BEA WebLogic Server™

WebLogic Security の紹介

BEA WebLogic Server バージョン 7.0
マニュアルの改訂： 2003 年 6 月 13 日

著作権

限定的権利条項

商標または登録商標

WebLogic Security の紹介

パート番号	マニュアルの日付	ソフトウェアのバージョン
なし	2003 年 6 月 13 日	BEA WebLogic Server バージョン 7.0

目次

このマニュアルの内容

対象読者 vii
e-docs Web サイト ix
このマニュアルの印刷方法 ix
関連情報 ix
サポート情報 x
表記規則 xi

1. WebLogic Security サービスの概要

このガイドの対象読者 1-1
WebLogic Security サービスの概要 1-3
WebLogic Security サービスの特徴 1-3
使いやすさとカスタマイズしやすさの両立 1-5
WebLogic Security の変更点 1-6

2. セキュリティの基礎概念

監査 2-1
認証 2-2
サブジェクトとプリンシパル 2-2
JAAS (Java Authentication and Authorization Service) 2-4
JAAS LoginModule 2-4
JAAS 制御フラグ 2-5
CallbackHandler 2-5
相互認証 2-6
ID アサーション プロバイダと LoginModule 2-7
ID アサーションとトークン 2-7
認証のタイプ 2-8
ユーザ名 / パスワード認証 2-8
証明書認証 2-9
境界認証 2-9
認可 2-12

WebLogic リソース....2-12
セキュリティ ポリシー....2-13
ContextHandler....2-14
アクセス決定....2-15
裁決....2-15
セキュア ソケット レイヤ (SSL)....2-16
SSL の機能....2-17
SSL トンネリング....2-18
一方向または双方向 SSL 認証....2-19
国内向け SSL と輸出向け SSL....2-20
デジタル証明書....2-21
認証局....2-22
ホスト名検証....2-23
トラスト マネージャ....2-23
非対称鍵アルゴリズム....2-24
対称鍵アルゴリズム....2-25
メッセージ ダイジェスト アルゴリズム....2-26
暗号スイート....2-26
ファイアウォール....2-28
接続フィルタ....2-29
境界認証....2-29
J2EE と WebLogic セキュリティ....2-29
SDK 1.3 セキュリティ パッケージ....2-30
Java Secure Socket Extension (JSSE)....2-30
Java Authentication and Authorization Service (JAAS)....2-30
Java セキュリティ マネージャ....2-31
Java Cryptography Architecture と Java Crytography Extension (JCE).. 2-32
CSIv2 (Common Secure Interoperability Version 2)....2-32

3. セキュリティ レルム

セキュリティ レルムの概要....3-1
ユーザ....3-2
グループ....3-3
セキュリティ ロール....3-4

セキュリティ ポリシー 3-4
セキュリティ プロバイダ 3-5
セキュリティ プロバイダ データベース 3-5
セキュリティ プロバイダ データベースとは 3-6
セキュリティ レルムとセキュリティ プロバイダ データベース 3-7
組み込み LDAP サーバ 3-7
セキュリティ プロバイダのタイプ 3-8
認証プロバイダ 3-9
ID アサーション プロバイダ 3-10
プリンシパル検証プロバイダ 3-11
認可プロバイダ 3-12
裁決プロバイダ 3-13
ロール マッピング プロバイダ 3-13
監査プロバイダ 3-15
資格マッピング プロバイダ 3-15
キーストア プロバイダ 3-16
レルム アダプタ プロバイダ 3-16
セキュリティ プロバイダのまとめ 3-17
セキュリティ プロバイダとセキュリティ レルム 3-18

4. WebLogic Security サービスのアーキテクチャ

アーキテクチャの概要 4-1
WebLogic Security フレームワーク 4-1
認証プロセス 4-3
ID アサーション プロセス 4-4
プリンシパル検証プロセス 4-5
認可プロセス 4-6
裁決プロセス 4-7
ロール マッピング プロセス 4-7
監査プロセス 4-9
資格マッピング プロセス 4-10
セキュリティ サービス プロバイダ インタフェース (SSPI) 4-11
WebLogic セキュリティ プロバイダ 4-12
WebLogic 認証プロバイダ 4-14
WebLogic ID アサーション プロバイダ 4-14

WebLogic プリンシパル検証プロバイダ 4-15
WebLogic 認可プロバイダ 4-16
WebLogic 裁決プロバイダ 4-16
WebLogic ロール マッピング プロバイダ 4-17
WebLogic 監査プロバイダ 4-18
WebLogic 資格マッピング プロバイダ 4-18
WebLogic キーストア プロバイダ.....4-18
WebLogic レルム アダプタ プロバイダ 4-19
アーキテクチャによってユーザにもたらされるメリット.....4-21
アプリケーション開発者.....4-21
サーバ管理者 / アプリケーション管理者.....4-22
サードパーティ セキュリティ サービス プロバイダ.....4-22

5. 用語

このマニュアルの内容

このマニュアルでは、新しい WebLogic Security サービスの特徴についてまとめ、WebLogic Security サービスのアーキテクチャと機能の概要を示します。WebLogic Security サービスを理解する上で基本となるマニュアルです。

このマニュアルの内容は以下のとおりです。

- 第 1 章「WebLogic Security サービスの概要」では、このマニュアルの対象読者について解説するとともに、WebLogic Security サービスの概要、その主な特徴、およびこのリリースでの変更点について簡単に説明します。
- 第 2 章「セキュリティの基礎概念」では、BEA WebLogic Server™ セキュリティに関する基礎概念について説明します。この章には、監査、認証、認可、セキュア ソケット レイヤ (SSL)、ファイアウォール、および J2EE と WebLogic セキュリティの関係に関する説明が記述されています。
- 第 3 章「セキュリティ レルム」では、WebLogic リソースの保護に使用するセキュリティ レルムに関する説明が記述されています。
- 第 4 章「WebLogic Security サービスのアーキテクチャ」では、WebLogic Server のセキュリティ アーキテクチャについて説明します。この章には、WebLogic Security フレームワーク、セキュリティ サービス プロバイダ インタフェース (SSPI)、および製品の一部として含まれる WebLogic セキュリティ プロバイダに関する説明が記述されています。
- 第 5 章「用語」では、WebLogic Server のセキュリティに関するドキュメントで扱う主な用語を定義します。

対象読者

このマニュアルは、以下の読者を対象としています。

- アプリケーション設計者 — セキュリティの目標を設定し、組織の全体的なセキュリティ アーキテクチャを設計するだけでなく、WebLogic Server のセキュリティ機能を評価して最適な実装方法を判断する設計者のことです。ア

プリケーション設計者は、セキュリティ システムや最先端のセキュリティ技術とツールだけでなく、Java プログラミング、Java セキュリティ、およびネットワーク セキュリティにも精通しています。

- セキュリティ開発者 —WebLogic Server に統合されるセキュリティ製品のシステム アーキテクチャとインフラストラクチャの定義と、WebLogic Server で使用するカスタム セキュリティ プロバイダの開発を主な業務とする開発者のことです。アプリケーション設計者と連携して、セキュリティ アーキテクチャを確実に設計に従って、セキュリティ ホールが発生しないように実装します。また、セキュリティが確実に正しくコンフィグレーションされるよう、サーバ管理者とも連携します。セキュリティ開発者は、認証、認可、監査 (AAA)、Java Management eXtension (JMX) などの Java に対する深い知識、および WebLogic Server とセキュリティ プロバイダの機能に対する実践的な知識をはじめとしたセキュリティ概念をしっかりと理解しています。
- アプリケーション開発者 — クライアント アプリケーションの開発、Web アプリケーションおよびエンタープライズ JavaBean (EJB) へのセキュリティ機能の付加、および他のエンジニアリング チームや品質保証 (QA) チーム、データベース チームとの連携によるセキュリティ機能の実装などを主な業務とする Java プログラマのことです。アプリケーション開発者は、サーブレット、JSP、JSEE などの J2EE コンポーネントを含む Java、および Java セキュリティについて実用的かつ深い知識を備えています。
- サーバ管理者 — アプリケーション設計者と密接に連携しながら、サーバおよびサーバ上で動作するアプリケーションのセキュリティ方式の設計、潜在的なセキュリティ リスクの特定、およびセキュリティ上の問題を防止するコンフィグレーションの提案を行う管理者のことです。関連する責務として、重要なプロダクション システムの保守、セキュリティ レルムのコンフィグレーションと管理、サーバ リソースとアプリケーション リソースへの認証および認可方式の実装、セキュリティ機能のアップグレード、およびセキュリティ プロバイダのデータベースの保守などが含まれる場合もあります。サーバ管理者は、Web アプリケーションと EJB のセキュリティ、公開鍵セキュリティ、および SSL を含む、Java セキュリティ アーキテクチャについて深い知識を備えています。
- アプリケーション管理者 — サーバ管理者と共同でセキュリティ コンフィグレーション、認証および認可方式を実装および管理したり、定義されたセキュリティ レルムでデプロイされているアプリケーション リソースへのアクセスを設定および管理したりする管理者のことです。アプリケーション管理者は、セキュリティの概念や Java セキュリティ アーキテクチャの一般的な知識を持っています。アプリケーション管理者は、Java、XML、デプロイ

メント記述子を理解し、サーバ ログおよび監査ログでセキュリティ イベントを特定できます。

e-docs Web サイト

BEA 製品のドキュメントは、BEA の Web サイトで入手できます。BEA のホーム ページで [製品のドキュメント] をクリックします。

このマニュアルの印刷方法

Web ブラウザの [ファイル | 印刷] オプションを使用すると、Web ブラウザからこのマニュアルを一度に 1 章ずつ印刷できます。

このマニュアルの PDF 版は、WebLogic Server の Web サイトで入手できます。PDF を Adobe Acrobat Reader で開くと、マニュアルの全体 (または一部分) を書籍の形式で印刷できます。PDF を表示するには、WebLogic Server ドキュメントのホーム ページを開き、[ドキュメントのダウンロード] をクリックして、印刷するマニュアルを選択します。

Adobe Acrobat Reader をインストールしていない場合は、Adobe の Web サイト (http://www.adobe.co.jp) で無料で入手できます。

関連情報

BEA WebLogic Server の以下のマニュアルには、WebLogic Security サービスに関係のある情報が記載されています。

- 『WebLogic Security の管理』— このマニュアルでは、WebLogic Server のセキュリティをコンフィグレーションする方法と、互換性セキュリティの使い方について説明します。

- 『WebLogic Security サービスの開発』— このマニュアルでは、セキュリティ ベンダおよびアプリケーション開発者を対象に、WebLogic Server 用のカスタム セキュリティ プロバイダを開発するのに必要な情報を提供します。
- 『プロダクション環境のロックダウン』— このマニュアルでは、WebLogic Server をプロダクション環境にデプロイする前に検討すべき重要なセキュリティ対策について説明します。
- 『WebLogic リソースのセキュリティ』— このマニュアルでは、さまざまなタイプの WebLogic リソースを紹介し、WebLogic Server を使用してそれらのリソースを保護するための情報を提供します。このバージョンでは、特に URL (Web) アプリケーション リソースおよびエンタープライズ JavaBean (EJB) リソースの保護について説明します。
- 『BEA WebLogic Server 7.0 へのアップグレード』— このマニュアルでは、以前のバージョンの BEA WebLogic Server を WebLogic Server 7.0 にアップグレードするために必要な手順およびその他の情報を示します。また、アプリケーションを WebLogic Server の以前のバージョンからバージョン 7.0 に移動する方法についても説明します。
- 『セキュリティ FAQ』— このマニュアルでは、WebLogic Server セキュリティに関してよく寄せられる質問とその回答を紹介します。
- WebLogic クラスに関する Javadoc のページ — このページでは、WebLogic Server のこのリリースで提供およびサポートされている WebLogic セキュリティ パッケージのリファレンスを提供します。

サポート情報

BEA のドキュメントに関するユーザからのフィードバックは弊社にとって非常に重要です。質問や意見などがあれば、電子メールで docsupport-jp@beasys.com までお送りください。寄せられた意見については、ドキュメントを作成および改訂する BEA の専門の担当者が直に目を通します。

電子メールのメッセージには、ご使用のソフトウェアの名前とバージョン、およびドキュメントのタイトルと日付をお書き添えください。

本バージョンの BEA WebLogic Server について不明な点がある場合、または BEA WebLogic Server のインストールおよび動作に問題がある場合は、BEA WebSupport (www.bea.com) を通じて BEA カスタマ サポートまでお問い合わせください。カスタマ サポートへの連絡方法については、製品パッケージに同梱されているカスタマ サポート カードにも記載されています。

カスタマ サポートでは以下の情報をお尋ねしますので、お問い合わせの際はあらかじめご用意ください。

- お名前、電子メール アドレス、電話番号、ファクス番号
- 会社の名前と住所
- お使いの機種とコード番号
- 製品の名前とバージョン
- 問題の状況と表示されるエラー メッセージの内容

表記規則

このマニュアルでは、全体を通して以下の表記規則が使用されています。

表記法	適用
〔Ctrl〕+ 〔Tab〕	複数のキーを同時に押すことを示す。
斜体	強調または書籍のタイトルを示す。

表記法	適用

等幅テキスト	コード サンプル、コマンドとそのオプション、データ構造体とそのメンバー、データ型、ディレクトリ、およびファイル名とその拡張子を示す。等幅テキストはキーボードから入力するテキストも示す。
	例:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`\tux\data\ap`
	`.java`
	`config.xml`
	`float`

太字の等幅テキスト	コード内の変数を示す。 例: `void **commit** ( )`
斜体の等幅テキスト	コード内の変数を示す。 例: `String *expr*`
すべて大文字のテキスト	デバイス名、環境変数、および論理演算子を示す。 例: LPT1 BEA_HOME OR
{ }	構文の中で複数の選択肢を示す。
[ ]	構文の中で任意指定の項目を示す。実際には、この括弧は入力しない。 例: `java [options] weblogic.Server [args...]`
|	構文の中で相互に排他的な選択肢を区切る。

表記法	適用
...	コマンドラインで以下のいずれかを示す。 ■ 引数を複数回繰り返すことができる。 ■ 任意指定の引数が省略されている。 ■ パラメータや値などの情報を追加入力できる。 実際には、この省略符号は入力しない。 例： `java [options] weblogic.Server [args...]`
. . .	コード サンプルまたは構文で項目が省略されていることを示す。 実際には、この省略符号は入力しない。

1 WebLogic Security サービスの概要

BEA WebLogic Server™ マニュアル セットに収められている他のセキュリティ関連マニュアルでは特定の作業 (たとえば、WebLogic セキュリティのプログラミング、カスタム セキュリティ プロバイダの開発、WebLogic Security サービスの管理など) の手順をユーザに説明しているのに対して、この入門編は WebLogic Security サービスのすべてのユーザを対象としています。そのため、このマニュアルは WebLogic Security サービスを理解する上で基本となるものです。

注意： WebLogic Security サービスには、固有の用語が多数あります。第 5 章「用語」で用語を理解してから、このマニュアルをお読みください。

以下の各節では、WebLogic Security サービスとその特徴を紹介します。

- 1-1 ページの「このガイドの対象読者」
- 1-3 ページの「WebLogic Security サービスの概要」
- 1-3 ページの「WebLogic Security サービスの特徴」
- 1-5 ページの「使いやすさとカスタマイズしやすさの両立」
- 1-6 ページの「WebLogic Security の変更点」

このガイドの対象読者

このマニュアルは、以下の読者を対象としています。

- アプリケーション設計者 — セキュリティの目標を設定し、組織の全体的なセキュリティ アーキテクチャを設計するだけでなく、WebLogic Server のセキュリティ機能を評価して最適な実装方法を判断する設計者のことです。アプリケーション設計者は、セキュリティ システムや最先端のセキュリティ技

