedures to be used to prevent in appropriate access to the TOE. BEA also provides documentation that describes the steps necessary to install WLI in accordance with the evaluated configuration.

These activities are documented in:

- BEA WebLogic Platform Version 8.1 Delivery and Operation, version 1.1, 4 March 2005
- Installing BEA WebLogic Platform 8.1 SP 6, June 2006

The Delivery and operation assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- ADO_DEL.1
- ADO_IGS.1

6.2.3 Development

BEA has documents describing all facets of the design of the TOE. These documents serve to describe all of the security functions of the TOE, the purpose and method of use of all interfaces both external and between subsystems, the architecture of the TOE (in terms of subsystems), and correspondence between the available design abstractions (including the ST).

These activities are documented in:

- BEA WebLogic Integration Version 8.1 Functional Specification, version 1.0, 31 October 2006
- BEA WebLogic Integration Version 8.1 High-level Design, version 1.0, 09 November 2006
- BEA WebLogic Integration Version 8.1 Representation Correspondence, version 1.0, 09 November 2006

The Development assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- ADV_FSP.1
- ADV_HLD.1

- ADV_RCR.1

6.2.4 Guidance documents

BEA provides administrator and user guidance on available tools and relevant parameters, how to utilize the TOE security functions, secure use assumptions, and warnings to administrators and users about actions that can compromise the security of the TOE.

BEA provides an extensive set of documentation describing the installation, configuration, management and operation of the TOE. This set comprises documentation for the WebLogic Server product (which is also part of the WebLogic Integration TOE) and for WebLogic Integration itself. The WebLogic Server documentation is available at http://edocs.bea.com/wls/docs81/index.html, while the WebLogic Integration documentation is available at http://edocs.bea.com/wli/docs81/interm/index.html. Additionally, the installation guide for WebLogic Platform, which also covers BEA WebLogic Integration, is available at http://edocs.bea.com/platform/docs81/install/index.html.

The guidance documentation included in the TOE is as follows:

- Installation Guidance
  - Installing BEA WebLogic Platform 8.1 SP6, June 2006
- WebLogic Server Guidance
  - Administration Console Online Help
  - Configuring and Managing WebLogic Server 8.1, 28 Jun 2006
  - Developing Web Applications for WebLogic Server 8.1, 28 Jun 2006
  - Introduction to WebLogic Security 8.1, 28 Jun 2006
  - Managing WebLogic Security 8.1, 28 Jun 2006
  - Programming WebLogic Enterprise JavaBeans 8.1, 28 Jun 2006
  - Programming WebLogic jCOM 8.1, 28 Jun 2006
  - Programming WebLogic Security 8.1, 28 Jun 2006
  - Programming WebLogic Server J2EE Connectors 8.1, 1 Jul 2003
  - Programming WebLogic Web Services 8.1, 28 Jun 2006
  - Securing a Production Environment 8.1, 28 Jun 2006
  - Securing WebLogic Resources 8.1, 28 Jun 2006
  - WebLogic Server Command Reference 8.1, 28 Jun 2006
- WebLogic Integration Guidance
  - Managing WebLogic Integration Solutions, 8.1, Oct 2005
  - Deploying WebLogic Integration Solutions, 8.1, 28 Jun 2006
  - Introducing Application Integration, 8.1, 28 Jun 2006
  - Introducing Trading Partner Integration, 8.1, 28 Jun 2006
  - Using the Application Integration Design Console, 8.1, 28 Jun 2006
  - Using Integration Controls, 8.1, Jan 2005

The Guidance documents assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- AGD_ADM.1
- AGD_USR.1

6.2.5 Life cycle support

BEA has a series of procedures that define the process for accepting and acting upon user reports of security flaws. These procedures describe the acceptance criteria for security flaws, how all security flaws are tracked and the status of the fix for each security flaw. The procedures also explain how information about flaws and corrections is made available to users of the TOE.

These activities are documented in:

- BEA WebLogic Platform, Version 8.1, Flaw Remediation (ALC_FLR), v0-1-03, 4 January 2006

The Life cycle support assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- ALC_FLR.1

6.2.6 Tests

The test documents describe the overall test plan, testing procedures, the tests themselves, including expected and actual results. In addition, these documents describe how the functional specification has been appropriately tested.

