Oracle® Fusion Middleware

Interoperability Guide for Oracle Web Services Manager

11*g* Release 1 (11.1.1.5)

E16098-05

April 2011

This document describes how to implement the most common Oracle WSM interoperability scenarios.

ORACLE

Oracle Fusion Middleware Interoperability Guide for Oracle Web Services Manager, 11*g* Release 1 (11.1.1.5)

E16098-05

Copyright © 2007, 2011, Oracle and/or its affiliates. All rights reserved.

Primary Author: Oracle Corporation

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

3 Interoperability with Oracle Containers for J2EE (OC4J) 10*g* Security Environments

4 Interoperability with Oracle WebLogic Server 11*g* Web Service Security Environments

5 Interoperability with Microsoft WCF/.NET 3.5 Security Environments

6 Interoperability with Oracle Service Bus 10*g* Security Environments

7 Interoperability with Axis 1.4 and WSS4J 1.5.8 Security Environments

8 Interoperability with Oracle GlassFish Enterprise Server Release 3.0.1

Preface

This preface describes the document accessibility features and conventions used in this guide—*Oracle Fusion Middleware Interoperability Guide for Oracle Web Services Manager*.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/support/contact.html` or visit `http://www.oracle.com/accessibility/support.html` if you are hearing impaired.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Overview of Oracle WSM Interoperability

This guide describes interoperability of Oracle Web Services Manager (Oracle WSM) with various security stacks. Each chapter includes the following information:

- Overview of each security stack
- An explanation of the usage scenarios

For details regarding limitations and known problems, see *Oracle Fusion Middleware Release Notes*.

1.1 About Oracle WSM Policies

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box.

For more details about the predefined policies, see "Predefined Policies" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

For information about configuring and attaching policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

1.2 Oracle WSM Interoperability Scenarios

Table 1–1 describes the most common Oracle WSM interoperability scenarios.

Table 1–1 Common Oracle WSM Interoperability Scenarios

Security Stack	Oracle WSM 11g Policies	Interoperability Scenario
Oracle WSM 10*g*	oracle/wss10_message_protection_service_policy oracle/wss10_message_protection_client_policy	"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 2-3
Oracle WSM 10*g*	oracle/wss10_username_token_with_message_protection_service_policy oracle/wss10_username_token_with_message_protection_client_policy	"Username Token with Message Protection (WS-Security 1.0)" on page 2-6
Oracle WSM 10*g*	oracle/wss10_saml_token_with_message_protection_service_policy oracle/wss10_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 2-9

Table 1–1 (Cont.) Common Oracle WSM Interoperability Scenarios

Security Stack	Oracle WSM 11g Policies	Interoperability Scenario
Oracle WSM 10*g*	oracle/wss10_x509_token_with_message_protection_service_policy oracle/wss10_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 2-12
Oracle WSM 10*g*	oracle/wss_username_token_over_ssl_service_policy oracle/wss_username_token_over_ssl_client_policy	"Username Token Over SSL" on page 2-14
Oracle WSM 10*g*	oracle/wss_saml_token_over_ssl_service_policy oracle/wss_saml_token_over_ssl_client_policy	"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 2-16
OC4J 10*g*	oracle/wss10_message_protection_service_policy oracle/wss10_message_protection_client_policy	"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 3-3
OC4J 10*g*	oracle/wss10_username_token_with_message_protection_service_policy oracle/wss10_username_token_with_message_protection_client_policy	"Username Token with Message Protection (WS-Security 1.0)" on page 3-6
OC4J 10*g*	oracle/wss10_saml_token_with_message_protection_service_policy oracle/wss10_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 3-10
OC4J 10*g*	oracle/wss10_x509_token_with_message_protection_service_policy oracle/wss10_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 3-13
OC4J 10*g*	oracle/wss_username_token_over_ssl_service_policy OR oracle/wss_saml_or_username_token_over_ssl_service_policy oracle/wss_username_token_over_ssl_client_policy	"Username token over SSL" on page 3-16
OC4J 10*g*	oracle/wss_saml_token_over_ssl_service_policy OR oracle/wss_saml_or_username_token_over_ssl_service_policy oracle/wss_saml_token_over_ssl_client_policy	"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 3-19
Oracle WebLogic Server 11*g*	oracle/wss11_username_token_with_message_protection_service_policy oracle/wss11_username_token_with_message_protection_client_policy	"Username Token With Message Protection (WS-Security 1.1)" on page 4-4

Table 1–1 (Cont.) Common Oracle WSM Interoperability Scenarios

Security Stack	Oracle WSM 11g Policies	Interoperability Scenario
Oracle WebLogic Server 11*g*	oracle/wss10_username_token_with_message_protection_service_policy oracle/wss10_username_token_with_message_protection_client_policy	"Username Token With Message Protection (WS-Security 1.0)" on page 4-7
Oracle WebLogic Server 11*g*	oracle/wss_username_token_over_ssl_service_policy	"Username Token Over SSL" on page 4-9
Oracle WebLogic Server 11*g*	oracle/wss_username_token_over_ssl_service_policy	"Username Token Over SSL with MTOM" on page 4-10
Oracle WebLogic Server 11*g*	oracle/wss_saml_token_over_ssl_service_policy	"SAML Token (Sender Vouches) Over SSL" on page 4-10
Oracle WebLogic Server 11*g*	oracle/wss11_saml20_token_with_message_protection_service_policy oracle/wss11_saml20_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) Over SSL with MTOM" on page 4-11
Oracle WebLogic Server 11*g*	oracle/wss11_saml20_token_with_message_protection_service_policy oracle/wss11_saml20_token_with_message_protection_client_policy	"SAML Token 2.0 (Sender Vouches) With Message Protection (WS-Security 1.1)" on page 4-12
Oracle WebLogic Server 11*g*	oracle/wss11_saml_token_with_message_protection_service_policy oracle/wss11_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15
Oracle WebLogic Server 11*g*	oracle/wss11_saml_token_with_message_protection_service_policy oracle/wss11_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1) and MTOM" on page 4-18
Oracle WebLogic Server 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy oracle/wss10_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 4-19
Oracle WebLogic Server 11*g*	oracle/wss10_x509_token_with_message_protection_service_policy oracle/wss10_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 4-22
Oracle WebLogic Server 11*g*	oracle/wss11_x509_token_with_message_protection_service_policy oracle/wss11_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 4-25
Microsoft WCF/.NET 3.5	oracle/wsmtom_service_policy oracle/wsmtom_client_policy	"Message Transmission Optimization Mechanism (MTOM)" on page 5-2
Microsoft WCF/.NET 3.5	oracle/wss11_username_token_with_message_protection_service_policy OR oracle/wss11_saml_or_username_token_with_message_protection_service_policy oracle/wss11_username_token_with_message_protection_client_policy	"Username Token With Message Protection (WS-Security 1.1)" on page 5-5

Table 1–1 (Cont.) Common Oracle WSM Interoperability Scenarios

Security Stack	Oracle WSM 11g Policies	Interoperability Scenario
Microsoft WCF/.NET 3.5	oracle/wss_saml_or_username_token_over_ssl_service_policy OR oracle/wss_username_token_over_ssl_service_policy	"Username Token Over SSL" on page 5-12
Microsoft WCF/.NET 3.5	oracle/wss11_x509_token_with_message_protection_service_policy oracle/wss11_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 5-14
Microsoft WCF/.NET 3.5	oracle/wss11_kerberos_with_message_protection_service_policy	"Kerberos with Message Protection" on page 5-20
Oracle Service Bus 10*g*	wss10_username_token_with_message_protection_service_policy wss10_username_token_with_message_protection_client_policy	"Username Token with Message Protection (WS-Security 1.0)" on page 6-2
Oracle Service Bus 10*g*	oracle/wss10_saml_token_with_message_protection_service_policy oracle/wss10_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 6-6
Oracle Service Bus 10*g*	oracle/wss_saml_or_username_token_over_ssl_service_policy	"SAML or Username Token Over SSL" on page 6-10
Oracle Service Bus 10*g*	oracle/wss10_x509_token_with_message_protection_service_policy oracle/wss10_x509_token_with_message_protection_client_policy	"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 6-13
Axis 1.4 and WSS4J 1.5.8	oracle/wss10_username_token_with_message_protection_service_policy oracle/wss10_username_token_with_message_protection_client_policy	"Username Token with Message Protection (WS-Security 1.0)" on page 7-3
Axis 1.4 and WSS4J 1.5.8	oracle/wss10_saml_token_with_message_protection_service_policy oracle/wss10_saml_token_with_message_protection_client_policy	"SAML Token with Message Protection (WS-Security 1.0)" on page 7-6
Axis 1.4 and WSS4J 1.5.8	oracle/wss_username_token_over_ssl_service_policy oracle/wss_username_token_over_ssl_client_policy	"Username Token Over SSL" on page 7-9
Axis 1.4 and WSS4J 1.5.8	oracle/wss_saml_token_over_ssl_service_policy oracle/wss_saml_token_over_ssl_client_policy	"SAML Token (Sender Vouches) Over SSL" on page 7-11
GlassFish Enterprise Server	oracle/wss11_username_token_with_message_protection_service_policy oracle/wss11_username_token_with_message_protection_client_policy	"Username Token with Message Protection (WS-Security 1.1)" on page 8-1

Table 1–1 (Cont.) Common Oracle WSM Interoperability Scenarios

Security Stack	Oracle WSM 11g Policies	Interoperability Scenario
GlassFish Enterprise Server	oracle/wss11_saml_token_with_message_protection_service_policy oracle/wss11_saml_token_with_message_protection_client_policy	"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 8-4

2

Interoperability with Oracle WSM 10g Security Environments

This chapter includes the following topics:

2.1 Overview of Interoperability with Oracle WSM 10*g* Security Environments

In Oracle WSM 10*g*, you specify *policy steps* at each policy enforcement point. The policy enforcement points in Oracle WSM 10*g* include Gateways and Agents.

Each policy step is a fine-grained operational task that addresses a specific security operation, such as authentication and authorization; encryption and decryption; security signature, token, or credential verification; and transformation. Each operational task is performed on either the Web service request or response. For more details about the Oracle WSM 10*g* policy steps, see "Oracle Web Services Manager Policy Steps" in *Oracle Web Services Manager Administrator's Guide 10g (10.1.3.4)* at `http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/policy_steps.htm#BABIAHEG`.

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see Predefined Policies. For information about configuring and attaching policies, see Configuring Policies and Attaching Policies to Web Services.

Table 2–1 summarizes the most common Oracle WSM 10*g* interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

For more information about:

- Oracle WSM 11*g* policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*
- Oracle WSM 10*g* policy steps, see "Oracle Web Services Manager Policy Steps" in *Oracle Web Services Manager Administrator's Guide 10g (10.1.3.4)* at `http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/policy_steps.htm#BABIAHEG`

Note: In the following scenarios, ensure that you are using a keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

Review "A Note About Oracle WSM 10g Gateways" on page 2-3 and "A Note About Third-party Software" on page 2-3 for important information about your usage of Oracle WSM 10g Gateways and third-party software.

Table 2–1 Interoperability With Oracle WSM 10g Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11g Policies	Oracle WSM 10g Policies
"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 2-3	Oracle WSM 10*g*—>Oracle WSM 11*g*	oracle/wss10_message_protection_service_policy	Request pipeline: Sign Message and Encrypt Response pipeline: Decrypt and Verify Signature
"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 2-3	Oracle WSM 11*g*—>Oracle WSM 10*g*	oracle/wss10_message_protection_client_policy	Request pipeline: Decrypt and Verify Signature Response pipeline: Sign Message and Encrypt
"Username Token with Message Protection (WS-Security 1.0)" on page 2-6	Oracle WSM 10*g*—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	Request pipeline: Sign Message and Encrypt Response pipeline: Decrypt and Verify Signature
"Username Token with Message Protection (WS-Security 1.0)" on page 2-6	Oracle WSM 11*g*—>Oracle WSM 10*g*	oracle/wss10_username_token_with_message_protection_client_policy	Request pipeline: ▪ Decrypt and Verify Signature ▪ Extract Credentials (configured as WS-BASIC) ▪ File Authenticate Response pipeline: Sign Message and Encrypt
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 2-9	Oracle WSM 10*g*—>Oracle WSM 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy	Request pipeline: ▪ Extract Credentials (configured as WS-BASIC ▪ SAML—Insert WSS 1.0 Sender-Vouches Token ▪ Sign and Encrypt Response pipeline: Decrypt and Verify Signature

Table 2–1 (Cont.) Interoperability With Oracle WSM 10g Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11g Policies	Oracle WSM 10g Policies
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 2-9	Oracle WSM 11*g*—>Oracle WSM 10*g*	oracle/wss10_saml_token_with_message_protection_client_policy	Request pipeline: ▪ XML Decrypt ▪ SAML—Verify WSS 1.0 Token Response pipeline: Sign Message and Encrypt
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 2-12	Oracle WSM 10*g*—>Oracle WSM 11*g*	oracle/wss10_x509_token_with_message_protection_service_policy	Request pipeline: Sign Message and Encrypt Response pipeline: Decrypt and Verify Signature
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 2-12	Oracle WSM 11*g*—>Oracle WSM 10*g*	oracle/wss10_x509_token_with_message_protection_client_policy	Request pipeline: Decrypt and Verify Response pipeline: Sign Message and Encrypt
"Username Token Over SSL" on page 2-14	Oracle WSM 10*g*—>Oracle WSM 11*g*	wss_username_token_over_ssl_service_policy	N/A
"Username Token Over SSL" on page 2-14	Oracle WSM 11*g*—>Oracle WSM 10*g*	wss_username_token_over_ssl_client_policy	Request pipeline: ▪ Extract Credentials ▪ File Authenticate
"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 2-16	Oracle WSM 10*g*—>Oracle WSM 11*g*	oracle/wss_saml_token_over_ssl_service_policy	Request pipeline: ▪ Extract Credentials ▪ SAML—Insert WSS 1.0 Sender-Vouches Token
"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 2-16	Oracle WSM 11*g*—>Oracle WSM 10*g*	oracle/wss_saml_token_over_ssl_client_policy	Request pipeline: ▪ Extract Credentials ▪ File Authenticate

The following sections provide additional interoperability information about using Oracle WSM 10*g* Gateways and third-party software with Oracle WSM 11*g*.

2.2 A Note About Oracle WSM 10*g* Gateways

As described in Examining the Rearchitecture of Oracle WSM in Oracle Fusion Middleware, Oracle Fusion Middleware 11*g* Release 1 (11.1.1) does not include a Gateway component. You can continue to use the Oracle WSM 10g Gateway components with Oracle WSM 10*g* policies in your applications.

2.3 A Note About Third-party Software

As described in Examining the Rearchitecture of Oracle WSM in Oracle Fusion Middleware, Oracle WSM 10*g* supports policy enforcement for third-party application servers, such as IBM WebSphere and Red Hat JBoss. Oracle Fusion Middleware 11*g* Release 1 (11.1.1) only supports Oracle WebLogic Server. You can continue to use the third-party application servers with Oracle WSM 10*g* policies.

2.4 Anonymous Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement anonymous authentication with message protection that conforms to the WS-Security 1.0 standard:

- "Configuring Oracle WSM 10g Client and Oracle WSM 11g Web Service" on page 2-4
- "Configuring Oracle WSM 11g Client and Oracle WSM 10g Web Service" on page 2-5

2.4.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.4.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the policy: oracle/wss10_message_protection_service_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

2. Edit the policy settings, as follows:
   a. Disable the Include Timestamp configuration setting.
   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Attach the policy to a Web service.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see Attaching Policies to Web Services. For more information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.4.1.2 Configuring Oracle WSM 10*g* Client

1. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at:
   `http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm`
2. Attach the following policy step to the request pipeline: Sign Message and Encrypt.
3. Configure the Sign Message and Encrypt policy step in the request pipeline, as follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.
4. Attach the following policy step to the response pipeline: Decrypt and Verify Signature.

5. Configure the Decrypt and Verify Signature policy step in the response pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
6. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.
7. Invoke the Web service.

2.4.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.4.2.1 Configuring Oracle WSM 10*g* Web Service

1. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
2. Attach the following policy step in the request pipeline: Decrypt and Verify Signature
3. Configure the Decrypt and Verify Signature policy step in the request pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
4. Attach the following policy step in the response pipeline: Sign Message and Encrypt
5. Configure the Sign Message and Encrypt policy response pipeline as follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.

2.4.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM Gateway.
2. Create a copy of the following policy: oracle/wss10_message_protection_client_policy.

> **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Disable the Include Timestamp configuration setting.

b. Leave the default configuration set for all other configuration settings.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Attach the policy to the Web service client.

For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Oracle WSM Policies to Web Service Clients" in JDeveloper Online Help.

4. Configure the policy, as described in "oracle/wss10_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

5. Invoke the Web service.

2.5 Username Token with Message Protection (WS-Security 1.0)

The following sections describe how to implement username token with message protection that conforms to the WS-Security 1.0 standard:

- "Configuring Oracle WSM 10g Client and Oracle WSM 11g Web Service" on page 2-6
- "Configuring Oracle WSM 11g Client and Oracle WSM 10g Web Service" on page 2-7

2.5.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.5.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the following policy: oracle/wss10_username_token_with_message_protection_service_policy.

Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Disable the Include Timestamp configuration setting.

b. Leave the default configuration set for all other configuration settings.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to a Web service.

For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more

information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.5.1.2 Configuring Oracle WSM 10*g* Client

1. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
2. Attach the following policy step to the request pipeline: Sign Message and Encrypt
3. Configure the Sign Message and Encrypt policy step in the request pipeline, as follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Set Encrypted Content to ENVELOPE.
   d. Set Signed Content to ENVELOPE.
   e. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.
4. Attach the following policy step to the response pipeline: Decrypt and Verify Signature.
5. Configure the Decrypt and Verify Signature policy step in the response pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
6. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.
7. Select the **Include Header** checkbox against WS-Security and provide valid credentials.
8. Invoke the Web service.

2.5.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.5.2.1 Configuring Oracle WSM 10*g* Web Service

1. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
2. Attach the following policy steps in the request pipeline:

   - Decrypt and Verify Signature

   - Extract Credentials (configured as WS-BASIC)

   - File Authenticate

> **Note:** You can substitute File Authenticate with LDAP Authenticate, Oracle Access Manager Authenticate, Active Directory Authenticate, or SiteMinder Authenticate.

3. Configure the Decrypt and Verify Signature policy step in the request pipeline, as follows:
   a. Configure the keystore properties for extracting credentials. The configuration should be in accordance with the keystore used on the server side.
4. Configure the Extract Credentials policy step in the request pipeline, as follows:
   a. Set the Credentials location to WS-BASIC.
5. Configure the File Authenticate policy step in the request pipeline to use valid credentials.
6. Attach the following policy step in the response pipeline: Sign Message and Encrypt.
7. Configure the Sign Message and Encrypt policy response pipeline, as follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.

2.5.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM Gateway.
2. Create a copy of the following policy: oracle/wss10_username_token_with_message_protection_client_policy.

   > **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:
   a. Disable the Include Timestamp configuration setting.
   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Attach the policy to the Web service client.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Oracle WSM Policies to Web Service Clients" in JDeveloper Online Help.

4. Configure the policy, as described in "oracle/wss10_username_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
5. Invoke the Web service.

2.6 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)

The following sections describe how to implement SAML token (sender vouches) with message protection that conforms to the WS-Security 1.0 standard:

- "Configuring Oracle WSM 10g Client and Oracle WSM 11g Web Service" on page 2-9
- "Configuring Oracle WSM 11g Client and Oracle WSM 10g Web Service" on page 2-10

2.6.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.6.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the following policy: oracle/wss10_saml_token_with_message_protection_service_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Disable the Include Timestamp configuration setting.
   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to the Web service.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.6.1.2 Configuring Oracle WSM 10g Client

1. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at:
   `http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm`
2. Attach the following policy steps in the request pipeline:

- Extract Credentials (configured as WS-BASIC)

- SAML—Insert WSS 1.0 Sender-Vouches Token

- Sign Message and Encrypt

3. Configure the Extract Credentials policy step in the request pipeline, as follows:
   a. Set the Credentials location to WS-BASIC.
4. Configure the SAML—Insert WSS 1.0 Sender-Vouches Token policy step in the request pipeline, as follows:
   a. Set Subject Name Qualifier to www.oracle.com.
   b. Set Assertion Issuer as www.oracle.com.
   c. Set Subject Format as UNSPECIFIED.
   d. Set other signing properties, as required.
5. Configure the Sign Message and Encrypt policy step in the request pipeline, as follows:
   a. Set the Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
6. Attach the following policy step in the response pipeline: Decrypt and Verify Signature.
7. Configure the Decrypt and Verify Signature policy step in the response pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
8. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.
9. Select **Include Header** checkbox against WS-Security and provide valid credentials.
10. Invoke the Web service.

2.6.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.6.2.1 Configuring Oracle WSM 10g Web Service

1. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
2. Attach the following policy steps in the request pipeline:

   - XML Decrypt

   - SAML—Verify WSS 1.0 Token

3. Configure the XML Decrypt policy step in the request pipeline, as follows:
   a. Configure the keystore properties for XML decryption. The configuration should be in accordance with the keystore used on the server side.
4. Configure the SAML—Verify WSS 1.0 Token policy step in the request pipeline, as follows:
   a. Set the Trusted Issuer Name as www.oracle.com.
5. Attach the following policy step in the response pipeline: Sign Message and Encrypt.
6. Configure the Sign Message and Encrypt policy step in the response pipeline, follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.

2.6.2.2 Configuring Oracle WSM 11g Client

1. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM Gateway.
2. Create a copy of the following policy: oracle/wss10_saml_token_with_message_protection_client_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:
   a. Disable the Include Timestamp configuration setting.
   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Attach the policy to the Web service client.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Oracle WSM Policies to Web Service Clients" in JDeveloper Online Help.
4. Configure the policy, as described in "oracle/wss10_saml_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
5. Invoke the Web service.

2.7 Mutual Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement mutual authentication with message protection that conform to the WS-Security 1.0 standards:

2.7.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.7.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the following policy: oracle/wss10_x509_token_with_message_protection_service_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Disable the Include Timestamp configuration setting.

   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to the Web service.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.7.1.2 Configuring Oracle WSM 10*g* Client

1. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at:
   `http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm`

2. Attach the following policy step in the request pipeline: Sign Message and Encrypt.

3. Configure the Sign Message and Encrypt policy step in the request pipeline, as follows:

   a. Set Encryption Algorithm to AES-128.

   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.

c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.

4. Attach the following policy step in the response pipeline: Decrypt and Verify Signature.
5. Configure the Decrypt and Verify Signature policy step in the response pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
6. Update the following property in the gateway-config-installer.properties file located at *ORACLE_HOME*/j2ee/*oc4j_instance*/applications/gateway/gateway/WEB-INF:

   ```
   pep.securitysteps.signBinarySecurityToken=true
   ```

7. Restart Oracle WSM 10*g* Gateway.
8. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.
9. Invoke the Web service.

2.7.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.7.2.1 Configuring Oracle WSM 10*g* Web Service

1. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
2. Attach the following policy steps in the request pipeline: Decrypt and Verify.
3. Configure the Decrypt and Verify Signature policy step in the request pipeline, as follows:
   a. Configure the keystore properties for decryption and signature verification. The configuration should be in accordance with the keystore used on the server side.
4. Attach the following policy steps in the response pipeline: Sign Message and Encrypt.
5. Configure the Sign Message and Encrypt policy step in the response pipeline, as follows:
   a. Set Encryption Algorithm to AES-128.
   b. Set Key Transport Algorithm to RSA-OAEP-MGF1P.
   c. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side.

2.7.2.2 Configuring Oracle WSM 11g Client

1. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM Gateway.
2. Create a copy of the following policy: oracle/wss10_x509_token_with_message_protection_client_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Disable the Include Timestamp configuration setting.

   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Attach the policy to the Web service client.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Configure the policy, as described in "oracle/wss10_x509_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
5. Invoke the Web service.

2.8 Username Token Over SSL

This section describes how to implement username token over SSL in the following scenarios:

- "Configuring Oracle WSM 10g Client and Oracle WSM 11g Web Service" on page 2-14
- "Configuring Oracle WSM 11g Client and Oracle WSM 10g Web Service" on page 2-15

For more information about:

- Configuring SSL on WebLogic Server, see Configuring SSL on WebLogic Server (One-Way) and Configuring SSL on WebLogic Server (Two-Way).
- Configuring SSL on OC4J, see `http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm`.

2.8.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.8.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for SSL.

For more information, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the following policy: wss_username_token_over_ssl_service_policy.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.8.1.2 Configuring Oracle WSM 10*g* Client

1. Configure the server for SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

2. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm

3. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.

4. Select the **Include Header** checkbox against WS-Security and provide valid credentials.

5. Invoke the Web service.

2.8.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.8.2.1 Configuring Oracle WSM 10*g* Web Service

1. Configure the server for SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

2. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm

3. Attach the following policy steps to the request pipeline:

   - Extract Credentials

   - File Authenticate

> **Note:** You can substitute File Authenticate with LDAP Authenticate, Oracle Access Manager Authenticate, Active Directory Authenticate, or SiteMinder Authenticate.