術とツールだけでなく、Java プログラミング、Java セキュリティ、およびネットワーク セキュリティにも精通しています。

- セキュリティ開発者 —WebLogic Server に統合されるセキュリティ製品のシステム アーキテクチャとインフラストラクチャの定義と、WebLogic Server で使用するカスタム セキュリティ プロバイダの開発を主な業務とする開発者のことです。アプリケーション設計者と連携して、セキュリティ アーキテクチャを確実に設計に従って、セキュリティ ホールが発生しないように実装します。また、セキュリティが確実に正しくコンフィグレーションされるよう、サーバ管理者とも連携します。セキュリティ開発者は、認証、認可、監査 (AAA)、Java Management eXtension (JMX) などの Java に対する深い知識、および WebLogic Server とセキュリティ プロバイダの機能に対する実践的な知識をはじめとしたセキュリティ概念をしっかりと理解しています。
- アプリケーション開発者 — クライアント アプリケーションの開発、Web アプリケーションおよびエンタープライズ JavaBean (EJB) へのセキュリティ機能の付加、および他のエンジニアリング チームや品質保証 (QA) チーム、データベース チームとの連携によるセキュリティ機能の実装などを主な業務とする Java プログラマのことです。アプリケーション開発者は、サーブレット、JSP、JSEE などの J2EE コンポーネントを含む Java、および Java セキュリティについて実用的かつ深い知識を備えています。
- サーバ管理者 — アプリケーション設計者と密接に連携しながら、サーバおよびサーバ上で動作するアプリケーションのセキュリティ方式の設計、潜在的なセキュリティ リスクの特定、およびセキュリティ上の問題を防止するコンフィグレーションの提案を行う管理者のことです。関連する責務として、重要なプロダクション システムの保守、セキュリティ レルムのコンフィグレーションと管理、サーバ リソースとアプリケーション リソースへの認証および認可方式の実装、セキュリティ機能のアップグレード、およびセキュリティ プロバイダのデータベースの保守などが含まれる場合もあります。サーバ管理者は、Web アプリケーションと EJB のセキュリティ、公開鍵セキュリティ、および SSL を含む、Java セキュリティ アーキテクチャについて深い知識を備えています。
- アプリケーション管理者 — サーバ管理者と共同でセキュリティ コンフィグレーション、認証および認可方式を実装および管理したり、定義されたセキュリティ レルムでデプロイされているアプリケーション リソースへのアクセスを設定および管理したりする管理者のことです。アプリケーション管理者は、セキュリティの概念や Java セキュリティ アーキテクチャの一般的な知識を持っています。アプリケーション管理者は、Java、XML、デプロイ

メント記述子を理解し、サーバ ログおよび監査ログでセキュリティ イベントを特定できます。

WebLogic Security サービスの概要

セキュリティのデプロイメント、管理、および保守は、Web を使用する顧客に対して新しいさまざまなサービスを提供している IT (情報技術) 企業にとって大きな課題です。IT 企業は、Web ベースのユーザに世界規模のネットワークを提供できるよう、システムとそのデータの機密性、整合性、および可用性の保持という基本的な問題に取り組まなければなりません。セキュリティに対する取り組みは、ネットワーク自体から個々のクライアント マシンに至るまで、システムのすべてのコンポーネントに関係します。インフラストラクチャ全体のセキュリティの実現は、確立されたセキュリティ ポリシーと周知徹底された手順に加え、不断の警戒が要求される大変な作業です。

WebLogic Server のこのリリースでは、セキュリティ アーキテクチャが全面的に設計し直されており、Web を介して使用できるアプリケーション用にユニークでセキュアな基盤が用意されています。企業は、WebLogic Server の新しいセキュリティ機能を利用することで、Web 上で使用できるアプリケーションの作成というセキュリティ課題に対処できるように設計された、包括的で柔軟性の高いセキュリティ インフラストラクチャのメリットを得ることができます。WebLogic セキュリティは、WebLogic Server アプリケーションを保護するためにスタンドアロンで使用することも、最高レベルのセキュリティ管理ソリューションを表す企業全体のセキュリティ管理システムの一部として使用することもできます。

WebLogic Security サービスの特徴

WebLogic Server のセキュリティ アーキテクチャはオープンで柔軟性が高いため、すべてのレベルのユーザが利点を生かすことができ、アプリケーション サーバに対して高度なセキュリティ設計を導入できます。アプリケーション サーバのユニークなセキュリティ ソリューション、および明確なセキュリティ ポリシーと文書で確立された手順を有している企業であれば、サーバとそのデータの機密性、整合性、および可用性を確保できます。

WebLogic Security サービスの主な特徴は、次のとおりです。

- 包括的で標準に基づいた設計である。
- WebLogic Server がホストになるアプリケーション向けに、メインフレームから Web ブラウザまでのエンド ツー エンドのセキュリティを提供する。
- レガシー セキュリティ方式を WebLogic Server セキュリティに統合することによって、企業の既存投資を活用できる。
- セキュリティ ツールが柔軟性の高い統一したシステムに統合されるので、企業全体のセキュリティ管理が容易である。
- 企業のビジネス ルールをセキュリティ ポリシーにマッピングするため、ビジネス要件に対するアプリケーション セキュリティのカスタマイズが容易である。
- セキュリティ ポリシーの更新が容易である。
- カスタマイズされたセキュリティ ソリューションへの適合が容易である。
- モジュール化されたアーキテクチャなので、特定の企業の要件を満たすようにセキュリティ インフラストラクチャを時間を追って変更できる。
- 遷移方法またはアップグレード パスの一部として、複数のセキュリティ プロバイダのコンフィグレーションをサポートしている。
- セキュリティの詳細とアプリケーションのインフラストラクチャが分離しているので、要件の変更に応じたセキュリティのデプロイメント、管理、保守、および変更が容易である。
- デフォルトの WebLogic セキュリティ プロバイダによって、そのまま使用できるセキュリティ方式が用意されている。
- WebLogic カスタム セキュリティ プロバイダを使用してカスタマイズできる。
- WebLogic Server Administration Console を使用して、セキュリティ ルール、セキュリティ ポリシー、およびセキュリティ プロバイダを統一して管理できる。
- JAAS (Java Authentication and Authorization Service)、JSSE (Java Secure Socket Extension)、JCE (Java Crytography Extension) などの標準の J2EE セキュリティ技術をサポートしている。

使いやすさとカスタマイズしやすさの両立

WebLogic Security サービスのコンポーネントおよびサービスでは、エンド ユーザと管理者による使いやすさと管理しやすさ、およびアプリケーション開発者とセキュリティ開発者によるカスタマイズしやすさの両立が求められます。以下の節では例を示します。

使いやすさ : セキュアな WebLogic Server 環境では、エンド ユーザに対して、ユーザ認証用のシングル サインオン (ユーザ ID の確認) のみを要求します。ユーザは、アプリケーションのリソースを含む WebLogic Server ドメイン内では再認証の必要はありません。シングル サインオンを利用すれば、ユーザはセッションごとに 1 回だけドメインにログオンすればよくなり、各リソースやアプリケーションにアクセスしようとするたびに個別にログオンを要求されることはなくなります。

開発者や管理者に対しては、新しくドメイン コンフィグレーション ウィザードが用意されています。ドメイン コンフィグレーション ウィザードは、管理サーバ、管理対象サーバ、および必要に応じてクラスタを含む新しいドメインの作成と、個々のサーバを追加することによる既存のドメインの拡張を支援します。ドメイン コンフィグレーション ウィザードは、`config.xml` ファイルと新しいドメインに追加するよう選択したサーバの起動スクリプトも自動的に生成します。

管理しやすさ : WebLogic Server 環境でアプリケーションをコンフィグレーションおよびデプロイする管理者は、製品に付属している WebLogic セキュリティ プロバイダを使用できます。これらのデフォルトのプロバイダは、そのままの状態で、必要なすべてのセキュリティ機能をサポートしています。管理者は、WebLogic Server に用意されているセキュリティ ストア (特殊用途の組み込み LDAP ディレクトリ サーバ) にセキュリティ データを保存できます。WebLogic Server のセキュリティのコンフィグレーションと管理を簡素化するために、堅牢なデフォルトのセキュリティ コンフィグレーションが用意されています。

カスタマイズしやすさ : アプリケーション開発者に対しては、WebLogic セキュリティ API と、JAAS (Java Authentication and Authorization Service)、JSSE (Java Secure Socket Extension) などの J2EE セキュリティ標準がサポートされています。これらの API と標準を使用して、WebLogic Server に接続するアプリケーションに対して、きめ細かい、カスタマイズされたセキュリティ環境を作成できます。

セキュリティ開発者は、セキュリティ サービス プロバイダ インタフェース (SSPI) を使用して、WebLogic Server 環境向けのカスタム セキュリティ プロバイダを開発できます。

WebLogic Security の変更点

WebLogic Server 6.x で提供されていたセキュリティに関しては、多くのセキュリティ機能が変更になりました。

表 1-1 に、それらの相違点をまとめておきます。

表 1-1 WebLogic Security 機能の変更点

WebLogic Server 6.x	WebLogic Server 7.0
セキュリティ API	既存のセキュリティ API の多くが、このリリースでは推奨されない。BEA では、それらに相当する J2EE 標準インタフェースを使用して同様の機能をアプリケーションに実装することをお勧めする。 非推奨 API の詳細なリストについては、『WebLogic Security プログラマーズ ガイド』の「セキュリティ API」を参照。
JAAS 認証	JAAS 認証は機能拡張され、IIOP および T3 の各クライアント用の LoginModule が用意されている。
監査	WebLogic Server デプロイメントに監査機能を追加するのに、`weblogic.security.Audit` インタフェースの実装を作成する必要はなくなった。製品に付属している WebLogic 監査プロバイダを利用すれば、記録したいデータをカスタマイズすることができる。

表 1-1 WebLogic Security 機能の変更点 (続き)

WebLogic Server 6.x	WebLogic Server 7.0
weblogic.xml、weblogic-ejb-jar.xml、および weblogic-ra.xml ファイルでのセキュリティ要件の定義	機能拡張が施され、セキュリティ要件が WebLogic Server Administration Console からでも指定できるようになった。
システム パスワード	このリリースの WebLogic Server には特定の System アカウントは存在しない。
アクセス制御リスト (ACL)	WebLogic Server の以前のリリースで使用されていた ACL はこのリリースでは推奨されない。セキュリティ ポリシーが ACL の代わりとなる。
ユーザおよびグループ	ユーザおよびグループは依然として使用される。ただし、リソースに ACL を割り当てるのではなく、ユーザ、グループ、あるいはセキュリティ ロールに WebLogic リソースへのアクセスを許可するセキュリティ ポリシーを作成するようになった。
6.x セキュリティ レルム (ファイル レルム、キャッシング レルム、LDAP レルム、Windows NT レルム、UNIX レルム、および RDBMS レルム)	WebLogic Server の以前のリリースで使用されていたセキュリティ レルムは、このリリースでは推奨されない。WebLogic 認証および認可プロバイダが、ファイル レルム、キャッシング レルム、および LDAP セキュリティ レルムで提供されるのと同じ機能を提供する。 レルム アダプタ プロバイダを利用すれば、認証 / 認可の新しい枠組みに移行する際に、既存の Windows NT、UNIX、および RDBMS セキュリティ レルムを引き続き使用できるようになる。
この機能は、WebLogic Server の以前のリリースでは利用不可。	複数のセキュリティ プロバイダをサポート。
SSL	WebLogic Server における SSL のサポートが更新され、JSSE 標準と TLS (Transport Layer Security) v1 プロトコルが提供されている。

表 1-1 WebLogic Security 機能の変更点 (続き)

WebLogic Server 6.x	WebLogic Server 7.0
この機能は、WebLogic Server の以前のリリースでは利用不可。	J2EE JKS キーストアをサポート。

2　セキュリティの基礎概念

この章では、WebLogic Server のセキュリティに関する以下の基礎概念について説明します。

- 2-1 ページの「監査」
- 2-2 ページの「認証」
- 2-12 ページの「認可」
- 2-16 ページの「セキュア ソケット レイヤ (SSL)」
- 2-28 ページの「ファイアウォール」
- 2-29 ページの「J2EE と WebLogic セキュリティ」

監査

監査とは、リクエストの操作とそれらのリクエストの結果に関する情報を、否認防止を目的として収集、格納、および配布するプロセスのことです。言い換えれば、監査はコンピュータのアクティビティの電子的な記録を提供するものです。WebLogic Server のセキュリティ アーキテクチャでは、監査プロバイダを使用して監査サービスを提供します。

監査プロバイダがコンフィグレーションされている場合、WebLogic Security フレームワークは、セキュリティ操作 (認証や認可など) が実行される前とされた後に、監査プロバイダを呼び出し、監査イベントの記録を有効にします。個々のイベントを監査するかどうかの決定は監査プロバイダによって行われ、特定の監査基準または重大度レベルあるいはその両方に基づいて決定することができます。監査情報を記載した記録は、LDAP サーバ、データベース、シンプル ファイルなどの出力リポジトリに書き出すことができます。

認証

認証とは、呼び出し側が、特定のユーザまたはシステムの代わりに動作していることを証明する際に使用するメカニズムのことです。認証は、ユーザ名とパスワードの組み合わせなどの資格を使用して「あなたは誰」という問いに答えます。

WebLogic Server で、認証プロバイダはユーザまたはシステム プロセスの ID を証明するために使用します。認証プロバイダでは、ID 情報を記憶したり、転送したり、その情報が必要な場合に (サブジェクトを通じて) システムのさまざまなコンポーネントで利用できるようにしたりします。認証プロセスでは、プリンシパル検証プロバイダが、サブジェクト内に格納されるプリンシパル (ユーザおよびグループ) のセキュリティを強化するため、それらのプリンシパルに署名して信頼性を検証します。

以下の節では、認証の概念と機能について説明します。

- 2-2 ページの「サブジェクトとプリンシパル」
- 2-4 ページの「JAAS (Java Authentication and Authorization Service)」
- 2-5 ページの「CallbackHandler」
- 2-6 ページの「相互認証」
- 2-7 ページの「ID アサーション プロバイダと LoginModule」
- 2-7 ページの「ID アサーションとトークン」
- 2-8 ページの「認証のタイプ」

サブジェクトとプリンシパル

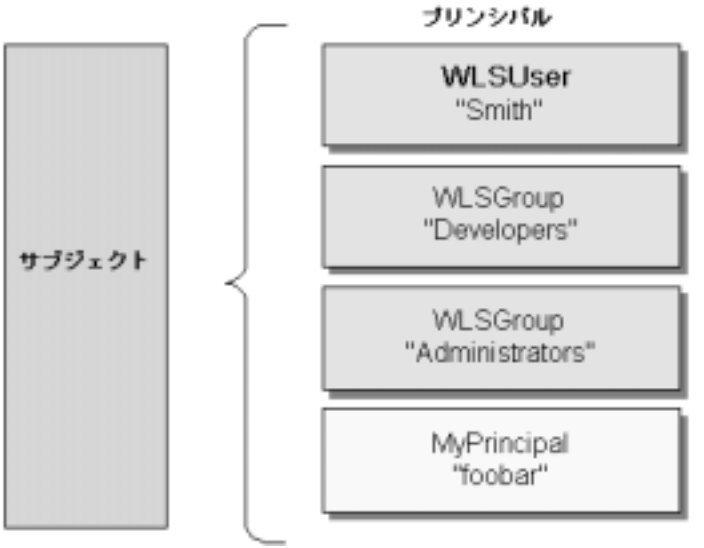
サブジェクトとプリンシパルは密接に関係しています。

プリンシパルは、認証の結果としてユーザまたはグループに割り当てられる ID です。WebLogic Server のようなアプリケーション サーバでは、ユーザもグループもプリンシパルとして使用することができます。JAAS (Java Authentication and Authorization Service) では、プリンシパルなどの認証情報のコンテナとして**サブジェクト**を使用することになっています。

図 2-1 に、ユーザ、グループ、プリンシパル、およびサブジェクト間の関係を示します。

注意： このリリースの WebLogic Server では、WebLogic Server 6.x ユーザに代わってサブジェクトが使用されます。

図 2-1 ユーザ、グループ、プリンシパル、およびサブジェクトの間の関係



認証に成功すると、その一環として、プリンシパルは署名され、その後の使用に備えてサブジェクトに格納されます。プリンシパルに署名するのはプリンシパル検証プロバイダであり、実際にプリンシパルをサブジェクトに格納するのは LoginModule です。後で、サブジェクト内に格納されたプリンシパルに呼び出し側がアクセスしようとすると、そのプリンシパルが署名されてから変更されていないことをプリンシパル検証プロバイダが確認したうえで、そのプリンシパルが呼び出し側に返されます (それ以外のセキュリティ条件がすべて満たされたと仮定する)。

WebLogic Server ユーザまたはグループを表すプリンシパルは、`weblogic.security.spi` パッケージの `WLSUser` インタフェースと `WLSGroup` インタフェースを実装する必要があります。

JAAS (Java Authentication and Authorization Service)