These activities are documented in:

- BEA WebLogic Integration Version 8.1 Test Documentation, version 1.0, 01 December 2006
- BEA WebLogic Integration Version 8.1 Test Documentation Addendum, version 1.0, 07 December 2006

The Tests assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- ATE_COV.1
- ATE_FUN.1
- ATE_IND.2

6.2.7 Vulnerability assessment

BEA has conducted a strength of function analysis wherein all permutational or probabilistic security mechanisms have been identified and analyzed resulting in a demonstration that all of the relevant mechanisms fulfill the minimum strength of function claim, SOF-Basic.

BEA performs regular vulnerability analyses of the entire TOE (including documentation) to identify and correct weaknesses that can be exploited in the TOE. The analysis explains why uncorrected vulnerabilities are not exploitable in the intended environment of the TOE.

These activities are documented in:

- BEA WebLogic Server Version 8.1 Common Criteria Vulnerability Assessment (AVA), v0-1-02, 17 Feb 2006
- BEA WebLogic Integration Version 8.1 Vulnerability Assessment, version 1.0, 17 November 2006
- BEA WebLogic Platform 8.1 Vulnerability Assessment (AVA) Addendum, Version 1.0, 07 May 2007

The Vulnerability assessment assurance measure satisfies the following EAL 2 augmented with ALC_FLR.1 assurance requirements:

- AVA_SOF.1
- AVA_VLA.1

7. Protection Profile Claims

There is no Protection Profile claim in this Security Target.

8. Rationale

This section provides the rationale for completeness and consistency of the Security Target. The rationale addresses the following areas:

- Security Objectives;
- Security Functional Requirements;
- Security Assurance Requirements;
- Strength of Functions;
- Requirement Dependencies;
- TOE Summary Specification; and,
- PP Claims.

8.1 Security Objectives Rationale

This section shows that all secure usage assumptions, organizational security policies, and threats are completely covered by security objectives. In addition, each objective counters or addresses at least one assumption, organizational security policy, or threat.

8.1.1 Security Objectives Rationale for the TOE and Environment

This section provides evidence demonstrating the coverage of organizational policies and usage assumptions by the security objectives.

	T.BYPASS	T.EXCESS_AUTHORITY	T.NO_TIME	T.NOCRYPTO	T.STORAGE	T.TAMPER	T.TSF_COMPROMISE	T.UNACCOUNTABLE	T.UNAUTHORIZED_ACCESS	T.UNDETECTED_ACTIONS	T.UNIDENTIFIED_USERS	A.NO_EVIL	A.NO_UNTRUSTED	A.PHYSICAL
O.AUDIT_GENERATION								X		X				
O.CRYPTOGRAPHY				X										
O.ID_AND_AUTH								X			X			
O.MANAGE							X							
O.MEDIATE									X					
O.ROLES		X												
O.SELF_PROTECTION	X				X	X	X							
OE.CRYPTOGRAPHY				X										
OE.JAVA	X					X	X							
OE.OS	X		X		X	X	X	X						
ON.NO_EVIL												X		
ON.NO_UNTRUSTED													X	
ON.PHYSICAL														X

Table 4 Environment to Objective Correspondence

8.1.1.1 T.BYPASS

An attacker may be able to bypass TOE protection mechanisms through unprotected interfaces in order to inappropriately access protected data and services.

This Threat is satisfied by ensuring that:

- O.SELF_PROTECTION: The TSF operates within its own domain, protecting itself at the interfaces that it offers to ensure that it is not subject to interference, tampering, or inappropriate disclosure of its information.
- OE.JAVA: The IT environment ensures that the Java 2 Security Sandbox will provide for separate domains for security providers and application code within the JVM.
- OE.OS: The IT environment ensures that the underlying operating system will protect TSF code and data structures from unauthorized modification and prevent TSF security functions from being bypassed through the OS interfaces.

8.1.1.2 T.EXCESS_AUTHORITY

An unauthorized user may be able to exercise administrator authorities to inappropriately manage the TOE.

This Threat is satisfied by ensuring that:

- O.ROLES: The TSF distinguishes administrative roles so that administrative functions can be restricted to users acting in those roles.