4. Configure the Extract Credentials policy step in the request pipeline, as follows:
   a. Configure the Credentials Location as WS-BASIC.
5. Configure the File Authentication policy step in the request pipeline with the appropriate credentials.

2.8.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM Gateway.

   Ensure that when generating the client, HTTP is specified in the URL along with the HTTP port number.
2. Create a copy of the following policy: oracle/wss_username_token_over_ssl_client_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:
   a. Disable the Include Timestamp configuration setting.
   b. Leave the default configuration set for all other configuration settings.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Attach the policy to the Web service client.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Oracle WSM Policies to Web Service Clients" in JDeveloper Online Help.
4. Configure the policy, as described in "oracle/wss_username_token_over_ssl_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
5. Invoke the Web service.

2.9 SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)

The following sections describe how to implement SAML token (sender vouches) over SSL that conforms to the WS-Security 1.0 standard:

- "Configuring Oracle WSM 10g Client and Oracle WSM 11g Web Service" on page 2-17
- "Configuring Oracle WSM 11g Client and Oracle WSM 10g Web Service" on page 2-18

For more information about:

- Configuring SSL on WebLogic Server, see Configuring SSL on WebLogic Server (One-Way) and Configuring SSL on WebLogic Server (Two-Way).

- Configuring SSL on OC4J, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

2.9.1 Configuring Oracle WSM 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WSM 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

2.9.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for two-way SSL.

   For more information, see Configuring SSL on WebLogic Server (Two-Way).

2. Create a copy of the following policy: oracle/wss_saml_token_over_ssl_service_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Disable the Include Timestamp configuration setting.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Attach the policy.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Policies to Web Services" in JDeveloper Online Help.

2.9.1.2 Configuring Oracle WSM 10g Client

1. Configure the server for two-way SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

2. Register the Web service (above) with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm

3. Attach the following policy steps to the request pipeline:

   - Extract Credentials

   - SAML—Insert WSS 1.0 Sender-Vouches Token

4. Configure the Extra Credentials policy step in the request pipeline, as follows:

   a. Configure the Credentials Location as WS-BASIC.

5. Configure the SAML—Insert WSS 1.0 Sender-Vouches Token policy step in the request pipeline, as follows:
   a. Configure the Subject Name Qualifier as www.oracle.com.
   b. Configure the Assertion Issuer as www.oracle.com.
   c. Configure the Subject Format as UNSPECIFIED.
   d. Configure the Sign the assertion as false.
6. Navigate to the Oracle WSM Test page and enter the virtualized URL of the Web service.
7. Select **Include Header** checkbox against WS-Security and provide valid credentials.
8. Invoke the Web service.

2.9.2 Configuring Oracle WSM 11*g* Client and Oracle WSM 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WSM 10*g* Web service, perform the following steps:

2.9.2.1 Configuring Oracle WSM 10*g* Web Service

1. Configure the server for two-way SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.
2. Register the Web service with the Oracle WSM 10*g* Gateway. See "Registering Web Services to an Oracle WSM Gateway" in the *Oracle WSM Administrator's Guide 10g* at: http://download.oracle.com/docs/cd/E12524_01/web.1013/e12575/gateways.htm
3. Attach the policy step: SAML—Verify WSS 1.0 Token
4. Configure the SAML—Verify WSS 1.0 Token policy step in the request pipeline, as follows:
   a. Under Signature Verification Properties, set Allow signed assertions only to false.
   b. Set the Trusted Issuer Name to www.oracle.com.

2.9.2.2 Configuring Oracle WSM 11*g* Client

1. Configure the server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. Create a client proxy using the virtualized URL of the Web service registered on the Oracle WSM gateway.
3. Create a copy of the following policy: oracle/wss_saml_token_over_ssl_client_policy.

> **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Disable the Include Timestamp configuration setting.

b. Leave the default configuration set for all other configuration settings.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4. Attach the policy to the Web service client.

   For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about attaching the policy at design time using JDeveloper, see "Attaching Oracle WSM Policies to Web Service Clients" in JDeveloper Online Help.

5. Configure the policy, as described in "oracle/wss_saml_token_over_ssl_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6. Invoke the Web service.

3

Interoperability with Oracle Containers for J2EE (OC4J) 10*g* Security Environments

This chapter describes the most common Oracle Containers for J2EE (OC4J) 10*g* interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

This chapter contains the following sections:

3.1 Overview of Interoperability with OC4J 10*g* Security Environments

In OC4J 10*g*, you configure your security environment.

- For information about using Application Server Control to configure the Web service, see *Oracle Application Server Advanced Web Services Developer's Guide* at `http://download.oracle.com/docs/cd/B31017_01/web.1013/b28975/toc.htm`.
- For information about using JDeveloper to develop and configure your client-side application, see the JDeveloper online help.
- For information about how to modify the XML-based deployment descriptor files, see *Oracle Application Server Web Services Security Guide 10g (10.1.3.1.0)* at: `http://download.oracle.com/docs/cd/B31017_01/web.1013/b28976/toc.htm`

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see "Predefined Policies" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For information about configuring and attaching policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

Table 3–1 summarizes the most common OC4J 10*g* interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

For information about configuring and attaching Oracle WSM 11*g* policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

Note: In the following scenarios, ensure that you are using a keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

Table 3–1 Interoperability With OC4J 10g Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	OC4J 10*g* Policies
"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 3-3	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss10_message_protection_service_policy	See "Configuring OC4J 10g Client" on page 3-3
"Anonymous Authentication with Message Protection (WS-Security 1.0)" on page 3-3	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss10_message_protection_client_policy	See "Configuring OC4J 10g Web Service" on page 3-4
"Username Token with Message Protection (WS-Security 1.0)" on page 3-6	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	See "Configuring OC4J 10g Client" on page 3-6
"Username Token with Message Protection (WS-Security 1.0)" on page 3-6	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss10_username_token_with_message_protection_client_policy	See "Configuring OC4J 10g Web Service" on page 3-8
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 3-10	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy	See "Configuring OC4J 10g Client" on page 3-10
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 3-10	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss10_saml_token_with_message_protection_client_policy	See "Configuring OC4J 10g Web Service" on page 3-11
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 3-13	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss10_x509_token_with_message_protection_service_policy	See "Configuring OC4J 10g Client" on page 3-13
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 3-13	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss10_x509_token_with_message_protection_client_policy	See "Configuring OC4J 10g Web Service" on page 3-15
"Username token over SSL" on page 3-16	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss_username_token_over_ssl_service_policy OR oracle/wss_saml_or_username_token_over_ssl_service_policy	See "Configuring OC4J 10g Client" on page 3-17
"Username token over SSL" on page 3-16	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss_username_token_over_ssl_client_policy	See "Configuring OC4J 10g Web Service" on page 3-18
"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 3-19	OC4J10*g*—>Oracle WSM 11*g*	oracle/wss_saml_token_over_ssl_service_policy OR oracle/wss_saml_or_username_token_over_ssl_service_policy	See "Configuring OC4J 10g Client" on page 3-20

Table 3–1 (Cont.) Interoperability With OC4J 10g Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	OC4J 10*g* Policies
"SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)" on page 3-19	Oracle WSM 11*g*—>OC4J10*g*	oracle/wss_saml_token_over_ssl_client_policy	See "Configuring OC4J 10g Web Service" on page 3-21

3.2 Anonymous Authentication with Message Protection (WS-Security 1.0)

This section describes how to implement anonymous authentication with message protection that conforms to the WS-Security 1.0 standard in the following scenarios:

- "Configuring OC4J 10g Client and Oracle WSM 11g Web Service" on page 3-3
- "Configuring Oracle WSM 11g Client and OC4J 10g Web Service" on page 3-4

3.2.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.2.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service application.
2. Attach the following policy to the entry point of the Web service: oracle/wss10_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.2.1.2 Configuring OC4J 10*g* Client

1. Create a client proxy for the Web service (above) using Oracle JDeveloper.
2. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.
3. Click **Authentication** in the Proxy Editor navigation bar and set the following options:
   - Select **No Authentication**.
4. Click **Inbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Verify Inbound Signed Request Body**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
   - Select all options under **Acceptable Signature Algorithms**.
5. Click **Outbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Sign Outbound Messages**.
   - Select **Add Timestamp to Outbound Messages** and **Creation Time Required in Timestamp**.

- Enter the **Expiration Time** (in seconds).

6. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Decrypt Inbound Message Content**.
   - Select all options under **Acceptable Signature Algorithms**.
7. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Encrypt Outbound Messages**.
   - Set the Algorithm to **AES-128**.
8. Click **Keystore Options** in the Proxy Editor navigation bar and configure the keystore properties, as required.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
9. Click **OK** to close the wizard.
10. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in next section.
11. Invoke the Web service method from the client.

Editing the <appname>Binding_Stub.xml File

1. Provide the keystore password and sign and encryption key passwords.
2. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd" local-part="Timestamp" />
...
```

3. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

4. In the outbound encryption, specify the key transport algorithm, as follows:

```
<outbound><encrypt>
<keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
...
```

3.2.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.2.2.1 Configuring OC4J 10*g* Web Service

1. Create and deploy a Web service application.

2. Use Application Server Control to secure the deployed Web service.
3. Click **Authentication** tab and ensure that no options are selected.
4. Click **Integrity** tab of the Inbound Policies page and set the following options:
   - Select **Require Message Body to Be Signed**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
5. Click **Integrity** tab of the Outbound Policies page and set the following options:
   - Select **Sign Body Element of Message**.
   - Set the **Signature Method** to **RSA-SHA1**.
   - Select **Add Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
6. Click **Confidentiality** tab of the Inbound Policies page and set the following options:
   - Select **Require Encryption of Message Body**.
7. Click **Confidentiality** tab of the Outbound Policies page and set the following options:
   - Select **Encrypt Body Element of Message**.
   - Set the **Encryption Method** to **AES-128**.
   - Set the public key to encrypt.
8. Configure the keystore properties and identity certificates.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
9. Edit the wsmgmt.xml deployment descriptor file, as described in Editing the wsmgmt.xml File.

3.2.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy for the OC4J 10*g* Web service.
2. Attach the following policy: oracle/wss10_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in "oracle/wss10_username_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Invoke the Web service method from the client.

Editing the wsmgmt.xml File

Edit the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
```

```
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

2. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

3. In the outbound encryption, specify the key transport algorithm, as follows:

```
<outbound><encrypt>
<keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
...
```

3.3 Username Token with Message Protection (WS-Security 1.0)

The following sections describe how to implement username token with message protection that conforms to the WS-Security 1.0 standard:

- "Configuring OC4J 10g Client and Oracle WSM 11g Web Service" on page 3-6
- "Configuring Oracle WSM 11g Client and OC4J 10g Web Service" on page 3-8

3.3.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.3.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create an Oracle WSM 11*g* Web service.
2. Attach the following policy to the Web service: oracle/wss10_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.3.1.2 Configuring OC4J 10*g* Client

1. Create a client proxy for the Web service (above) using Oracle JDeveloper.
2. Specify the username and password in the client proxy, as follows:

```
port.setUsername(<username>)
port.setPassword(<password>)
```

3. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.
4. Click **Authentication** in the Proxy Editor navigation bar and set the following options:
   - Select **Use Username to Authenticate**.
   - Deselect **Add Nonce** and **Add Creation Time**.

5. Click **Inbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Verify Inbound Signed Request Body**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
   - Select all options under **Acceptable Signature Algorithms**.
6. Click **Outbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Sign Outbound Messages**.
   - Select **Add Timestamp to Outbound Messages** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
7. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Decrypt Inbound Message Content**.
   - Select all options under **Acceptable Signature Algorithms**.
8. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Encrypt Outbound Messages**.
   - Set the Algorithm to **AES-128**.
9. Click **Keystore Options** in the Proxy Editor navigation bar and configure the keystore properties, as required.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
10. Click **OK** to close the wizard.
11. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in Editing the <appname>Binding_Stub.xml File.
12. Invoke the Web service.

Editing the <appname>Binding_Stub.xml File

Edit the <appname>Binding_Stub.xml file, as follows:

1. Provide the keystore password and sign and encryption key passwords.
2. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp" />
...
```

3. In the outbound signature, specify that the timestamp and UsernameToken should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
```

```
utility-1.0.xsd" local-part="Timestamp"/>
 <tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-secext-1.0.xsd" local-part="UsernameToken"/>
...
```

4. In the outbound encryption, specify the key transport algorithm, as follows:

```
<outbound><encrypt>
<keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
...
```

5. In the outbound encryption, specify that the UsernameToken should be encrypted, as follows:

```
<outbound>/<encrypt>/<tbe-elements>
<tbe-element local-part="UsernameToken"
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-secext-1.0.xsd" mode="CONTENT"/>
...
```

3.3.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.3.2.1 Configuring OC4J 10*g* Web Service

1. Create and deploy a JAX-RPC Web service on OC4J.
2. Use Application Server Control to secure the deployed Web service.
3. Click **Authentication** tab and set the following options:
   - Select **Use Username/Password Authentication**.
   - Set **Password** to **Plain Text**.
4. Click **Integrity** tab in Inbound Policies page and set the following options:
   - Select **Require Message Body to Be Signed**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
5. Click **Integrity** tab in Outbound Policies page and set the following options:
   - Select **Sign Body Element of Message**.
   - Set the **Signature Method** to **RSA-SHA1**.
   - Select **Add Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
6. Click **Confidentiality** tab in the Inbound Policies page and set the following options:
   - Select **Require Encryption of Message Body**.
7. Click **Confidentiality** tab in the Outbound Policies page and set the following options:
   - Select **Encrypt Body Element of Message**.

- Set the **Encryption Method** to **AES-128**.
- Set the public key to encrypt.

8. Configure the keystore properties and identity certificates.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

9. Edit the wsmgmt.xml deployment descriptor file, as described in Editing the wsmgmt.xml File.

3.3.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy for the OC4J 10*g* Web service.
2. Attach the following policy: oracle/wss10_username_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Configure the policy, as described in "oracle/wss10_username_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Invoke the Web service method from the client.

Editing the wsmgmt.xml File

Edit the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

2. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

3. In the outbound encryption, specify that the UsernameToken should be encrypted, as follows:

```
<outbound>/<encrypt>/<tbe-elements>
<tbe-element local-part="UsernameToken"
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-secext-1.0.xsd" mode="CONTENT"/>
...
```

3.4 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)

The following sections describe how to implement SAML token sender vouches with message protection that conforms to the WS-Security 1.0 standard:

3.4.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.4.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create an Oracle WSM 11g Web service.
2. Attach the following policy to the Web service: oracle/wss10_saml_token__with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.4.1.2 Configuring OC4J 10*g* Client

1. Create a client proxy for the Web service (above) using Oracle JDeveloper.
2. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.
3. Click **Authentication** in the Proxy Editor navigation bar and set the following options:
   - Select **Use SAML Token**.
   - Click **SAML Details**.
   - Select **Sender Vouches Confirmation** and **Use Signature**.
   - Enter the username that needs to be propagated as the **Default Subject Name**.
   - Enter www.oracle.com as the **Default Issuer Name**.
4. Click **Inbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Verify Inbound Signed Request Body**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
   - Select all options under **Acceptable Signature Algorithms**.
5. Click **Outbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Sign Outbound Messages**.
   - Select **Add Timestamp to Outbound Messages** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).

6. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Decrypt Inbound Message Content**.
   - Select all options under **Acceptable Signature Algorithms**.
7. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Encrypt Outbound Messages**.
   - Set the Algorithm to **AES-128**.
8. Click **Keystore Options** in the Proxy Editor navigation bar and configure the keystore properties, as required.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
9. Click **OK** to close the wizard.
10. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in Editing the <appname>Binding_Stub.xml File.
11. Invoke the Web service method.

Editing the <appname>Binding_Stub.xml File

Edit the <appname>Binding_Stub.xml file, as follows:

1. Provide the keystore password and sign and encryption key passwords.
2. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp" />
...
```

3. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

4. In the outbound encryption, specify the key transport algorithm, as follows:

```
<outbound><encrypt>
<keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
...
```

3.4.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.4.2.1 Configuring OC4J 10*g* Web Service

1. Create and deploy a JAX-RPC Web service on OC4J.

2. Use the Application Server Control to secure the deployed Web service.
3. Click **Authentication** in navigation bar and set the following options:
   - Select **Use SAML Authentication**.
   - Select **Accept Sender Vouches**.
   - Deselect **Verify Signature**.
4. Click **Inbound Integrity** in the navigation bar and set the following option:
   - Select **Require Message Body To Be Signed**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
5. Click **Outbound Integrity** in the navigation bar and select the following options:
   - Select **Sign Body Element of Message**.
   - Set the **Signature Method** to **RSA-SHA1**.
   - Select **Add Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
6. Click **Inbound Confidentiality** in the navigation bar and set the following option:
   - Deselect **Require Encryption of Message Body**.
7. Click **Outbound Confidentiality** in the navigation bar and set the following option:
   - Select **Encrypt Body Element of Message**.
   - Set the **Encryption Method** to **AES-128**.
   - Set the public key to encrypt.
8. Configure the keystore properties and identity certificates.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

   For more information, see the Oracle Fusion Middleware Administrator's Guide.
9. Edit the wsmgmt.xml deployment descriptor file, as described in Editing the wsmgmt.xml File.
10. Invoke the Web service.

3.4.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy for the OC4J 10*g* Web service.
2. Attach the following policy: oracle/wss10_saml_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in "oracle/wss10_saml_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Invoke the Web service method from the client.

Editing the wsmgmt.xml File

Edit the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

2. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

3. In the outbound encryption, specify that the UsernameToken should be encrypted, as follows:

```
<outbound>/<encrypt>/<tbe-elements>
<tbe-element local-part="UsernameToken"
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-secext-1.0.xsd" mode="CONTENT"/>
...
```

3.5 Mutual Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement mutual authentication with message protection that conforms to the WS-Security 1.0 standard:

- "Configuring OC4J 10g Client and Oracle WSM 11g Web Service" on page 3-13
- "Configuring Oracle WSM 11g Client and OC4J 10g Web Service" on page 3-15

3.5.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.5.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service application.
2. Attach the following policy to the Web service: oracle/wss10_x509_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.5.1.2 Configuring OC4J 10*g* Client

1. Create a client proxy for the Web service (above) using Oracle JDeveloper.
2. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.

3. Click **Authentication** in the Proxy Editor navigation bar and set the following options:
   - Select **Use X509 To Authenticate**.
4. Click **Inbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Verify Inbound Signed Request Body**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
   - Select all options under **Acceptable Signature Algorithms**.
5. Click **Outbound Integrity** in the Proxy Editor navigation bar and set the following options:
   - Select **Sign Outbound Messages**.
   - Select **Add Timestamp to Outbound Messages** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
6. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Decrypt Inbound Message Content**.
   - Select all options under **Acceptable Signature Algorithms**.
7. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and set the following options:
   - Select **Encrypt Outbound Messages**.
   - Set the Algorithm to **AES-128**.
8. Click **Keystore Options** in the Proxy Editor navigation bar and configure the keystore properties, as required.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
9. Click **OK** to close the wizard.
10. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in Editing the <appname>Binding_Stub.xml File.
11. Invoke the Web service.

Editing the <appname>Binding_Stub.xml File

Edit the <appname>Binding_Stub.xml file, as follows:

1. Provide the keystore password and sign and encryption key passwords.
2. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp" />
...
```

3. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

4. In the outbound encryption, specify the key transport algorithm, as follows:

```
<outbound><encrypt>
<keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
...
```

3.5.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.5.2.1 Configuring OC4J 10*g* Web Service

1. Create and deploy a JAX-RPC Web service on OC4J.
2. Use the Application Server Control to secure the deployed Web service.
3. Click **Authentication** taband set the following options:
   - Select **Use X509 Certificate Authentication**.
4. Click **Integrity** tab of the Inbound Policies page and set the following options:
   - Select **Require Message Body to Be Signed**.
   - Select **Verify Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
5. Click **Integrity** tab of the Outbound Policies page and set the following options:
   - Select **Sign Body Element of Message**.
   - Set the **Signature Method** to **RSA-SHA1**.
   - Select **Add Timestamp** and **Creation Time Required in Timestamp**.
   - Enter the **Expiration Time** (in seconds).
6. Click **Confidentiality** tab of the Inbound Policies page and set the following options:
   - Select **Require Encryption of Message Body**.
7. Click **Confidentiality** tab of the Outbound Policies page and set the following options:
   - Select **Encrypt Body Element of Message**.
   - Set the **Encryption Method** to **AES-128**.
   - Set the public key to encrypt.
8. Configure the keystore properties and identity certificates.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

9. Edit the wsmgmt.xml deployment descriptor file, as described in Editing the wsmgmt.xml File.

3.5.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy to the OC4J 10*g* Web service.
2. Attach the following policy: oracle/wss10_x509_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in "oracle/wss10_x509_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Invoke the Web service.

Editing the wsmgmt.xml File

Edit the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the inbound signature, specify the following:

```
<inbound><verify-signature><tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

2. In the outbound signature, specify that the timestamp should be signed, as follows:

```
<outbound>/<signature>/<tbs-elements>
<tbs-element
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-utility-1.0.xsd" local-part="Timestamp"/>
...
```

3. In the outbound encryption, specify that the UsernameToken should be encrypted, as follows:

```
<outbound>/<encrypt>/<tbe-elements>
<tbe-element local-part="UsernameToken"
name-space="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity
-secext-1.0.xsd" mode="CONTENT"/>
...
```

3.6 Username token over SSL

The following sections describe how to implement username token over SSL:

- "Configuring OC4J 10g Client and Oracle WSM 11g Web Service" on page 3-17
- "Configuring Oracle WSM 11g Client and OC4J 10g Web Service" on page 3-18

For information about:

- Configuring SSL on WebLogic Server, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
- Configuring SSL on OC4J, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

3.6.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.6.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach one of the following policies to the Web service:

   oracle/wss_username_token_over_ssl_service_policy

   oracle/wss_username_or_saml_token_over_ssl_service_policy

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.6.1.2 Configuring OC4J 10*g* Client

1. Create a client proxy for the Web service (above) using Oracle JDeveloper.

   Ensure that the Web service endpoint references the URL with HTTPS and SSL port configured on Oracle WebLogic Server.

2. Add the following code excerpt to initialize two-way SSL (at the beginning of the client proxy code):

```
HostnameVerifier hv = new HostnameVerifier()
httpsURLConnection.setDefaultHostnameVerifier(hv);
System.setProperty("javax.net.ssl.trustStore","<trust_store>");
System.setProperty("javax.net.ssl.trustStorePassword","<trust_store
_password>");
System.setProperty("javax.net.ssl.keyStore","<key_store>");
System.setProperty("javax.net.ssl.keyStorePassword","<key_store_password>");
System.setProperty("javax.net.ssl.keyStoreType","JKS");
```

3. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.
4. Click **Authentication** in the Proxy Editor navigation bar and set the following options:
   - Select **Use Username to Authenticate**.
   - Deselect **Add Nonce** and **Add Creation Time**.
5. Click **Inbound Integrity** in the Proxy Editor navigation bar and deselect all options.

6. Click **Outbound Integrity** in the Proxy Editor navigation bar and deselect all options.
7. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and deselect all options.
8. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and deselect all options.
9. Click **Keystore Options** in the Proxy Editor navigation bar and configure the keystore properties, as required.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
10. Click **OK** to close the wizard.
11. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in Editing the <appname>Binding_Stub.xml File.
12. Invoke the Web service.

Editing the <appname>Binding_Stub.xml File

Edit the <appname>Binding_Stub.xml file, as follows:

1. Provide the keystore password and sign and encryption key passwords.
2. In the outbound signature, specify that the timestamp should be signed, as follows (and remove all other tags):

```
<outbound>
   <signature>
      <add-timestamp created="true" expiry="<Expiry_Time>"/>
   </signature>
...
```

3.6.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.6.2.1 Configuring OC4J 10*g* Web Service

1. Configure the server for SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.
2. Use the Application Server Control to secure the deployed Web service.
3. Click **Authentication** tab and set the following options:
   - Select **Use Username/Password Authentication**.
4. Click **Integrity** tab of the Inbound Policies page and deselect all options.
5. Click **Integrity** tab of the Outbound Policies page and deselect all options.
6. Click **Confidentiality** tab of the Inbound Policies page and deselect all options.
7. Click **Confidentiality** tab of the Outbound Policies page and deselect all options.

8. Edit the wsmgmt.xml deployment descriptor file, as described in Editing the wsmgmt.xml File.

3.6.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy to the OC4J 10*g* Web service using clientgen.

   Ensure that the Web service endpoint references the URL with HTTPS and SSL port configured on Oracle WebLogic Server.