クライアントが認証の必要なアプリケーション、アプレット、エンタープライズ JavaBean (EJB)、あるいはサーブレットのいずれであろうと、WebLogic Server では、JAAS (Java Authentication and Authorization Service) クラスを用いて、信頼性とセキュリティを確保しつつクライアントに対する認証を行います。JAAS では PAM (プラグイン可能な認証モジュール) フレームワークの Java 版を実装しており、それを使用することでアプリケーションは基礎となる認証技術からの独立性を保つことができるようになります。このため、PAM フレームワークを利用することで、アプリケーションに修正を加えることなく新しいまたは更新版の認証技術を使用することができます。

WebLogic Server は、リモートのファット クライアントの認証および内部の認証で JAAS を使用します。したがって、JAAS に直に関与する必要があるのは、カスタム認証プロバイダの開発者とファット クライアント アプリケーションの開発者だけです。シン クライアントのユーザまたはコンテナ内のファット クライアント アプリケーション (サーブレットからエンタープライズ JavaBeans を呼び出すものなど) の開発者は、JAAS を直接使用したり、その知識を身につけたりする必要はありません。

JAAS LoginModule

LoginModule は、認証処理における「馬車馬」のような存在です。すべての LoginModule は、セキュリティ レルム内のユーザの認証と、サブジェクト内への必要なプリンシパル (ユーザ / グループ) の格納を担当します。境界認証に使用されない LoginModule も、提示された証明データ (たとえば、ユーザのパスワード) が正しいかどうかを確認します。

セキュリティ レルムに複数の認証プロバイダがコンフィグレーションされている場合、各認証プロバイダの LoginModule は同じサブジェクトにプリンシパルを格納します。したがって、ある認証プロバイダの LoginModule によって、WebLogic Server ユーザ (`WLSUser` インタフェースの実装) を表すプリンシパル「Joe」が追加された場合、セキュリティ レルム内の他のすべての認証プロバイダは、「Joe」に出会ったときに同じ人物を参照する必要があります。つまり、他の認証プロバイダの LoginModule は、同じ人物を参照するために、WebLogic Server ユーザを表す別のプリンシパル (「Joseph」など) をサブジェクトに追加

しようとしてはなりません。ただし、別の認証プロバイダの LoginModule は、WLSUser 以外のタイプのプリンシパルを「Joseph」という名前で追加することができます。

JAAS 制御フラグ

セキュリティ レルムに複数の認証プロバイダがコンフィグレーションされている場合、認証プロバイダの制御フラグ属性で認証プロバイダの実行順序を決定します。次に、制御フラグ属性の値を示します。

- REQUIRED— この LoginModule は成功する必要があります。失敗しても、認証はコンフィグレーションされている認証プロバイダの LoginModule リストにある次の LoginModule に進みます。これはデフォルト設定です。
- REQUISITE— この LoginModule は成功する必要があります。他の認証プロバイダがコンフィグレーションされている場合、この LoginModule が成功すると、認証は LoginModule リストにある次の LoginModule に進みます。それ以外の場合、制御はアプリケーションに戻されます。
- SUFFICIENT— この LoginModule は成功する必要はありません。成功した場合、制御はアプリケーションに戻されます。他の認証プロバイダがコンフィグレーションされている場合、この LoginModule が失敗すると、認証は LoginModule リストにある次の LoginModule に進みます。
- OPTIONAL— ユーザはこれらの認証プロバイダの認証テストに合格してもしなくてもかまいません。ただし、セキュリティ レルムでコンフィグレーションされているすべての認証プロバイダで JAAS 制御フラグが OPTIONAL に設定されている場合は、ユーザはコンフィグレーションされているプロバイダのいずれか 1 つの認証テストに合格しなければなりません。

CallbackHandler

CallbackHandler は、可変個の引数を複合オブジェクトとしてメソッドに渡すことができるようにする高度に柔軟な JAAS 規格です。CallbackHandler のタイプは、NameCallback、PasswordCallback、および TextInputCallback の 3 つで、いずれも javax.security.auth.callback パッケージの一部です。NameCallback と PasswordCallback は、それぞれユーザ名とパスワードを返します。TextInputCallback は、ユーザがログイン フォームの追加フィールド (

ユーザ名とパスワードを取得するフィールド以外のフィールド) に入力したデータにアクセスするために使用します。TextInputCallback はフォームの追加フィールドにつき 1 つ必要で、各 TextInputCallback のプロンプト文字列はフォームのフィールド名と一致する必要があります。WebLogic Server は、フォームベースの Web アプリケーション ログインに対してだけ TextInputCallback を使用します。

アプリケーションは、CallbackHandler を実装し、それを基礎となるセキュリティ サービスに渡すことで、それらのサービスがアプリケーションと対話してユーザ名やパスワードなど特定の認証データを取得したり、エラーや警告メッセージなど特定の情報を表示したりできるようにします。

CallbackHandler は、アプリケーションに依存する形で実装されます。たとえば、グラフィカル ユーザ インタフェース (GUI) を持つアプリケーションに対する実装では、ウィンドウを表示して必要な情報をプロンプトで要求したり、エラー メッセージを表示したりできます。要求された情報をユーザからではなく、代替ソースから取得するように実装することもできます。

基礎となるセキュリティ サービスは、個々の Callbacks を CallbackHandler に渡すことによってさまざまなタイプの情報に対するリクエストを行います。CallbackHandler 実装は、渡された Callbacks に応じて情報を取得したり表示したりする方法を決定します。たとえば、基礎となるサービスがユーザを認証するためにユーザ名とパスワードを必要としていれば、NameCallback および PasswordCallback を使用します。その後、CallbackHandler は逐次的にユーザ名およびパスワードの入力を求めるか、または単一のウィンドウで両方の入力を求めるかを選択できます。

相互認証

相互認証を使用すると、クライアントとサーバの両方で相互に認証が必要になります。この認証は、証明書などの形態の証明データを使用して行うことができます。WebLogic Server では、相互認証の一形態である、双方向の SSL 認証がサポートされています。ただし厳密には、相互認証は SSL 認証よりも上位のプロトコル スタックで行われます。詳細については、2-19 ページの「一方向または双方向 SSL 認証」を参照してください。

ID アサーション プロバイダと LoginModule

LoginModule と一緒に用いると、ID アサーション プロバイダはシングル サインオンをサポートします。たとえば、ID アサーション プロバイダはデジタル証明書からトークンを生成することができ、そのトークンをシステム内で回すことができるため、ユーザは何度もサインオンを求められることはありません。

ID アサーション プロバイダが使用する LoginModule は、以下のことが可能です。

- 開発したカスタム認証プロバイダの一部となる。
- BEA が開発し WebLogic Server と一緒にパッケージ化した WebLogic 認証プロバイダの一部になる。
- サードパーティのセキュリティ ベンダの認証プロバイダの一部となる。

単純認証の場合と違って、ID アサーション プロバイダが使用する LoginModule は証明データ (ユーザ名とパスワードなど) を検証しません。単にユーザが存在することを検証するだけです。

ID アサーションとトークン

ID アサーション プロバイダでは、有効なトークンを WebLogic Server ユーザにマップする、ユーザ名マッパーがサポートされています。ID アサーション プロバイダは、ユーザまたはシステム プロセスの ID を断定するために使用する特定のトークン タイプをサポートするために開発します。ID アサーション プロバイダは複数のトークン タイプをサポートするように開発できますが、WebLogic Server 管理者はただ 1 つの「アクティブ」なトークン タイプを検証するように ID アサーション プロバイダをコンフィグレーションする必要があります。同じトークン タイプを検証する複数の ID アサーション プロバイダを 1 つのセキュリティ レルムに組み込むことも可能ですが、実際に検証を行うのは 1 つの ID アサーション プロバイダだけです。

注意： X.501 および X.509 証明書に対して WebLogic ID アサーション プロバイダを使用するには、`weblogic.security.providers.authentication.UserNameMapper` イ

ンタフェースの実装を提供する必要があります。詳細については、『WebLogic Security サービスの開発』の「カスタム ID アサーション プロバイダを開発する必要があるか」を参照してください。

認証のタイプ

WebLogic Server ユーザが、保護されている WebLogic リソースへのアクセスを要求する場合、必ず認証を受けなければなりません。このため、各ユーザは資格 (パスワードなど) を WebLogic Server に提示する必要があります。WebLogic Server 配布キットに含まれている WebLogic 認証プロバイダでは、以下のタイプの認証がサポートされています。

- 2-8 ページの「ユーザ名 / パスワード認証」
- 2-9 ページの「証明書認証」
- 2-9 ページの「境界認証」

WebLogic Server では、WebLogic Server 製品の一部として提供される WebLogic 認証プロバイダ、またはさまざまなタイプの認証を実行するカスタム セキュリティ プロバイダを使用できます。WebLogic 認証プロバイダと認証のコンフィグレーション方法については、4-10 ページの「資格マッピング プロセス」および『WebLogic Security の管理』の以下の節を参照してください。

- 「セキュリティ プロバイダのコンフィグレーション」
- 「SSL のコンフィグレーション」

ユーザ名 / パスワード認証

ユーザ名 / パスワード認証では、ユーザ ID とパスワードをユーザに要求し、WebLogic Server に送ります。WebLogic Server は受け取った情報をチェックして、信頼性を確認できれば、保護されている WebLogic リソースへのアクセスを許可します。

セキュア ソケット レイヤ (SSL)、または Hyper-Text Transfer Protocol with SSL (HTTPS) を使用すると、ユーザ名 / パスワード認証にさらに高度なレベルのセキュリティを提供できます。SSL は、クライアントと WebLogic Server との間で

転送されるデータを暗号化するので、ユーザ ID とパスワードはクリア テキストでは転送されません。したがって、WebLogic Server は、ユーザの ID およびパスワードの機密性を保持したままユーザを認証できます。

証明書認証

SSL または HTTPS クライアントのリクエストが送信されると、WebLogic Server は、クライアントに対してデジタル証明書を提示して応答します。クライアントによってそのデジタル証明書が確認されると、SSL 接続が確立されます。デジタル証明書は、WebLogic Server の身元を検証するエンティティ (信頼性のある認証局) によって発行されます。

相互認証の一形態である、双方向の SSL 認証を使用することもできます。双方向の SSL 認証では、クライアントとサーバの両方が証明書を提示しないと、両者の間で接続スレッドが有効になりません。2-19 ページの「一方向または双方向 SSL 認証」を参照してください。

注意： 双方向の SSL 認証は、WebLogic Server 製品の一部として提供される WebLogic 認証プロバイダでサポートされています。

境界認証

境界認証は、アプリケーション サーバ ドメインの外側にいるリモート ユーザの ID を認証するプロセスです。

以下の節では、境界認証について説明します。

- 2-9 ページの「境界認証の仕組み」
- 2-10 ページの「WebLogic Server が境界認証をサポートする仕組み」

境界認証の仕組み

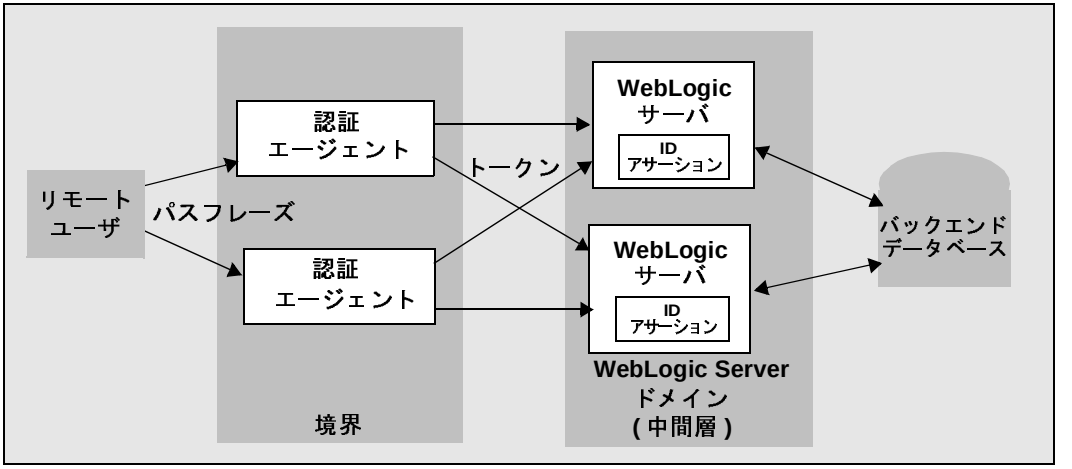
境界認証は通常、リモート ユーザが断定された ID と、検証の実行に使用される特定の形態の対応する証明データ (一般的にはパスフレーズの形態、具体的にはパスワード、クレジット カード番号、暗証番号などの個人を識別する情報) を指定することによって行われます。

実際に ID を保証するエンティティである、**認証エージェント**は、さまざまな形態を取ることができます。たとえば、仮想プライベート ネットワーク (VPN)、ファイアウォール、企業認証サービスといったグローバルな ID サービスの形態を取ることができます。認証エージェントのこれらの形態には共通の特徴があります。これらはすべて、認証を受けたユーザに関する情報を調べるために後で提示する必要があるアーティファクトつまり**トークン**を生成するという認証プロセスを実行します。現時点では、トークンの形式はベンダごとに異なりますが、XML を使用した標準のトークン形式の定義が進められています。さらに、属性証明書に対しては、デジタル証明書用の X.509 標準に基づいた標準があります。しかし、結局のところ、アプリケーションやそれが構築されているインフラストラクチャがこの概念をサポートするよう設計されていない場合は、企業のネットワーク内のアプリケーションに対してリモート ユーザは再認証を行う必要があります。

WebLogic Server が境界認証をサポートする仕組み

WebLogic Server は、ID アサーションのサポートを介して境界までシングル サインオン概念を拡張できるよう設計されています (図 2-2 を参照)。WebLogic Security フレームワークの重要な一部分として提供されている ID アサーションの概念に基づくことで、WebLogic Server では Checkpoint の OPSEC、新しく登場した SAML (Security Assertion Markup Language)、または Common Secure Interoperability (CSI) v2 などのプロトコルに対する拡張といった境界認証方式によって提供される認証メカニズムを使用して、この機能を実現できます。

図 2-2　境界認証



境界認証をサポートするには、1 つまたは複数のトークン フォーマットをサポートするように設計された ID アサーション プロバイダを使用する必要があります。複数の異なる ID アサーション プロバイダを登録して使用できます。トークンは、WebLogic Server がサポートするさまざまなプロトコルによって提供されるメカニズムを使用して、通常のビジネス リクエストの一部として転送されます。WebLogic Server でリクエストが受け取られると、プロトコル メッセージを処理するエンティティによってメッセージ内のトークンが認識されます。この情報は、WebLogic Security フレームワークの呼び出しに使用されます。WebLogic Security フレームワークは、トークンの検証を処理する適切な ID アサーション プロバイダを呼び出します。トークンを検証し、そのトークンの信頼性を確立するために必要なすべてのアクションの実行と、保証されたユーザ ID の提供は、ID アサーション プロバイダ実装によって行われます。ユーザはアプリケーションに対して再認証を行う必要がありません。

認可

認可とは、ユーザの ID などの情報に基づいて、ユーザと WebLogic リソースとのやり取りを管理するプロセスのことです。言い換えれば、認可とは「自分は何にアクセスできますか」という質問に答えるものです。WebLogic Server では、認可プロバイダはユーザと WebLogic リソースとの対話を制限して、整合性、機密性、および可用性を確保します。

以下の節では、認可の概念と機能について説明します。

- 2-12 ページの「WebLogic リソース」
- 2-13 ページの「セキュリティ ポリシー」
- 2-14 ページの「ContextHandler」
- 2-15 ページの「アクセス決定」
- 2-15 ページの「裁決」

WebLogic リソース

WebLogic リソースは、基底の WebLogic Server エンティティを表す構造化オブジェクトです。セキュリティ ロールとセキュリティ ポリシーを使用すると、これらのオブジェクトを権限のないアクセスから保護できます。

WebLogic リソースは階層化されています。このため、セキュリティ ロールとセキュリティ ポリシーは自由なレベルで定義できます。たとえば、エンタープライズ アプリケーション (EAR) 全体、複数の EJB を含むエンタープライズ JavaBean (EJB) JAR、その JAR 内の特定のエンタープライズ JavaBean (EJB)、その EJB 内の単一のメソッドなどに対してセキュリティ ロールとセキュリティ ポリシーを定義できます。

以下の WebLogic リソースの実装を使用できます。

- 管理リソース
- アプリケーション リソース
- コンポーネント オブジェクト モデル (COM) リソース

- エンタープライズ情報システム (EIS) リソース
- エンタープライズ JavaBean (EJB) リソース
- Java Database Connectivity (JDBC) リソース
- Java Message Service (JMS) リソース
- Java Naming and Directory Interface (JNDI) リソース
- サーバ リソース
- Uniform Resource Locator (URL) リソース