8.1.1.3 T.NO_TIME

Those responsible for the TOE may not be able to determine the sequence of audited security relevant events.

This Threat is satisfied by ensuring that:

- OE.OS: The IT environment ensures that the underlying operating system will provide support for reliable time stamps. Note that the TOE makes a call to the operating system (IT environment) to obtain the time based on the system clock.

8.1.1.4 T.NOCRYPTO

An attacker may be able to observe authentication data transmitted in the clear due to cryptographic services not being available.

This Threat is satisfied by ensuring that:

- O.CRYPTOGRAPHY: The TSF ensures that the use of cryptographic services for digital encapsulation.
- OE.CRYPTOGRAPHY: The IT environment ensures that the use of cryptographic and key management services for encryption, authentication, key generation, and key destruction.

8.1.1.5 T.STORAGE

An attacker may be able to cause the loss or destruction of Audit and other TSF data.

This Threat is satisfied by ensuring that:

- O.SELF_PROTECTION: The TSF will maintain a domain for its own execution to help ensure that it can effectively control the resources it protects.
- OE.OS: The IT environment ensures that the underlying operating system will protect TSF code and data structures from unauthorized modification and provide files for the storage of audit records.

8.1.1.6 T.TAMPER

An attacker may be able to inappropriately modify or otherwise tamper with TSF programs and data.

This Threat is satisfied by ensuring that:

- O.SELF_PROTECTION: The TSF will maintain a domain for its own execution so that it can protect itself at the interfaces it offers.
- OE.JAVA: The IT environment ensures that the Java 2 Security Sandbox will provide for separate domains for security providers and application code within the JVM.
- OE.OS: The IT environment ensures that the underlying operating system will protect TSF code and data structures from unauthorized modification.

8.1.1.7 T.TSF_COMPROMISE

A user or process may cause TSF data or executable code to be inappropriately accessed (viewed, modified, or deleted).

This Threat is satisfied by ensuring that:

- O.MANAGE: The TOE ensures that administrator functions and facilities are protected from unauthorized use.
- O.SELF_PROTECTION: The TSF will maintain a domain for its own execution so that it can protect itself and its data.
- OE.JAVA: The IT environment ensures that the Java 2 Security Sandbox will provide for separate domains for security providers and application code within the JVM.
- OE.OS: The IT environment ensures that the underlying operating system will protect TSF code and data structures from unauthorized modification and prevent TSF security functions from being bypassed through the OS interfaces

8.1.1.8 T.UNACCOUNTABLE

Users of the TOE may not be held accountable for their security-relevant actions.

This Threat is satisfied by ensuring that:

- O.AUDIT_GENERATION: The TOE ensures that the TOE can detect and create records of security-relevant events associated with users.
- O.ID_AND_AUTH: The TOE ensures that only identified and authenticated users can access logical services of the TOE.
- OE.OS: The IT environment ensures the audit trail is protected to help ensure accountability.

8.1.1.9 T.UNAUTHORIZED_ACCESS

A user may gain access to user data for which they are not authorized according to the TOE security policies.

This Threat is satisfied by ensuring that:

- O.MEDIATE: The TOE ensures that user data is protected in accordance with its security policies.

8.1.1.10 T.UNDETECTED_ACTIONS

The administrator may not have the ability to detect potential security violations, thus limiting the administrator's ability to identify and take action against a possible security breach.

This Threat is satisfied by ensuring that:

- O.AUDIT_GENERATION: The TOE ensures that the TOE will provide the capability to detect and create records of security-relevant events associated with users. The administrator can review the audit records with a text editor to look for potential security violations.

8.1.1.11 T.UNIDENTIFIED_USERS

An attacker may gain access to the TOE without being reliably identified allowing them to gain unauthorized access to data or TOE resources.

This Threat is satisfied by ensuring that:

- O.ID_AND_AUTH: The TOE ensures that only identified and authenticated users can access protected security-relevant functions or data within the TOE.

8.1.1.12 A.NO_EVIL

Administrators are non-hostile, appropriately trained, and follow all administrator guidance.