2. Add the following code excerpt to initialize two-way SSL (at the beginning of the client proxy code):

```
HostnameVerifier hv = new HostnameVerifier()
httpsURLConnection.setDefaultHostnameVerifier(hv);
System.setProperty("javax.net.ssl.trustStore","<trust_store>");
System.setProperty("javax.net.ssl.trustStorePassword","<trust_store
_password>");
System.setProperty("javax.net.ssl.keyStore","<key_store>");
System.setProperty("javax.net.ssl.keyStorePassword","<key_store_password>");
System.setProperty("javax.net.ssl.keyStoreType","JKS");
```

3. Attach the following policy: oracle/wss_username_token_over_ssl_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4. Configure the policy, as described in "oracle/wss_username_token_over_ssl_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

5. Invoke the Web service.

Editing the wsmgmt.xml File

Edit the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the outbound signature, specify that the timestamp should be signed, as follows (and remove all other tags):

```
<outbound>
   <signature>
      <add-timestamp created="true" expiry="<Expiry_Time>"/>
   </signature>
...
```

3.7 SAML Token (Sender Vouches) Over SSL (WS-Security 1.0)

The following sections describe how to implement SAML token (sender vouches) over SSL that conforms to the WS-Security 1.0 standard:

- "Configuring OC4J 10g Client and Oracle WSM 11g Web Service" on page 3-20
- "Configuring Oracle WSM 11g Client and OC4J 10g Web Service" on page 3-21

For information about:

- Configuring SSL on WebLogic Server, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

- Configuring SSL on OC4J, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

3.7.1 Configuring OC4J 10*g* Client and Oracle WSM 11*g* Web Service

To configure OC4J 10*g* client and Oracle WSM 11*g* Web service, perform the following steps:

3.7.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the following policy to the Web service:

   oracle/wss_saml_token_over_ssl_service_policy OR

   oracle/wss_username_or_saml_token_over_ssl_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3.7.1.2 Configuring OC4J 10*g* Client

1. Configure the server for two-way SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.

2. Create a client proxy for the Web service (above) using Oracle JDeveloper.

   Ensure that the Web service endpoint references the URL with HTTPS and SSL port configured on Oracle WebLogic Server.

3. Add the following code excerpt to initialize two-way SSL (at the beginning of the client proxy code):

```
HostnameVerifier hv = new HostnameVerifier()
httpsURLConnection.setDefaultHostnameVerifier(hv);
System.setProperty("javax.net.ssl.trustStore","<trust_store>");
System.setProperty("javax.net.ssl.trustStorePassword","<trust_store
_password>");
System.setProperty("javax.net.ssl.keyStore","<key_store>");
System.setProperty("javax.net.ssl.keyStorePassword","<key_store_password>");
System.setProperty("javax.net.ssl.keyStoreType","JKS");
```

4. Use the Oracle JDeveloper wizard to secure the proxy by right-clicking on the proxy project and selecting **Secure Proxy**.

5. Click **Authentication** in the Proxy Editor navigation bar and set the following options:

   - Select **Use SAML Token**.
   - Click **SAML Details**.
   - Select **Sender Vouches Confirmation**.
   - Enter a valid username as the **Default Subject Name**.

6. Click **Inbound Integrity** in the Proxy Editor navigation bar and set the following option:
   - Deselect **Verify Inbound Signed Message Body**.
7. Click **Outbound Integrity** in the Proxy Editor navigation bar and deselect all options.
8. Click **Inbound Confidentiality** in the Proxy Editor navigation bar and set the following option:
   - Deselect **Decrypt Inbound Message Content**.
9. Click **Outbound Confidentiality** in the Proxy Editor navigation bar and set the following option:
   - Deselect **Encrypt Outbound Message**.
10. Provide required information for the keystore to be used.
11. Click **OK** to close the wizard.
12. In the Structure pane, click **<appname>Binding_Stub.xml** and edit the file as described in Editing the <appname>Binding_Stub.xml File.
13. Invoke the Web service.

Editing the <appname>Binding_Stub.xml File

Edit the <appname>Binding_Stub.xml file, as follows:

1. Provide the keystore password and sign and encryption key passwords.
2. In the outbound signature, specify that the timestamp should be signed, as follows (and remove all other tags):

```
<outbound>
   <signature>
      <add-timestamp created="true" expiry="<Expiry_Time>"/>
   </signature>
...
```

3.7.2 Configuring Oracle WSM 11*g* Client and OC4J 10*g* Web Service

To configure Oracle WSM 11*g* client and OC4J 10*g* Web service, perform the following steps:

3.7.2.1 Configuring OC4J 10*g* Web Service

1. Configure the server for two-way SSL.

   For more information, see http://download.oracle.com/docs/cd/B14099_19/web.1012/b14013/configssl.htm.
2. Use the Application Server Control to secure the deployed Web service.
3. Click **Authentication** in navigation bar and set the following options:
   - Select **Use SAML Authentication**.
   - Select **Accept Sender Vouches**.
   - Deselect **Verify Signature**.
4. Click **Integrity** tab of the Inbound Policies page and deselect all options.

5. Click **Integrity** tab of the Outbound Policies page and deselect all options.
6. Click **Confidentiality** tab of the Inbound Policies page and deselect all options.
7. Click **Confidentiality** tab of the Outbound Policies page and deselect all options.
8. Edit the wsmgmt.xml deployment descriptor file, as described in Edit the wsmgmt.xml File.

3.7.2.2 Configuring Oracle WSM 11*g* Client

1. Configure the server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Create a client proxy to the OC4J 10*g* Web service.

   Ensure that the Web service endpoint references the URL with HTTPS and SSL port configured on Oracle WebLogic Server.

3. Attach the following policy: oracle/wss_saml_token_over_ssl_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4. Configure the policy, as described in "oracle/wss_saml_token_over_ssl_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

5. Invoke the Web service.

Edit the wsmgmt.xml File

Editing the wsmgmt.xml file in *ORACLE_HOME*/j2ee/*oc4j_instance*/config, as follows:

1. In the outbound signature, specify that the timestamp should be signed, as follows (and remove all other tags):

```
<outbound>
   <signature>
      <add-timestamp created="true" expiry="<Expiry_Time>"/>
   </signature>
...
```

4

Interoperability with Oracle WebLogic Server 11*g* Web Service Security Environments

This chapter contains the following sections:

4.1 Overview of Interoperability with Oracle WebLogic Server 11*g* Web Service Security Environments

In Oracle Fusion Middleware 11*g*, you can attach both Oracle WSM and Oracle WebLogic Server Web service policies to WebLogic Java EE Web services.

For more details about the predefined Oracle WSM 11*g* policies, see the following sections in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*:

- "Attaching Policies to Web Services"
- "Configuring Policies"
- "Predefined Policies"

For more details about the predefined Oracle WebLogic Server 11*g* Web service policies, see:

- "Attaching Policies to WebLogic Web Services and Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*
- *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

Table 4–1 summarizes the most common Oracle WebLogic Server 11*g* Web service policy interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

Table 4–1 Interoperability With Oracle WebLogic Server 11g Web Services Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	Oracle WebLogic Server 11*g* Policies
"Username Token With Message Protection (WS-Security 1.1)" on page 4-4	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss11_username_token_with_message_protection_service_policy	▪ Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token With Message Protection (WS-Security 1.1)" on page 4-4	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_username_token_with_message_protection_client_policy	▪ Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token With Message Protection (WS-Security 1.1) and MTOM" on page 4-6	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	▪ Wssp1.2-wss10_username_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token With Message Protection (WS-Security 1.1) and MTOM" on page 4-6	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_username_token_with_message_protection_client_policy	▪ Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token With Message Protection (WS-Security 1.0)" on page 4-7	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	▪ Wssp1.2-wss10_username_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token With Message Protection (WS-Security 1.0)" on page 4-7	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss10_username_token_with_message_protection_client_policy	▪ Wssp1.2-wss10_username_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Username Token Over SSL" on page 4-9	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss_username_token_over_ssl_service_policy	Wssp1.2-2007-Https-UsernameToken-Plain.xml
"Username Token Over SSL with MTOM" on page 4-10	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss_username_token_over_ssl_service_policy	Wssp1.2-2007-Https-UsernameToken-Plain.xml
"SAML Token (Sender Vouches) Over SSL" on page 4-10	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss_saml_token_over_ssl_service_policy	Wssp1.2-2007-Saml1.1-Sender Vouches-Https.xml

Table 4–1 (Cont.) Interoperability With Oracle WebLogic Server 11g Web Services Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	Oracle WebLogic Server 11*g* Policies
"SAML Token (Sender Vouches) Over SSL with MTOM" on page 4-11	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss_saml_token_over_ssl_service_policy	Wssp1.2-2007-Saml1.1-Sender Vouches-Https.xml
"SAML Token 2.0 (Sender Vouches) With Message Protection (WS-Security 1.1)" on page 4-12	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss11_saml_token_with_message_protection_service_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token 2.0 (Sender Vouches) With Message Protection (WS-Security 1.1)" on page 4-12	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_saml_token_with_message_protection_client_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss11_saml_token_with_message_protection_service_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_saml_token_with_message_protection_client_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1) and MTOM" on page 4-18	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss11_saml_token_with_message_protection_service_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1) and MTOM" on page 4-18	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_saml_token_with_message_protection_client_policy wsmtom_policy	▪ Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 4-19	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy	▪ Wssp1.2-wss10_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml

Table 4–1 (Cont.) Interoperability With Oracle WebLogic Server 11g Web Services Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	Oracle WebLogic Server 11*g* Policies
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 4-19	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss10_saml_token_with_message_protection_client_policy	▪ Wssp1.2-wss10_saml_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 4-22	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss10_x509_token_with_message_protection_service_policy	▪ Wssp1.2-wss10_x509_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 4-22	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss10_x509_token_with_message_protection_client_policy	▪ Wssp1.2-wss10_x509_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 4-25	Oracle WebLogic Server 11*g*—>Oracle WSM 11*g*	oracle/wss11_x509_token_with_message_protection_service_policy	▪ Wssp1.2-wss11_x509_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml
"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 4-25	Oracle WSM 11*g*—>Oracle WebLogic Server 11*g*	oracle/wss11_x509_token_with_message_protection_client_policy	▪ Wssp1.2-wss11_x509_token_with_message_protection_owsm_policy.xml ▪ Wssp1.2-2007-SignBody.xml ▪ Wssp1.2-2007-EncryptBody.xml

4.2 Username Token With Message Protection (WS-Security 1.1)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.1 standard in the following interoperability scenarios:

- "Configuring JSE Client Using Oracle WebLogic Server 11g Security Policy and Oracle WSM 11g Web Service" on page 4-4
- "Configuring the Client and Oracle WebLogic Server 11g Web Service" on page 4-5

4.2.1 Configuring JSE Client Using Oracle WebLogic Server 11*g* Security Policy and Oracle WSM 11*g* Web Service

To configure a JSE (or JEE) client that uses Oracle WebLogic Server 11*g* security policy and Oracle WSM 11*g* Web service, perform the following steps:

4.2.1.1 Configuring Oracle WSM 11g Web Service

1. Create a Web service.

2. Attach the following policy to the Web service: oracle/wss11_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.2.1.2 Configuring the Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*

2. Attach the following policies:
   - Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml
3. Provide the configuration for the server (encryption key) in the client, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the encryption key configured for the Web service.
4. Invoke the Web service method from the client.

4.2.2 Configuring the Client and Oracle WebLogic Server 11*g* Web Service

To configure a JSE (or JEE) client that uses Oracle WSM 11*g* policies and Oracle WebLogic Server 11*g* Web service, perform the following steps:

4.2.2.1 Configuring Oracle WebLogic Server 11*g* Web Service

1. Create a Web service.
2. Attach the following policies:
   - Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Configure message-level security, as described in:

   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   You only need to configure the Confidentiality Key for a WS-Security 1.1 policy.

5. Deploy the Web service.

   See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

4.2.2.2 Configuring the Client

1. Create a client proxy to the Web service (above).
2. Attach the following policy to the Web service client: oracle/wss11_username_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in "oracle/wss11_username_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Specify keystore.recipient.alias in the client configuration.
5. Ensure that the keystore.recipient.alias keys specified for the client exist as trusted certificate entry in the trust store configured for the Web service.
6. Provide a valid username and password as part of the configuration.
7. Invoke the Web service method from the client.

4.3 Username Token With Message Protection (WS-Security 1.1) and MTOM

This section describes how to implement username token with message protection that conforms to the WS-Security 1.1 standard and uses Message Transmission Optimization Mechanism (MTOM) in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service"
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service"

4.3.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

1. Configure the Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service as described in "Username Token With Message Protection (WS-Security 1.1)" on page 4-4.
2. To enable MTOM communication, use the @MTOM annotation in the Web service in Step 2 of "Configuring the Client" on page 4-5.

4.3.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

1. Configure the Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service as described in "Username Token With Message Protection (WS-Security 1.1)" on page 4-4.
2. To enable MTOM communication, perform one of the following:
   - Use the @MTOM annotation in the Web service in Step 2 of "Configuring Oracle WebLogic Server 11g Web Service" on page 4-5.
   - In Step 3 of "Configuring the Client" on page 4-6, attach wsmtom_policy from the Management tab.

4.4 Username Token With Message Protection (WS-Security 1.0)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-7
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-8

> **Note:** WS-Security 1.0 policy is supported for legacy applications only. Use WS-Security 1.1 policy for maximum performance. For more information, see "Username Token With Message Protection (WS-Security 1.1)" on page 4-4.

4.4.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.4.1.1 Configuring Oracle WSM 11g Web Service

1. Create a Web service.
2. Attach the following policy to the Web service: oracle/wss10_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.4.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*
2. Attach the following policies:
   - Wssp1.2-wss10_username_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

3. Configure the client for server (encryption key) and client certificates, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the decryption key configured for the Web service.

4. Invoke the Web service method from the client.

4.4.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

4.4.2.1 Configuring Oracle WebLogic Server 11g Web Service

1. Create a Web service.
2. Attach the following policies:
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-wss10_username_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Configure message-level security, as described in:

   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
5. Deploy the Web service.

   See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

4.4.2.2 Configuring Oracle WSM 11g Client

1. Create a client proxy to the Web service (above).
2. Attach the following policy to the Web service client: oracle/wss10_username_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in Oracle Fusion Middleware Security and Administrator's Guide for Web Services.
3. Configure the policy, as described in "oracle/wss10_username_token_with_message_protection_client_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Ensure that you use different keys for client (sign and decrypt key) and keystore recipient alias (server public key used for encryption). Ensure that the recipient

alias is in accordance with the keys defined in the Web service policy security configuration.

5. Ensure that the signing and encryption keys specified for the client exist as trusted certificate entries in the trust store configured for the Web service.
6. Provide a valid username and password as part of the configuration.
7. Invoke the Web service method from the client.

4.5 Username Token Over SSL

The following section describes how to implement username token over SSL, describing the following interoperability scenario:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-9

4.5.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.5.1.1 Configuring Oracle WSM 11g Web Service

1. Configure the server for one-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. Create a Web service.
3. Attach the following policy: oracle/wss_username_token_over_ssl_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.5.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a client proxy for the Web service (above) using clientgen. Provide a valid username and password as part of the configuration for this policy in the client proxy.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*.
2. Configure WebLogic Server for SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Attach Wssp1.2-2007-Https-UsernameToken-Plain.xml to the Web service client.
5. Provide the truststore and other required System properties in the SSL client, as described in "Using SSL Authentication in Java Clients" in *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.

6. Invoke the Web service.

4.6 Username Token Over SSL with MTOM

The following section describes how to implement username token over SSL with Message Transmission Optimization Mechanism (MTOM) in the following interoperability scenario:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-10

4.6.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

1. Configure the Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service as described in "Username Token Over SSL" on page 4-9.
2. To enable MTOM communication, use the @MTOM annotation in the Web service in Step 4 of "Configuring Oracle WebLogic Server 11g Client" on page 4-9.

4.7 SAML Token (Sender Vouches) Over SSL

The following section describes how to implement SAML token sender vouches with SSL. It describes the following interoperability scenario:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-10

4.7.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.7.1.1 Configuring Oracle WSM 11g Web Service

1. Configure the oracle/wss_saml_token_over_ssl_service_policy policy for two-way SSL, as described in "oracle/wss_saml_token_over_ssl_service_policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. Create a Web service.
3. Attach the following policy to the Web service: oracle/wss_saml_token_over_ssl_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.7.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*.

2. Configure Oracle WebLogic Server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Configure identity and trust stores, as described in "Configure Identity and Trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*

4. Attach Wssp1.2-2007-Saml1.1-SenderVouches-Https.xml to the Web service client.

5. Configure a SAML credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAMLCredentialMapperV2.

   Select the new provider, click on Provider Specific, and configure it as follows:

   a. Set Issuer URI to www.oracle.com.

   b. Set Name Qualifier to www.oracle.com.

6. Restart Oracle WebLogic Server.

7. Create a SAML relying party, as described in "Create a SAML 1.1 Relying Party" and "Configure a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Set the Profile to WSS/Sender-Vouches.

8. Configure the SAML relying party, as described in "Configure a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Configure the SAML relying party as follows (leave other values set to the defaults):

   - Target URL: <url_used_to_access_Web_service>
   - Description: <your_description>

   Select the Enabled checkbox and click **Save**.

   Ensure the Target URL is set to the URL used for the client Web service.

9. Create a servlet and call the proxy code from the servlet.

10. Use BASIC authentication so that the authenticated subject can be created.

11. Provide the truststore and other required System properties in the SSL client, as described in "Using SSL Authentication in Java Clients" in *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.

12. Invoke the Web application client.

    Enter the credentials of the user whose identity is to be propagated using the SAML token.

4.8 SAML Token (Sender Vouches) Over SSL with MTOM

The following section describes how to implement SAML token sender vouches over SSL with MTOM. It describes the following interoperability scenario:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-12

4.8.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

1. Configure the Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service as described in "SAML Token (Sender Vouches) Over SSL" on page 4-10.
2. To enable MTOM communication, use the @MTOM annotation in the Web service in Step 4 of "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-10.

4.9 SAML Token 2.0 (Sender Vouches) With Message Protection (WS-Security 1.1)

This section describes how to implement SAML 2.0 token sender vouches with message protection that conforms to the WS-Security 1.1 standard in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-12
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-14

4.9.1 Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.9.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a JAX-WS Web service.
2. Attach the following policy to the Web service: oracle/wss11_saml20_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.9.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a J2EE client for the deployed Web service using JDeveloper. Create a Web project and create a proxy using WSDL proxy.
2. Attach the following policies:
   - Wssp1.2-2007-Saml2.0-SenderVouches-Wss1.1.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

Extract weblogic.jar to a folder and provide the absolute path to the above policies files.

3. Add servlet to above web project.
4. Configure the client for server (encryption key) and client certificates, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the decryption key configured for the Web service.
5. Secure the Web application client using BASIC Authentication. For more information, see "Developing BASIC Authentication Web Applications" in *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.
6. Deploy the J2EE Web application client.

   See "Deploying Web Services Applications" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
7. Configure a SAML credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the Oracle WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAML2CredentialMapper.

   Select the new provider, click on Provider Specific, and configure it as follows:

   a. Set Issuer URI to www.oracle.com.
   b. Set Name Qualifier to www.oracle.com.
8. Restart WebLogic Server.
9. To create a new service provider partner, perform the following steps:

   a. Select the credential mapper created in Step 7 in the WebLogic Administration Console, and then select the Management tab.
   b. Select New, and then select New Webservice Service Provider Partner.
   c. Provide a name, and select Finish.
10. Configure the service provider partner as follows:

    a. Select the service provide partner created in Step 9.
    b. Select the Enabled check box.
    c. Provide the Audience URI.
    d. Set Issuer URI to www.oracle.com.
    e. Set Target URL to <url_used_to_access_Web_service>.
    f. Set Profile to WSS/Sender-Vouches.
11. Invoke the Web application client.

    Enter the credentials of the user whose identity is to be propagated using SAML token.

4.9.2 Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

4.9.2.1 Configuring Oracle WebLogic Server 11g Web Service

1. Create a Web service.
2. Attach the following policies:
   - Wssp1.2-2007-Saml2.0-SenderVouches-Wss1.1.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure the keystore properties for message signing and encryption. The configuration should be in accordance with the keystore used on the server side. Create the trust store out of the keystore by exporting both keys, and trust both of them while importing into trust store. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
4. Configure message-level security, as described in:
   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Since this is a WS-Security 1.1 policy, you need to configure Confidentiality Key only.
5. Attach new configuration using the annotation:

   @WssConfiguration(value="my_security_configuration") where my_security_configuration is the name of the Web Security Configuration created in Step 4. For more information, see "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
6. Deploy the Web service.

   See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.
7. Create a SAML Identity Asserter, as described in "Configuring Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAML2IdentityAsserter.
8. Restart WebLogic Server.
9. To add the identity provider to the identity assertor created in Step 7, perform the following steps:

a. Select the identity assertor created in Step 7 in the WebLogic Administration Console.

b. Create a new identity provider partner, select New, and then select New Webservice Identity Provider Partner.

c. Provide a name, and select Finish.

10. Configure the identity provider as follows:

a. Select the identity provide partner created in Step 9.

b. Select the Enabled check box.

c. Provide the Audience URI. For example: target:*:/saml20WLSWS-Project1-context-root/Class1Port

d. Set Issuer URI to www.oracle.com.

e. Set Target URL to <url_used_to_access_Web_service>.

f. Set Profile to WSS/Sender-Vouches.

4.9.2.2 Configuring Oracle WSM 11g Client

1. Generate a client using JDeveloper for the Web service created in "Configuring Oracle WSM 11g Client" on page 4-15. Create a Web project and then select New, and create a client proxy using the WSDL.
2. Add a servlet in the above project.
3. Attach the following policy to the Web service client: oracle/wss11_saml20_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Specify keystore.recipient.alias in the client configuration.

   Ensure that keystore.recipient.alias is the same as the decryption key specified for the Web service.
5. Ensure that the keystore.recipient.alias keys specified for the client exist as trusted certificate entry in the trust store configured for the Web service.
6. In JDeveloper, secure web project with Form-based authentication using the Configure ADF Security Wizard.
7. Invoke the Web application client.

4.10 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)

This section describes how to implement SAML token sender vouches with message protection that conforms to the WS-Security 1.1 standard in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-16
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-17

4.10.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.10.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service.
2. Attach the following policy to the Web service: oracle/wss11_saml_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.10.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*
2. Attach the following policies:
   - Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml
3. Configure the client for server (encryption key) and client certificates, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the decryption key configured for the Web service.
4. Secure the Web application client using BASIC Authentication. For more information, see "Developing BASIC Authentication Web Applications" in *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.
5. Deploy the Web service client.

   See "Deploying Web Services Applications" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
6. Configure a SAML credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the Oracle WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAMLCredentialMapperV2.

   Select the new provider, click on Provider Specific, and configure it as follows:

   a. Set Issuer URI to www.oracle.com.

   b. Set Name Qualifier to www.oracle.com.
7. Restart WebLogic Server.

8. Create a SAML relying party, as described in "Create a SAML 1.1 Relying Party" and "Configure a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Set the Profile to WSS/Sender-Vouches.

9. Configure the SAML relying party, as described in "Configure a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Ensure the Target URL is set to the URL used for the client Web service.

10. Invoke the Web application client.

    Enter the credentials of the user whose identity is to be propagated using SAML token.

4.10.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

4.10.2.1 Configuring Oracle WebLogic Server 11g Web Service

1. Create a Web service.
2. Attach the following policies:
   - Wssp1.2-wss11_saml_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Configure message-level security, as described in:
   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*
   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Since this is a WS-Security 1.1 policy, you need to configure Confidentiality Key only.
5. Deploy the Web service.

   See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.
6. Create a SAMLIdentityAsserterV2 authentication provider, as described in "Configuring Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAMLCredentialMapperV2.

7. Restart WebLogic Server.
8. Select the authentication provider created in step 5.
9. Create a SAML asserting party, as described in "Create a SAML 1.1 Asserting Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Set Profile to WSS/Sender-Vouches.
10. Configure the SAML asserting party, as described in "Configure a SAML 1.1 Asserting Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

    Configure the SAML asserting party as follows:

    a. Set Issuer URI to www.oracle.com.

    b. Set Target URL to <url_used_to_access_Web_service>.

4.10.2.2 Configuring Oracle WSM 11g Client

1. Create a client proxy to the Web service (above).
2. Attach the following policy to the Web service client: oracle/wss11_saml_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in oracle/wss11_saml_token_with_message_protection_client_policy.
4. Specify keystore.recipient.alias in the client configuration.

   Ensure that keystore.recipient.alias is the same as the decryption key specified for the Web service.
5. Ensure that the keystore.recipient.alias keys specified for the client exist as trusted certificate entry in the trust store configured for the Web service.
6. Provide a valid username whose identity needs to be propagated using SAML token in the client configuration.
7. Invoke the Web application client.

   Enter the credentials of the user whose identity is to be propagated using SAML token.

4.11 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1) and MTOM

This section describes how to implement SAML token with sender vouches and message protection that conforms to the WS-Security 1.1 standard and uses Message Transmission Optimization Mechanism (MTOM) in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-19
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-19

4.11.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

1. Configure the Oracle WebLogic Server 11g client and Oracle WSM 11g Web service as described in "SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15.
2. To enable MTOM communication, use the @MTOM annotation in the Web service in Step 2 of "Configuring Oracle WebLogic Server 11g Client" on page 4-16.

4.11.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

1. Configure the Oracle WSM 11*g* client and Oracle WebLogic Server 11g Web service as described in "SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15.
2. To enable MTOM communication, perform one of the following:
   - Use the @MTOM annotation in the Web service in Step 2 of "Configuring Oracle WebLogic Server 11g Web Service" on page 4-17.
   - In Step 2 of "Configuring Oracle WSM 11g Client" on page 4-18, attach wsmtom_policy from the Management tab.

4.12 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)

This section describes how to implement SAML token with sender vouches and message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-19
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-21

> **Note:** WS-Security 1.0 policy is supported for legacy applications only. Use WS-Security 1.1 policy for maximum performance. For more information, see "SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)" on page 4-15.