注意： これらの WebLogic リソースの実装の詳細については、WebLogic Server 7.0 API リファレンス Javadoc を参照してください。詳細については、『WebLogic リソースのセキュリティ』の「WebLogic リソースのタイプ」を参照してください。

セキュリティ ポリシー

以前のバージョンの WebLogic Server では、アクセス制御リスト (ACL) を使って WebLogic リソースを保護していました。このリリースの WebLogic Server では、WebLogic リソースに「誰がアクセスできるか」という問いに対し、ACL ではなくセキュリティ ポリシーが答えます。セキュリティ ポリシーは、WebLogic リソースと、1 つまたは複数のユーザ、グループ、またはセキュリティ ロールとの間に関連付けを定義するときに作成されます。また、セキュリティ ポリシーに時間制約を定義することもできます。WebLogic リソースは、セキュリティ ポリシーが割り当てられるまでは保護されません。

セキュリティ ポリシーは、定義済みの任意の WebLogic リソース (EJB リソースや JNDI リソースなど)、または WebLogic リソースの特定のインスタンス (Web アプリケーション内の EJB メソッドやサーブレット) の属性または操作に割り当てることができます。セキュリティ ポリシーを WebLogic リソースのタイプに割り当てると、そのリソースのすべての新しいインスタンスがそのセキュリティ ポリシーを継承します。個々のリソースまたは属性に割り当てられたセキュリティ ポリシーは、WebLogic リソースのタイプに割り当てられたセキュリティ ポリシーをオーバーライドします。定義済みの WebLogic リソースのリストについては、2-12 ページの「WebLogic リソース」を参照してください。

セキュリティ ポリシーは、認可プロバイダのデータベースに格納されます。デフォルトでは、WebLogic 認可プロバイダがコンフィグレーションされ、セキュリティ ポリシーは組み込み LDAP サーバに格納されます。

ユーザまたはグループを使用してセキュリティ ポリシーを作成する場合は、そのユーザまたはグループがデフォルト セキュリティ レルムでコンフィグレーションされた認証プロバイダのセキュリティ プロバイダ データベースで定義されていなければなりません。セキュリティ ロールを使用してセキュリティ ポリシーを作成する場合は、そのセキュリティ ロールがデフォルト セキュリティ レルムでコンフィグレーションされたロール マッピング プロバイダのセキュリティ プロバイダ データベースで定義されていなければなりません。WebLogic 認証プロバイダおよびロール マッピング プロバイダは、デフォルトでは組み込み LDAP サーバ内のデータベースにコンフィグレーションされています。

デフォルトで、WebLogic Server には WebLogic リソース用のセキュリティ ポリシーが定義されています。これらのセキュリティ ポリシーは、セキュリティ ロールとデフォルト グローバル グループに基づいています。また、必要に応じてセキュリティ ポリシーはユーザに基づくこともできます。セキュリティ ポリシーは、ユーザまたはグループではなく、セキュリティ ロールに基づいて作成するようにしてください。セキュリティ ロールに基づいてセキュリティ ポリシーを作成すると、ユーザまたはグループに付与されるセキュリティ ロールに基づいてアクセスを管理できます。管理の方法としてはこちらの方が効率的です。WebLogic リソースのデフォルト セキュリティ ポリシーのリストについては、『WebLogic リソースのセキュリティ』の「デフォルト セキュリティ ポリシー」を参照してください。

ContextHandler

ContextHandler は、リソース コンテナから追加のコンテキストとコンテナ固有の情報を取得し、アクセスやロール マッピングを決定するセキュリティ プロバイダにこれらの情報を提供する高性能の WebLogic クラスです。`ContextHandler` インタフェースを使用すると、内部 WebLogic リソース コンテナで WebLogic Security フレームワーク呼び出しに追加情報を渡すことができます。その結果、セキュリティ プロバイダは、特定のメソッドの引数で提供される以上のコンテキスト情報を取得できるようになります。ContextHandler は本質的には名前と値のリストなので、セキュリティ プロバイダは検索する名前を認識しておく必要があります。つまり、ContextHandler の使用には、WebLogic リ

ソース コンテナとセキュリティ プロバイダの間の緊密な連携が不可欠です。ContextHandler の名前と値の各組み合わせは、**コンテキスト要素**と呼ばれ、`ContextElement` オブジェクトによって表されます。

現在、2 種類の WebLogic リソース コンテナ (サーブレット コンテナと EJB コンテナ) が ContextHandler を WebLogic Security フレームワークに渡します。そのため、URL (Web) および EJB リソース タイプには、異なるコンテキスト要素が含まれます。これらのコンテキスト要素の値は、認可プロバイダまたはロール マッピング プロバイダで調べることができます。監査イベントをポストするためにセキュリティ プロバイダが実装される場合に使用する `AuditContext` インタフェースの実装によっても、コンテキスト要素の値を調べることができます。

特定のコンテキスト要素の値に関する詳細については、『WebLogic Security サービスの開発』の「ContextHandler と WebLogic リソース」を参照してください。

アクセス決定

認証プロバイダの LoginModule と同じく、**アクセス決定**は、「アクセスは許可されるのか」という質問に答える認可プロバイダのコンポーネントです。具体的には、指定された操作を WebLogic リソースに対して実行するパーミッションをサブジェクトが持っているかどうかが、アプリケーション内の特定のパラメータを用いてアクセス決定に質問されます。この情報が与えられると、アクセス決定はそれに対する回答を `PERMIT`、`DENY`、または `ABSTAIN` のいずれかで返します。

裁決

裁決では、セキュリティ レルムで複数の認可プロバイダがコンフィグレーションされている場合に発生するおそれのある認可上の衝突を、それぞれの認可プロバイダのアクセス決定の結果を比較検討することによって解消します。WebLogic Server では、裁決プロバイダを使用して、複数のアクセス決定から返される結果を調停し、`PERMIT` か `DENY` の最終判定を下します。また、裁決プロバイダでは、単一の認可プロバイダのアクセス決定から `ABSTAIN` の回答が返されたときにどうすべきかを指定することもできます。

セキュア ソケット レイヤ (SSL)

SSL は、Web 経由で接続されたアプリケーション間でのセキュアな通信を可能にします。SSL 通信のコンポーネントの詳細、およびそれぞれのコンポーネントが必要な理由については、Netscape Communications Corporation が提供する「How SSL Works」(`http://developer.netscape.com/tech/security/ssl/howitworks.html`) を参照してください。

WebLogic Server では、SSL 通信が完全にサポートされています。デフォルトでは、WebLogic Server では一方向 SSL 認証がコンフィグレーションされています。Administration Console を使用して、WebLogic Server に対して双方向 SSL 認証をコンフィグレーションできます。

WebLogic Server で SSL を使用するには、プライベート キー、対となる公開鍵を含むデジタル証明書、および対となる公開鍵を含むデジタル証明書に組み込まれたデータを検証できる信頼性のある認証局 (CA) の少なくとも 1 つによって署名されたデジタル証明書が必要となります。デジタル証明書に署名した認証局が不明な場合は、WebLogic Server に信頼性のあるルート CA をインストールしなければならないこともあります。信頼性のあるルート CA とは、有名な認証局によって署名された CA です。

サーバのデジタル証明書を取得するには、公開鍵、プライベート キー、および公開鍵を含む証明書署名リクエスト (CSR) を生成します。認証局に CSR リクエストを提出し、署名された証明書を取得する手順に従ってデジタル証明書を取得します。

必要となるプライベート キー、デジタル証明書、および信頼性のある CA をすべて取得したら、WebLogic Server が ID 検証にそれらを使用できるよう、それらを格納する必要があります。プライベート キーと対になる公開鍵を含むデジタル証明書は、ドメイン ディレクトリ内のファイルにのみ格納できます。プライベート キーは、ドメイン ディレクトリ内のファイルまたは WebLogic キーストア プロバイダを介してアクセスされるキーストアのいずれかに格納できます。信頼性のある CA の証明書はファイルまたはキーストアに格納できます。プライベート キー、公開鍵、および証明書の要件と手順の詳細については、『WebLogic Security の管理』の「SSL のコンフィグレーション」を参照してください。

ブラウザを使用して WebLogic Server アプリケーションと接続する場合に SSL を使用するには、単純に HTTPS とセキュア ポート (ポート番号 7002) を URL で指定します。次に例を示します。

```
https://localhost:7002/examplesWebApp/SnoopServlet.jsp
```

localhost は、Web アプリケーションをホストしているシステムの名前です。

この節では、以下の内容について説明します。

- 2-17 ページの「SSL の機能」
- 2-18 ページの「SSL トンネリング」
- 2-19 ページの「一方向または双方向 SSL 認証」
- 2-20 ページの「国内向け SSL と輸出向け SSL」
- 2-21 ページの「デジタル証明書」
- 2-22 ページの「認証局」
- 2-23 ページの「ホスト名検証」
- 2-23 ページの「トラスト マネージャ」
- 2-24 ページの「非対称鍵アルゴリズム」
- 2-25 ページの「対称鍵アルゴリズム」
- 2-26 ページの「メッセージ ダイジェスト アルゴリズム」
- 2-26 ページの「暗号スイート」

SSL の機能

WebLogic Server では、SSL の pure-Java 実装が提供されます。一般的に、SSL は以下の機能を提供します。

- 通信を行うアプリケーションが互いの身元を認証するために利用するメカニズム
- アプリケーション間でやり取りするデータの暗号化

SSL を使用する場合、対象 (サーバ) は常に発信元 (クライアント) に対して自身を認証します。対象が要求した場合は、発信元が自身を対象に対して認証することもあります。ネットワーク経由で送信されるデータは暗号化されるので、予定されている宛先以外には解釈できません。SSL 接続はデジタル証明書をやり取りするアプリケーション間のハンドシェークで始まり、使用する暗号化アルゴリズムの取り決め、そのセッションの残りで使用する暗号キーの生成と続きます。

具体的には、SSL が提供するセキュリティ機能は以下のとおりです。

- サーバ認証 —WebLogic Server は、信頼性のある認証局が発行したデジタル証明書を使用して、クライアントに対して認証します。SSL では最低でも、サーバが自身のデジタル証明書を使用してクライアントに対して認証を行うことが必要になります。デジタル証明書の提示がクライアントに要求されない場合の接続タイプは、いわゆる一方向 SSL 認証です。
- クライアント識別確認 — 場合によっては、WebLogic Server に対するデジタル証明書の提示がクライアントに要求されることもあります。WebLogic Server で、そのデジタル証明書が信頼性のある認証局によって発行されたものであることが確認されると、SSL 接続が確立されます。デジタル証明書が提示されなかったり、確認されなかったりした場合、SSL 接続は確立されません。この接続タイプは、相互認証の一形態である、いわゆる双方向の SSL 認証です。
- 機密性 — クライアントのリクエストとサーバの応答は暗号化され、ネットワーク経由でやり取りされるデータの機密性が保持されます。
- データの整合性 — クライアントと WebLogic Server との間のデータ フローを、第三者のユーザ ID 検証による改ざんから保護します。

Web ブラウザを使って WebLogic Server と通信する場合は、Hypertext Transfer Protocol with SSL (HTTPS) を利用して、ネットワーク通信の安全性を確保できます。

SSL トンネリング

SSL は、IP ベースのプロトコル上をトンネリングされます。トンネリングとは、各 SSL レコードをカプセル化し、別のプロトコル上でレコードを送信するために必要なヘッダと一緒にパッケージ化することです。SSL は、以下のように Web ブラウザおよび Java クライアントで使用できます。

- Web ブラウザと WebLogic Server との SSL 通信は、HTTPS パケットにカプセル化されて転送されます。次に例を示します。

  ```
  https://myserver.com/mypage.html
  ```

  WebLogic Server では、SSL バージョン 3 をサポートする Web ブラウザで HTTPS を使用できます。WebLogic Server の Java 仮想マシン (JVM) では、現在、HTTPS アダプタがサポートされていません。したがって、WebLogic Server は、Web ブラウザでの SSL 実装に依存します。

- SSL を使って WebLogic Server に接続する Java クライアントは、BEA の多重化 T3 プロトコル上をトンネリングします。次に例を示します。

  ```
  t3s://myserver.com:7002/mypage.html
  ```

  WebLogic Server 内で動作する Java クライアントは、他の WebLogic Server に T3S 接続するか、または Web サーバやセキュアなプロキシ サーバなど SSL をサポートする他のサーバに HTTPS 接続します。

一方向または双方向 SSL 認証

WebLogic Server では、一方向と双方向の SSL 認証がサポートされています。一方向 SSL 認証では、対象 (サーバ) が発信元 (クライアント) に対してデジタル証明書を提示して身元を証明する必要があります。クライアントは、2 つのチェックを行って、デジタル証明書を検証します。

1. デジタル証明書が信頼性のある認証局のリストにあるかどうか。
2. 証明書にあるホスト名がサーバ名に一致するかどうか。

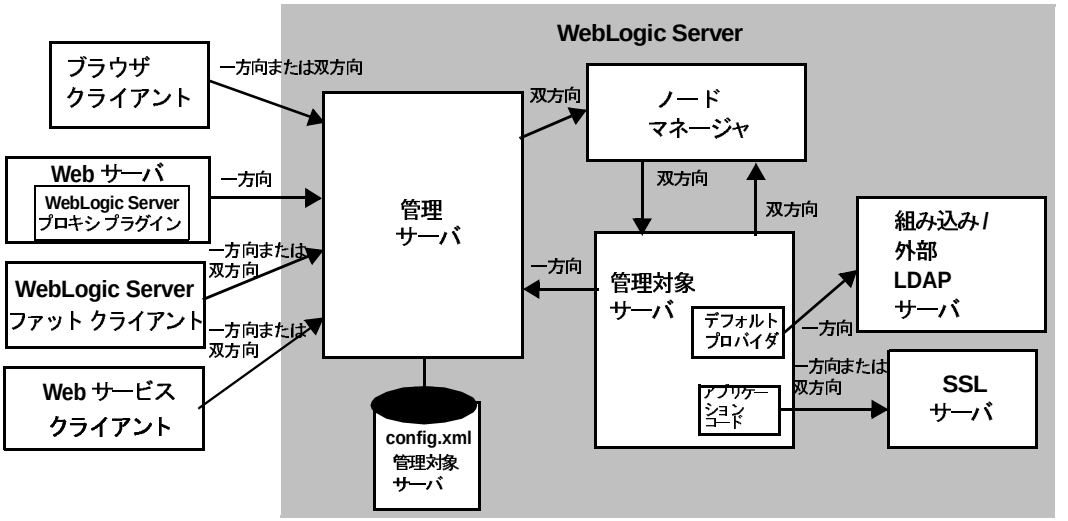
上記のチェックの両方で true が返されると、SSL 接続が確立します。

双方向の SSL 認証では、クライアントとサーバの両方がデジタル証明書を提示しないと、両者の間で SSL 接続が有効になりません。そのため、この場合、WebLogic Server はクライアントに対して自身を認証するだけでなく (証明書認証の最低限の要件)、要求側のクライアントにも認証を要求します。双方向 SSL 認証は、アクセスを許可する対象を信頼されたクライアントに制限する場合に便利です。

図 2-3 に、WebLogic Server の SSL 接続と、一方向 SSL、双方向 SSL、またはその両方をサポートする接続を示します。Web ブラウザ クライアント、Web サーバ、ファット クライアント、Web サービス クライアント、および SSL サーバの

接続に対して一方向 SSL または双方向 SSL のいずれかをコンフィグレーションできます。WebLogic Server は、SSL 接続が一方向または双方向のどちらにコンフィグレーションされているかを判別します。SSL は、Administration Console を使用してコンフィグレーションします。

図 2-3　WebLogic Server が SSL 接続をサポートする仕組み



注意：図中の SSL サーバは、任意の J2EE 準拠サーバ

国内向け SSL と輸出向け SSL

WebLogic Server は、輸出向けと国内向けのどちらの強度の SSL でも使用できます。

- 輸出向け SSL は、512 ビットの証明書と 40 ビットおよび 50 ビットのバルク データ暗号化をサポートします。
- 国内向け SSL は、768 ビットと 1024 ビットの証明書、および 128 ビットのバルク データ暗号化をサポートします。

標準の WebLogic Server 配布キットは、輸出向けの強度の SSL だけをサポートします。国内向けバージョンについては、BEA の販売代理店を通じてお求めください。