This Assumption is satisfied by ensuring that:

- ON.NO_EVIL: The environment ensures that sites using the TOE shall ensure that administrators are non-hostile, appropriately trained, and follow all administrator guidance.

8.1.1.13 A.NO_UNTRUSTED

There are no untrusted user accounts or malicious software on the server platform.

This Assumption is satisfied by ensuring that:

- ON.NO_UNTRUSTED: The environment ensures that those responsible for the TOE will ensure that there are no untrusted user accounts or software on the server platform.

8.1.1.14 A.PHYSICAL

Physical security, commensurate with the value of the TOE and the data it contains, is provided by the IT environment.

This Assumption is satisfied by ensuring that:

- ON.PHYSICAL: The environment ensures that physical security will be provided within the domain for the value of the IT assets protected by the operating system and the value of the stored, processed, and transmitted information.

8.2 Security Requirements Rationale

This section provides evidence supporting the internal consistency and completeness of the components (requirements) in the Security Target. Note that **Table 5** indicates the requirements that effectively satisfy the individual objectives. .

8.2.1 Security Functional Requirements Rationale

All Security Functional Requirements (SFR) identified in this Security Target are fully addressed in this section and each SFR is mapped to the objective for which it is intended to satisfy.

	O.AUDIT_GENERATION	O.CRYPTOGRAPHY	O.ID_AND_AUTH	O.MANAGE	O.MEDIATE	O.ROLES	O.SELF_PROTECTION	OE.CRYPTOGRAPHY	OE.JAVA	OE.OS
FAU_GEN.1	X									
FAU_GEN.2	X									
FCS_COP_EX.1		X								
FDP_ACC.1					X					
FDP_ACF.1					X					
FIA_AFL.1			X							
FIA_ATD.1			X							
FIA_UAU.1			X							
FIA_UAU.5			X							
FIA_UID.1			X							
FIA_USB.1			X							
FMT_MOF.1				X						
FMT_MSA.1				X						
FMT_MSA.3				X						
FMT_SMF.1				X						
FMT_SMR.1				X		X				
FPT_RVM.1a							X			
FPT_SEP.1a							X			
FAU_SAR.1										X
FAU_STG.1										X
FCS_CKM.1a								X		
FCS_CKM.1b								X		
FCS_CKM.4								X		
FCS_COP.1a								X		
FCS_COP.1b								X		
FCS_COP.1c								X		
FMT_MSA.2								X		
FPT_RVM.1b									X	X
FPT_SEP.1b									X	X
FPT_STM.1										X

Table 5 Objective to Requirement Correspondence

8.2.1.1 O.AUDIT_GENERATION

The TOE will provide the capability to detect and create records of security relevant events associated with users.

This TOE Security Objective is satisfied by ensuring that:

- FAU_GEN.1: The TSF is required to generate audit records for the auditable events identified in FAU_GEN.1.
- FAU_GEN.2: The TSF is required to ensure audit records are associated with the applicable user.

8.2.1.2 O.CRYPTOGRAPHY

The TOE shall provide the capability invoke cryptographic services for digital encapsulation of critical user data.

This TOE Security Objective is satisfied by ensuring that:

- FCS_COP_EX.1: The TSF is required to digitally encapsulate traffic using an appropriate mechanism.

8.2.1.3 O.ID_AND_AUTH

The TOE will provide identification and authentication mechanisms that control logical access to the TOE.

This TOE Security Objective is satisfied by ensuring that:

- FIA_AFL.1: The TSF is required to allow an administrator to define a threshold for incorrect login attempts after which the TSF will lock the user's account to help mitigate the chance of an inappropriate user login.
- FIA_ATD.1: The TSF is required to associate a username, password, and groups with each user so that it can effectively identify and authenticate the user and subsequently assign the appropriate authorities.
- FIA_UAU.1: The TSF is required to ensure that users are authenticated prior to allowing access to resources, except those specifically permitted by an administrator.
- FIA_UAU.5: The TSF is required to offer alternate user authentication mechanisms to support a variety of authentication scenarios.
- FIA_UID.1: The TSF is required to ensure that users are identified prior to allowing access to resources, except those specifically permitted by an administrator.
- FIA_USB.1: The TSF is required to ensure that user attributes are appropriately assigned to subjects acting on behalf of the corresponding user.