4.12.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the following steps:

4.12.1.1 Configuring Oracle WSM 11g Web Service

1. Create a Web service.

2. Attach the following policy to the Web service: oracle/wss10_saml_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.12.1.2 Configuring Oracle WebLogic Server 11g Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*

2. Attach the following policies:
   - Wssp1.2-wss10_saml_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml
3. Configure the client for server (encryption key) and client certificates, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the decryption key configured for the Web service.
4. Secure the Web application client using BASIC Authentication. For more information, see "Developing BASIC Authentication Web Applications" in *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.
5. Deploy the Web service client.

   See "Deploying Web Services Applications" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
6. Configure a SAML credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAMLCredentialMapperV2.
7. Select the SAMLCredentialMapperV2, click on Provider Specific, and configure it as follows:
   a. Set Issuer URI to www.oracle.com.
   b. Set Name Qualifier to www.oracle.com.
8. Restart WebLogic Server.
9. Create a SAML relying party, as described in "Create a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Set the profile to WSS/Sender-Vouches.
10. Configure the SAML relying party, as described in "Configure a SAML 1.1 Relying Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

    Ensure the target URL is set to the URL used for the client Web service.

11. Invoke the Web application client and enter the appropriate credentials.

4.12.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the following steps:

4.12.2.1 Configuring Oracle WebLogic Server 11g Web Service

1. Create a Web service.
2. Attach the following policies:
   - Wssp1.2-wss10_saml_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Configure message-level security, as described in:
   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*
   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Since this is a WS-Security 1.1 policy, you need to configure Confidentiality Key only.
5. Deploy the Web service.

   See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.
6. Create a SAMLIdentityAsserterV2 authentication provider, as described in "Configuring Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   In the WebLogic Server Administration Console, navigate to Security Realms > RealmName > Providers > Credential Mapping page and create a New Credential Mapping Provider of type SAMLCredentialMapperV2.
7. Restart WebLogic Server.
8. Select the authentication provider created in step 5.
9. Create a SAML asserting party, as described in "Create a SAML 1.1 Asserting Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
   - Set Profile to WSS/Sender-Vouches.
10. Configure a SAML asserting party, as described in "Configure a SAML 1.1 Asserting Party" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

Configure the SAML asserting party as follows (leave other values set to the defaults):

a. Set Issuer URI to www.oracle.com.

b. Set Target URL to <url_used_by_client>.

4.12.2.2 Configuring Oracle WSM 11g Client

1. Create a client proxy to the Web service (above).
2. Attach the following policy to the Web service client: oracle/wss10_saml_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Configure the policy, as described in oracle/wss10_saml_token_with_message_protection_client_policy.
4. Ensure that you use different keys for client (sign and decrypt key) and keystore recipient alias (server public key used for encryption). Ensure that the recipient alias is in accordance with the keys defined in the Web service policy security configuration.
5. Ensure that the signing and encryption keys specified for the client exist as trusted certificate entries in the trust store configured for the Web service.
6. Provide valid username whose identity needs to be propagated using SAML token in the client configuration.
7. Invoke the Web service method.

4.13 Mutual Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement mutual authentication with message protection that conform to the WS-Security 1.0 standards:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-22
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-23

4.13.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 10*g* client and Oracle WSM 11*g* Web service, perform the steps in the following sections.

4.13.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service.
2. Attach the following policy to the Web service: oracle/wss10_x509_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.13.1.2 Configuring Oracle WebLogic Server 11*g* Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*

2. Attach the following policies:
   - Wssp1.2-wss10_x509_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml
3. Provide the configuration for the server (encryption key) in the client, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the encryption key configured for the Web service.
4. Invoke the Web service method from the client.

4.13.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the steps in the following sections.

4.13.2.1 Configuring Oracle WebLogic Server 11*g* Web Service

1. Create a JAX-WS Web service.
2. Attach the following policies:
   - Wssp1.2-wss10_x509_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
4. Configure message-level security, as described in:

   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*

   You only need to configure the Confidentiality Key for a WS-Security 1.0 policy.
5. Create and configure token handlers for X.509 and for username token. In WebLogic Administration Console, navigate to the Web Service Security page of the domain and create the token handlers as follows:
   - Create a token handle for username token and configure the following:
     - Name: <name>

  - Class name: weblogic.xml.crypto.wss.UsernameTokenHandler
  - Token Type: ut
  - Handling Order: 1
- Create a token handler for X.509 and configure the following:
  - Name: <name>
  - Class name: weblogic.xml.crypto.wss.BinarySecurityTokenHandler
  - Token Type: x509
  - Handling Order: 0
- For the X.509 token handler, add the following properties:
  - Name: UserX509ForIdentity
  - Value: true
  - IsEncrypted: False

  For more information on token handlers, see "Create a token handler of a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

6. Configure a credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Create a PKICredentialMapper and configure it as follows (leave all other values set to the defaults):

   - Keystore Provider: N/A
   - Keystore Type: jks
   - Keystore File Name: default_keystore.jks
   - Keystore Pass Phrase: <password>
   - Confirm Keystore Pass Phrase: <password>

7. Configure Authentication, as described in "Configure Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select the **Authentication** tab and configure as follows:

   - Click **DefaultIdentityAsserter** and add **X.509** to **Chosen** active types
   - Click **Provider Specific** and configure the following:
     - Default User Name Mapper Attribute Type: CN
     - Active Types: X.509
     - Use Default User Name Mapper: True

8. If the users are not added, add the Common Name (CN) user specified in the certificate as described in "Create users" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

9. Restart Oracle WebLogic Server.

10. Deploy the Web service.

    See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

4.13.2.2 Configuring Oracle WSM 10*g* Client

1. Create a client proxy for the Web service using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*.

2. Attach the following policy to the client: wss10_x509_token_with_message_protection_client_policy

3. Provide the configuration for the server (encryption key) in the client, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the encryption key configured for the Web service.

4. Invoke the Web service method from the client.

4.14 Mutual Authentication with Message Protection (WS-Security 1.1)

The following sections describe how to implement mutual authentication with message protection that conform to the WS-Security 1.1 standards:

- "Configuring Oracle WebLogic Server 11g Client and Oracle WSM 11g Web Service" on page 4-22
- "Configuring Oracle WSM 11g Client and Oracle WebLogic Server 11g Web Service" on page 4-23

4.14.1 Configuring Oracle WebLogic Server 11*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle WebLogic Server 11*g* client and Oracle WSM 11*g* Web service, perform the steps in the following sections.

4.14.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a JAX-WS Web service.

2. Attach the following policy to the Web service: oracle/wss11_x509_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4.14.1.2 Configuring Oracle WebLogic Server 11*g* Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*

2. Attach the following policies:

   - Wssp1.2-wss11_x509_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

3. Provide the configuration for the server (encryption key) in the client, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the encryption key configured for the Web service.

4. Invoke the Web service method from the client.

4.14.2 Configuring Oracle WSM 11*g* Client and Oracle WebLogic Server 11*g* Web Service

To configure Oracle WSM 11*g* client and Oracle WebLogic Server 11*g* Web service, perform the steps in the following sections:

4.14.2.1 Configuring Oracle WebLogic Server 11*g* Web Service

1. Create a JAX-WS Web service.
2. Attach the following policies:
   - Wssp1.2-wss11_x509_token_with_message_protection_owsm_policy.xml
   - Wssp1.2-2007-SignBody.xml
   - Wssp1.2-2007-EncryptBody.xml

   For more information, see "Updating the JWS File with @Policy and @Policies Annotations" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.
3. Configure identity and trust stores, as described in "Configure identity and trust" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*
4. Configure message-level security, as described in:

   - "Configuring Message-Level Security" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*

   - "Create a Web Service security configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   You only need to configure the Confidentiality Key for a WS-Security 1.1 policy.
5. Create and configure token handlers for X.509 and for username token. In WebLogic Administration Console, navigate to the Web Service Security page of the domain and create the token handlers as follows:
   - Create a token handle for username token and configure the following:
     - Name: <name>
     - Class name: weblogic.xml.crypto.wss.UsernameTokenHandler
     - Token Type: ut
     - Handling Order: 1
   - Create a token handler for X.509 and configure the following:
     - Name: <name>
     - Class name: weblogic.xml.crypto.wss.BinarySecurityTokenHandler
     - Token Type: x509
     - Handling Order: 0

- For the X.509 token handler, add the following properties:
  - Name: UserX509ForIdentity
  - Value: true
  - IsEncrypted: False

  For more information on token handlers, see "Create a token handler of a Web Service security configuration" in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help.

6. Configure a credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Create a PKICredentialMapper and configure it as follows (leave all other values set to the defaults):

   - Keystore Provider: N/A
   - Keystore Type: jks
   - Keystore File Name: default_keystore.jks
   - Keystore Pass Phrase: <password>
   - Confirm Keystore Pass Phrase: <password>

7. Configure Authentication, as described in "Configure Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select the **Authentication** tab and configure as follows:

   - Click **DefaultIdentityAsserter** and add **X.509** to **Chosen** active types
   - Click **Provider Specific** and configure the following:
     - Default User Name Mapper Attribute Type: CN
     - Active Types: X.509
     - Use Default User Name Mapper: True

8. If the users are not added, add the Common Name (CN) user specified in the certificate as described in "Create users" in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help.
9. Restart Oracle WebLogic Server.
10. Deploy the Web service.

    See *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*.

4.14.2.2 Configuring Oracle WSM 11*g* Client

1. Create a client proxy for the Web service (above) using clientgen.

   For more information, see "Using the clientgen Ant Task to Generate Client Artifacts" in *Oracle Fusion Middleware Getting Started With JAX-WS Web Services for Oracle WebLogic Server*

2. Attach the following policy to the client: wss11_x509_token_with_message_protection_client_policy

Note: Edit the policy as follows:

```
 <orasp:x509-token
orasp:sign-key-ref-mech="thumbprint"

orasp:enc-key-ref-mech="thumbprint"/>
```

3. Provide the configuration for the server (encryption key) in the client, as described in "Updating a Client Application to Invoke a Message-Secured Web Service" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server*.

   Ensure that the encryption key specified is in accordance with the encryption key configured for the Web service.

4. Invoke the Web service method from the client.

5

Interoperability with Microsoft WCF/.NET 3.5 Security Environments

This chapter contains the following sections:

5.1 Overview of Interoperability with Microsoft WCF/.NET 3.5 Security Environments

In conjunction with Microsoft, Oracle has performed interoperability testing to ensure that the Web service security policies created using Oracle WSM 11*g* can interoperate with Web service policies configured using Microsoft Windows Communication Foundation (WCF)/.NET 3.5 Framework and vice versa.

For more information about Microsoft WCF/.NET 3.5 Framework, see `http://msdn.microsoft.com/en-us/netframework/aa663324.aspx`.

For more details about the predefined Oracle WSM 11*g* policies, see the following topics in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*:

- "Attaching Policies to Web Services"
- "Configuring Policies"
- "Predefined Policies"

Table 5–1 summarizes the most common Microsoft .NET 3.5 interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

Note: In the following scenarios, ensure that you are using a keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

In addition, ensure that the keys use the proper extensions, including DigitalSignature, Non_repudiation, Key_Encipherment, and Data_Encipherment.

Table 5–1 Interoperability With Microsoft WCF/.NET 3.5 Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11g Policies	Microsoft WCF/.NET 3.5 Policies
"Message Transmission Optimization Mechanism (MTOM)" on page 5-2	Microsoft WCF/.NET 3.5—>Oracle WSM 11*g*	oracle/wsmtom_service_policy	See "Configuring Microsoft WCF/.NET 3.5 Client" on page 5-3
"Message Transmission Optimization Mechanism (MTOM)" on page 5-2	Oracle WSM 11*g*—>Microsoft WCF/.NET 3.5	oracle/wsmtom_client_policy	See "Configuring Microsoft WCF/.NET 3.5 Web Service" on page 5-4
"Username Token With Message Protection (WS-Security 1.1)" on page 5-5	Microsoft WCF/.NET 3.5—>Oracle WSM 11*g*	oracle/wss11_username_token_with_message_protection_service_policy OR oracle/wss11_saml_or_username_token_with_message_protection_service_policy	See "Configuring Microsoft WCF/.NET 3.5 Client" on page 5-6
"Username Token With Message Protection (WS-Security 1.1)" on page 5-5	Oracle WSM 11*g*—>Microsoft WCF/.NET 3.5	oracle/wss11_username_token_with_message_protection_client_policy	See "Configuring Microsoft WCF/.NET 3.5 Web Service" on page 5-9
"Username Token Over SSL" on page 5-12	Microsoft WCF/.NET 3.5—>Oracle WSM 11*g*	oracle/wss_saml_or_username_token_over_ssl_service_policy OR oracle/wss_username_token_over_ssl_service_policy	See "Configuring Microsoft WCF/.NET 3.5 Client" on page 5-13
"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 5-14	Microsoft WCF/.NET 3.5—>Oracle WSM 11*g*	oracle/wss11_x509_token_with_message_protection_service_policy	See "Configuring Microsoft WCF/.NET 3.5 Client" on page 5-15
"Mutual Authentication with Message Protection (WS-Security 1.1)" on page 5-14	Oracle WSM 11*g*—>Microsoft WCF/.NET 3.5	oracle/wss11_x509_token_with_message_protection_client_policy	See "Configuring Microsoft WCF/.NET 3.5 Web Service" on page 5-18
"Kerberos with Message Protection" on page 5-20	Microsoft WCF/.NET 3.5—>Oracle WSM 11*g*	oracle/wss11_kerberos_with_message_protection_service_policy	See "Configuring Microsoft WCF/.NET 3.5 Client" on page 5-21

5.2 Message Transmission Optimization Mechanism (MTOM)

This section describes how to implement MTOM in the following interoperability scenarios:

- "Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service" on page 5-3
- "Configuring Oracle WSM 11g Client and Microsoft WCF/.NET 3.5 Web Service" on page 5-4

5.2.1 Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11*g* Web Service

To configure Microsoft WCF/.NET 3.5 client and Oracle WSM 11g Web service, perform the steps described in the following sections:

5.2.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service application.
2. Attach the following policy to the Web service: oracle/wsmotom_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Deploy the application.

5.2.1.2 Configuring Microsoft WCF/.NET 3.5 Client

1. Use the SVCUtil utility to create a client proxy and configuration file from the deployed Web service. See "Example app.config File for MTOM Interoperability" on page 5-3.
2. Run the client program.

Example app.config File for MTOM Interoperability

The following provides an example of the app.config file:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.serviceModel>
        <bindings>
            <customBinding>
                <binding name="CustomBinding_IMTOMService">
                    <mtomMessageEncoding maxReadPoolSize="64"
                     maxWritePoolSize="16"
                        messageVersion="Soap12" maxBufferSize="65536"
                        writeEncoding="utf-8">
                        <readerQuotas maxDepth="32" maxStringContentLength=
                         "8192" maxArrayLength="16384"
                            maxBytesPerRead="4096" maxNameTableCharCount="16384" />
                    </mtomMessageEncoding>
                    <httpTransport manualAddressing="false" maxBufferPoolSize="524288"
                        maxReceivedMessageSize="65536" allowCookies="false"
                           authenticationScheme="Anonymous"
                        bypassProxyOnLocal="false" hostNameComparisonMode="StrongWildcard"
                        keepAliveEnabled="true" maxBufferSize="65536"
                           proxyAuthenticationScheme="Anonymous"
                        realm="" transferMode="Buffered"
                           unsafeConnectionNtlmAuthentication="false"
                        useDefaultWebProxy="true" />
                </binding>
            </customBinding>
        </bindings>
        <client>
          <endpoint address="<endpoint_url>"
              binding="customBinding" bindingConfiguration="CustomBinding_IMTOMService"
              contract="IMTOMService" name="CustomBinding_IMTOMService" >
          </endpoint>
        </client>
    </system.serviceModel>
</configuration>
```

5.2.2 Configuring Oracle WSM 11*g* Client and Microsoft WCF/.NET 3.5 Web Service

To configure Oracle WSM 11*g* client and Microsoft WCF/.NET 3.5 Web service, perform the steps described in the following sections:

5.2.2.1 Configuring Microsoft WCF/.NET 3.5 Web Service

1. Create a .NET Web service.

   For more information, see "How to: Define a Windows Communication Foundation Service Contract" at `http://msdn.microsoft.com/en-us/library/ms731835.aspx`.

   For an example of a .NET Web service, see "Example of .NET Web Service for MTOM Interoperability" on page 5-4.

2. Deploy the application.

5.2.2.2 Configuring Oracle WSM 11*g* Client

1. Using JDeveloper, create a SOA composite that consumes the .NET Web service. For more information, see the *Developer's Guide for SOA Suite*.
2. Attach the following policy to the Web service client: oracle/wsmtom_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

Example of .NET Web Service for MTOM Interoperability

The following provides an example of the .NET Web service for MTOM interoperability.

```
static void Main(string[] args)
{
    string uri = "http://host:port/TEST/MTOMService/SOA/MTOMService";
    // Step 1 of the address configuration procedure: Create a URI to serve as the base address.
    Uri baseAddress = new Uri(uri);

    // Step 2 of the hosting procedure: Create ServiceHost
    ServiceHost selfHost = new ServiceHost(typeof(MTOMService), baseAddress);

    try {
        HttpTransportBindingElement hb = new HttpTransportBindingElement();
        hb.ManualAddressing = false;
        hb.MaxBufferPoolSize = 2147483647;
        hb.MaxReceivedMessageSize = 2147483647;
        hb.AllowCookies = false;
        hb.AuthenticationScheme = System.Net.AuthenticationSchemes.Anonymous;
        hb.KeepAliveEnabled = true;
        hb.MaxBufferSize = 2147483647;
        hb.ProxyAuthenticationScheme = System.Net.AuthenticationSchemes.Anonymous;
        hb.Realm = "";
        hb.TransferMode = System.ServiceModel.TransferMode.Buffered;
        hb.UnsafeConnectionNtlmAuthentication = false;
        hb.UseDefaultWebProxy = true;
        MtomMessageEncodingBindingElement me = new MtomMessageEncodingBindingElement();
        me.MaxReadPoolSize=64;
```

```
        me.MaxWritePoolSize=16;
        me.MessageVersion=System.ServiceModel.Channels.MessageVersion.Soap12;
        me.WriteEncoding = System.Text.Encoding.UTF8;
        me.MaxWritePoolSize = 2147483647;
        me.MaxBufferSize = 2147483647;
        me.ReaderQuotas.MaxArrayLength = 2147483647;
        CustomBinding binding1 = new CustomBinding();
        binding1.Elements.Add(me);
        binding1.Elements.Add(hb);
        ServiceEndpoint ep = selfHost.AddServiceEndpoint(typeof(IMTOMService), binding1,
               "MTOMService");
        EndpointAddress myEndpointAdd = new EndpointAddress(new Uri(uri),
        EndpointIdentity.CreateDnsIdentity("WSMCert3"));
        ep.Address = myEndpointAdd;

        // Step 4 of the hosting procedure: Enable metadata exchange.
        ServiceMetadataBehavior smb = new ServiceMetadataBehavior();
        smb.HttpGetEnabled = true;
        selfHost.Description.Behaviors.Add(smb);
        using (ServiceHost host = new ServiceHost(typeof(MTOMService)))
        {
            System.ServiceModel.Description.ServiceDescription svcDesc =
                 selfHost.Description;
            ServiceDebugBehavior svcDebug =
                  svcDesc.Behaviors.Find<ServiceDebugBehavior>();
            svcDebug.IncludeExceptionDetailInFaults = true;
        }

        // Step 5 of the hosting procedure: Start (and then stop) the service.
        selfHost.Open();
        Console.WriteLine("The service " + uri + " is ready.");
        Console.WriteLine("Press <ENTER> to terminate service.");
        Console.WriteLine();
        Console.ReadLine();
        // Close the ServiceHostBase to shutdown the service.
        selfHost.Close();
    }
    catch (CommunicationException ce)
    {
        Console.WriteLine("An exception occurred: {0}", ce.Message);
        selfHost.Abort();
    }
}
```

5.3 Username Token With Message Protection (WS-Security 1.1)

This section describes how to implement username token with message protection that conforms to WS-Security 1.1 in the following interoperability scenarios:

- "Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service" on page 5-5
- "Configuring Oracle WSM 11g Client and Microsoft WCF/.NET 3.5 Web Service" on page 5-9

5.3.1 Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11*g* Web Service

To configure Microsoft WCF/.NET 3.5 client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

5.3.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service application.
2. Attach one of the following policies to the Web service:

   oracle/wss11_username_token_with_message_protection_service_policy

   oracle/wss11_saml_or_username_token_with_message_protection_service_policy

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Export the X.509 certificate file from the keystore on the service side to a .cer file (for example, *alice.cer*) using the following command:

   ```
   keytool -export -alias alice -file C:\alice.cer -keystore default-keystore.jks
   ```

5.3.1.2 Configuring Microsoft WCF/.NET 3.5 Client

1. Import the certificate file (exported previously) to the keystore on the client server using Microsoft Management Console (mmc). For information, see "How to: View Certificates with the MMC Snap-in" at http://msdn.microsoft.com/en-us/library/ms788967.aspx.
   a. Open a command prompt.
   b. Type **mmc** and press ENTER.
   c. Select **File > Add/Remove snap-in**.
   d. Select **Add and Choose Certificates**.

      Note: To view certificates in the local machine store, you must be in the Administrator role.

   e. Select **Add**.
   f. Select **My user account and finish**.
   g. Click **OK**.
   h. Expand **Console Root > Certificates -Current user > Personal > Certificates**.
   i. Right-click on **Certificates** and select **All tasks > Import** to launch Certificate import Wizard.
   j. Click **Next**, select **Browse**, and navigate to the .cer file that was exported previously.
   k. Click **Next** and accept defaults and finish the wizard.
2. Generate a .NET client using the WSDL of the Web service.

   For more information, see "How to: Create a Windows Communication Foundation Client" at http://msdn.microsoft.com/en-us/library/ms733133.aspx.

Note: SVCUtil does not support some security policy assertions such as <sp:SignedParts>. As a workaround:

- Detach the policy
- Generate proxy using SVCUtil
- Attach the policy back

3. In the Solution Explorer of the client project, add a reference by right-clicking on references, selecting Add reference, and browsing to C:\Windows\Microsoft .NET framework\v3.0\Windows Communication Framework\System.Runtime.Serialization.dll.
4. Edit the app.config file in the .NET project to update the certificate file and disable replays, as described in "Edit the app.config File" on page 5-7.
5. Compile the project.
6. Open a command prompt and navigate to the project's Debug folder.
7. Enter <client_project_name>.exe and press Enter.

Edit the app.config File

Edit the app.config file to update the certificate file and disable replays, as shown in the following example (changes are identified in **bold**). If you follow the default key setup, then <certificate_cn> should be set to alice.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.serviceModel>
    <behaviors>
      <endpointBehaviors>
         <behavior name="secureBehaviour">
           <clientCredentials>
             <serviceCertificate>
               <defaultCertificate findValue="<certificate_cn>"
                storeLocation="CurrentUser" storeName="My"
                x509FindType="FindBySubjectName"/>
             </serviceCertificate>
           </clientCredentials>
         </behavior>
      </endpointBehaviors>
    </behaviors>
  <bindings>
    <customBinding>
      <binding name="HelloWorldSoapHttp">
      <security defaultAlgorithmSuite="Basic128"
       authenticationMode="UserNameForCertificate"
       requireDerivedKeys="false" securityHeaderLayout="Lax"
       includeTimestamp="true"
       keyEntropyMode="CombinedEntropy"
       messageProtectionOrder="SignBeforeEncrypt"
       messageSecurityVersion=
"WSSecurity11WSTrustFebruary2005WSSecureConversationFebruary2005WSSecurityPolicy11
BasicSecurityProfile10"
       requireSignatureConfirmation="true">
     <localClientSettings
       cacheCookies="true"
       detectReplays="false"
```

```
      replayCacheSize="900000"
      maxClockSkew="00:05:00"
      maxCookieCachingTime="Infinite"
      replayWindow="00:05:00"
      sessionKeyRenewalInterval="10:00:00"
      sessionKeyRolloverInterval="00:05:00"
      reconnectTransportOnFailure="true"
      timestampValidityDuration="00:05:00"
      cookieRenewalThresholdPercentage="60" />
    <localServiceSettings detectReplays="true"
      issuedCookieLifetime="10:00:00"
      maxStatefulNegotiations="128"
      replayCacheSize="900000"
      maxClockSkew="00:05:00"
      negotiationTimeout="00:01:00"
      replayWindow="00:05:00"
      inactivityTimeout="00:02:00"
      sessionKeyRenewalInterval="15:00:00"
      sessionKeyRolloverInterval="00:05:00"
      reconnectTransportOnFailure="true"
      maxPendingSessions="128"
      maxCachedCookies="1000"
      timestampValidityDuration="00:05:00" />
    <secureConversationBootstrap /></security>
    <textMessageEncoding
     maxReadPoolSize="64"
     maxWritePoolSize="16"
     messageVersion="Soap11"
     writeEncoding="utf-8">
       <readerQuotas
        maxDepth="32"
        maxStringContentLength="8192"
        maxArrayLength="16384"
        maxBytesPerRead="4096"
        maxNameTableCharCount="16384" />
    </textMessageEncoding>
    <HttpTransport
     manualAddressing="false"
     maxBufferPoolSize="524288"
     maxReceivedMessageSize="65536"
     allowCookies="false"
     authenticationScheme="Anonymous"
     bypassProxyOnLocal="false"
     hostNameComparisonMode="StrongWildcard"
     keepAliveEnabled="true"
     maxBufferSize="65536"
     proxyAuthenticationScheme="Anonymous"
     realm=""
     transferMode="Buffered"
     unsafeConnectionNtlmAuthentication="false"
     useDefaultWebProxy="true" />
     </binding>
   </customBinding>
 </bindings>
   <client>
     <endpoint address="<endpoint_url>"
      binding="customBinding"
      bindingConfiguration="HelloWorldSoapHttp"
      contract="HelloWorld"
      name="HelloWorldPort"
```

```
      behaviorConfiguration="secureBehaviour" >
       <identity>
         <dns value="<certificate_cn>"/>
       </identity>
      </endpoint>
    </client>
  </system.serviceModel>
</configuration>
```

5.3.2 Configuring Oracle WSM 11*g* Client and Microsoft WCF/.NET 3.5 Web Service

To configure Oracle WSM 11*g* client and Microsoft WCF/.NET 3.5 Web service, perform the steps described in the following sections:

5.3.2.1 Configuring Microsoft WCF/.NET 3.5 Web Service

1. Create a .NET Web service.

   For more information, see "How to: Define a Windows Communication Foundation Service Contract" at http://msdn.microsoft.com/en-us/library/ms731835.aspx.