注意： 国内向け強度の WebLogic Server が必要であり、その資格を満たしている場合には、国内向けの WebLogic Server ソフトウェア ライセンスを受け取ることになります。そのライセンスは WebLogic Server 配布キットのインストール時に使用します。

米国政府は、2000 年初めに暗号化ソフトウェアの輸出に関する規制を緩和したので、国内向けバージョンの WebLogic Server はほとんどの国でご使用いただけます。

暗号の強度が高いので、国内向け WebLogic Server 配布キットをお勧めします。

注意： 輸出向け強度の WebLogic Server 配布キットを使って証明書署名リクエスト (CSR) (証明書に対して電子的に生成されるリクエスト) を生成する場合は、国内向け強度の SSL 接続に対応できないので、国内向けの証明書を提示するクライアントを認証できません。

詳細については、『WebLogic Security の管理』の「SSL のコンフィグレーション」を参照してください。

デジタル証明書

デジタル証明書とは、インターネットなどのネットワークを経由して、プリンシパルおよびエンティティのユニークな ID を確認するための電子的なドキュメントのことです。デジタル証明書は、認証局として知られる信頼された第三者によって確認されたユーザまたはエンティティの ID を、特定の公開鍵に安全にバインドします。公開鍵とプライベート キー (秘密鍵) を組み合わせることで、デジタル証明書のオーナにユニークな ID が提供されます。

デジタル証明書を使用すると、特定の公開鍵が実際に特定のユーザまたはエンティティに属しているかどうかを確認できます。デジタル証明書の受信側は、証明書内の公開鍵を使用して、デジタル署名が対応するプライベート キーで作成されたものかどうかを確認します。こうした確認が終わると、この推論のチェーンによって、対応するプライベート キーの所有者がデジタル証明書に名前のあるサブジェクトであること、そしてそのデジタル署名を作成したのがそのサブジェクトであることを確認できます。

一般に、デジタル証明書には以下のようにさまざまな情報が入っています。

- サブジェクト (保持者、オーナ) の名前と、デジタル証明書を使用する Web サーバの URL や個人の電子メール アドレスなど、そのサブジェクトの ID をユニークに確認するために必要なその他の情報
- 主体の公開鍵
- デジタル証明書を発行した認証局の名前
- シリアル番号
- (開始日と終了日で定義される) デジタル証明書が有効性を持つ期間 (有効期間)

最も広く使われているデジタル証明書のフォーマットは、ITU-T X.509 国際標準で定義されているものです。X.509 標準に準拠していればどのアプリケーションを使っても、デジタル証明書を読み書きできます。WebLogic Server の公開鍵インフラストラクチャ (PKI) では、X.509 バージョン 3 (X.509v3) に準拠したデジタル証明書が認識されます。Verisign や Entrust などの認証局から証明書を取得することをお勧めします。

詳細については、『WebLogic Security の管理』の「SSL のコンフィグレーション」を参照してください。

認証局

デジタル証明書は、認証局によって発行されます。デジタル証明書および公開鍵の発行先の ID を保証する信頼された第三者組織または企業はすべて、認証局になることができます。認証局は、デジタル証明書を作成する際に、証明書が改ざんされればわかるようにプライベート キーを使って署名します。認証局は、署名したデジタル証明書を要求側に返します。

要求側は、認証局の公開鍵を使って発行認証局の署名を確認します。認証局の公開鍵は、低レベル認証局の公開鍵の妥当性を保証する高レベル認証局から発行された証明書を提供することで利用できるようになります。この方式によって、認証局の階層が構築されます。この階層は、最終的に、ルート証明書と呼ばれる、最上位の自己署名証明書にたどり着きます。この証明書では、証明にそれ以外の一切の公開鍵が不要です。ルート証明書は、信頼性のある (ルート) 認証局によって発行されます。

受信側がすでに信頼性を確認済みの、該当認証局よりも高レベルの認証局が署名した該当認証局の公開鍵が入ったデジタル証明書を持っている場合、暗号化されたメッセージの受信側は、認証局の公開鍵を再帰的に信頼します。この点では、

デジタル証明書はデジタルの信頼関係の踏み台です。結局、信頼する必要があるのは、ごく少数の最上位認証局の公開鍵だけです。証明書のチェーンを通じて、多数のユーザのデジタル署名の信頼を確立できます。

通信中のエンティティの ID は、このようにしてデジタル署名によって確立できますが、デジタル署名による信頼は、信頼性を確認するための公開鍵の範囲にとどまります。

詳細については、『WebLogic Security の管理』の「SSL のコンフィグレーション」を参照してください。

ホスト名検証

ホスト名検証は、SSL 接続先のホスト名が予定していた通信先、または許可された通信先であることを確認するプロセスです。Web クライアント (Web ブラウザ、WebLogic クライアント、またはクライアントとして機能している WebLogic Server) が別のアプリケーション サーバとの SSL 接続を要求する場合に、介在者の攻撃を防ぎます。

WebLogic Server の SSL ハンドシェーク機能としてのデフォルトの動作は、SSL サーバのデジタル証明書の SubjectDN にある一般名と、SSL 接続の開始に使用する SSL サーバのホスト名を比較することです。これらの名前が一致しない場合は SSL 接続が中断されます。

トラスト マネージャ

SSL クライアントが SSL サーバに接続すると、SSL サーバは認証のためにデジタル証明書チェーンをクライアントに提示します。提示されたチェーンに無効なデジタル証明書が含まれている場合もあります。SSL 仕様では、クライアントが無効な証明書を検出した場合、SSL 接続が中断されることになっています。しかし、Web ブラウザは、無効な証明書を無視するかどうかをユーザに確認し、証明書チェーン内の残りの証明書を使用して SSL サーバを認証できるかどうかを判別するため、チェーンの検証を継続します。トラスト マネージャを使用すると、どのような場合に SSL 接続を継続するか (または中止するか) を制御でき、上記のような矛盾した動作をなくすことができます。トラスト マネージャを使用すると、SSL 接続を続行する前にカスタム検証を実行できます。たとえ

ば、トラスト マネージャを使用して、特定の地域 (町、州、国など) のユーザやその他の特殊な属性を持つユーザだけが SSL 接続を介してアクセスを取得できるように指定することができます。

WebLogic Server は `weblogic.security.SSL.TrustManagerJSSE` インタフェースを提供します。このインタフェースを使用すると、ピアのデジタル証明書内での検証エラーをオーバーライドし、SSL ハンドシェークを継続できます。また、サーバのデジタル証明書チェーンで付加的な検証を実行することで、SSL ハンドシェークを中止することもできます。

注意 : このインタフェースは新しいスタイルの証明書を受け取り、このリリースの WebLogic Server では非推奨の `weblogic.security.SSL.TrustManager` インタフェースの代わりに使用されます。

非対称鍵アルゴリズム

非対称鍵 (公開鍵ともいう) アルゴリズムは、公開鍵、プライベート キーという数学的には互いに関連性を持ちながらも異なる 2 つの鍵を利用します。

- 公開鍵 (一般に配布) は、デジタル署名を確認したり、データを外見上解釈不能な形式に変換したりするために使用されます。
- プライベート キー (非公開のまま保管) は、デジタル署名を作成したり、データを元の形式に戻したりするために使用されます。

WebLogic Server の公開鍵インフラストラクチャ (PKI) もデジタル署名のアルゴリズムをサポートします。デジタル署名アルゴリズムは、デジタル署名を生成するためだけの公開鍵アルゴリズムのことです。

WebLogic Server では、Rivest、Shamir、Adelman (RSA) アルゴリズムがサポートされています。

公開鍵とプライベート キーの保存については、3-16 ページの「キーストア プロバイダ」を参照してください。

対称鍵アルゴリズム

対称鍵アルゴリズムでは、メッセージの暗号化と解読に同じ鍵を使用します。共通鍵暗号化システムは対称鍵アルゴリズムを使用して、通信中の 2 つのエンティティ間でやり取りされるメッセージを暗号化します。対称暗号化は、公開鍵暗号作成法よりも少なくとも 1000 倍の早さで処理を実行できます。

ブロック暗号は、プレーン テキスト (暗号化されていないテキスト) の固定長ブロックを同じ長さの暗号テキスト (暗号化済みテキスト) データ ブロックに変換する対称鍵アルゴリズムの一種です。この変換は、ランダムに生成したセッション キーの値に従って発生します。固定長はブロック サイズと呼ばれます。

WebLogic Server の PKI は以下の対称鍵アルゴリズムをサポートしています。

- DES-CBC (Data Encryption Standard - Cipher Block Chaining)

  DES-CBC は、Cipher Block Chaining (CBC) モードの 64 ビット ブロックの暗号文です。56 ビットの鍵 (8 パリティ ビットをフル 64 ビット鍵から除去します) を使用します。

- 2 キー形式トリプル DES (Data Encryption Standard)

  2 キー形式トリプル DES は、Encrypt-Decrypt-Encrypt (EDE) モードの 128 ビット ブロックの暗号です。2 キー形式トリプル DES では、2 つの 56 ビット鍵 (実効では 112 ビットの鍵 1 つ) を利用します。

  現在では、トリプル DES による DES 暗号鍵を保護および転送する方式が一般的となっています。つまり、入力データ (この場合には シングル DES 鍵) を暗号化し、解読してから、もう一度暗号化します (暗号化 - 解読 - 暗号化処理)。同じキーを 2 回の暗号化処理に使います。

- RC4 (Rivest's Cipher 4)

  RC4 は、サイズが 40 ～ 128 ビットの鍵を使用する可変キー サイズ ブロック暗号です。DES よりも処理が高速で、キー サイズを 40 ビットにして輸出可能です。米国企業の海外子会社および海外支社には、56 ビットのキー サイズが許可されています。WebLogic Server PKI によってキー長が 128 ビットに制限されていますが、米国では、事実上キー長の制限なしに RC4 を使用できます。

注意： WebLogic Server ユーザは、このアルゴリズムのリストを拡張したり変更したりすることができません。

メッセージ ダイジェスト アルゴリズム

WebLogic Server は、MD5 および SHA (Secure Hash Algorithm) メッセージ ダイジェスト アルゴリズムをサポートしています。MD5 も SHA も、広く知られた一方向のハッシュ アルゴリズムです。一方向のハッシュ アルゴリズムは、メッセージを、メッセージ ダイジェストまたはハッシュ値と呼ばれる数字の固定文字列に変換します。

MD5 は高速処理可能な 128 ビット ハッシュで、32 ビット マシンで使用することを想定しています。MD5 と比べて、SHA は 160 ビット ハッシュを用いてより高度なセキュリティを提供しますが、処理速度は低下します。

暗号スイート

暗号スイートとは、鍵交換アルゴリズム、対称暗号化アルゴリズム、およびセキュア ハッシュ アルゴリズムを含む SSL 暗号方式の一種です。通信の整合性を保護するために使用します。たとえば、`RSA_WITH_RC4_128_MD5` という暗号スイートは、鍵交換用に RSA、バルク暗号化用に 128 ビット キーを使う RC4、およびメッセージ ダイジェスト用に MD5 を使用します。

WebLogic Server で使用されている暗号化パッケージは FIPS 140-1 に準拠しています。

WebLogic Server でサポートされている暗号スイートは、表 2-1 のとおりです。

表 2-1 でサポートされている SSL 暗号スイート WebLogic Server

暗号スイート	対称鍵の強度
SSL_RSA_WITH_RC4_128_SHA	128
SSL_RSA_WITH_RC4_128_MD5	128
TLS_RSA_WITH_DES_CBC_SHA	56
SSL_RSA_EXPORT_WITH_RC4_40_MD5	40
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA	40
SSL_RSA_WITH_3DES_EDE_CBC_SHA	112
SSL_RSA_WITH_NULL_SHA	0
SSL_RSA_WITH_NULL_MD5	0
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA	56
TLS_RSA_EXPORT124_WITH_RC4_56_SHA	56

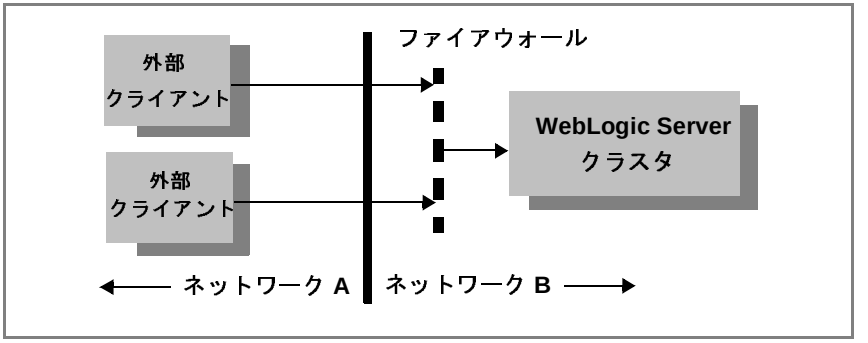
WebLogic Server のライセンスによって、通信を保護するために使用する暗号スイートの強度 (国内向けまたは輸出向け) が決まります。WebLogic Server config.xml ファイルで定義されている暗号スイートの強度がライセンスに指定されている強度を超える場合、サーバはライセンスに指定されている強度を使用します。たとえば、持っているライセンスが輸出向けで、国内向け暗号スイートを使用するよう config.xml ファイルで定義した場合は、サーバは国内向け暗号スイートを拒否し、輸出向け暗号スイートを使用します。

ファイアウォール

ファイアウォールとは、2 つのネットワーク間のトラフィックを制限する機能のことです。ファイアウォールは、ソフトウェアとハードウェア (ルータや専用のゲートウェイ マシンなど) を組み合わせて作成できます。ファイアウォールは、プロトコル、要求されたサービス、ルーティング情報、および送信元と送信先のホストまたはネットワークに基づいてトラフィックの通過を許可または拒否するフィルタを利用します。また、認証されたユーザについてもアクセスを許可する場合があります。

図 2-4 に、WebLogic Server クラスタ宛てのトラフィックをフィルタ処理するファイアウォールを使った代表的な設定を示します。

図 2-4　代表的なファイアウォールの設定



WebLogic Server では、ファイアウォールと組み合わせて、以下の機能を使用できます。

- 2-29 ページの「接続フィルタ」
- 2-29 ページの「境界認証」

接続フィルタ

WebLogic Server 接続フィルタを使用して、プロトコル、IP アドレス、および DNS ノード名に基づいてネットワーク トラフィックをフィルタ処理するファイアウォールを設定できます。詳細については、『WebLogic Security プログラマーズ ガイド』の「ネットワーク接続フィルタを使用したアプリケーション サーバリソースの保護」を参照してください。

境界認証

ID アサーション プロバイダを使用して、**境界認証** (トークンを使用する特別なタイプの認証) を設定できます。WebLogic Server のセキュリティ アーキテクチャは、境界に基づく認証 (Web サーバ、ファイアウォール、VPN) を実行し、複数のセキュリティ トークン タイプ / プロトコル (SOAP、IIOP-CSIv2) を処理する ID アサーション プロバイダをサポートします。

J2EE と WebLogic セキュリティ

ユーザ認証とユーザ認可を実装および使用するために、BEA WebLogic Server では、Java 2 Platform、Enterprise Edition (J2EE) の SDK バージョン 1.3 のセキュリティ サービスが利用されます。他の J2EE コンポーネントと同様、セキュリティ サービスも標準化されたモジュール コンポーネントに基づいています。BEA WebLogic Server は、標準に従ってこれらの Java セキュリティ サービス メソッドを実装し、細かなアプリケーションの動作をプログラミングを必要とせずに自動的に処理する拡張を追加します。

BEA WebLogic Server の SDK 1.3 セキュリティのサポートにより、アプリケーションの開発者は、Sun Microsystems のセキュリティ領域の最新の強化拡張と開発を利用できます。このため、企業投資を主に Java プログラミング技術に傾注できます。定義され、文書化された Java 標準に従うことで、WebLogic Server のセキュリティ サポートでは Java 開発者に対する共通の基準が確立されます。WebLogic Server によって提供される新技術は、SDK 1.3 のサポートに基づいています。

この節では、以下の内容について説明します。

- 2-30 ページの「SDK 1.3 セキュリティ パッケージ」
- 2-32 ページの「CSIv2 (Common Secure Interoperability Version 2)」

SDK 1.3 セキュリティ パッケージ

WebLogic Server は、以下の SDK 1.3 セキュリティ パッケージに準拠し、それらのパッケージをサポートしています。

- 2-30 ページの「Java Secure Socket Extension (JSSE)」
- 2-30 ページの「Java Authentication and Authorization Service (JAAS)」
- 2-31 ページの「Java セキュリティ マネージャ」
- 2-32 ページの「Java Cryptography Architecture と Java Crytography Extension (JCE)」