8.2.1.4 O.MANAGE

The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.

This TOE Security Objective is satisfied by ensuring that:

- FMT_MOF.1: The TSF is required to restrict the ability to determine and control the current behavior of the applicable security functions to the administrator.
- FMT_MSA.1: The TSF is required to restrict the ability to modify user attributes, administrator roles, groups and group membership, and security policy settings to administrators.
- FMT_MSA.3: The TSF is required to restrict the ability to manage default initial values to the administrator.
- FMT_SMF.1: The TSF is required to provide the security management functions necessary to support effective security management of the TOE.
- FMT_SMR.1: The TSF is required to maintain a set of administrator roles and their association with users.

8.2.1.5 O.MEDIATE

The TOE will protect user data in accordance with its security policies.

This TOE Security Objective is satisfied by ensuring that:

- FDP_ACC.1: The TSF is required to enforce its access control SFP on all subjects, objects, and operations defined in FDP_ACC.1.
- FDP_ACF.1: The TSF is required to enforce the access control rules associated with the access control SFP (see FDP_ACF.1).

8.2.1.6 O.ROLES

The TOE will support administrator roles that are differentiated from users not allowed to perform administrative operations.

This TOE Security Objective is satisfied by ensuring that:

- FMT_SMR.1: The TSF is required to support administrator roles that can be assigned to users.

8.2.1.7 O.SELF_PROTECTION

The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure.

This TOE Security Objective is satisfied by ensuring that:

- FPT_RVM.1a: The TSF is required to ensure that the enforcement functions cannot be bypassed as realized at its own interfaces.
- FPT_SEP.1a: The TSF is required to maintain its own domain, protected from interference and tampering from the untrusted subjects it is intended to service.

8.2.1.8 OE.CRYPTOGRAPHY

The IT environment shall provide cryptographic services for encryption, authentication, and key management services for key generation and key destruction.

This IT Environment Security Objective is satisfied by ensuring that:

- FCS_CKM.1a: The IT environment is required to generate encryption keys using an appropriate mechanism.
- FCS_CKM.1b: The IT environment is required to generate authentication keys using an appropriate mechanism.
- FCS_CKM.4: The IT environment is required to destroy keys appropriately.
- FCS_COP.1a: The IT environment is required to encrypt/decrypt using an appropriate mechanism.
- FCS_COP.1b: The IT environment is required to perform cryptographic authentication using an appropriate mechanism.
- FCS_COP.1c: The IT environment is required to perform cryptographic integrity using appropriate hashing mechanisms.
- FMT_MSA.2: The IT environment is required to ensure the entry of only secure values.

8.2.1.9 OE.JAVA

The Java 2 Security Sandbox will provide for separate domains for security providers and application code within the JVM.

This IT Environment Security Objective is satisfied by ensuring that:

- FPT_RVM.1b: The IT environment is required to ensure its security functions cannot be bypassed.
- FPT_SEP.1b: The IT environment is required to protect itself from tampering and to separate its subjects.

8.2.1.10 OE.OS

The underlying operating system will protect TSF code and data structures from unauthorized modification and prevent TSF security functions from being bypassed through the OS interfaces. The operating system will provide protected files for the storage of audit records and also tools for review of the audit records. The operating system platform will provide reliable time stamps.

This IT Environment Security Objective is satisfied by ensuring that:

- FAU_SAR.1: The IT environment is required to provide the means to review the audit trail.
- FAU_STG.1: The IT environment is responsible to protect the audit trail.
- FPT_RVM.1b: The IT environment is required to ensure its security functions cannot be bypassed to help ensure that the TOE itself is instantiated in a secure manner within the IT environment.
- FPT_SEP.1b: The IT environment is required to protect itself from tampering and to separate its subjects, including separating the TOE from other subjects known to the IT environment.
- FPT_STM.1: The IT environment is required to provide reliable time stamps.

8.3 Security Assurance Requirements Rationale

EAL 2 was selected as the assurance level because the TOE is a commercial product whose users require a low to moderate level of independently assured security. WebLogic Integration is targeted at an environment with good physical access security and competent administrators, where EAL 2 should provide adequate assurance. Within such environments it is assumed that attackers will have little attack potential. As such, EAL2 is appropriate to provide the assurance necessary to counter the limited potential for attack.