   Be sure to create a custom binding for the Web service using the SymmetricSecurityBindingElement. For an example, see "Example .NET Web Service Client" on page 5-10.

2. Create and import a certificate file to the keystore on the Web service server.

   Using VisualStudio, the command would be similar to the following:

   ```
   makecert -r -pe -n "CN=wsmcert3" -sky exchange -ss my C:\wsmcert3.cer
   ```

   This command creates and imports a certificate in mmc.

   If the command does not provide expected results, then try the following sequence of commands. You need to download Windows Developer Kit (WDK) at http://www.microsoft.com/whdc/devtools/WDK/default.mspx.

   ```
   makecert -r -pe -n "CN=wsmcert3" -sky exchange -ss my -sv wscert3.pvk
   C:\wsmcert3.cer
   pvk2pfx.exe -pvk wscert3.pvk -spc wsmcert3.cer -pfx PRF_WSMCert3.pfx -pi
   welcome1
   ```

   Then, in mmc, import PRF_WSMCert3.pfx.

3. Import the certificate created on the Web service server to the client server using the keytool command. For example:

   ```
   keytool -import -alias wsmcert3 -file C:\wsmcert3.cer -keystore <owsm_client_
   keystore>
   ```

4. Right-click on the Web service Solution project in Solutions Explorer and click **Open Folder In Windows Explorer**.

5. Navigate to the bin/Debug folder.

6. Double-click on the <project>.exe file. This command will run the Web service at the URL provided.

5.3.2.2 Configuring Oracle WSM 11*g* Client

1. Using JDeveloper, create a SOA composite that consumes the .NET Web service. For more information, see the *Developer's Guide for SOA Suite*.

2. In JDeveloper, create a partner link using the WSDL of the .NET service.

3. Attach the following policy to the Web service client: oracle/wss11_username_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

4. Provide configurations for the csf-key and keystore.recipient.alias.

   You can specify this information when attaching the policy, by overriding the policy configuration. For more information, see "Attaching Clients Permitting Overrides" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*

   Ensure that you configure the keystore.recipient.alias as the alias of the certificate imported in step 1 (wsmcert3). For example:

```
<wsp:PolicyReference URI="oracle/wss11_username_token_with_message_protection_
client_policy"
    orawsp:category="security" orawsp:status="enabled"/>
  <property name="csf-key" type="xs:string"
    many="false">basic.credentials</property>
  <property name="keystore.recipient.alias" type="xs:string"
    many="false">wsmcert3</property>
```

5.3.2.3 Example .NET Web Service Client

```
static void Main(string[] args)
{
    // Step 1 of the address configuration procedure: Create a URI to serve as the
    // base address.
    // Step 2 of the hosting procedure: Create ServiceHost
    string uri = "http://<host>:<port>/TEST/NetService";
    Uri baseAddress = new Uri(uri);

    ServiceHost selfHost = new ServiceHost(typeof(CalculatorService), baseAddress);

    try
    {
        SymmetricSecurityBindingElement sm =
            SymmetricSecurityBindingElement.CreateUserNameForCertificateBindingElement();
        sm.DefaultAlgorithmSuite = System.ServiceModel.Security.SecurityAlgorithmSuite.Basic128;
        sm.SetKeyDerivation(false);
        sm.SecurityHeaderLayout = SecurityHeaderLayout.Lax;
        sm.IncludeTimestamp = true;
        sm.KeyEntropyMode = SecurityKeyEntropyMode.CombinedEntropy;
        sm.MessageProtectionOrder = MessageProtectionOrder.SignBeforeEncrypt;
        sm.MessageSecurityVersion =
        MessageSecurityVersion.WSSecurity11WSTrustFebruary2005WSSecureConversationFebruary2005
        WSSecurityPolicy11BasicSecurityProfile10;
        sm.RequireSignatureConfirmation = true;
        sm.LocalClientSettings.CacheCookies = true;
        sm.LocalClientSettings.DetectReplays = true;
        sm.LocalClientSettings.ReplayCacheSize = 900000;
        sm.LocalClientSettings.MaxClockSkew = new TimeSpan(00, 05, 00);
        sm.LocalClientSettings.MaxCookieCachingTime = TimeSpan.MaxValue;
        sm.LocalClientSettings.ReplayWindow = new TimeSpan(00, 05, 00); ;
        sm.LocalClientSettings.SessionKeyRenewalInterval = new TimeSpan(10, 00, 00);
        sm.LocalClientSettings.SessionKeyRolloverInterval = new TimeSpan(00, 05, 00); ;
```

```
sm.LocalClientSettings.ReconnectTransportOnFailure = true;
sm.LocalClientSettings.TimestampValidityDuration = new TimeSpan(00, 05, 00); ;
sm.LocalClientSettings.CookieRenewalThresholdPercentage = 60;
sm.LocalServiceSettings.DetectReplays = false;
sm.LocalServiceSettings.IssuedCookieLifetime = new TimeSpan(10, 00, 00);
sm.LocalServiceSettings.MaxStatefulNegotiations = 128;
sm.LocalServiceSettings.ReplayCacheSize = 900000;
sm.LocalServiceSettings.MaxClockSkew = new TimeSpan(00, 05, 00);
sm.LocalServiceSettings.NegotiationTimeout = new TimeSpan(00, 01, 00);
sm.LocalServiceSettings.ReplayWindow = new TimeSpan(00, 05, 00);
sm.LocalServiceSettings.InactivityTimeout = new TimeSpan(00, 02, 00);
sm.LocalServiceSettings.SessionKeyRenewalInterval = new TimeSpan(15, 00, 00);
sm.LocalServiceSettings.SessionKeyRolloverInterval = new TimeSpan(00, 05, 00);
sm.LocalServiceSettings.ReconnectTransportOnFailure = true;
sm.LocalServiceSettings.MaxPendingSessions = 128;
sm.LocalServiceSettings.MaxCachedCookies = 1000;
sm.LocalServiceSettings.TimestampValidityDuration = new TimeSpan(15, 00, 00);
HttpTransportBindingElement hb = new HttpTransportBindingElement();
hb.ManualAddressing = false;
hb.MaxBufferPoolSize = 524288;
hb.MaxReceivedMessageSize = 65536;
hb.AllowCookies = false;
hb.AuthenticationScheme = System.Net.AuthenticationSchemes.Anonymous;
hb.KeepAliveEnabled = true;
hb.MaxBufferSize = 65536;
hb.ProxyAuthenticationScheme = System.Net.AuthenticationSchemes.Anonymous;
hb.Realm = "";
hb.TransferMode = System.ServiceModel.TransferMode.Buffered;
hb.UnsafeConnectionNtlmAuthentication = false;
hb.UseDefaultWebProxy = true;
TextMessageEncodingBindingElement tb1 = new TextMessageEncodingBindingElement();
tb1.MaxReadPoolSize = 64;
tb1.MaxWritePoolSize = 16;
tb1.MessageVersion = System.ServiceModel.Channels.MessageVersion.Soap12;
tb1.WriteEncoding = System.Text.Encoding.UTF8;
CustomBinding binding1 = new CustomBinding(sm);
binding1.Elements.Add(tb1);
binding1.Elements.Add(hb);
ServiceEndpoint ep = selfHost.AddServiceEndpoint(typeof(ICalculator), binding1,
  "CalculatorService");

EndpointAddress myEndpointAdd = new EndpointAddress(
new Uri(uri),
EndpointIdentity.CreateDnsIdentity("WSMCert3"));
ep.Address = myEndpointAdd;

// Step 4 of the hosting procedure: Enable metadata exchange.
ServiceMetadataBehavior smb = new ServiceMetadataBehavior();
smb.HttpGetEnabled = true;
selfHost.Description.Behaviors.Add(smb);
selfHost.Credentials.ServiceCertificate.SetCertificate(StoreLocation.CurrentUser,
   StoreName.My,
X509FindType.FindBySubjectName, "WSMCert3");
selfHost.Credentials.ClientCertificate.Authentication.CertificateValidationMode =
    X509CertificateValidationMode.PeerOrChainTrust;
selfHost.Credentials.UserNameAuthentication.UserNamePasswordValidationMode =
    UserNamePasswordValidationMode.Custom;
CustomUserNameValidator cu = new CustomUserNameValidator();
selfHost.Credentials.UserNameAuthentication.CustomUserNamePasswordValidator = cu;
using (ServiceHost host = new ServiceHost(typeof(CalculatorService)))
```

```
        {
            System.ServiceModel.Description.ServiceDescription svcDesc = selfHost.Description;
            ServiceDebugBehavior svcDebug = svcDesc.Behaviors.Find<ServiceDebugBehavior>();
            svcDebug.IncludeExceptionDetailInFaults = true;
        }

        // Step 5 of the hosting procedure: Start (and then stop) the service.
        selfHost.Open();
        Console.WriteLine("The Calculator service is ready.");
        Console.WriteLine("Press <ENTER> to terminate service.");
        Console.WriteLine();
        Console.ReadLine();
        selfHost.Close();
    }
    catch (CommunicationException ce)
    {
         Console.WriteLine("An exception occurred: {0}", ce.Message);
         selfHost.Abort();
     }
}
```

5.4 Username Token Over SSL

This section describes how to implement username token over SSL in the following interoperability scenario:

- "Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service" on page 5-12

5.4.1 Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11*g* Web Service

To configure Microsoft WCF/.NET 3.5 client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

5.4.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Create a copy of one of the following policies:

   oracle/wss_username_token_over_ssl_service_policy

   oracle/wss_saml_or_username_token_over_ssl_service_policy

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Disable the Creation Time Required configuration setting.

   b. Disable the Nonce Required configuration setting.

   c. Leave the default configuration set for all other configuration settings.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Attach the policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

5.4.1.2 Configuring Microsoft WCF/.NET 3.5 Client

1. Generate a .NET client using the WSDL of the Web service.

   For more information, see "How to: Create a Windows Communication Foundation Client" at http://msdn.microsoft.com/en-us/library/ms733133.aspx.

2. In the Solution Explorer of the client project, add a reference by right-clicking on references, selecting Add reference, and browsing to C:\Windows\Microsoft .NET framework\v3.0\Windows Communication Framework\System.Runtime.Serialization.dll.

3. Edit the app.config file, as described in "Edit the app.config File" on page 5-13.

4. Compile the project.

5. Open a command prompt and navigate to the project's Debug folder.

6. Type <client_project_name>.exe and press Enter.

Edit the app.config File

Edit the app.config file to update the certificate file and disable replays, as shown in the following example (changes are identified in **bold**):

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.serviceModel>
        <bindings>
            <customBinding>
                <binding name="BPELProcess1Binding">
                  <security defaultAlgorithmSuite="Basic128"
                   authenticationMode="UserNameOverTransport"
                   requireDerivedKeys="false" securityHeaderLayout="Lax" includeTimestamp="true"
                   keyEntropyMode="CombinedEntropy" messageProtectionOrder="SignBeforeEncrypt"
                   messageSecurityVersion="WSSecurity11WSTrustFebruary2005WSSecureConversation
                   February2005WSSecurityPolicy11BasicSecurityProfile10"
                   requireSignatureConfirmation="true">
                    <localClientSettings cacheCookies="true" detectReplays="true"
                      replayCacheSize="900000" maxClockSkew="00:05:00"
                      maxCookieCachingTime="Infinite"
                      replayWindow="00:05:00" sessionKeyRenewalInterval="10:00:00"
                      sessionKeyRolloverInterval="00:05:00" reconnectTransportOnFailure="true"
                      timestampValidityDuration="00:05:00"
                      cookieRenewalThresholdPercentage="60"/>
                    <localServiceSettings detectReplays="true" issuedCookieLifetime="10:00:00"
                        maxStatefulNegotiations="128" replayCacheSize="900000"
                        maxClockSkew="00:05:00"
                        negotiationTimeout="00:01:00" replayWindow="00:05:00"
                        inactivityTimeout="00:02:00"
                        sessionKeyRenewalInterval="15:00:00"
                        sessionKeyRolloverInterval="00:05:00"
                        reconnectTransportOnFailure="true" maxPendingSessions="128"
```

```
                maxCachedCookies="1000" timestampValidityDuration="00:05:00" />
              <secureConversationBootstrap />
            </security>
            <textMessageEncoding maxReadPoolSize="64" maxWritePoolSize="16"
                  messageVersion="Soap11" writeEncoding="utf-8">
                  <readerQuotas maxDepth="32" maxStringContentLength="8192"
                   maxArrayLength="16384"
                   maxBytesPerRead="4096" maxNameTableCharCount="16384" />
            </textMessageEncoding>
            <httpsTransport manualAddressing="false" maxBufferPoolSize="524288"
                 maxReceivedMessageSize="65536" allowCookies="false"
                 authenticationScheme="Anonymous"
                 bypassProxyOnLocal="false" hostNameComparisonMode="StrongWildcard"
                 keepAliveEnabled="true" maxBufferSize="65536"
                 proxyAuthenticationScheme="Anonymous"
                 realm="" transferMode="Buffered"
                 unsafeConnectionNtlmAuthentication="false"
                 useDefaultWebProxy="true"  requireClientCertificate="false"/>
          </binding>
        </customBinding>
      </bindings>
      <client>
          <endpoint address="
 https://host:port/soa-infra/services/default/IO_NET6/bpelprocess1_client_ep"
 binding="customBinding" bindingConfiguration="BPELProcess1Binding"
 contract="BPELProcess1" name="BPELProcess1_pt" />
      </client>
  </system.serviceModel>
</configuration>
```

5.5 Mutual Authentication with Message Protection (WS-Security 1.1)

The following sections describe how to implement mutual authentication with message protection that conform to the WS-Security 1.1 standards:

- "Configuration Prerequisites for Interoperability" on page 5-14
- "Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service" on page 5-15
- "Configuring Oracle WSM 11g Client and Microsoft WCF/.NET 3.5 Web Service" on page 5-18

Configuration Prerequisites for Interoperability

1. Export the X.509 certificate file from the keystore on the service side to a .cer file (for example, *alice.cer*) using the following command:

   ```
   keytool -export -alias alice -file C:\alice.cer -keystore default-keystore.jks
   ```

2. Import the certificate file (exported previously) to the keystore on the client server using Microsoft Management Console (mmc). For information, see "How to: View Certificates with the MMC Snap-in" at http://msdn.microsoft.com/en-us/library/ms788967.aspx.

   a. Open a command prompt.

   b. Type **mmc** and press ENTER.

   c. Select **File > Add/Remove snap-in**.

   d. Select **Add and Choose Certificates**.

Note: To view certificates in the local machine store, you must be in the Administrator role.

e. Select **Add**.

f. Select **My user account and finish**.

g. Click **OK**.

h. Expand **Console Root > Certificates -Current user > Personal > Certificates**.

i. Right-click on **Certificates** and select **All tasks > Import** to launch Certificate import Wizard.

j. Click **Next**, select **Browse**, and navigate to the .cer file that was exported previously.

k. Click **Next** and accept defaults and finish the wizard.

5.5.1 Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11*g* Web Service

To configure Microsoft WCF/.NET 3.5 client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

5.5.1.1 Configuring Oracle WSM 11g Web Service

1. Create a SOA composite and deploy it.
2. In Enterprise Manager, clone the following policy:

   oracle/wss11_x509_token_with_message_protection_service_policy

   Rename it as follows: wss11_x509_token_with_message_protection_service_policy_net
3. Export wss11_x509_token_with_message_protection_service_policy_net. Change encrypted="true" to "false", and import it back.

```
<orasp:x509-token orasp:enc-key-ref-mech="thumbprint"
orasp:is-encrypted="false" orasp:is-signed="false"
orasp:sign-key-ref-mech="direct"/>
```

4. Using Enterprise Manager, attach the policy to the Web service.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

5.5.1.2 Configuring Microsoft WCF/.NET 3.5 Client

1. Use the SVCUtil utility to create a client proxy (see "Sample Client Program" on page 5-18) and configuration file from the deployed Web service.
2. In the Solution Explorer of the client project, add a reference by right-clicking on references, selecting Add reference, and browsing to C:\Windows\Microsoft .NET framework\v3.0\Windows Communication Framework\System.Runtime.Serialization.dll.

   a. Create a configuration file: app.config. Add the following code after the <system.serviceModel> element.

```
<configuration>
  <system.serviceModel>
```

```
      <behaviors>
        <endpointBehaviors>
          <behavior name="secureBehaviour">
            <clientCredentials>
              <serviceCertificate>
                <defaultCertificate findValue="<certificate_cn>"
storeLocation="CurrentUser" storeName="My"
x509FindType="FindBySubjectName"/>
              </serviceCertificate>
            </clientCredentials>
          </behavior>
        </endpointBehaviors>
      </behaviors>
        <bindings>
            <customBinding>
```

b. Modify the endpoint behavior as follows:

```
<endpoint address="http://<server>:<port>//MyWebService1SoapHttpPort"
              binding="customBinding"
 bindingConfiguration="MyWebService1SoapHttp"
              contract="MyWebService1" name="MyWebService1SoapHttpPort"
 behaviorConfiguration="secureBehaviour" >
            <identity>
              <dns value="<certificate_cn>"/>
            </identity>
          </endpoint>
```

c. Disable the message replay detection as follows:

```
<localClientSettings cacheCookies="true" detectReplays="false"
                            replayCacheSize="900000"
 maxClockSkew="00:05:00" maxCookieCachingTime="Infinite"
```

d. Create a custom binding as shown below:

```
<security defaultAlgorithmSuite="Basic128"
 authenticationMode="MutualCertificate"
```

e. "Sample app.config File" on page 5-16 provides an example of the configuration file.

3. Compile the project.
4. Open a command prompt and navigate to the project's Debug folder.
5. Enter <client_project_name>.exe and press Enter.

Sample app.config File

The following provides an example of the app.config file:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.serviceModel>
      <behaviors>
        <endpointBehaviors>
          <behavior name="secureBehaviour">
            <clientCredentials>
              <serviceCertificate>
                <defaultCertificate findValue="<certificate_cn>"
                                    storeLocation="CurrentUser"
                                    storeName="My"
```

```
                              x509FindType="FindBySubjectName"/>
        </serviceCertificate>

    </clientCredentials>
    </behavior>
  </endpointBehaviors>
</behaviors>
  <bindings>

    <customBinding>
      <binding name="BPELProcess1Binding">
        <security defaultAlgorithmSuite="Basic128" authenticationMode="MutualCertificate"
            requireDerivedKeys="false" securityHeaderLayout="Lax" includeTimestamp="true"
            keyEntropyMode="CombinedEntropy" messageProtectionOrder="SignBeforeEncrypt"

            messageSecurityVersion="WSSecurity11WSTrustFebruary2005WSSecureConversation
            February2005WSSecurityPolicy11BasicSecurityProfile10"
            requireSignatureConfirmation="true">
          <localClientSettings cacheCookies="true" detectReplays="false"
              replayCacheSize="900000" maxClockSkew="00:05:00"
              maxCookieCachingTime="Infinite"
              replayWindow="00:05:00" sessionKeyRenewalInterval="10:00:00"
              sessionKeyRolloverInterval="00:05:00" reconnectTransportOnFailure="true"
              timestampValidityDuration="00:05:00" cookieRenewalThresholdPercentage="60" />
          <localServiceSettings detectReplays="true" issuedCookieLifetime="10:00:00"
              maxStatefulNegotiations="128" replayCacheSize="900000" maxClockSkew="00:05:00"
              negotiationTimeout="00:01:00" replayWindow="00:05:00"
              inactivityTimeout="00:02:00"
              sessionKeyRenewalInterval="15:00:00" sessionKeyRolloverInterval="00:05:00"
              reconnectTransportOnFailure="true" maxPendingSessions="128"
              maxCachedCookies="1000" timestampValidityDuration="00:05:00" />
          <secureConversationBootstrap />
        </security>
        <textMessageEncoding maxReadPoolSize="64" maxWritePoolSize="16"
            messageVersion="Soap11" writeEncoding="utf-8">
          <readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
              maxBytesPerRead="4096" maxNameTableCharCount="16384" />
        </textMessageEncoding>
        <httpTransport manualAddressing="false" maxBufferPoolSize="524288"
            maxReceivedMessageSize="65536" allowCookies="false"
            authenticationScheme="Anonymous"
            bypassProxyOnLocal="false" hostNameComparisonMode="StrongWildcard"
            keepAliveEnabled="true" maxBufferSize="65536"
            proxyAuthenticationScheme="Anonymous"
            realm="" transferMode="Buffered" unsafeConnectionNtlmAuthentication="false"
            useDefaultWebProxy="true" />
      </binding>
    </customBinding>

  </bindings>
  <client>
    <endpoint address="<endpoint_url>"
        binding="customBinding" bindingConfiguration="BPELProcess1Binding"
        contract="BPELProcess1" name="BPELProcess1_pt"  >
      <identity>
        <dns value=<certificate_cn>/>
      </identity>
    </endpoint>
  </client>
```

```
    </system.serviceModel>
</configuration>
```

Sample Client Program

```
 namespace IO_NET10_client
{
    class Program
    {
        static void Main(string[] args)
        {

            BPELProcess1Client client = new BPELProcess1Client();

            client.ClientCredentials.ClientCertificate.SetCertificate(
                    StoreLocation.CurrentUser,
                    StoreName.My,
                    X509FindType.FindBySubjectName, "WSMCert3");

             client.ClientCredentials.ServiceCertificate.SetDefaultCertificate(
                       StoreLocation.CurrentUser,
                       StoreName.My,
                    X509FindType.FindBySubjectName, "Alice");

            process proc = new process();
            proc.input = "Test wss11_x509_token_with_message_protection_policy -
";
            Console.WriteLine(proc.input);
            processResponse response = client.process(proc);

            Console.WriteLine(response.result.ToString());
            Console.WriteLine("Press <ENTER> to terminate Client.");
            Console.ReadLine();
          }
    }
}
```

5.5.2 Configuring Oracle WSM 11*g* Client and Microsoft WCF/.NET 3.5 Web Service

To configure Oracle WSM 11*g* client and Microsoft WCF/.NET 3.5 Web Service, perform the steps described in the following sections:

- "Configuring Microsoft WCF/.NET 3.5 Web Service" on page 5-9
- "Configuring Oracle WSM 11g Client" on page 5-19

5.5.2.1 Configuring Microsoft WCF/.NET 3.5 Web Service

1. Create a .NET Web service.

   For more information, see "How to: Define a Windows Communication Foundation Service Contract" at http://msdn.microsoft.com/en-us/library/ms731835.aspx.

   For an example of a .NET Web service, see "Example .NET Web Service Client" on page 5-10.

2. Create a custom binding for the Web service using the SymmetricSecurityBindingElement.

For more information, see "How to: Create a Custom Binding Using the SecurityBindingElement" at http://msdn.microsoft.com/en-us/library/ms730305.aspx.

3. The following is a sample of the SymmetricSecurityBindingElement object:

```
SymmetricSecurityBindingElement sm =
(SymmetricSecurityBindingElement)SecurityBindingElement.CreateMutualCertificate
BindingElement();

sm.DefaultAlgorithmSuite =
System.ServiceModel.Security.SecurityAlgorithmSuite.Basic128;sm.SetKeyDerivati
on(false);
sm.SecurityHeaderLayout = SecurityHeaderLayout.Lax;sm.IncludeTimestamp =
true;
sm.KeyEntropyMode = SecurityKeyEntropyMode.CombinedEntropy;
sm.MessageProtectionOrder =
MessageProtectionOrder.SignBeforeEncrypt;sm.MessageSecurityVersion =
MessageSecurityVersion.WSSecurity11WSTrustFebruary2005WSSecureConversation
February2005WSSecurityPolicy11BasicSecurityProfile10;
sm.RequireSignatureConfirmation =
true;
```

4. Deploy the application.

5.5.2.2 Configuring Oracle WSM 11g Client

1. Using JDeveloper, create a SOA composite that consumes the .NET Web service. For more information, see the *Developer's Guide for SOA Suite*.
2. In JDeveloper, create a partner link using the WSDL of the .NET service and add the import as follows:

```
<wsdl:import namespace="<namespace>" location="<WSDL location>"/>
```

3. In Enterprise Manager, clone the policy: wss11_x509_token_with_message_protection_service_policy. Rename it as follows: wss11_x509_token_with_message_protection_service_policy_net
4. Export wss11_x509_token_with_message_protection_service_policy_net. Change encrypted="true" to "false", and import it back

```
<orasp:x509-token orasp:enc-key-ref-mech="thumbprint" orasp:is-encrypted="true"
 orasp:is-signed="false" orasp:sign-key-ref-mech="direct"/>
```

5. Attach the policy to the Web service client.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6. Provide configurations for the keystore.recipient.alias.