Java Secure Socket Extension (JSSE)

JSSE は、SSL と TLS v1 プロトコルをサポートおよび実装し、それらのプロトコルの機能をプログラムで利用可能にするパッケージのセットです。WebLogic Server では、WebLogic Server クライアント、および他のサーバとの間で転送されるデータを暗号化するために、セキュア ソケット レイヤ (SSL) がサポートされています。

JSSE は SSL 機能のクラスのコア セットを提供し、Certicom などの企業はそれらのクラスに対する拡張を提供します。WebLogic Server では、SSL の実装に Certicom JSSE 拡張を使用します。

Java Authentication and Authorization Service (JAAS)

JAAS は、ユーザベースの認証およびアクセス制御のためのフレームワークを提供するパッケージのセットです。WebLogic Server では、JAAS の認証用クラスだけを使用します。

JAAS は以下の場合に使用されます。

- リモート Java クライアント認証
- Web コンテナや EJB コンテナ内、および WebLogic 認証プロバイダや ID アサーション プロバイダ内における WebLogic Server インスタンス間の内部認証

JAAS の詳細については、2-4 ページの「JAAS (Java Authentication and Authorization Service)」を参照してください。

Java セキュリティ マネージャ

Java セキュリティ マネージャは、Sun Microsystems, Inc. によって開発された、Java 仮想マシン (JVM) のセキュリティ マネージャです。セキュリティ マネージャは、Java API と連携し `java.lang.SecurityManager` クラスを通じてセキュリティ境界を定義します。`SecurityManager` クラスは、プログラマが各自の Java アプリケーション用のカスタム セキュリティ ポリシーを確立することを可能にします。

Java セキュリティ マネージャと WebLogic Server を一緒に使用すると、JVM 内で実行されている WebLogic リソースのセキュリティを強化できます。WebLogic Server での Java セキュリティ マネージャを使用した WebLogic リソースの保護は、省略可能なセキュリティ手順です。

Java セキュリティ マネージャを使用すると、WebLogic リソースを保護するために、以下のセキュリティ タスクを行うことができます。

- `weblogic.policy` ファイルの修正
- EJB およびリソース アダプタに対するアプリケーション型のセキュリティ ポリシーの設定

  このタスクは、Java セキュリティ ポリシー ファイルを使用して行います。
- 特定の EJB およびリソース アダプタに対するアプリケーション固有のセキュリティ ポリシーの設定

  このタスクは、デプロイメント記述子 (`weblogic.xml`、`weblogic-ejb-jar.xml`、および `rar.xml`) を使用して行います。

これらのタスクに対する Java セキュリティ マネージャの使用方法の詳細については、『WebLogic Security プログラマーズ ガイド』の「Java セキュリティ マネージャを使用しての WebLogic リソースの保護」を参照してください。

Java Cryptography Architecture と Java Crytography Extension (JCE)

Sun Microsystems, Inc. によって開発された、これらのセキュリティ API は、Java プラットフォーム向けの暗号機能へのアクセスおよび暗号機能の開発と、暗号、鍵の生成と合意、および Message Authentication Code (MAC) アルゴリズムの実装の開発に使用するフレームワークを提供します。

WebLogic Server ではこれらのセキュリティ API が完全にサポートされています。

CSIv2 (Common Secure Interoperability Version 2)

WebLogic Server では、IIOP (Internet Inter-ORB) (GIOP バージョン 1.2) および CORBA CSIv2 (Common Secure Interoperability version 2) 仕様に基づいた EJB (Enterprise JavaBean) 相互運用性プロトコルがサポートされています。WebLogic Server で CSIv2 をサポートすることにより、以下のことが可能になります。

- J2EE (Java 2 Enterprise Edition) バージョン 1.3 の参照実装と相互運用できる。
- WebLogic Server IIOP クライアントで T3 クライアントの場合と同じようにユーザ名とパスワードを指定できるようになる。
- GSSAPI (Generic Security Services Application Programming Interface) 初期コンテキスト トークンをサポートできるようになる。今回のリリースでは、ユーザ名 / パスワードと GSSUP (Generic Security Services Username Password) トークンのみがサポートされています。

注意： WebLogic Server における CSIv2 実装は、J2EE CTS (Compatibility Test Suite) 適合性テストに合格しています。

CSIv2 実装への外部インタフェースは、CORBA オブジェクトのユーザ名とパスワードを取得する JAAS LoginModule です。JAAS LoginModule は、WebLogic Java クライアント、あるいは別の J2EE アプリケーション サーバに対するクライ

アントの役目をする WebLogic Server インスタンスで使用することができます。CSIv2 サポート用の JAAS LoginModule は `UsernamePasswordLoginModule` という名前で、`weblogic.security.auth.login` パッケージに格納されています。

3　セキュリティ レルム

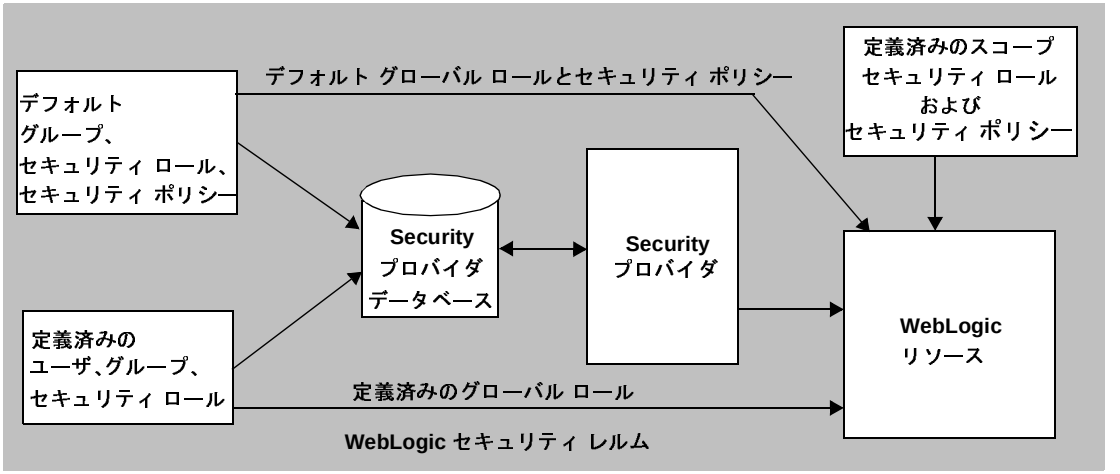
この章では、以下の内容について説明します。

- 3-1 ページの「セキュリティ レルムの概要」
- 3-2 ページの「ユーザ」
- 3-3 ページの「グループ」
- 3-4 ページの「セキュリティ ロール」
- 3-4 ページの「セキュリティ ポリシー」
- 3-5 ページの「セキュリティ プロバイダ」

セキュリティ レルムの概要

WebLogic Server のこのリリースでは、セキュリティ レルムが WebLogic リソースを保護するためのメカニズムを構成します。各セキュリティ レルムは、コンフィグレーション済みのセキュリティ プロバイダ、ユーザ、グループ、セキュリティ ロール、およびセキュリティ ポリシーで構成されます (図 3-1 を参照)。ユーザはセキュリティ レルムで定義されていないと、そのセキュリティ レルムに属する WebLogic リソースにアクセスできません。ユーザが特定の WebLogic リソースにアクセスしようとした場合、WebLogic Server は、関連するセキュリティ レルムでそのユーザに割り当てられているセキュリティ ロールと、その特定の WebLogic リソースのセキュリティ ポリシーをチェックして、ユーザを認証および認可します。

図 3-1　WebLogic Server のセキュリティ レルム



ユーザ

ユーザとは、myrealm などのセキュリティ レルムで認証されるエンティティのことです (図 3-1 を参照)。ユーザは、アプリケーション エンド ユーザなどの人物でも、クライアント アプリケーションやその他の WebLogic Server インスタンスなどのソフトウェア エンティティでもかまいません。認証の結果として、ユーザには ID つまりプリンシパルが割り当てられます。各ユーザには、セキュリティ レルムの中で固有の ID が与えられます。ユーザはセキュリティ ロールと関連付けられているグループに入れるか、またはセキュリティ ロールと直に関連付けることができます。

ユーザが WebLogic Server にアクセスしようとする場合、ユーザは、JAAS LoginModule を通じて証明データ (たとえば、パスワードまたはデジタル証明書) をセキュリティ レルム内にコンフィグレーションされている認証プロバイダに提示します。WebLogic Server がユーザ名と資格に基づいてユーザの ID を確認できた場合、WebLogic Server はユーザの代わりにコードを実行するスレッドをそのユーザに割り当てられたプリンシパルに関連付けます。ただし、スレッドがコードの実行を始める前に、WebLogic Server は WebLogic リソースのセキュリ

ティ ポリシーとユーザに割り当てられているプリンシパルをチェックして、ユーザが処理を続行するために必要なパーミッションを持っていることを確認します。

WebLogic 認証プロバイダを使用し、ユーザを定義する場合は、そのユーザのパスワードも定義します。WebLogic Server では、すべてのパスワードがハッシュ化されています。その後、クライアントのリクエストを受け取ると、WebLogic Server はクライアントが提示するパスワードをハッシュ化して、ハッシュ化済みパスワードと一致するかどうか比較します。

注意： すべてのユーザ名とグループはセキュリティ レルムの中で固有でなければなりません。

詳細については、『WebLogic リソースのセキュリティ』の「ユーザとグループ」を参照してください。

グループ

グループとは、論理的に整理したユーザの集合のことです (図 3-1 を参照)。一般に、グループ メンバーどうしは何らかの共通点を持っています。たとえば、企業の販売スタッフを、販売員と販売マネージャという 2 つのグループに分けることができます。このように分けることで、職務によって販売スタッフの WebLogic リソースへのアクセス レベルを異なるものにできます。

グループを管理する方が、多くのユーザを個々に管理するよりも効率的です。たとえば、管理者はユーザをグループに所属させ、グループにセキュリティ ロールを割り当ててから、そのセキュリティ ロールをセキュリティ ポリシーを介して WebLogic リソースに関連付けることで、50 ユーザのパーミッションを一度に指定できます。

すべてのユーザ名とグループはセキュリティ レルムの中で固有でなければなりません。

詳細については、『WebLogic リソースのセキュリティ』の「ユーザとグループ」を参照してください。

セキュリティ ロール

セキュリティ ロールは、特定の条件に基づいてユーザまたはグループに付与される特権です (図 3-1 を参照)。グループと同様、セキュリティ ロールを使用すると、複数のユーザによる WebLogic リソースへのアクセスを一度に制限できます。ただし、グループとは異なり、セキュリティ ロールには以下のような特長があります。

- ユーザ名、グループ メンバーシップ、または時刻などの条件に基づいて動的に計算されてユーザまたはグループに付与される。
- 常に WebLogic Server ドメイン全体を対象とするグループとは異なり、WebLogic Server ドメインの単一のアプリケーションに属する特定の WebLogic リソースを対象にできる。

セキュリティ ロールをユーザまたはグループに付与すると、そのセキュリティ ロールを付与されている限り、そのユーザまたはグループには定義されたアクセス特権が与えられます。複数のユーザまたはグループに単一のセキュリティ ロールを付与することができます

注意： WebLogic Server 6.x では、セキュリティ ロールは Web アプリケーションと エンタープライズ JavaBean (EJB) だけに適用されました。このリリースの WebLogic Server では、セキュリティ ロールは、定義されているすべての WebLogic リソースに対して適用されます。

セキュリティ ロールの詳細については、『WebLogic リソースのセキュリティ』の「セキュリティ ロール」を参照してください。

セキュリティ ポリシー

セキュリティ ポリシーとは、WebLogic リソースと、1 つまたは複数のユーザ、グループ、またはセキュリティ ロールとの間の関連付けです。セキュリティ ポリシーによって、WebLogic リソースに対する不正なアクセスを防ぐことができます。WebLogic リソースは、セキュリティ ポリシーを作成するまでは保護されません。

注意： セキュリティ ポリシーは、WebLogic Server 6.x で WebLogic リソースを保護するために使用していたアクセス制御リスト (ACL) に代わるものです。

セキュリティ ポリシーの詳細については、2-13 ページの「セキュリティ ポリシー」を参照してください。

セキュリティ プロバイダ

セキュリティ プロバイダは、WebLogic リソースを保護するためにアプリケーションにセキュリティ サービスを提供するモジュールです (図 3-1 を参照)。WebLogic Server 製品の一部として提供されるセキュリティ プロバイダを使用することも、サードパーティ セキュリティ ベンダからカスタム セキュリティ プロバイダを購入することも、独自にカスタム セキュリティ プロバイダを開発することもできます。カスタム セキュリティ プロバイダの開発方法については、『WebLogic Security サービスの開発』を参照してください。

この節では、以下の内容について説明します。

- 3-5 ページの「セキュリティ プロバイダ データベース」
- 3-8 ページの「セキュリティ プロバイダのタイプ」
- 3-18 ページの「セキュリティ プロバイダとセキュリティ レルム」

セキュリティ プロバイダ データベース

以下の節では、セキュリティ プロバイダ データベースとは何か、セキュリティ レルムがセキュリティ プロバイダ データベースの使用にどのように影響するのかについて説明します。

- 3-6 ページの「セキュリティ プロバイダ データベースとは」
- 3-7 ページの「セキュリティ レルムとセキュリティ プロバイダ データベース」
- 3-7 ページの「組み込み LDAP サーバ」

セキュリティ プロバイダ データベースとは

セキュリティ プロバイダ データベースには、一部のセキュリティ プロバイダがセキュリティ サービスを提供するために使用するユーザ、グループ、セキュリティ ポリシー、セキュリティ ロール、および資格が格納されます (図 3-1 を参照)。たとえば、認証プロバイダではユーザとグループについての情報、認可プロバイダではセキュリティ ポリシーについての情報、ロール マッピング プロバイダではセキュリティ ロールについての情報、資格マッピング プロバイダではエンタープライズ情報システム (EIS) にアクセスするためにリソース アダプタで使用する資格についての情報が必要になります。これらのセキュリティ プロバイダが正しく機能するためには、この情報をデータベースでアクセスできるようにする必要があります。

セキュリティ プロバイダ データベースとしては、WebLogic セキュリティ プロバイダで使用されるような組み込み LDAP サーバ、Web 上で使用できるサンプル カスタム セキュリティ プロバイダで使用されるような properties ファイル、またはすでに使用しているプロダクション レベルの顧客側で用意されたデータベースを使用できます。

注意： サンプル カスタム セキュリティ プロバイダは、BEA dev2dev Web サイトのメッセージ URL `http://dev2dev.bea.com/code/wls.jsp` で入手できます。

セキュリティ プロバイダ データベースは、セキュリティ プロバイダを初めて使用するとき (セキュリティ プロバイダを格納するセキュリティ レルムがデフォルト (アクティブ) セキュリティ レルムとして設定される前) に初期化する必要があります。この初期化は、以下の場合に行うことができます。

- WebLogic Server のインスタンスが起動するとき
- セキュリティ プロバイダの MBean の 1 つが呼び出されるとき

セキュリティ プロバイダ データベースは、少なくとも、WebLogic Server で提供されるデフォルトのグループ、セキュリティ ポリシー、セキュリティ ロールで初期化されます。詳細については、『WebLogic Security サービスの開発』の「セキュリティ プロバイダと WebLogic リソース」を参照してください。

セキュリティ レルムとセキュリティ プロバイダ データベース

同じセキュリティ レルムで同じタイプの複数のセキュリティ プロバイダがコンフィグレーションされている場合、それらのセキュリティ プロバイダでは同じセキュリティ プロバイダ データベースを使用できます。この動作は、すべての WebLogic セキュリティ プロバイダおよびサンプル セキュリティ プロバイダ (BEA dev2dev Web サイトの `http://dev2dev.bea.com/code/wls.jsp` で入手可能) でも同じように機能します。

たとえば、デフォルト セキュリティ レルム (`myrealm`) で 2 つの WebLogic 認証プロバイダをコンフィグレーションする場合、それらの WebLogic 認証プロバイダは両方とも組み込み LDAP サーバの同じ位置をセキュリティ プロバイダ データベースとして使用し、したがって同じユーザとグループを使用します。さらに、WebLogic 認証プロバイダの 1 つにユーザまたはグループを追加すると、他の WebLogic 認証プロバイダでもそのユーザまたはグループが表示されます。