The base assurance level was augmented to EAL 2 augmented with ALC_FLR.1, because flaw remediation procedures provide greater assurance that security-related bugs will be fixed in a widely distributed commercial product.

8.4 Strength of Functions Rationale

The overall strength of function claim of SOF-Basic is believed to be commensurate with the overall assurance claim of EAL 2 augmented with ALC_FLR.1. The only applicable security function is Identification and Authentication where passwords are used by users as evidence of their claimed identities. The intent is that the password mechanism meets or exceeds SOF-Basic and the evidence can be found in the strength of function analysis included in BEA WebLogic Vulnerability Analysis.

8.5 Requirement Dependency Rationale

The following table demonstrates that all dependencies among the claimed security requirements are satisfied, except for ADV_SPM.1, and therefore, with the following rationale, the requirements work together to accomplish the overall objectives defined for the TOE and its IT environment.

FMT_MSA.2 as defined in the Common Criteria as being dependent upon ADV_SPM.1. However, FMT_MSA.2 is included in this Security Target only because of a dependency of FCS_COP.1. In the case of the TOE described in this security target, the cryptographic mechanism does not require the entry of secure values and therefore there is no need to document otherwise applicable constraints for secure values in a security policy model (per ADV_SPM.1).

ST Requirement	CC Dependencies	ST Dependencies
FAU_GEN.1	FPT_STM.1	FPT_STM.1
FAU_GEN.2	FAU_GEN.1 and FIA_UID.1	FAU_GEN.1 and FIA_UID.1
FCS_COP_EX.1	(FDP_ITC.1 or FCS_CKM.1) and FCS_CKM.4 and FMT_MSA.2	FCS_CKM.1a and FCS_CKM.1b and FCS_CKM.4 and FMT_MSA.2
FDP_ACC.1	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1 and FMT_MSA.3	FDP_ACC.1 and FMT_MSA.3
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_ATD.1	none	none
FIA_UAU.1	FIA_UID.1	FIA_UID.1
FIA_UAU.5	none	none
FIA_UID.1	none	none
FIA_USB.1	FIA_ATD.1	FIA_ATD.1
FMT_MOF.1	FMT_SMR.1 and FMT_SMF.1	FMT_SMR.1 and FMT_SMF.1
FMT_MSA.1	FMT_SMR.1 and FMT_SMF.1 and (FDP_ACC.1 or FDP_IFC.1)	FMT_SMR.1 and FMT_SMF.1 and FDP_ACC.1
FMT_MSA.3	FMT_MSA.1 and FMT_SMR.1	FMT_MSA.1 and FMT_SMR.1
FMT_SMF.1	none	none
FMT_SMR.1	FIA_UID.1	FIA_UID.1
FPT_RVM.1a	none	none
FPT_SEP.1a	none	none
FAU_SAR.1	FAU_GEN.1	FAU_GEN.1
FAU_STG.1	FAU_GEN.1	FAU_GEN.1