   You can specify this information when attaching the policy, by overriding the policy configuration. For more information, see "Attaching Clients Permitting Overrides" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

   Ensure that you configure the keystore.recipient.alias as the alias of the certificate imported in step 4 (wsmcert3).

7. Invoke the Web service method from the client.

5.6 Kerberos with Message Protection

This section describes how to implement kerberos with message protection in the following interoperability scenarios:

- "Configuration Prerequisites for Interoperability" on page 5-20
- "Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service" on page 5-20

5.6.1 Configuration Prerequisites for Interoperability

Perform the following prerequisite steps:

1. Configure the Key Distribution Center (KDC) and Active Directory (AD). For more information, see the section "To Configure Windows Active Directory and Domain Controller" (the domain controller can serve as KDC) at http://docs.sun.com/app/docs/doc/820-3746/gisgm?l=en&a=view.
2. Set up the Kerberos configuration file krb5.conf in c:\winnt as shown in Example 5–1.

Example 5–1 Sample Kerberos Configuration File

```
[logging]
default = c:\log\krb5libs.log
kdc = c:\log\krb5kdc.log
admin_server = c:\log\kadmind.log
[libdefaults]
default_realm = MYCOMPANY.LOCAL
dns_lookup_realm = false
dns_lookup_kdc = false
default_tkt_enctypes = rc4-hmac
default_tgs_enctypes = rc4-hmac
permitted_enctypes = rc4-hmac
kdc = <hostname>
[realms]
MYCOMPANY.LOCAL =
{ kdc = <hostname>:<portnumber>  admin_server = <hostname>:<portnumber>
  default_domain = <domainname>
}
 [domain_realm]
.<domainname> = MYCOMPANY.LOCAL
 <domainname> = MYCOMPANY.LOCAL
[appdefaults]
pam =
{   debug = false  ticket_lifetime = 36000  renew_lifetime = 36000  forwardable =
 true  krb4_convert = false }
```

5.6.2 Configuring Microsoft WCF/.NET 3.5 Client and Oracle WSM 11g Web Service

To configure Microsoft WCF/.NET 3.5 client and Oracle WSM 11g Web service, perform the steps described in the following sections:

5.6.2.1 Configuring Oracle WSM 11g Web Service

1. Create a Web service application.
2. Copy the following policy: wss11_kerberos_with_message_protection_service_policy.

3. Edit the policy settings to set Algorithm Suite to Basic128Rsa15.
4. Attach the policy to the web service. For more information about attaching the policy at deployment time using Fusion Middleware Control, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
5. Deploy the application.

5.6.2.2 Configuring Microsoft WCF/.NET 3.5 Client

1. Create a user in AD to represent the host where the web service is hosted. By default the user account is created with RC4-HMAC encryption. For example, foobar with user name as "HTTP/foobar".
2. Use the following ktpass command to create a keytab file on the Windows AD machine where the KDC is running:

   ```
   ktpass -princ HTTP/foobar@MYCOMPANY.LOCAL -pass Oracle123
   -mapuser foobar -out foobar.keytab -ptype KRB5_NT_PRINCIPAL
   -kvno 4
   ```

   where HTTP/foobar is the SPN, mapped to a user "foobar". Do not set "/desonly or cyrpto as "des-cbc-crc". MYCOMPANY.LOCAL is the default Realm for the KDC and is available in the krb5.ini file. The pass password must match the password created during the user creation.

   Use FTP binary mode to move the generated keytab file to the machine where the SOA Composite Web service is hosted.
3. Use the following setSpn command to map the service principal to the user:

   ```
   setSpn -A HTTP/foobar@MYCOMPANY.LOCAL foobar
   ```

   ```
   setSpn -L foobar
   ```

   Only one SPN must be mapped to the user. If there are multiple SPNs mapped to the user, remove them using the command `setSpn -D <spname> <username>`.
4. Use the SVCUtil utility to create a client proxy and configuration file from the deployed Web service.

   Add the files generatedProxy.cs and app.config by right clicking the application (in the Windows Explorer) and selecting **Add Existing Item**.

   In the endpoint element of the app.config, add an "identity" element with service principal name as "HTTP/foobar@MYCOMPANY.LOCAL" (the same value used for creating keytab).

   ```
   <client>
          <endpoint address="http://host:port/HelloServicePort"
              binding="customBinding"
   bindingConfiguration="NewHelloSoap12HttpPortBinding"
              contract="NewHello" name="HelloServicePort">
          <identity>
            <servicePrincipalName value ="HTTP/foobar@MYCOMPANY.LOCAL"/>
          </identity>
          </endpoint>

        </client>
   ```

   A sample binding is provided in Example 5–2.

Example 5–2 Sample Binding

```
<customBinding>
                <binding name="NewHelloSoap12HttpPortBinding">
                    <!--Added by User: Begin-->
                  <security defaultAlgorithmSuite="Basic128"
                   authenticationMode="Kerberos"
                    requireDerivedKeys="false" securityHeaderLayout="Lax"
                    includeTimestamp="true"
                    keyEntropyMode="CombinedEntropy"
                    messageProtectionOrder="SignBeforeEncrypt"
                   messageSecurityVersion="WSSecurity11WSTrustFebruary2005
                   WSSecureConversationFebruary2005WSSecurityPolicy11BasicSecurity
                    Profile10"
                  requireSignatureConfirmation="true">
                    <localClientSettings cacheCookies="true" detectReplays="true"
                        replayCacheSize="900000" maxClockSkew="00:05:00"
                        maxCookieCachingTime="Infinite"
                        replayWindow="00:05:00"
                        sessionKeyRenewalInterval="10:00:00"
                        sessionKeyRolloverInterval="00:05:00"
                        reconnectTransportOnFailure="true"
                        timestampValidityDuration="00:05:00"
                        cookieRenewalThresholdPercentage="60" />
                    <localServiceSettings detectReplays="true"
                        issuedCookieLifetime="10:00:00"
                        maxStatefulNegotiations="128" replayCacheSize="900000"
                        maxClockSkew="00:05:00"
                        negotiationTimeout="00:01:00" replayWindow="00:05:00"
                        inactivityTimeout="00:02:00"
                        sessionKeyRenewalInterval="15:00:00"
                        sessionKeyRolloverInterval="00:05:00"
                        reconnectTransportOnFailure="true"
                        maxPendingSessions="128"
                        maxCachedCookies="1000"
                        timestampValidityDuration="00:05:00" />
                    <secureConversationBootstrap />
                  </security>
                  <!--Added by User: End-->
                    <textMessageEncoding maxReadPoolSize="64"
                       maxWritePoolSize="16"
                        messageVersion="Soap12" writeEncoding="utf-8">
                        <readerQuotas maxDepth="32" maxStringContentLength="8192"
                          maxArrayLength="16384"
                            maxBytesPerRead="4096" maxNameTableCharCount="16384"
/>
                    </textMessageEncoding>
                  <!--Added by User: Begin-->
                  <httpTransport manualAddressing="false"
                       maxBufferPoolSize="524288"
                        maxReceivedMessageSize="65536" allowCookies="false"
                        authenticationScheme="Anonymous"
                        bypassProxyOnLocal="false"
                        hostNameComparisonMode="StrongWildcard"
                        keepAliveEnabled="true" maxBufferSize="65536"
                        proxyAuthenticationScheme="Anonymous"
                        realm="" transferMode="Buffered"
                        unsafeConnectionNtlmAuthentication="false"
                        useDefaultWebProxy="true" />
                  <!--Added by User: End-->
                </binding>
```

```
</customBinding>
```

The svcutil.exe utility will not work if the Web service has an Oracle WSM policy attached to it. Detach the policy from the service before running this utility to generate the proxy and re-attach once all artifacts are generated successfully.

5. Run the client program.

6

Interoperability with Oracle Service Bus 10*g* Security Environments

This chapter contains the following sections:

6.1 Overview of Interoperability with Oracle Service Bus 10*g* Security Environments

In Oracle Service Bus 10*g*, you attach policies to configure your security environment for inbound and outbound requests. Oracle Service Bus uses the underlying WebLogic security framework as building blocks for its security services. For information about configuring and attaching policies, see "Using WS-Policy in Oracle Service Bus Proxy and Business Services" in *Oracle Service Bus Security Guide* at `http://download.oracle.com/docs/cd/E13159_01/osb/docs10gr3/security/ws_policy.html`.

> **Note:** Ensure that you have downloaded and applied the TYBN and U37Z patches released for Oracle Service Bus 10.3 using the patch tool.

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box.

For more details about the predefined policies, see "Predefined Policies" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

For more information about configuring and attaching policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

Table 6–1 summarizes the most common Oracle Service Bus 10*g* interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

For more information about:

- Configuring and attaching Oracle WSM 11*g* policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
- Configuring and attaching Oracle Service Bus 10*g* policies, see "Using WS-Policy in Oracle Service Bus Proxy and Business Services" in *Oracle Service Bus Security Guide* at `http://download.oracle.com/docs/cd/E13159_01/osb/docs10gr3/security/ws_policy.html`.

Note: In the following scenarios, ensure that you are using a keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

In addition, ensure that the keys use the proper extensions, including DigitalSignature, Non_repudiation, Key_Encipherment, and Data_Encipherment.

Table 6–1 Interoperability With Oracle Service Bus 10g Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11*g* Policies	Oracle Service Bus 10*g* Policies
"Username Token with Message Protection (WS-Security 1.0)" on page 6-2	Oracle Service Bus 10*g*—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	Encrypt.xml Sign.xml
"Username Token with Message Protection (WS-Security 1.0)" on page 6-2	Oracle WSM 11*g*—>Oracle Service Bus 10*g*	oracle/wss10_username_token_with_message_protection_client_policy	Encrypt.xml Sign.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 6-6	Oracle Service Bus 10*g*—>Oracle WSM 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy	Encrypt.xml Sign.xml
"SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)" on page 6-6	Oracle WSM 11*g*—>Oracle Service Bus 10*g*	oracle/wss10_saml_token_with_message_protection_client_policy	Encrypt.xml Sign.xml
"SAML or Username Token Over SSL" on page 6-10	Oracle Service Bus 10*g*—>Oracle WSM 11*g*	oracle/wss_saml_or_username_token_over_ssl_service_policy	Auth.xml
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 6-13	Oracle Service Bus 10*g*—>Oracle WSM 11*g*	oracle/wss10_x509_token_with_message_protection_service_policy	Encrypt.xml Sign.xml
"Mutual Authentication with Message Protection (WS-Security 1.0)" on page 6-13	Oracle WSM 11*g*—>Oracle Service Bus 10*g*	oracle/wss10_x509_token_with_message_protection_client_policy	Encrypt.xml Sign.xml

6.2 Username Token with Message Protection (WS-Security 1.0)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

- "Configuration Prerequisites for Interoperability" on page 6-3
- "Configuring Oracle Service Bus 10g Client and Oracle WSM 11g Web Service" on page 6-3
- "Configuring Oracle WSM 11g Client and Oracle Service Bus 10g Web Service" on page 6-5

Configuration Prerequisites for Interoperability

Perform the following prerequisite steps for the WebLogic Server on which Oracle Service Bus is running:

1. Copy the default-keystore.jks and trust.jks files to your domain directory.

   The default-keystore.jks is used to store public and private keys for SOAP messages within the WebLogic Domain. The trust.jks is used to store private keys, digital certificates, and trusted certificate authority certificates that are used to establish and verify identity and trust in the WebLogic Server environment.

2. Invoke the WebLogic Administration Console, as described in Accessing Oracle WebLogic Administration Console.

3. Configure the Custom Identity and Custom Trust keystores, as described in "Configuring keystores" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

4. Configure SSL, as described in "Set up SSL" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Specify the private key alias, as required. For example: `oratest`.

5. Configure a credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Create a PKICredentialMapper and configure it as follows (leave all other values set to the defaults):

   - Keystore Provider: N/A
   - Keystore Type: jks
   - Keystore File Name: default_keystore.jks
   - Keystore Pass Phrase: <password>
   - Confirm Keystore Pass Phrase: <password>

6. Restart Oracle WebLogic Server.

7. Invoke the OSB Console. For example:

   ```
   http://<host name>:<port number>/sbconsole
   ```

8. Create a ServiceKeyProvider.

9. Specify Encryption Key and Digital Signature Key, as required.

   You must use different keys on the Oracle WSM and Oracle Service Bus servers. You can use the same key for encryption and signing, if desired.

6.2.1 Configuring Oracle Service Bus 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle Service Bus 10*g* client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

6.2.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the following policy: wss10_username_token_with_message_protection_service_policy.

> **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Set Encryption Key Reference Mechanism to issuerserial.

b. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.

c. Enable the Include Timestamp configuration setting.

d. Set Is Encrypted to **false** for the Username token element only.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to the Web service.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6.2.1.2 Configuring Oracle Service Bus 10*g* Client

1. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, copy the files to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.

2. Edit the encryption algorithm in myEncrypt.xml file to prevent encryption compliance failure, as follows:

```
<wssp:Target>
   <wssp:EncryptionAlgorithm
     URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
   <wssp:MessageParts
     Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">
      wsp:Body()
   </wssp:MessageParts>
</wssp:Target>
```

3. Edit the mySign.xml policy file attached to the Oracle Service Bus business service **request** only to sign the Username token by including the following target:

```
<wssp:Target>
   <wssp:DigestAlgorithm URI=
    "http://www.w3.org/2000/09/xmldsig#sha1" />
   <wssp:MessageParts Dialect=
    "http://www.bea.com/wls90/security/policy/wsee#part">
      wls:SecurityHeader(wsse:UsernameToken)
   </wssp:MessageParts>
</wssp:Target>
```

4. Edit the mySign.xml policy file attached to the Oracle Service Bus business service **response** only to specify that the security token is unsigned:

```
<wssp:Integrity SignToken="false">
```

   Also, for SOA clients only, comment out the target for system headers, as shown:

```
<!-- wssp:Target>
  <wssp:DigestAlgorithm
   URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
   Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
   wls:SystemHeaders()
  </wssp:MessageParts>
</wssp:Target -->
```

5. Invoke the Web service method from the client.

6.2.2 Configuring Oracle WSM 11*g* Client and Oracle Service Bus 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle Service Bus 10*g* Web Service, perform the steps described in the following sections:

6.2.2.1 Configuring Oracle Service Bus 10*g* Web Service

1. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, copy the files to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.

2. Edit the encryption algorithm in the myEncrypt.xml file to prevent encryption compliance failure, as follows:

```
<wssp:Target>
   <wssp:EncryptionAlgorithm
     URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
   <wssp:MessageParts
     Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">
      wsp:Body()
   </wssp:MessageParts>
</wssp:Target>
```

3. Edit the mySign.xml policy file attached to the proxy service **request** only to specify that the security token is unsigned:

```
<wssp:Integrity SignToken="false">
```

   Also, for SOA clients only, comment out the target for system headers, as shown:

```
<!-- wssp:Target>
  <wssp:DigestAlgorithm
   URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
   Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
   wls:SystemHeaders()
  </wssp:MessageParts>
</wssp:Target -->
```

4. Create a Web service application that invokes the Oracle Service Bus routing service.

6.2.2.2 Configuring Oracle WSM 11*g* Client

1. Create a copy of the following policy: wss10_username_token_with_message_protection_client_policy.

> **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Set Encryption Key Reference Mechanism to issuerserial.

b. Set Recipient Encryption Key Reference Mechanism to issuerserial.

c. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.

d. Disable the Include Timestamp configuration setting.

e. Set Is Encrypted to **false**.

f. Leave the default configuration set for message signing and encryption.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to the Web service client.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Invoke the Web service from the client.

6.3 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)

This section describes how to implement SAML token (sender vouches) with message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

Configuration Prerequisites for Interoperability

Perform the following prerequisite steps for the WebLogic Server on which Oracle Service Bus is running:

1. Copy the default-keystore.jks and trust.jks files to your domain directory.

   The default-keystore.jks is used to store public and private keys for SOAP messages within the WebLogic Domain. The trust.jks is used to store private keys, digital certificates, and trusted certificate authority certificates that are used to establish and verify identity and trust in the WebLogic Server environment.

2. Invoke the WebLogic Administration Console, as described in Accessing Oracle WebLogic Administration Console.

3. Create a SAMLIdentityAsserterV2 authentication provider, as described in "Configuring Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
4. Restart WebLogic Server to add the new provider to the Administration Server's Runtime MBean server.
5. Select the authentication provider created in step 3.
6. Create and configure a SAML asserting party, as described in "SAML Identity Asserter V2: Create an Asserting Party" and "SAML Identity Asserter V2: Asserting Party: Configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Configure the SAML asserting party as follows (leave other values set to the defaults):

   - Profile: WSS/Sender-Vouches
   - Target URL: <OSB Proxy Service Endpoint URI>
   - Issuer URI: www.oracle.com

   Select the Enabled checkbox and click **Save**.
7. Create a SamlCredentialMapperV2 credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select SamlCredentialMapperV2 from the drop-down list and name the credential mapper, for example, UC2_SamlCredentialMapperV2.
8. Restart WebLogic Server.
9. Configure the credential mapper as follows (leave other values set to the defaults):

   - Issuer URI: www.oracle.com

     Note: This value is specified in the policy file.
   - Name Qualifier: oracle.com
10. Create and configure a SAML relying party, as described in "SAML Credential Mapping Provider V2: Create a Relying Party" and "SAML Credential Mapping Provider V2: Relying Party: Configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

    Configure the SAML relying party as follows (leave other values set to the defaults):

    - Profile: WSS/Sender-Vouches
    - Target URL: <Oracle WSM 11g Web Service>
    - Description: <your_description>

    Select the Enabled checkbox and click **Save**.
11. Restart WebLogic Server.

6.3.1 Configuring Oracle Service Bus 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle Service Bus 10*g* client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

6.3.1.1 Configuring Oracle WSM 11*g* Web Service

1. Create a copy of the following policy: oracle/wss10_saml_token_with_message_protection_service_policy.
   a. Set Encryption Key Reference Mechanism to issuerserial.
   b. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.
   c. Set Is Encrypted to **false** for the Username token element only.
   d. Leave the default configuration set for message signing and encryption.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. Attach the policy to the Web service.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6.3.1.2 Configuring Oracle Service Bus 10*g* Client

1. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.
2. Edit the encryption algorithm in the myEncrypt.xml file to prevent encryption compliance failure, as follows:

```
<wssp:Target>
   <wssp:EncryptionAlgorithm
     URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
   <wssp:MessageParts
     Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">
      wsp:Body()
   </wssp:MessageParts>
</wssp:Target>
```

3. Edit the mySign.xml file attached to the Oracle Service Bus business service **request** only to sign the SAML assertion by including the following target:

```
<wssp:Target>
   <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
   <wssp:MessageParts Dialect=
    "http://www.bea.com/wls90/security/policy/wsee#part">
      wls:SecurityHeader(wsse:Assertion)
   </wssp:MessageParts>
</wssp:Target>
```

4. Edit the mySign.xml file attached to the Oracle Service Bus business service **response** only to specify that the security token is unsigned, as follows:

```
<wssp:Integrity SignToken="false">
```

   Also, for SOA clients only, comment out the target for system headers, as shown:

```
<!-- wssp:Target>
  <wssp:DigestAlgorithm
   URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
   Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
```

```
      wls:SystemHeaders()
    </wssp:MessageParts>
  </wssp:Target -->
```

5. Use the custom SAML policy file defined in Example 6–1.
6. Invoke the Web service from the client.

The following defines the custom SAML policy to be used:

Example 6–1 Custom SAML Policy

```
<?xml version="1.0"?>
<wsp:Policy
   xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
   xmlns:wssp="http://www.bea.com/wls90/security/policy"
   xmlns:wsu="
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
"
   xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
   wsu:Id="custom_saml">
   <wssp:Identity xmlns:wssp="http://www.bea.com/wls90/security/policy">
      <wssp:SupportedTokens>
         <wssp:SecurityToken
          TokenType=
"http://docs.oasis-open.org/wss/2004/01/oasis-2004-01-saml-token-profile-1.0#SAMLA
ssertionID">
            <wssp:Claims>
               <wssp:ConfirmationMethod>
                  sender-vouches
               </wssp:ConfirmationMethod>
            </wssp:Claims>
         </wssp:SecurityToken>
      </wssp:SupportedTokens>
   </wssp:Identity>
   </wsp:Policy>
```

6.3.2 Configuring Oracle WSM 11*g* Client and Oracle Service Bus 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle Service Bus 10*g* Web Service, perform the steps described in the following sections:

6.3.2.1 Configuring Oracle Service Bus 10*g* Web Service

1. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.

2. Edit the encryption algorithm in the myEncrypt.xml policy file to prevent encryption compliance failure, as follows:

```
<wssp:Target>
   <wssp:EncryptionAlgorithm
     URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
   <wssp:MessageParts
     Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">
      wsp:Body()
   </wssp:MessageParts>
</wssp:Target>
```

3. Edit the mySign.xml policy file attached to the proxy service **request** only to specify that the security token is unsigned:

```
<wssp:Integrity SignToken="false">
```

Also, for SOA clients only, comment out the target for system headers, as shown:

```
<!-- wssp:Target>
  <wssp:DigestAlgorithm
   URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
   Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
   wls:SystemHeaders()
  </wssp:MessageParts>
</wssp:Target -->
```

4. Use the custom SAML policy file defined in Example 6–1.

6.3.2.2 Configuring Oracle WSM 11*g* Client

1. Create a copy of the following policy: wss10_saml_token_with_message_protection_client_policy.

> **Note:** Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

Edit the policy settings, as follows:

a. Set Encryption Key Reference Mechanism to issuerserial.

b. Set Recipient Encryption Key Reference Mechanism to issuerserial.

c. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.

d. Disable the Include Timestamp configuration setting.

e. Leave the default configuration set for message signing and encryption.

For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the policy to the Web service client.

For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Invoke the Web service from the client.

6.4 SAML or Username Token Over SSL

This section describes how to implement the SAML or username token over SSL policy in the following interoperability scenario:

Note: The interoperability scenario described in this section also applies to the SAML Token Over SSL and Username Token Over SSL policies.

Configuration Prerequisites for Interoperability

See "Configuration Prerequisites for Interoperability" on page 6-3 for configuration information on the username token.

See "Configuration Prerequisites for Interoperability" on page 6-6 for configuration information on the SAML token.

SAML Prerequisites for Interoperability

For SAML, perform the following prerequisite steps for the WebLogic Server on which Oracle Service Bus is running:

1. Create a SamlCredentialMapperV2 credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select SamlCredentialMapperV2 from the drop-down list and name the credential mapper; for example, UC2_SamlCredentialMapperV2.

2. Restart WebLogic Server.

3. Configure the credential mapper as follows (leave other values set to the defaults):

   - Issuer URI: www.oracle.com

     Note: This value is specified in the policy file.

   - Name Qualifier: oracle.com

4. Create and configure a SAML relying party, as described in "SAML Credential Mapping Provider V2: Create a Relying Party" and "SAML Credential Mapping Provider V2: Relying Party: Configuration" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Configure the SAML relying party as follows (leave other values set to the defaults):

   - Profile: WSS/Sender-Vouches
   - Target URL: <Oracle WSM 11g Web Service>
   - Description: <your_description>

   Select the Enabled checkbox and click **Save**.

5. Restart WebLogic Server.

6.4.1 Configuring Oracle Service Bus 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle Service Bus 10*g* client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

6.4.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

- If the service policy is Username Token Over SSL, set **Two Way Client Cert Behavior** to "Client Certs Requested and Not Enforced."
- If the service policy is SAML Token Over SSL, set **Two Way Client Cert Behavior** to "Client Certs Requested and Enforced."

2. Create a copy of the following policy: *wss_saml_or_username_token_over_ssl_service_policy*.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   - For *wss_username_token_over_ssl_service_policy*, disable the Create Element and Nonce configuration settings.
   - For *wss_saml_token_over_ssl_service_policy*, disable the Include Timestamp configuration setting.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Use Fusion Middleware Control to import the policy.
4. Use JDeveloper to create a simple SOA composite.
5. Attach the copy of the *wss_saml_or_username_token_over_ssl_service_policy* policy to the composite and deploy it.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6.4.1.2 Configuring Oracle Service Bus 10*g* Client

Both the SAML token client and the username token client are supported.

1. Configure the server for two-way SSL.

   For more information, see "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

   - If the client policy is the equivalent of Username Token Over SSL, then set **Two Way Client Cert Behavior** to "Client Certs Requested and Not Enforced."
   - If the client policy is the equivalent of SAML Token Over SSL, then set **Two Way Client Cert Behavior** to "Client Certs Requested and Enforced."