注意： 同じタイプの 2 つの WebLogic セキュリティ プロバイダ (またはサンプル セキュリティ プロバイダ) が 2 つの異なるセキュリティ レルムでコンフィグレーションされている場合は、それぞれが独自のセキュリティ プロバイダ データベースを使用します。一度にアクティブにできるのは、1 つのセキュリティ レルムだけです。

独自に開発したカスタム セキュリティ プロバイダ (またはサードパーティ セキュリティ ベンダのカスタム セキュリティ プロバイダ) は、セキュリティ プロバイダの各インスタンスが独自のデータベースを使用するように、またはセキュリティ レルムのすべてのセキュリティ プロバイダ インスタンスが同じデータベースを共有するように設計できます。これは、既存のシステムとセキュリティ要件に基づいて決断する必要のある設計上の決定事項です。セキュリティ プロバイダに影響する設計上の決定事項の詳細については、『WebLogic Security サービスの開発』の「設計上の考慮事項」を参照してください。

組み込み LDAP サーバ

組み込み LDAP サーバは、WebLogic セキュリティ プロバイダ用のユーザ、グループ、セキュリティ ロール、およびセキュリティ ポリシーを格納するデータベースとして使用されます。組み込み LDAP サーバは、包括的な LDAP サーバです。以下のアクセス機能と格納機能がサポートされています。

- LDAP サーバ内のエントリへのアクセスとエントリの変更
- LDAP ブラウザを使用した、LDAP サーバに対するセキュリティ データのインポートとエクスポート
- WebLogic セキュリティ プロバイダによる読み込みおよび書き込みアクセス

  注意： WebLogic Server では、組み込み LDAP サーバへの属性の追加はサポートされていません。

表 3-1 に、各 WebLogic セキュリティ プロバイダによる組み込み LDAP サーバの使用方法を示します。

表 3-1 組み込み LDAP サーバの使用

WebLogic セキュリティ プロバイダ	組み込み LDAP サーバの使用
認証	ユーザおよびグループ情報を格納。
ID アサーション	ユーザおよびグループ情報を格納。
認可	セキュリティ ロールおよびセキュリティ ポリシーを格納。
裁決	なし。
ロール マッピング	指定された WebLogic リソースについて要求側に付与されるロール群を計算によって取得することで、動的ロール関連付けをサポート。
監査	なし。
資格マッピング	ユーザ名 / パスワードの資格マッピング情報を格納。

セキュリティ プロバイダのタイプ

以下の節では、WebLogic Server で使用できるセキュリティ プロバイダのタイプを説明します。

- 3-9 ページの「認証プロバイダ」

- 3-10 ページの「ID アサーション プロバイダ」
- 3-11 ページの「プリンシパル検証プロバイダ」
- 3-12 ページの「認可プロバイダ」
- 3-13 ページの「裁決プロバイダ」
- 3-13 ページの「ロール マッピング プロバイダ」
- 3-15 ページの「監査プロバイダ」
- 3-15 ページの「資格マッピング プロバイダ」
- 3-16 ページの「キーストア プロバイダ」
- 3-16 ページの「レルム アダプタ プロバイダ」
- 3-17 ページの「セキュリティ プロバイダのまとめ」

注意： 複数のタイプのプロバイダを結合した単一のセキュリティ プロバイダを開発することはできません。たとえば、認可とロール マッピングを行う 1 つのセキュリティ プロバイダを開発することはできません。

認証プロバイダ

認証プロバイダを使用すると、WebLogic Server ではユーザを検証することで信頼を確立できます。WebLogic Server のセキュリティ アーキテクチャは、ユーザ名とパスワードの認証、証明書に基づく認証 (WebLogic Server を直接用いる)、および HTTP 証明書に基づく認証 (外部の Web サーバを介して行う) を実行する認証プロバイダをサポートします。

注意： ID アサーション プロバイダは、境界に基づく認証と複数のセキュリティ トークン タイプ / プロトコルを処理する特殊なタイプの認証プロバイダです。詳細については、3-10 ページの「ID アサーション プロバイダ」を参照してください。

認証プロバイダには、ユーザまたはシステムの認証を実際に実行する LoginModule が組み込まれています。認証プロバイダは、プリンシパル (ユーザ / グループ) に対する署名と信頼性の確認によって追加のセキュリティを提供するプリンシパル検証プロバイダも使用します。プリンシパル検証プロバイダの詳細については、『WebLogic Security サービスの開発』の「プリンシパル検証プロバイダ」を参照してください。

セキュリティ レルムには少なくとも 1 つの認証プロバイダが必要であり、セキュリティ レルムで複数の認証プロバイダをコンフィグレーションすることもできます。複数の認証プロバイダをコンフィグレーションすると、それぞれが異なる認証を実行する複数の LoginModule を使用できます。管理者は、各認証プロバイダをコンフィグレーションして、ユーザがシステムにログインするときに複数の LoginModule がどのように呼び出されるのかを指定します。認証プロバイダは認証で使用されるプリンシパルにセキュリティを追加するので、プリンシパル検証プロバイダは認証プロバイダからアクセス可能である必要があります。

認証プロバイダおよび LoginModule の詳細については、『WebLogic Security サービスの開発』の認証プロバイダ」を参照してください。

ID アサーション プロバイダ

ID アサーションでは、リクエストの外部に存在することのある、クライアントから提供されるトークンを使用してクライアントの ID が確立されます。したがって、ID アサーション プロバイダの機能は、トークンを検証してユーザ名にマップすることになります。このマッピングがいったん完了すれば、認証プロバイダの LoginModule を使用してユーザ名がプリンシパルに変換されます。ID アサーション プロバイダを使用すると、WebLogic Server でユーザを検証して信頼を確立できます。

ID アサーション プロバイダは、ユーザまたはシステム プロセスがトークンを使用してそれぞれの ID を証明する特殊な形態の認証プロバイダです (つまり境界認証)。ID アサーション プロバイダ用の LoginModule を作成すれば、認証プロバイダの代わりに ID アサーション プロバイダを使用できます。また、認証プロバイダの LoginModule を使用する場合は、認証プロバイダに加えて ID アサーション プロバイダも使用できます。ID アサーション プロバイダは境界認証を有効にし、シングル サインオンをサポートします。

WebLogic Server のセキュリティ アーキテクチャは、境界に基づく認証 (Web サーバ、ファイアウォール、VPN) を実行する ID アサーション プロバイダをサポートします。ID アサーション プロバイダでは、Kerberos、SAML (Security Assertion Markup Language)、Microsoft Passport といった異なるタイプのトークンをサポートでき、複数のセキュリティ トークン プロトコル (SOAP、IIOP-CSIv2) を処理できます。認証プロバイダの LoginModule と一緒に使用すると、ID アサーション プロバイダはシングル サインオンをサポートします。たと

えば、ID アサーション プロバイダはデジタル証明書からトークンを生成することができ、そのトークンをシステム内で回すことができるため、ユーザは何度もサインオンを求められることはありません。

注意： X.501 および X.509 証明書に対して WebLogic ID アサーション プロバイダを使用するには、`weblogic.security.providers.authentication.UserNameMapper` インタフェースの実装を提供する必要があります。詳細については、『WebLogic Security サービスの開発』の「カスタム ID アサーション プロバイダを開発する必要があるか」を参照してください。

ID アサーション プロバイダはセキュリティ レルムで複数をコンフィグレーションできますが、必須ではありません。ID アサーション プロバイダでは複数のトークン タイプをサポートできますが、1 度にアクティブになるのは特定の ID アサーション プロバイダごとに 1 つのトークン タイプだけです。たとえば、特定の ID アサーション プロバイダは Kerberos と SAML の両方をサポートできますが、システムをコンフィグレーションする管理者は ID アサーション プロバイダでどちらのトークン タイプ (Kerberos または SAML) がアクティブになるのかを選択する必要があります。コンフィグレーションされている ID アサーション プロバイダが 1 つしかなく、Kerberos トークンを処理するように設定されているが、SAML トークン タイプもサポートする必要がある場合は、SAML を処理できる ID アサーション プロバイダをもう 1 つコンフィグレーションし、そのアクティブ トークン タイプとして SAML を設定する必要があります。

ID アサーション プロバイダの詳細については、『WebLogic Security サービスの開発』の「ID アサーション プロバイダ」を参照してください。

プリンシパル検証プロバイダ

プリンシパル検証プロバイダは、主に認証プロバイダの「ヘルパー」として機能する特殊なセキュリティ プロバイダです。一部の LoginModule は RMI クライアントの代わりにリモートで実行することができ、プログラムによるサーバ呼び出しから次のサーバ呼び出しまでの間、認証済みのサブジェクトがクライアント アプリケーション コードに保持できるので、認証プロバイダはプリンシパル検証プロバイダを利用してそのサブジェクト内に格納されているプリンシパルのセキュリティを保護する追加対策を講じます。

プリンシパル検証プロバイダは、プリンシパルに対する署名と信頼性の確認によってそのようなセキュリティを保護する追加対策を講じます。この**プリンシパル検証**によって、信頼度は向上し、悪意あるプリンシパルによる改ざんの可能性

が低下します。サブジェクトのプリンシパルの検証は、WebLogic Server で各呼び出しの RMI クライアント リクエストがデマーシャリングされる際に行われます。サブジェクトのプリンシパルの信頼性も、認可判定の際に検証されます。

セキュリティ レルムには少なくとも 1 つの認証プロバイダが必要なので、セキュリティ レルムにはプリンシパル検証プロバイダも 1 つ必要です。複数の認証プロバイダがある場合は、それぞれに対応するプリンシパル検証プロバイダが必要です。

注意： Administration Console を使用して、プリンシパル検証プロバイダを直接コンフィグレーションすることはできません。WebLogic Server では認証プロバイダのコンフィグレーション時に必要なプリンシパル検証プロバイダがコンフィグレーションされます。

プリンシパル検証プロバイダの詳細については、『WebLogic Security サービスの開発』の「プリンシパル検証プロバイダ」を参照してください。

認可プロバイダ

認可プロバイダは、ユーザまたはグループに付与されたセキュリティ ロールと、要求された WebLogic リソースのセキュリティ ポリシーに基づいて WebLogic リソースへのアクセスを制御します。WebLogic リソース、セキュリティ ロール、およびセキュリティ ポリシーの詳細については、『WebLogic リソースのセキュリティ』を参照してください。

認可プロバイダの中には、**アクセス決定**があります。アクセス決定は、WebLogic リソースに対して特定の操作を実行するパーミッションがサブジェクトにあるかどうかを実際に判断します。詳細については、『WebLogic Security サービスの開発』の「プリンシパル検証プロバイダ」を参照してください。

セキュリティ レルムには少なくとも 1 つの認可プロバイダが必要であり、セキュリティ レルムで複数の認可プロバイダをコンフィグレーションすることもできます。複数の認可プロバイダをコンフィグレーションすると、より高度なモジュール設計に従うことができます。たとえば、Web アプリケーションとエンタープライズ JavaBean (EJB) のマッピングを処理する 1 つの認可プロバイダと、その他のタイプの WebLogic リソースのマッピングを処理する別の認可プロバイダをコンフィグレーションできます。また、国内の従業員のパーミッションを処理する 1 つの認可プロバイダと、海外の従業員のパーミッションを処理する別の認可プロバイダをコンフィグレーションすることもできます

認可プロバイダおよびアクセス決定の詳細については、『WebLogic Security サービスの開発』の「認可プロバイダ」を参照してください。

裁決プロバイダ

認可プロバイダの一部として、アクセス決定は特定の WebLogic リソースにアクセスするためのパーミッションがサブジェクトにあるかどうかを判断します。したがって、複数の認可プロバイダがコンフィグレーションされている場合は、アクセス可能かどうかの問いに対してそれぞれが異なる回答を返す可能性があります。これらの回答は、`PERMIT`、`DENY`、`ABSTAIN` のいずれかです。複数の認可プロバイダのアクセス決定の回答が一致しない場合にどうするかを決定するのは、裁決プロバイダの主な役割です。裁決プロバイダは、各アクセス決定の回答を比較検討し最終結果を返すことで認可上の衝突を解消します。認証プロバイダが 1 つだけで、裁決プロバイダがない場合、その 1 つの認証プロバイダのアクセス決定から返される `ABSTAIN` は `DENY` のように処理されます。

注意： WebLogic 裁決プロバイダでは、ABSTAIN を PERMIT と DENY のどちらとして扱うかを Administration Console で指定できます。

裁決プロバイダは、コンフィグレーションされている認可プロバイダが複数の場合のみセキュリティ レルムでコンフィグレーションする必要があります。セキュリティ レルムでコンフィグレーションできる裁決プロバイダは 1 つだけです。

注意： デフォルトのセキュリティ レルムには認可プロバイダが 1 つしかないので、裁決プロバイダが提供されていても必要ありません。ただし、互換性レルムには認可プロバイダが 2 つあるので、そのレルムでは裁決プロバイダが必要です。

裁決プロバイダおよびアクセス決定の詳細については、『WebLogic Security サービスの開発』の「裁決プロバイダ」を参照してください。

ロール マッピング プロバイダ

ロール マッピング プロバイダでは、指定された WebLogic リソースについて要求側に付与されるセキュリティ ロール群を計算によって取得することで、動的ロール関連付けをサポートします。WebLogic Security フレームワークは、特定の WebLogic リソースにアクセスする必要があるときに特定のサブジェクトにどのセキュリティ ロールが適用されるのかを以下のようにして判断します。

- J2EE および WebLogic デプロイメント記述子ファイルからセキュリティ ロールを取得する
- ビジネス ロジックおよび現在の操作パラメータを使用してセキュリティ ロールを判断する

ロール マッピング プロバイダから認可プロバイダにこのセキュリティ ロール情報が提供されるので、認可プロバイダは、ロールベースのセキュリティを用いる WebLogic リソース (すなわち、Web アプリケーションやエンタープライズ JavaBean コンテナのリソース) に「アクセスできるか」という質問に答えることができます。

セキュリティ ロールは J2EE デプロイメント記述子に設定するか、WebLogic Server Administration Console を使って作成します。デプロイメント記述子に設定されたセキュリティ ロールはデプロイ時に適用されます (デプロイメント記述子を無視することにした場合を除く)。

セキュリティ レルムには少なくとも 1 つのロール マッピング プロバイダが必要であり、セキュリティ レルムで複数のロール マッピング プロバイダをコンフィグレーションすることもできます。複数のロール マッピング プロバイダをコンフィグレーションすると、従来のインフラストラクチャ要件の範囲内で作業することも (たとえばユーザおよびセキュリティ ロールの情報が含まれる各 LDAP サーバにつき 1 つのロール マッピング プロバイダをコンフィグレーションする)、または高度なモジュール設計に従うこともできます (たとえば Web アプリケーションとエンタープライズ JavaBean (EJB) のマッピングを処理する 1 つのロール マッピング プロバイダと、その他のタイプの WebLogic リソースのマッピングを処理する別のロール マッピング プロバイダをコンフィグレーションする)。

注意 : 複数のロール マッピング プロバイダがコンフィグレーションされている場合には、WebLogic Security フレームワークは、そのすべてのロール マッピング プロバイダから返されるセキュリティ ロール群の論理積を取ります。つまり、すべてのロール マッピング プロバイダからのセキュリティ ロール名が、重複を削除して 1 つのリストに結合されます。

ロール マッピング プロバイダの詳細については、『WebLogic Security サービスの開発』の「ロール マッピング プロバイダ」を参照してください。

監査プロバイダ

監査プロバイダは、リクエストの操作とそれらのリクエストの結果に関する情報を、否認防止を目的として収集、格納、および配布します。監査プロバイダは、特定の監査基準 (重大度など) に基づいて特定のイベントを監査するかどうかを決定します。また、監査プロバイダは LDAP バックエンド、データベース、シンプル ファイルなどの出力リポジトリに監査情報を書き込むことができます。セキュリティ担当者の呼び出しなどの特定のアクションも監査プロバイダの一部としてコンフィグレーションすることができます。

他のタイプのセキュリティ プロバイダ (認証や認可など) は、WebLogic Security フレームワークを介して呼び出しを行うことによって、セキュリティ操作の実行前と実行後に監査サービスを要求できます。詳細については、『WebLogic Security サービスの開発』の「カスタム セキュリティ プロバイダからのイベントの監査」を参照してください。

監査プロバイダはセキュリティ レルムで複数をコンフィグレーションできますが、必須ではありません。

監査プロバイダの詳細については、『WebLogic Security サービスの開発』の「監査プロバイダ」を参照してください。

資格マッピング プロバイダ

資格マップは、WebLogic Server で使用される資格から、レガシー システム (またはリモート システム) で使用される資格へのマッピングであり、そのシステムの特定のリソースへの接続方法を WebLogic Server に指示します。つまり、資格マップを使用することで、WebLogic Server が、認証済みのサブジェクトに代わってリモート システムにログインできるようになります。