ST Requirement	CC Dependencies	ST Dependencies
FCS_CKM.1a	(FCS_CKM.2 or FCS_COP.1) and FCS_CKM.4 and FMT_MSA.2	FCS_COP.1a and FCS_CKM.4 and FMT_MSA.2
FCS_CKM.1b	(FCS_CKM.2 or FCS_COP.1) and FCS_CKM.4 and FMT_MSA.2	FCS_COP.1b and FCS_CKM.4 and FMT_MSA.2
FCS_CKM.4	(FDP_ITC.1 or FCS_CKM.1) and FMT_MSA.2	FCS_CKM.1a and FCS_CKM.1b and FMT_MSA.2
FCS_COP.1a	(FDP_ITC.1 or FCS_CKM.1) and FCS_CKM.4 and FMT_MSA.2	FCS_CKM.1a and FCS_CKM.4 and FMT_MSA.2
FCS_COP.1b	(FDP_ITC.1 or FCS_CKM.1) and FCS_CKM.4 and FMT_MSA.2	FCS_CKM.1b and FCS_CKM.4 and FMT_MSA.2
FCS_COP.1c	(FDP_ITC.1 or FCS_CKM.1) and FCS_CKM.4 and FMT_MSA.2	FCS_CKM.1b and FCS_CKM.4 and FMT_MSA.2
FMT_MSA.2	ADV_SPM.1 and FMT_MSA.1 and FMT_SMR.1 and (FDP_ACC.1 or FDP_IFC.1)	[ADV_SPM.1 rationale above] and FMT_MSA.1 and FMT_SMR.1 and FDP_ACC.1
FPT_RVM.1b	none	none
FPT_SEP.1b	none	none
FPT_STM.1	none	none
ACM_CAP.2	none	none
ADO_DEL.1	none	none
ADO_IGS.1	AGD_ADM.1	AGD_ADM.1
ADV_FSP.1	ADV_RCR.1	ADV_RCR.1
ADV_HLD.1	ADV_FSP.1 and ADV_RCR.1	ADV_FSP.1 and ADV_RCR.1
ADV_RCR.1	none	none
AGD_ADM.1	ADV_FSP.1	ADV_FSP.1
AGD_USR.1	ADV_FSP.1	ADV_FSP.1
ALC_FLR.1	none	none
ATE_COV.1	ADV_FSP.1 and ATE_FUN.1	ADV_FSP.1 and ATE_FUN.1
ATE_FUN.1	none	none
ATE_IND.2	ADV_FSP.1 and AGD_ADM.1 and AGD_USR.1 and ATE_FUN.1	ADV_FSP.1 and AGD_ADM.1 and AGD_USR.1 and ATE_FUN.1
AVA_SOF.1	ADV_FSP.1 and ADV_HLD.1	ADV_FSP.1 and ADV_HLD.1
AVA_VLA.1	ADV_FSP.1 and ADV_HLD.1 and AGD_ADM.1 and AGD_USR.1	ADV_FSP.1 and ADV_HLD.1 and AGD_ADM.1 and AGD_USR.1

Table 6 Requirement Dependencies

8.6 Explicitly Stated Requirements Rationale

There is a single explicitly defined security functional requirement (SFR) in this Security Target: FCS_COP_EX.1. The requirement is explicitly stated in order to make it clear that a cryptographic function is invoked by the TOE in order to realize the applicable security-relevant result (i.e., encapsulation of traffic). The FCS SFRs in the CC are designed not only to identify a cryptographic operation, but also to specify the implementation (i.e., standard) corresponding to that operation. This explicit requirement is necessary because the TOE only invokes the operation, but does not implement it and there are no SFRs that allow the specification of only the invocation aspect. Note that the SFR has been modeled after SFR, but rather than specifying standards conformance indicates that the TOE invokes the IT environment function to perform specific cryptographic functions corresponding with specific algorithms with specific key sizes, as specified by the TOE. The explicit SFR is treated as an alternate version of FCS_COP and shares the same hierarchy and dependencies (as indicated in the table in the previous section).

8.7 TOE Summary Specification Rationale

Each subsection in Section 6, the TOE Summary Specification, describes a security function of the TOE. Each description is followed with rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The set of security functions work together to satisfy all of the security functions and assurance requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This Section in conjunction with Section 6, the TOE Summary Specification, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. **Table 7 Security Functions vs. Requirements Mapping** demonstrates the relationship between security requirements and security functions.

	Security audit	Cryptographic support	User data protection	Identification and authentication	Security management	Protection of the TSF
FAU_GEN.1	X					
FAU_GEN.2	X					
FCS_COP_EX.1		X				
FDP_ACC.1			X			
FDP_ACF.1			X			
FIA_AFL.1				X		
FIA_ATD.1				X		
FIA_UAU.1				X		
FIA_UAU.5				X		
FIA_UID.1				X		
FIA_USB.1				X		
FMT_MOF.1					X	
FMT_MSA.1					X	
FMT_MSA.3					X	
FMT_SMF.1					X	
FMT_SMR.1					X	
FPT_RVM.1a						X
FPT_SEP.1a						X

Table 7 Security Functions vs. Requirements Mapping

8.8 PP Claims Rationale

See Section 7, Protection Profile Claims.