2. In the Oracle Service Bus console, import the WSDL for the relying party. Make sure that there is no policy attached. (Policy assertions are not allowed on this service.)
3. For SAML token, create a business service.
   a. Attach the policy shown in Example 6–1, "Custom SAML Policy" to the request.
   b. Change the WSDL from HTTP to HTTPS.
4. For username token, create a business service.
   a. Attach the *auth.xml* policy to the request.

b. Change the WSDL from HTTP to HTTPS.

5. Create a service key provider.
6. Create a proxy service, and create a route to the business service.

   In **HTTP Transport Configuration**, set Authentication to "basic."

   On the **Security** page, associate the Service key provider. This is needed for Oracle Service Bus to send the client cert to SOA.
7. Run the proxy service from the Oracle Service Bus console with the username and password.

6.5 Mutual Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement mutual authentication with message protection that conform to the WS-Security 1.0 standards:

Configuration Prerequisites for Oracle WebLogic Server

Perform the following prerequisite steps for the Oracle WebLogic Server on which Oracle Service Bus is running:

1. Copy the default-keystore.jks and trust.jks files to your domain directory.

   The default-keystore.jks is used to store public and private keys for SOAP messages within the WebLogic Domain. The trust.jks is used to store private keys, digital certificates, and trusted certificate authority certificates that are used to establish and verify identity and trust in the Oracle WebLogic Server environment.
2. Invoke the WebLogic Administration Console, as described in Accessing Oracle WebLogic Administration Console.
3. Configure the Custom Identity and Custom Trust keystores, as described in "Configuring keystores" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.
4. Configure SSL, as described in "Set up SSL" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Specify the private key alias, as required. For example: `oratest`.
5. Configure a credential mapping provider, as described in "Configure Credential Mapping Providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Create a PKICredentialMapper and configure it as follows (leave all other values set to the defaults):

   - Keystore Provider: N/A
   - Keystore Type: jks
   - Keystore File Name: default_keystore.jks

- Keystore Pass Phrase: <password>
- Confirm Keystore Pass Phrase: <password>

6. Configure Authentication, as described in "Configure Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select the **Authentication** tab and configure as follows:

   - Click **DefaultIdentityAsserter** and add **X.509** to **Chosen** active types
   - Click **Provider Specific** and configure the following:
     - Default User Name Mapper Attribute Type: CN
     - Active Types: X.509
     - Use Default User Name Mapper: True

7. Configure a token handler to specify that a client invoking a message-secured Web service uses an X.509 certificate to establish their identity. In WebLogic Administration Console, navigate to the Web Service Security page of the domain and configure the inbound and outbound messages as follows:

   Note: Only username token with message protection or mutual authentication with message protection is available at any given time. Once you enable mutual authentication with message protection, username authentication will fail.

   - Click _SERVICE_BUS_INBOUND_WEB_SERVICE_SECURITY_MBEAN_ and select the Token Handler tab.
   - Click X.509 token handler and configure the following:
     - Name: UseX509ForIdentity
     - Value: True
   - Perform the same steps for the outbound Oracle Service Bus MBean: _ SERVICE_BUS_OUTBOUND_WEB_SERVICE_SECURITY_MBEAN_

8. If the users are not added, add the Common Name (CN) user specified in the certificate as described in "Create users" in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help.
9. Restart Oracle WebLogic Server.

Configuration Prerequisites for Oracle WSM

Perform the following prerequisite steps for the Oracle WSM using Oracle WebLogic Server Administration Console:

1. Configure Authentication, as described in "Configure Authentication and Identity Assertion providers" in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

   Select the **Authentication** tab and configure as follows:

   - Click **DefaultIdentityAsserter** and add **X.509** to **Chosen** active types
   - Click **Provider Specific** and configure the following:
     - Default User Name Mapper Attribute Type: CN

- Active Types: X.509
- Use Default User Name Mapper: True

2. If the users are not added, add the Common Name (CN) user specified in the certificate as described in "Create users" in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help.
3. Restart Oracle WebLogic Server.

6.5.1 Configuring Oracle Service Bus 10*g* Client and Oracle WSM 11*g* Web Service

To configure Oracle Service Bus 10*g* client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

- "Configuring Oracle WSM 11g Web Service" on page 6-15
- "Configuring Oracle Service Bus 10g Client" on page 6-15

6.5.1.1 Configuring Oracle WSM 11g Web Service

1. Create and deploy a SOA composite.
2. Create a copy of the following policy: wss10_x509_token_with_message_protection_service_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   Edit the policy settings, as follows:

   a. Set Encryption Key Reference Mechanism to issuerserial.
   b. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
3. Attach the policy to the Web service.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6.5.1.2 Configuring Oracle Service Bus 10g Client

1. Create an Oracle Service Bus business service.
2. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, copy the files to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.
3. Attach the X.509 policy to the Oracle Service Bus business service **request**.

```
<wsp:Policy
  xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
  xmlns:wssp="http://www.bea.com/wls90/security/policy"

xmlns:s0="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-ut
ility-1.0.xsd"
```

```
  s0:Id="X509Auth">
        <wssp:Identity xmlns:wssp="http://www.bea.com/wls90/security/policy">
            <wssp:SupportedTokens>
                <wssp:SecurityToken
TokenType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-p
rofile-1.0#X509v3"/>
            </wssp:SupportedTokens>
        </wssp:Identity>
</wsp:Policy>
```

4. Attach the Sign.xml policy file to the Oracle Service Bus business service **request**.
5. Edit the myEncrypt.xml policy and attach it to the Oracle Service Bus business service **request**.

```
<?xml version="1.0"?>
<wsp:Policy
 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
 xmlns:wssp="http://www.bea.com/wls90/security/policy"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-u
tility-1.0.xsd"
 xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
 wsu:Id="X509Encrypt">
  <wssp:Confidentiality>
    <wssp:KeyWrappingAlgorithm URI="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
    <wssp:Target>
      <wssp:EncryptionAlgorithm
URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
      <wssp:MessageParts
Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">wsp:Body()</wssp:Message
Parts>
    </wssp:Target>
    <wssp:KeyInfo/>
  </wssp:Confidentiality>
</wsp:Policy>
```

6. Edit the mySign.xml policy file attached to the Oracle Service Bus business service **response** to specify that the security token is unsigned:

```
<wssp:Integrity SignToken="false">
```

Also, for SOA clients only, comment out the target for system headers, as shown:

```
  <?xml version="1.0"?>
<wsp:Policy
  xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
  xmlns:wssp="http://www.bea.com/wls90/security/policy"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"
  xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
  wsu:Id="X509Sign">
  <wssp:Integrity SignToken="false">
    <wssp:SignatureAlgorithm URI="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
    <wssp:CanonicalizationAlgorithm
 URI="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    <!--wssp:Target>
      <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
      <wssp:MessageParts
 Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
       wls:SystemHeaders()
```

```
      </wssp:MessageParts>
    </wssp:Target-->
    <wssp:Target>
      <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
      <wssp:MessageParts
 Dialect="http://www.bea.com/wls90/security/policy/wsee#part">
        wls:SecurityHeader(wsu:Timestamp)
      </wssp:MessageParts>
    </wssp:Target>
    <wssp:Target>
      <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
      <wssp:MessageParts
 Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">
      wsp:Body()
      </wssp:MessageParts>
    </wssp:Target>
   </wssp:Integrity>
  <wssp:MessageAge/>
</wsp:Policy>
```

7. Attach the myEncrypt.xml policy file from Step 6 to the Oracle Service Bus business service **response**.
8. Create a ServiceKeyProvider.
9. Specify Encryption Key and Digital Signature Key, as required.

   You must use different keys on the Oracle WSM and Oracle Service Bus servers. You can use the same key for encryption and signing, if desired.
10. Create a proxy service, and create a route to the business service.

    On the **Security** page, associate the Service key provider. This is needed for Oracle Service Bus to send the client certificate to SOA.
11. Run the proxy service from the Oracle Service Bus console.

6.5.2 Configuring Oracle WSM 11*g* Client and Oracle Service Bus 10*g* Web Service

To configure Oracle WSM 11*g* client and Oracle Service Bus 10*g* Web Service, perform the steps described in the following sections:

- "Configuring Oracle Service Bus 10g Web Service" on page 6-17
- "Configuring Oracle WSM 11g Client" on page 6-19

6.5.2.1 Configuring Oracle Service Bus 10g Web Service

1. Create a Oracle Service Bus proxy service.
2. Create a copy of the Encrypt.xml and Sign.xml policy files.

   For example, to myEncrypt.xml and mySign.xml. It is not recommended to edit the predefined policy files directly.
3. Attach the X.509 policy as described in "Configuring Oracle Service Bus 10g Client" on page 6-15 to the proxy service **request**.
4. Edit the mySign.xml policy file attached to the proxy service **request** and comment out the target for system headers and timestamp, as shown:

```
<?xml version="1.0"?>
<wsp:Policy
```

```
  xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
  xmlns:wssp="http://www.bea.com/wls90/security/policy"

xmlns:s0="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-ut
ility-1.0.xsd"
  s0:Id="X509SignRequest">
  <wssp:Integrity
 xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
 xmlns:wssp="http://www.bea.com/wls90/security/policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
  <wssp:SignatureAlgorithm URI="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
  <wssp:CanonicalizationAlgorithm URI="http://www.w3.org/2001/10/xml-exc-c14n#"
 />
  <!-- wssp:Target>
  <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
Dialect="http://www.bea.com/wls90/security/policy/wsee#part">wls:SystemHeaders
()</wssp:MessageParts>
  </wssp:Target -->
  <!-- wssp:Target>
  <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
Dialect="http://www.bea.com/wls90/security/policy/wsee#part">wls:SecurityHeader
(wsu:Timestamp)</wssp:MessageParts>
  </wssp:Target -->
  <wssp:Target>
  <wssp:DigestAlgorithm URI="http://www.w3.org/2000/09/xmldsig#sha1" />
  <wssp:MessageParts
Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">wsp:Body()</wssp:Message
Parts>
  </wssp:Target>
</wsp:Policy>
```

5. Edit the encryption algorithm in the myEncrypt.xml file attached to the proxy service **request** as follows:

```
<?xml version="1.0"?>
<wsp:Policy
 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
 xmlns:wssp="http://www.bea.com/wls90/security/policy"

xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-u
tility-1.0.xsd"
 xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
 wsu:Id="X509Encrypt">
  <wssp:Confidentiality>
    <wssp:KeyWrappingAlgorithm URI="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
    <wssp:Target>
      <wssp:EncryptionAlgorithm
URI="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
      <wssp:MessageParts
Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">wsp:Body()</wssp:Message
Parts>
    </wssp:Target>
    <wssp:KeyInfo/>
  </wssp:Confidentiality>

</wsp:Policy>
```

6. Attach mySign.xml and myEncrypt.xml policy files from the previous steps to the proxy service **response**.
7. Create a Service Key Provider.

6.5.2.2 Configuring Oracle WSM 11g Client

1. Create a copy of the following policy: wss10_x509_token_with_message_ protection_client_policy.

   Note: Oracle recommends that you do not change the predefined policies so that you will always have a known set of valid policies to work with.

   In Enterprise Manager, edit the policy settings, as follows:

   a. Set Encryption Key Reference Mechanism to issuerserial.
   b. Set Recipient Encryption Key Reference Mechanism to issuerserial.
   c. Set Algorithm Suite to Basic128Rsa15 to match the algorithm suite used for Oracle Service Bus.
   d. Disable the Include Timestamp configuration setting.

   For more information, see "Creating a Web Service Policy from an Existing Policy" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. In Enterprise Manager, specify keystore.recipient.alias in the client configuration. Ensure that the keystore.recipient.alias keys specified for the client exist as trusted certificate entry in the trust store configured for the Web service.
3. Attach the policy to the Web service client.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
4. Invoke the Web service from the client.

7

Interoperability with Axis 1.4 and WSS4J 1.5.8 Security Environments

This chapter contains the following sections:

7.1 Overview of Interoperability With Axis 1.4 and WSS4J 1.5.8 Security Environments

In Axis 1.4 and WSS4J 1.5.8, you configure your security environment for inbound and outbound requests using handlers and deployment descriptors. For more information, see the *Axis Deployment Tutorial* at `http://ws.apache.org/wss4j/axis.html`.

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see "Predefined Policies" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about configuring and attaching policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

Table 7–1the most common Axis and WSS4J interoperability scenarios based on the following security requirements: authentication, message protection, and transport.

For more information about:

- Configuring and attaching Oracle WSM 11*g* policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
- Configuring and attaching policies on Axis and WSS4J, see the *Axis Deployment Tutorial* at `http://ws.apache.org/wss4j/axis.html`.

Table 7–1 Interoperability with Axis and WSS4J Security Environments

Interoperability Scenario	Client—>Web Service	Oracle WSM 11g Policies	Axis/WSS4J Policies
"Username Token with Message Protection (WS-Security 1.0)" on page 7-3	Axis/WSS4J—>Oracle WSM 11*g*	oracle/wss10_username_token_with_message_protection_service_policy	UsernameToken Timestamp Signature Encrypt
"Username Token with Message Protection (WS-Security 1.0)" on page 7-3	Oracle WSM 11*g*—>Axis/WSS4J	oracle/wss10_username_token_with_message_protection_client_policy	UsernameToken Timestamp Signature Encrypt
"SAML Token with Message Protection (WS-Security 1.0)" on page 7-6	Axis/WSS4J—>Oracle WSM 11*g*	oracle/wss10_saml_token_with_message_protection_service_policy	SAMLTokenUnsigned Timestamp Signature Encrypt
"SAML Token with Message Protection (WS-Security 1.0)" on page 7-6	Oracle WSM 11*g*—>Axis/WSS4J	oracle/wss10_saml_token_with_message_protection_client_policy	SAMLTokenUnsigned Timestamp Signature Encrypt
"Username Token Over SSL" on page 7-9	Axis/WSS4J—>Oracle WSM 11*g*	oracle/wss_username_token_over_ssl_service_policy	UsernameToken Timestamp
"Username Token Over SSL" on page 7-9	Oracle WSM 11*g*—>Axis/WSS4J	oracle/wss_username_token_over_ssl_client_policy	Timestamp UsernameToken
"SAML Token (Sender Vouches) Over SSL" on page 7-11	Axis/WSS4J—>Oracle WSM 11*g*	oracle/wss_saml_token_over_ssl_service_policy	SAMLTokenUnsigned Timestamp
"SAML Token (Sender Vouches) Over SSL" on page 7-11	Oracle WSM 11*g*—>Axis/WSS4J	oracle/wss_saml_token_over_ssl_client_policy	Timestamp SAMLTokenUnsigned

7.2 Required Files for Interoperability With Axis and WSS4J

Perform the following steps to create the handler and property files that are required in each of the Axis and WSS4J interoperability scenarios:

1. Create and compile a password callback class, PWCallback.java, that can resolve passwords required by username and keystore aliases.

   The deployment descriptors defined in the following sections, contain username information, but not password information. As a best practice, you should not store sensitive information such as passwords in clear text within the deployment descriptor. To obtain the password, the Axis handler calls the password callback class. This mechanism is similar to JAAS. For more information, see the WSS4J documentation at `http://ws.apache.org/wss4j`.

2. Create the keystore properties file, crypto.properties, as shown below. Include this file in the classes directory.

```
org.apache.ws.security.crypto.provider=org.apache.ws.security.components.crypto
.Merlin
org.apache.ws.security.crypto.merlin.keystore.type=jks
org.apache.ws.security.crypto.merlin.keystore.password=welcome1
org.apache.ws.security.crypto.merlin.file=default-keystore.jks
```

3. Create the saml.properties file, required for SAML interoperability scenarios only, as shown below.

```
org.apache.ws.security.saml.issuerClass=org.apache.ws.security.saml.SAMLIssuerI
mpl
org.apache.ws.security.saml.issuer.cryptoProp.file=crypto.properties
org.apache.ws.security.saml.issuer.key.name=orakey
org.apache.ws.security.saml.issuer.key.password=orakey
org.apache.ws.security.saml.issuer=www.oracle.com
org.apache.ws.security.saml.subjectNameId.name=weblogic
org.apache.ws.security.saml.authenticationMethod=password
```

```
org.apache.ws.security.saml.confirmationMethod=senderVouches
```

7.3 Username Token with Message Protection (WS-Security 1.0)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

- "Configuring Axis and WSS4J Client and Oracle WSM 11g Web Service" on page 7-3
- "Configuring Oracle WSM 11g Client and Axis and WSS4J Web Service" on page 7-4

7.3.1 Configuring Axis and WSS4J Client and Oracle WSM 11*g* Web Service

To configure Axis and WSS4J client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

7.3.1.1 Configuring Oracle WSM 11*g* Web Service

1. Attach the following policy to the Web service: oracle/wss10_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Deploy the Web service.

7.3.1.2 Configuring Axis and WSS4J Client

1. Build your Web service client proxy.
2. Create the password callback class, PWCallback.java, and keystore properties file, crypto.properties, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
3. Include the keystore file (for example, default-keystore.jks) and crypto.properties file directly under the classes folder.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

4. Edit the deployment descriptor, client_deploy.wsdd, similar to Example 7–1.

   In the example, the receiver decrypts, verifies, and validates the username token; the sender inserts a username token, timestamp, signs the body, username token, and timestamp, and encrypts the body and username token. As shown in the example, the encryption key transport is overridden to match the Oracle WSM default requirements

5. Set the following property within the client code to use the deployment descriptor defined in the previous step.

   ```
   System.setProperty("axis.ClientConfigFile", "client_deploy.wsdd");
   ```

6. Deploy the Web service client.

The following shows an example of the client_deploy.wsdd deployment descriptor.

Example 7–1 client_deploy.wsdd Deployment Descriptor

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
            xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
 <transport name="http"
  pivot="java:org.apache.axis.transport.http.HTTPSender"/>
  <globalConfiguration >
   <!-- wss10_username_token_with_message_protection -->
   <requestFlow>
     <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
       <parameter name="passwordCallbackClass"
        value="com.oracle.xmlns.ConfigOverride_jws.CO_SOA.BPELProcess1.PWCallback"/>
       <parameter name="passwordType" value="PasswordText"/>
       <parameter name="user" value="weblogic"/>
       <parameter name="action" value="UsernameToken Timestamp Signature Encrypt"/>
       <parameter name="encryptionKeyTransportAlgorithm"
        value="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
       <parameter name="encryptionKeyIdentifier" value="DirectReference" />
       <parameter name="encryptionPropFile" value="crypto.properties" />
       <parameter name="encryptionUser" value="orakey" />
       <parameter name="encryptionParts" value=
    "{Element}{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}
     UsernameToken;{Content}{http://schemas.xmlsoap.org/soap/envelope/}Body" />
       <parameter name="signatureUser" value="orakey" />
       <parameter name="signaturePropFile" value="crypto.properties" />
       <parameter name="signatureKeyIdentifier" value="DirectReference" />
       <parameter name="signatureParts" value=
    "{Element}{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}
UsernameToken;{Element}{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd}
Timestamp;{Element}{http://schemas.xmlsoap.org/soap/envelope/}Body" />
     </handler>
   </requestFlow>
   <responseFlow>
     <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
       <parameter name="passwordCallbackClass" value="com.oracle.xmlns.ConfigOverride_jws.CO
_SOA.BPELProcess1.PWCallback"/>
       <parameter name="action" value="Timestamp Signature Encrypt" />
       <parameter name="signaturePropFile" value="crypto.properties" />
       <parameter name="decryptionPropFile" value="crypto.properties" />
       <parameter name="enableSignatureConfirmation"  value="false" />
     </handler>
   </responseFlow>
  </globalConfiguration >
</deployment>
```

7.3.2 Configuring Oracle WSM 11*g* Client and Axis and WSS4J Web Service

To configure Oracle WSM 11*g* client and Axis and WSS4J Web Service, perform the steps described in the following sections:

7.3.2.1 Configuring Axis and WSS4J Web Service

1. Build your Web service.
2. Create the password callback class, PWCallback.java, and keystore properties file, crypto.properties, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.

3. Include the keystore file (for example, default-keystore.jks) and crypto.properties file directly under the classes folder.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

4. Edit the deployment descriptor, server_deploy.wsdd, as shown in Example 7–2.

   In the example, the receiver decrypts, verifies, and validates the username token; the sender inserts a username token, timestamp, signs the body, username token, and timestamp, and encrypts the body and username token. As shown in the example, the encryption key transport is overridden to match the Oracle WSM default requirements.

   Note: WSS4J enforces an order to the elements in the header. Ensure action ordering is updated in server_deploy.wsdd as shown in Example 7–2.

5. Deploy the Web service.

7.3.2.2 Configuring Oracle WSM 11*g* Client

1. Attach the following policy to the Web service: oracle/wss10_username_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. For JSE clients only, configure the Web service client properties, as follows:

   Note: This step is not required for JEE clients.

```
myPort.setProperty(ClientConstants.WSS_KEYSTORE_TYPE,"JKS");
myPort.setProperty(ClientConstants.WSS_KEYSTORE_LOCATION,
 "/keystore-path/default-keystore.jks");
myPort.setProperty(ClientConstants.WSS_KEYSTORE_PASSWORD, "welcome1");
myPort.setProperty(ClientConstants.WSS_RECIPIENT_KEY_ALIAS,"orakey");
...
```

   Where `setProperty` is defined as follows:

```
public void setProperty(String name, String value) {
   ((Stub) _port)._setProperty(name, value);
}
```

3. Deploy the Web service client.

The following shows an example of the server_deploy.wsdd deployment descriptor.

Example 7–2 server_deploy.wsdd Deployment Descriptor

```
<ns1:service name="HelloWorld" provider="java:RPC" style="wrapped" use="literal">
<!-- wss10_username_token_with_message_protection -->
<requestFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
      <parameter name="passwordCallbackClass" value="PWCallback1"/>
      <parameter name="user" value="wss4j"/>
      <parameter name="action" value="Signature UsernameToken Timestamp Encrypt"/>
      <parameter name="signaturePropFile" value="crypto.properties" />
```

```
      <parameter name="decryptionPropFile" value="crypto.properties" />
   </handler>
</requestFlow>
<responseFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
       <parameter name="passwordCallbackClass" value="PWCallback1"/>
       <parameter name="user" value="orakey"/>
       <parameter name="action" value="Timestamp Signature Encrypt"/>
       <parameter name="encryptionKeyTransportAlgorithm"
          value="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
       <parameter name="signaturePropFile" value="crypto.properties" />
       <parameter name="signatureKeyIdentifier" value="DirectReference" />
       <parameter name="signatureParts"
value="{Element}{http://schemas.xmlsoap.org/soap/envelope/}Body;{Element}
{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd}Timestamp" />
       <parameter name="encryptionKeyIdentifier" value="DirectReference" />
   </handler>
</responseFlow>
</ns1:service>
```

7.4 SAML Token with Message Protection (WS-Security 1.0)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.0 standard in the following interoperability scenarios:

- "Configuring Axis and WSS4J Client and Oracle WSM 11g Web Service" on page 7-6
- "Configuring Oracle WSM 11g Client and Axis and WSS4J Web Service" on page 7-8

7.4.1 Configuring Axis and WSS4J Client and Oracle WSM 11*g* Web Service

To configure Axis and WSS4J client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

7.4.1.1 Configuring Oracle WSM 11*g* Web Service

1. Attach the following policy to the Web service: oracle/wss10_saml_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Deploy the Web service.

7.4.1.2 Configuring Axis and WSS4J Client

1. Build your Web service client proxy.
2. Create the password callback class, PWCallback.java, keystore properties file, crypto.properties file, and saml.properties file, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
3. Include the keystore file (for example, default-keystore.jks) and crypto.properties file directly under the classes folder.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.

4. Edit the deployment descriptor, client_deploy.wsdd, similar to Example 7–3.

   In the example, the receiver decrypts, verifies, and validates the SAML token; the sender inserts a SAML token, timestamp, signs the body, SAML token, and timestamp, and encrypts the body. As shown in the example, the encryption key transport is overridden to match the Oracle WSM default requirements.

5. Set the following property within the client code to use the deployment descriptor defined in the previous step.

   ```
   System.setProperty("axis.ClientConfigFile", "client_deploy.wsdd");
   ```

6. Deploy the Web service client.

The following shows an example of the client_deploy.wsdd deployment descriptor.

Example 7–3 client_deploy.wsdd Deployment Descriptor

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
            xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
 <transport name="http"
  pivot="java:org.apache.axis.transport.http.HTTPSender"/>
  <globalConfiguration >
<!-- wss10_saml_token_with_message_protection -->
    <requestFlow>
      <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
        <parameter name="passwordCallbackClass"
         value="com.oracle.xmlns.ConfigOverride_jws.CO_SOA.BPELProcess1.PWCallback"/>
        <parameter name="passwordType" value="PasswordText"/>
        <parameter name="user" value="weblogic"/>
        <parameter name="action" value="Timestamp Signature SAMLTokenSigned Encrypt"/>
        <parameter name="samlPropFile" value="saml.properties"/>
        <parameter name="encryptionKeyTransportAlgorithm"
         value="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
        <parameter name="encryptionKeyIdentifier" value="DirectReference" />
        <parameter name="encryptionPropFile" value="crypto.properties" />
        <parameter name="encryptionUser" value="orakey" />
        <parameter name="encryptionParts"
         value="{Content}{http://schemas.xmlsoap.org/soap/envelope/}Body" />
        <parameter name="signatureUser" value="orakey" />
        <parameter name="signaturePropFile" value="crypto.properties" />
        <parameter name="signatureKeyIdentifier" value="DirectReference" />
        <parameter name="signatureParts" value="{Element}
          {http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd}
          Timestamp;{Element}
          {http://schemas.xmlsoap.org/soap/envelope/}Body" />
      </handler>
    </requestFlow>
    <responseFlow>
      <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
        <parameter name="passwordCallbackClass"
         value="com.oracle.xmlns.ConfigOverride_jws.CO_SOA.BPELProcess1.PWCallback" />
        <parameter name="action" value="Timestamp Signature Encrypt" />
        <parameter name="signaturePropFile" value="crypto.properties" />
        <parameter name="decryptionPropFile" value="crypto.properties" />
        <parameter name="enableSignatureConfirmation" value="false" />
    </handler>
  </responseFlow>
  </globalConfiguration >
</deployment>
```

7.4.2 Configuring Oracle WSM 11*g* Client and Axis and WSS4J Web Service

To configure Oracle WSM 11*g* client and Axis and WSS4J Web Service, perform the steps described in the following sections:

7.4.2.1 Configuring Axis and WSS4J Web Service

1. Build your Web service.
2. Create the password callback class, PWCallback.java, keystore properties file, crypto.properties file, and saml.properties file as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
3. Include the keystore file (for example, default-keystore.jks) and crypto.properties file directly under the classes folder.

   Ensure that you are using keystore with v3 certificates. By default, the JDK 1.5 keytool generates keystores with v1 certificates.
4. Edit the deployment descriptor, server_deploy.wsdd, as shown in Example 7–4.

   In the example, the receiver decrypts, verifies, and validates the SAML token; the sender inserts a SAML token, timestamp, signs the body, SAML token, and timestamp, and encrypts the body. As shown in the example, the encryption key transport is overridden to match the Oracle WSM default requirements.

   Note: WSS4J enforces an order to the elements in the header. Ensure action ordering is updated in server_deploy.wsdd as shown in Example 7–4.
5. Deploy the Web service.

7.4.2.2 Configuring Oracle WSM 11*g* Client

1. Attach the following policy to the Web service: oracle/wss10_saml_token_with_message_protection_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
2. For JSE clients only, configure the Web service client properties, as follows:

   Note: This step is not required for JEE clients.

   ```
   myPort.setProperty(ClientConstants.WSS_KEYSTORE_TYPE,"JKS");
   myPort.setProperty(ClientConstants.WSS_KEYSTORE_LOCATION,
    "/keystore-path/default-keystore.jks");
   myPort.setProperty(ClientConstants.WSS_KEYSTORE_PASSWORD, "welcome1");
   myPort.setProperty(ClientConstants.WSS_RECIPIENT_KEY_ALIAS,"orakey");
   ...
   ```

   Where `setProperty` is defined as follows:

   ```
   public void setProperty(String name, String value) {
      ((Stub) _port)._setProperty(name, value);
   }
   ```
3. Deploy the Web service client.

The following shows an example of the server_deploy.wsdd deployment descriptor.

Example 7–4 server_deploy.wsdd Deployment Descriptor

```
<ns1:service name="HelloWorld" provider="java:RPC" style="wrapped" use="literal">
<!-- wss10_username_token_with_message_protection -->
<requestFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
      <parameter name="passwordCallbackClass" value="PWCallback1"/>
      <parameter name="user" value="wss4j"/>
      <parameter name="action" value="Signature SAMLTokenUnsigned Timestamp Encrypt"/>
      <parameter name="signaturePropFile" value="crypto.properties" />
      <parameter name="decryptionPropFile" value="crypto.properties" />
   </handler>
</requestFlow>
<responseFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
       <parameter name="passwordCallbackClass" value="PWCallback1"/>
       <parameter name="user" value="orakey"/>
       <parameter name="action" value="Timestamp Signature Encrypt"/>
       <parameter name="encryptionKeyTransportAlgorithm"
          value="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
       <parameter name="signaturePropFile" value="crypto.properties" />
       <parameter name="signatureKeyIdentifier" value="DirectReference" />
       <parameter name="signatureParts"
value="{Element}{http://schemas.xmlsoap.org/soap/envelope/}Body;{Element}
{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd}Timestamp" />
       <parameter name="encryptionKeyIdentifier" value="DirectReference" />
   </handler>
</responseFlow>
</ns1:service>
```

7.5 Username Token Over SSL

This section describes how to implement username token over SSL in the following interoperability scenarios:

- "Configuring Axis and WSS4J Client and Oracle WSM 11g Web Service" on page 7-9
- "Configuring Oracle WSM 11g Client and Axis and WSS4J Web Service" on page 7-10

7.5.1 Configuring Axis and WSS4J Client and Oracle WSM 11*g* Web Service

To configure Axis and WSS4J client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

7.5.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the following policy to the Web service: oracle/wss_username_token_over_ssl_service_policy.

For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Deploy the Web service.

7.5.1.2 Configuring Axis and WSS4J Client

1. Build your Web service client proxy.
2. Create the password callback class, PWCallback.java, and keystore properties file, crypto.properties, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
3. Edit the deployment descriptor, client_deploy.wsdd, similar the example below. In the example, the receiver validates the username token and timestamp; the sender inserts a timestamp.

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
             xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
 <transport name="http"
  pivot="java:org.apache.axis.transport.http.HTTPSender"/>
<globalConfiguration >
<!-- wss_username_token -->
<requestFlow >
   <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
      <parameter name="action" value="UsernameToken Timestamp"/>
      <parameter name="user" value="weblogic"/>
      <parameter name="passwordCallbackClass"
       value="com.oracle.xmlns.ConfigOverride_jws.CO_
SOA.BPELProcess1.PWCallback"/>
      <parameter name="passwordType" value="PasswordText"/>
  </handler>
</requestFlow >
</globalConfiguration >
</deployment>
```

4. Set the following property within the client code to use the deployment descriptor defined in the previous step.

```
System.setProperty("axis.ClientConfigFile", "client_deploy.wsdd");
```

5. Deploy the Web service client.

7.5.2 Configuring Oracle WSM 11*g* Client and Axis and WSS4J Web Service

To configure Oracle WSM 11*g* client and Axis and WSS4J Web service, perform the steps described in the following sections:

7.5.2.1 Configuring Axis and WSS4J Web Service

1. Configure the server for SSL.
2. Build your Web service.
3. Create the password callback class, PWCallback.java, and crypto.properties file, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
4. Edit the deployment descriptor, server_deploy.wsdd, similar to the example below. In the example, the receiver validates the username token and the timestamp; the sender inserts a timestamp.

```
<ns1:service name="HelloWorld" provider="java:RPC" style="wrapped"
 use="literal">
<!-- wss_username_token_over_ssl -->
  <requestFlow>
    <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
      <parameter name="passwordCallbackClass" value="PWCallback1"/>
      <parameter name="action" value="Timestamp UsernameToken"/>
    </handler>
  </requestFlow>
  <responseFlow>
     <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
      <parameter name="action" value="Timestamp"/>
      </handler>
  </responseFlow>
</ns1:service>
```

5. Deploy the Web service.

7.5.2.2 Configuring Oracle WSM 11*g* Client

1. Attach the following policy to the Web service client: wss_username_token_over_ssl_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. For JSE clients only, configure the Web service client properties, as shown below. The username and password must be set by the client for generating the username token.

   Note: This step is not required for JEE clients.

   ```
   myPort.setUsername("wss4j");
   myPort.setPassword("security"););
   ```

3. Deploy the Web service client.

   When running the client, include the following client system property, where *default-keystore.jks* specifies the keystore that contains the certificate corresponding to the server certificate.

   ```
   -Djavax.net.ssl.trustStore=default-keystore.jks
   ```

7.6 SAML Token (Sender Vouches) Over SSL

This section describes how to implement SAML token (sender vouches) over SSL in the following interoperability scenarios:

- "Configuring Axis and WSS4J Client and Oracle WSM 11g Web Service" on page 7-11
- "Configuring Oracle WSM 11g Client and Axis and WSS4J Web Service" on page 7-12

7.6.1 Configuring Axis and WSS4J Client and Oracle WSM 11*g* Web Service

To configure Axis and WSS4J client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

7.6.1.1 Configuring Oracle WSM 11*g* Web Service

1. Configure the server for SSL.

   For more information, see "Configuring SSL on WebLogic Server (One-Way)" and "Configuring SSL on WebLogic Server (Two-Way)" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Attach the following policy to the Web service: wss_saml_token_over_ssl_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

3. Deploy the Web service.

7.6.1.2 Configuring Axis and WSS4J Client

1. Build your Web service client proxy.
2. Create the password callback class, PWCallback.java; keystore properties file, crypto.properties; and SAML properties file, saml.properties, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
3. Edit the deployment descriptor, client_deploy.wsdd, similar the example below. In the example, the receiver validates the SAML token and timestamp; the sender inserts a timestamp.

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
             xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
<transport name="http"
  pivot="java:org.apache.axis.transport.http.HTTPSender"/>
 <globalConfiguration >
<!-- wss_saml_token -->
<requestFlow >
   <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
      <parameter name="action" value="SAMLTokenSigned Timestamp"/>
      <parameter name="samlPropFile" value="saml.properties"/>
      <parameter name="user" value="weblogic"/>
      <parameter name="passwordCallbackClass"
 value="com.oracle.xmlns.ConfigOverride_jws.CO_SOA.BPELProcess1.PWCallback"/>
      <parameter name="passwordType" value="PasswordText"/>
      <parameter name="signatureUser" value="orakey" />
      <parameter name="signatureKeyIdentifier" value="DirectReference" />
      <parameter name="signaturePropFile" value="crypto.properties" />
   </handler>
</requestFlow >
</globalConfiguration >
</deployment>
```

4. Set the following property within the client code to use the deployment descriptor defined in the previous step.

```
System.setProperty("axis.ClientConfigFile", "client_deploy.wsdd");
```

5. Deploy the Web service client.

7.6.2 Configuring Oracle WSM 11*g* Client and Axis and WSS4J Web Service

To configure Oracle WSM 11*g* client and Axis and WSS4J Web service, perform the steps described in the following sections:

7.6.2.1 Configuring Axis and WSS4J Web Service

1. Configure the server for SSL.
2. Build your Web service.
3. Create the password callback class, PWCallback.java, and crypto.properties file, as described in "Required Files for Interoperability With Axis and WSS4J" on page 7-2.
4. Edit the deployment descriptor, server_deploy.wsdd, similar to the example below.

   In the example, the receiver validates the SAML token and the timestamp; the sender inserts a timestamp.

```
<ns1:service name="HelloWorld" provider="java:RPC" style="wrapped"
 use="literal">
<!-- wss_saml_token_over_ssl -->
<requestFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllReceiver">
      <parameter name="passwordCallbackClass" value="PWCallback1"/>
      <parameter name="action" value="Timestamp SAMLTokenUnsigned"/>
   </handler>
</requestFlow>
<responseFlow>
   <handler type="java:org.apache.ws.axis.security.WSDoAllSender" >
      <parameter name="action" value="Timestamp"/>
   </handler>
</responseFlow>
</ns1:service>
```

5. Deploy the Web service.

7.6.2.2 Configuring Oracle WSM 11g Client

1. Attach the following policy to the Web service client: wss_saml_token_over_ssl_client_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Service Clients" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. For JSE clients, configure the Web service client properties, as shown below. The username must be set by the client for generating the SAML assertion.

   Note: This step is not required for JEE clients.v

```
myPort.setUsername("wss4j");
```

3. Deploy the Web service client.

   When running the client, include the following client system property, where *default-keystore.jks* specifies the keystore that contains the certificate corresponding to the server certificate.

```
-Djavax.net.ssl.trustStore=default-keystore.jks
```

8

Interoperability with Oracle GlassFish Enterprise Server Release 3.0.1

This chapter contains the following sections:

8.1 Overview of Interoperability With Oracle GlassFish Security Environments

Oracle GlassFish Enterprise Server Release 3.0.1 is an open source application server for the Java EE platform. Metro is an open-source Web service stack that is a part of Oracle GlassFish Enterprise Server.

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see "Predefined Policies" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*. For more information about configuring and attaching policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

For more information about:

- Configuring and attaching Oracle WSM 11*g* policies, see "Configuring Policies" and "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.
- Configuring Oracle GlassFish, see `http://docs.sun.com/app/docs/coll/1343.9`.
- Configuring Metro Web services, see `http://metro.java.net/guide/`

8.2 Username Token with Message Protection (WS-Security 1.1)

This section describes how to implement username token with message protection that conforms to the WS-Security 1.1 standard in the following interoperability scenarios:

8.2.1 Configuring GlassFish Client and Oracle WSM 11*g* Web Service

To configure GlassFish client and Oracle WSM 11*g* Web service, perform the steps described in the following sections:

8.2.1.1 Configuration Prerequisites for Interoperability

Perform the following prerequisite steps:

1. Create a default-keystore.jks file with the following command:

```
$JAVA_HOME/bin/keytool -genkeypair -alias orakey -keypass welcome -keyalg RSA
 -dname "CN=orakey, O=oracle C=us" -keystore default-keystore.jks -storepass
 welcome
```

2. Copy default-keystore.jks to the domain's fmwconfig directory.
3. Create a file user in GlassFish with the following command:

```
$<GLASSFISHV3_HOME>/glassfish/bin/asadmin create-file-user
```

   For more information, see http://docs.sun.com/app/docs/doc/820-4495/6nfv4mkkl?a=view.

4. Import orakey from default-keystore.jks into GlassFish keystore and truststore. These are located in the directory `<domain-dir>/config`

```
$JAVA_HOME/bin/keytool -importkeystore -srckeystore
 <path-to>/default-keystore.jks -destkeystore
 <path-to-gf-domain>/config/cacerts.jks -srcalias  orakey -destalias orakey
 -srckeypass welcome -destkeypass changeit
```

5. Copy jps-config.xml and default-keystore.jks from the domain's fmwconfig directory into a local folder.

8.2.1.2 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service.
2. Attach the following policy to the Web service: oracle/wss11_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

8.2.1.3 Configuring GlassFish/Metro Client

1. Using NetBeans, create a Metro client by selecting **New Project > Java > Java Application**.

1. Provide a project name and location and select Finish.
2. Right click on the project. Select **New > Web service Client**. Follow the wizard and provide WSDL URL for service deployed in WebLogic.
3. Select **Edit Web Services Attributes**.
4. Check **Use Development Defaults** to include Metro libraries into the project.
5. Uncheck **Use Development Defaults**. Provide username subject and password.
6. For a Metro SE client:

a. Edit the truststore configuration. Select the same default-keystore.jks created in "Configuration Prerequisites for Interoperability" on page 8-2.

b. Drag and drop the Web service operation into main class, main method.

c. Right click on the project and choose run to execute the project.

7. For a Metro Java EE client:

a. Drag and drop the Web service operation into EJB or Servlet to invoke.

b. Deploy the application into GlassFish and invoke the Web service.

8.2.2 Configuring Oracle WSM 11*g* Client and GlassFish Web Service

To configure Oracle WSM 11*g* client and GlassFish Web service, perform the steps described in the following sections:

8.2.2.1 Configuration Prerequisites for Interoperability

Perform the following prerequisite steps:

1. Create a default-keystore.jks file with the following command:

```
$JAVA_HOME/bin/keytool -genkeypair -alias orakey -keypass welcome -keyalg RSA
 -dname "CN=orakey, O=oracle C=us" -keystore default-keystore.jks -storepass
 welcome
```

2. Copy default-keystore.jks to the domain's fmwconfig directory.

3. Save the credentials in credential store using WLST commands. For example:

```
$<ORACLE_HOME>/common/bin/wlst.sh
> connect()
> createCred(map="oracle.wsm.security", key="keystore-csf-key",
 user="keystore", password="welcome")
> createCred(map="oracle.wsm.security", key="sign-csf-key", user="orakey",
 password="welcome")
> createCred(map="oracle.wsm.security", key="enc-csf-key", user="orakey",
 password="welcome")
>createCred(map="oracle.wsm.security", key="glassfish.credentials" ,
 user="wlsUser" , password="welcome1" , description="Glassfish user
 credentials");
```

A file cwallet.sso is created in the directory `DOMAIN_HOME/config/fmwconfig`

4. Create a file user in GlassFish with the following command:

```
$<GLASSFISHV3_HOME>/glassfish/bin/asadmin create-file-user
```

For more information, see
http://docs.sun.com/app/docs/doc/820-4495/6nfv4mkkl?a=view.

5. Import orakey from default-keystore.jks into GlassFish keystore and truststore. These are located in the directory `<domain-dir>/config`

```
$JAVA_HOME/bin/keytool -importkeystore -srckeystore
 <path-to>/default-keystore.jks -destkeystore
 <path-to-gf-domain>/config/keystore.jks -srcalias  orakey -destalias orakey
 -srckeypass welcome -destkeypass changeit
```

6. Copy cwallet.sso, jps-config.xml and default-keystore.jks from the domain's fmwconfig directory into a local folder.

8.2.2.2 Configuring GlassFish/Metro Web Service

1. Create a Metro Web service. For more information, see http://metro.java.net/guide/Developing_with_NetBeans.html.
2. Configure the appropriate security mechanism. For more information, see http://metro.java.net/guide/Security_Mechanisms.html.

8.2.2.3 Configuring Oracle WSM 11*g* Client

1. Using JDeveloper, create a Web service proxy for the GlassFish service. Select the policy oracle/wss11_username_token_with_message_protection_client_policy in the wizard.
2. Set the csf-key to glassfish.credentials in the Override Properties option for the Web service proxy.
3. In the Web service proxy main class, set the system property of oracle.security.jps.config to jps-config.xml from Step 6 of "Configuration Prerequisites for Interoperability" on page 8-3.
4. Invoke the Web service.

Note: If you are using

- Oracle Service Bus business service, set the property overrides to glassfish.credentials in the Security page. For more information, see "Policy Overrides" in *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus* at http://download.oracle.com/docs/html/E15866_01/owsm.htm.
- SOA Web service reference, set the property overrides to glassfish.credentials in the Security page. For more information, see Section 37.2.2 "How to Override Policy Configuration Property Values" in Developer's Guide for SOA Suite at http://download.oracle.com/docs/cd/E15523_01/integration.1111/e10224/sca_policy.htm#CDDEIAFA.

8.3 SAML Token (Sender Vouches) with Message Protection (WS-Security 1.1)

The following sections describe how to implement SAML token (sender vouches) with message protection that conforms to the WS-Security 1.1 standard:

8.3.1 Configuring GlassFish Client and Oracle WSM 11*g* Web Service

To configure GlassFish client and Oracle WSM 11*g* Web Service, perform the steps described in the following sections:

8.3.1.1 Configuration Prerequisites for Interoperability

Perform the following prerequisite steps:

1. Create a default-keystore.jks file with the following command:

```
$JAVA_HOME/bin/keytool -genkeypair -alias orakey -keypass welcome -keyalg RSA
 -dname "CN=orakey, O=oracle C=us" -keystore default-keystore.jks -storepass
 welcome
```

2. Copy default-keystore.jks to the domain's fmwconfig directory.
3. Create a file user in GlassFish with the following command:

   ```
   $<GLASSFISHV3_HOME>/glassfish/bin/asadmin create-file-user
   ```

   For more information, see http://docs.sun.com/app/docs/doc/820-4495/6nfv4mkkl?a=view.
4. Add the user as described in "Create users" in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help.
5. Import orakey from default-keystore.jks into GlassFish keystore and truststore. These are located in the directory `<domain-dir>/config`

   ```
   $JAVA_HOME/bin/keytool -importkeystore -srckeystore
    <path-to>/default-keystore.jks -destkeystore
    <path-to-gf-domain>/config/cacerts.jks -srcalias  orakey -destalias orakey
    -srckeypass welcome -destkeypass changeit
   ```

6. Copy jps-config.xml and default-keystore.jks from the domain's fmwconfig directory into a local folder.

8.3.1.2 Configuring Oracle WSM 11*g* Web Service

1. Create a Web service.
2. Attach the following policy to the Web service: oracle/wss11_username_token_with_message_protection_service_policy.

   For more information about attaching the policy, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

8.3.1.3 Configuring GlassFish/Metro Client

1. Using NetBeans, create a Metro client by selecting **New Project > Java > Java Application**.

1. Provide a project name and location. Select the server to deploy and select Finish.
2. Right click on the project. Select **New > Web service Client**. Follow the wizard and provide WSDL URL for service deployed in WebLogic.
3. Create a SAML CallbackHandler that can be used with WSIT SAML Security Mechanisms supported by NetBeans. You can also download a sample from http://xwss.java.net/files/documents/4864/50700/SamlCallbackHandler.java.

   a. Place the file in the source folder of the project.

   b. Ensure issuer variable value is the same as in the jps-config.xml file created in Step 5 of "Configuration Prerequisites for Interoperability" on page 8-4.

   c. Set the urn reference to `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`.

   d. Set the user created in Step 3 and Step 4 of "Configuration Prerequisites for Interoperability" on page 8-6. For example, to set the user to wlsuser, modify the file as follows:

```
CN=wlsuser,OU=SU,O=wlsuser,L=Los Angeles,ST=CA,C=US
```

4. To configure the JVM, log on to the GlassFish Administration Console. For more information, see http://docs.sun.com/app/docs/doc/820-4334/gewgw?l=en&a=view&q=glassfish+admin+console.
   a. In the left pane, expand **Configuration** and click **JVM Setting**.
   b. In the right pane, click **JVM Option** tab.
   c. Click **Add JVM Option**. A new text field is displayed. Enter `-DWSIT_HOME=${com.sun.aas.installRoot}`.
   d. Click **Enterprise Server** in left pane.
   e. Click **Restart** in the right pane to restart the server.
5. Expand Web Services Reference node. Using NetBeans, right click **Service Reference** and select **Edit Web Services Attributes**.
6. For SAML Callback Handler option, click Browse and select the file from Step 3.
7. Set the alias in Keystore and Truststore.
8. Open index.jsp file. Right click and select **Web Service Client Reference**. Select Operation in **Select Operation to Invoke** dialog box and click ok.
9. Run the project.

8.3.2 Configuring Oracle WSM 11*g* Client and GlassFish Web Service

To configure Oracle WSM 11*g* client and GlassFish Web Service, perform the steps described in the following sections:

8.3.2.1 Configuration Prerequisites for Interoperability

Perform the following prerequisite steps:

1. Create a default-keystore.jks file with the following command:

```
$JAVA_HOME/bin/keytool -genkeypair -alias orakey -keypass welcome -keyalg RSA
 -dname "CN=orakey, O=oracle C=us" -keystore default-keystore.jks -storepass
 welcome
```

2. Copy default-keystore.jks to the domain's fmwconfig directory.
3. Save the credentials in credential store using WLST commands. For example:

```
$<ORACLE_HOME>/common/bin/wlst.sh
> connect()
> createCred(map="oracle.wsm.security", key="keystore-csf-key",
 user="keystore", password="welcome")
> createCred(map="oracle.wsm.security", key="sign-csf-key", user="orakey",
 password="welcome")
> createCred(map="oracle.wsm.security", key="enc-csf-key", user="orakey",
 password="welcome")
>createCred(map="oracle.wsm.security", key="glassfish.credentials" ,
 user="wlsUser" , password="welcome1" , description="Glassfish user
 credentials");
```

   A file cwallet.sso is created in the directory `DOMAIN_HOME/config/fmwconfig`
4. Create a file user in GlassFish with the following command:

```
$<GLASSFISHV3_HOME>/glassfish/bin/asadmin create-file-user
```

For more information, see
http://docs.sun.com/app/docs/doc/820-4495/6nfv4mkkl?a=view.

5. Import orakey from default-keystore.jks into GlassFish keystore and truststore. These are located in the directory <domain-dir>/config

```
$JAVA_HOME/bin/keytool -importkeystore -srckeystore
 <path-to>/default-keystore.jks -destkeystore
 <path-to-gf-domain>/config/keystore.jks -srcalias  orakey -destalias orakey
 -srckeypass welcome -destkeypass changeit
```

6. Copy cwallet.sso, jps-config.xml and default-keystore.jks from the domain's fmwconfig directory into a local folder.

8.3.2.2 Configuring GlassFish/Metro Web Service

1. Create a Metro Web service. For more information, see
http://metro.java.net/guide/Developing_with_NetBeans.html.
2. Configure the appropriate security mechanism. For more information, see
http://metro.java.net/guide/Security_Mechanisms.html.

8.3.2.3 Configuring Oracle WSM 11*g* Client

1. Using JDeveloper, create a Web service proxy for the GlassFish service. Select the policy oracle/wss11_saml_token_with_message_protection_client_policy in the wizard.
2. Set the path to jps-config.xml created in Step 6 of "Configuration Prerequisites for Interoperability" on page 8-6.
3. Set the USERNAME_PROPERTY as follows: `((BindingProvider) sAMLTokenEchoService).getRequestContext().put(BindingProvider .USERNAME_PROPERTY, "wlsUser");`
4. Invoke the Web service.