資格マッピング プロバイダでは、ユーザ名とパスワードの組み合わせ、Kerberos チケット、公開鍵証明書といった複数種の資格を処理できます。資格マッピングはデプロイメント記述子で設定しても、WebLogic Server Administration Console を使って設定してもかまいません。これらの資格マッピングはデプロイ時に適用されます (それらの資格マッピングを無視することにした場合を除く)。

セキュリティ レルムには少なくとも 1 つの資格マッピング プロバイダが必要であり、セキュリティ レルムで複数の資格マッピング プロバイダをコンフィグレーションすることもできます。複数の資格マッピング プロバイダがコンフィグ

レーションされている場合は、WebLogic Security フレームワークが各資格マッピング プロバイダに働きかけて、コンテナの要求するタイプの資格があるかどうかを確認してから、すべての資格を収集してリストとして返します。

資格マッピング プロバイダの詳細については、『WebLogic Security サービスの開発』の「資格マッピング プロバイダ」を参照してください。

キーストア プロバイダ

キーストアを使用すると、パスワードで保護された、プライベート キー (およびプライベート キーに関連付けられた公開鍵証明書) と信頼性のある認証局のストアを作成および管理できます。

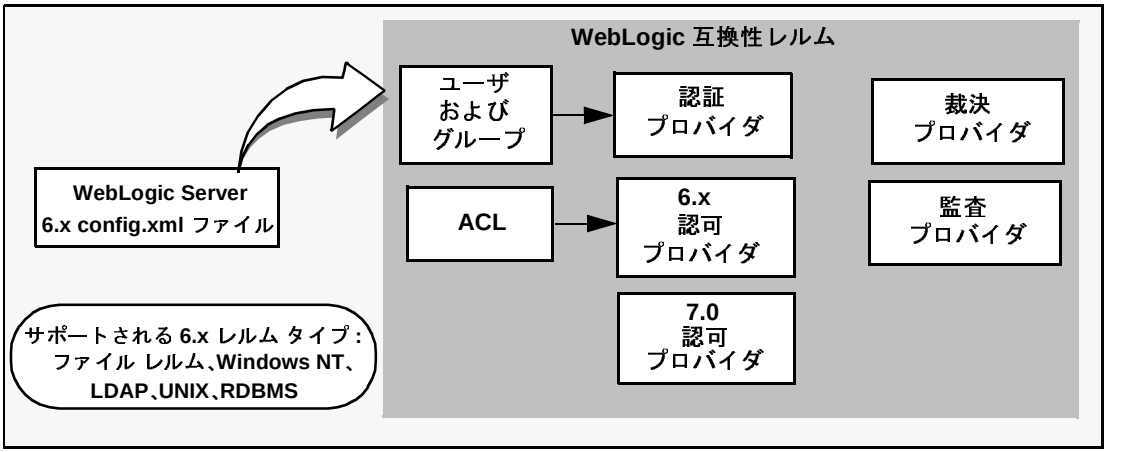
WebLogic Server 製品の一部として含まれる WebLogic キーストア プロバイダは、保護されたプライベート キーをキーストアから取得するために使用されます。

注意： WebLogic キーストア プロバイダのサポートは、WebLogic Server のこのリリースでは非推奨になりました。カスタム キーストア プロバイダの開発はサポートされていません。キーストア プロバイダは、下位互換性を保つためにのみサポートされています。代わりにキーストアを使用してください。キーストアの使用方法については、『WebLogic Security の管理』の「キーストアを作成してプライベート キーと信頼性のある認証局をそのキーストアにロードする」を参照してください。

レルム アダプタ プロバイダ

レルム アダプタ プロバイダは、既存のバージョン 6.x セキュリティ レルムを WebLogic Server のこのリリースのセキュリティ機能と併用できるようにすることで、バージョン 6.x WebLogic セキュリティ レルムとの下位互換性を提供するものです。レルム アダプタ プロバイダは、WebLogic Server 6.x で使用されるレルム API (`weblogic.security.acl`) を WebLogic Server のこのリリースで使用される API にマップします。図 3-2 に互換性レルムとサポートされるセキュリティ プロバイダのタイプを示します。

図 3-2 互換性レルム



セキュリティ プロバイダのまとめ

表 3-2 に、セキュリティ レルムで同じタイプの複数のセキュリティ プロバイダをコンフィグレーションできるかどうかを示します。

表 3-2 同じセキュリティ レルム内の同じタイプのプロバイダ

タイプ	複数のプロバイダをサポートするかどうか
認証プロバイダ	サポートする
ID アサーション プロバイダ	サポートする
プリンシパル検証プロバイダ	サポートする
認可プロバイダ	サポートする
裁決プロバイダ	サポートしない
ロール マッピング プロバイダ	サポートする
監査プロバイダ	サポートする
資格マッピング プロバイダ	サポートする

表 3-2 同じセキュリティ レルム内の同じタイプのプロバイダ (続き)

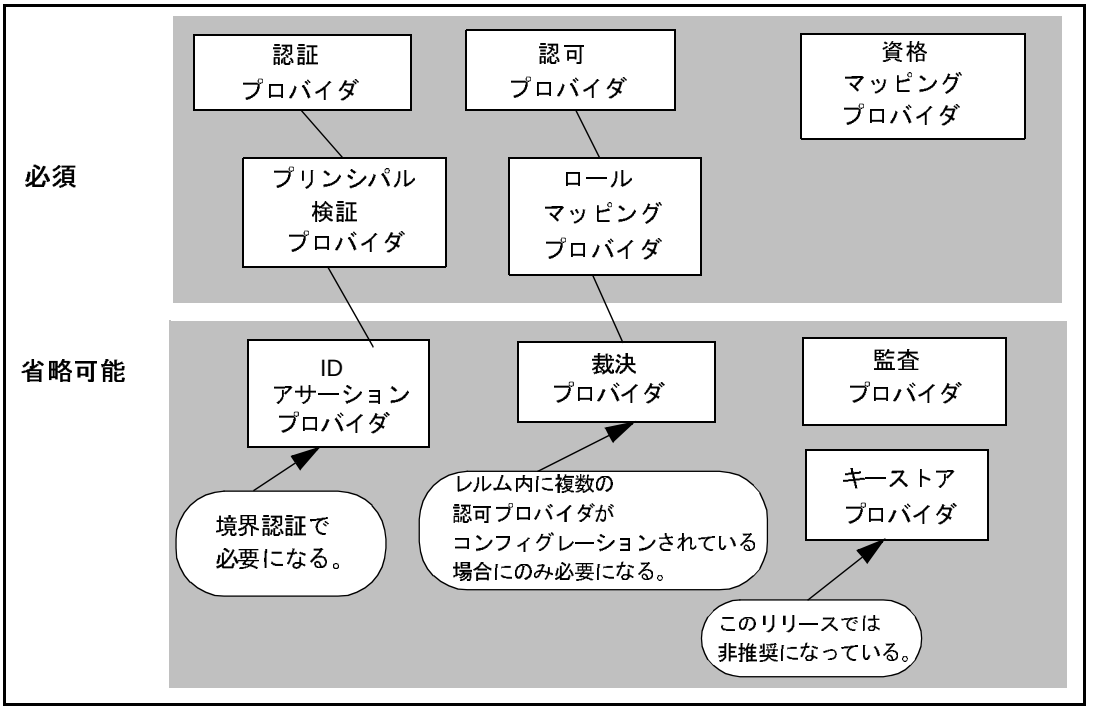
タイプ	複数のプロバイダをサポートするかどうか
キーストア プロバイダ	サポートする
レルム アダプタ プロバイダ	裁決プロバイダを除くすべてのタイプのレルム アダプタ プロバイダで、複数のプロバイダがサポートされています。サポートされるタイプについては、図 3-2 を参照してください。

セキュリティ プロバイダとセキュリティ レルム

すべてのセキュリティ プロバイダは、セキュリティ レルムのコンテキスト内に存在します。以前の WebLogic Server リリース 6.x を使用しているのでなければ、**デフォルト レルム** (つまり `myrealm` と呼ばれるアクティブなセキュリティ レルム) としてあらかじめ定義されている WebLogic Server セキュリティ レルムには図 3-3 に示す WebLogic セキュリティ プロバイダが用意されています。

注意 : リリース 6.x からリリース 7.0 へアップグレードしている場合には、最初にあらかじめ用意されているものは**互換性レルム** (最初はデフォルト レルムとして定義されている) で、それを使用すれば既存のコンフィグレーションをそのまま扱うことができます。ただし、6.x モデルは非推奨扱いになっているので、セキュリティ レルムを 7.0 モデルにアップグレードする必要があります。アップグレードの詳細については、『BEA WebLogic Server 7.0 へのアップグレード』の「WebLogic Server 6.x からバージョン 7.0 へのアップグレード」の章の「セキュリティのアップグレード」を参照してください。

図 3-3 セキュリティ レルムの WebLogic セキュリティ プロバイダ



注意： レルム アダプタ プロバイダは図 3-3 には表示されていません。図 3-2 に示したように、レルム アダプタ プロバイダは互換性レルム内でしか使用できないためです。

セキュリティ プロバイダは WebLogic Server セキュリティ レルムに「プラグイン」される個別のモジュールまたはコンポーネントなので、手間をかけずに追加、置換、または削除できます。WebLogic セキュリティ プロバイダ、独自に開発したカスタム セキュリティ プロバイダ、サードパーティ セキュリティ ベンダのセキュリティ プロバイダ、または以上 3 つの組み合わせを使用して完全に機能するセキュリティ レルムを作成できます。ただし、図 3-3 に示すように、一部のタイプのセキュリティ プロバイダは 7.0 セキュリティ レルムが正しく機能する上で必須の要素です。表 3-3 は、7.0 セキュリティ レルムが完全に機能するためにどのセキュリティ プロバイダをコンフィグレーションしなければならないかを説明しています。

表 3-3 セキュリティ レルムのセキュリティ プロバイダ

タイプ	必須 / 省略可能
認証プロバイダ	必須
ID アサーション プロバイダ	境界認証を使用する場合は必須
プリンシパル検証プロバイダ	必須
認可プロバイダ	必須
裁決プロバイダ	複数の認可プロバイダがコンフィグレーションされている場合は必須
ロール マッピング プロバイダ	必須
監査プロバイダ	省略可能
資格マッピング プロバイダ	必須
キーストア プロバイダ	省略可能 **注意：** WebLogic キーストア プロバイダは、WebLogic Server のこのリリースでは非推奨になっている。カスタム キーストア プロバイダの開発はサポートされていない。キーストア プロバイダは、下位互換性を保つためにのみサポートされている。代わりにキーストアを使用すること。キーストアの使用方法については、『WebLogic Security の管理』の「キーストアを作成してプライベート キーと信頼性のある認証局をそのキーストアにロードする」を参照してください。

セキュリティ レルムの詳細については、『WebLogic Security の管理』の以下の節を参照してください。

- 「セキュリティのコンフィグレーション手順」

- 「新しいセキュリティ レルムをデフォルト (アクティブ) セキュリティ レルムとして設定する」
- 「セキュリティ レルムの削除」

4 WebLogic Security サービスのアーキテクチャ

この章では、以下の内容について説明します。

- 4-1 ページの「アーキテクチャの概要」
- 4-21 ページの「アーキテクチャによってユーザにもたらされるメリット」

アーキテクチャの概要

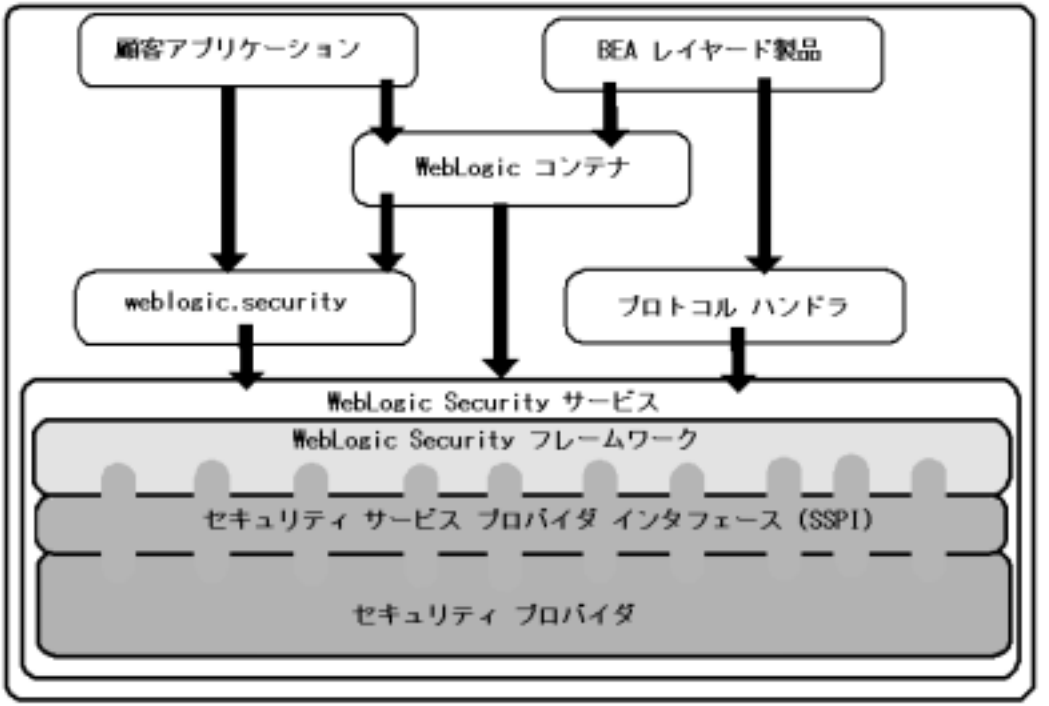
この節では、WebLogic Security サービスのアーキテクチャについて説明します。アーキテクチャは、以下の節で説明する 3 つの主要なコンポーネントで構成されています。

- 4-1 ページの「WebLogic Security フレームワーク」
- 4-11 ページの「セキュリティ サービス プロバイダ インタフェース (SSPI)」
- 4-12 ページの「WebLogic セキュリティ プロバイダ」

WebLogic Security フレームワーク

図 4-1 に、WebLogic Security フレームワークの概観を示します。フレームワークは、`weblogic.security.service` パッケージのインタフェース、クラス、および例外で構成されます。

図 4-1 WebLogic Security サービスのアーキテクチャ



WebLogic Security フレームワークの主な役割は、簡素化されたアプリケーション プログラミング インタフェース (API) を提供することです。API は、セキュリティ サービスを定義するセキュリティ開発者およびアプリケーション開発者によって使用されます。そのコンテキスト内で WebLogic Security フレームワークは、WebLogic コンテナ (Web および EJB)、リソース コンテナ、セキュリティ プロバイダの間の仲介役としても機能します。

以下の節では、WebLogic Security フレームワークを介した WebLogic コンテナ、リソース コンテナ、各セキュリティ プロバイダの間の対話について説明します。

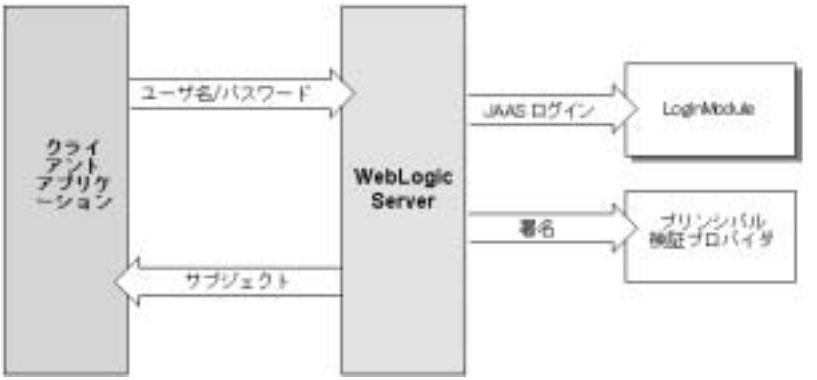
- 4-3 ページの「認証プロセス」
- 4-4 ページの「ID アサーション プロセス」
- 4-5 ページの「プリンシパル検証プロセス」

- 4-6 ページの「認可プロセス」
- 4-7 ページの「裁決プロセス」
- 4-7 ページの「ロール マッピング プロセス」
- 4-9 ページの「監査プロセス」
- 4-10 ページの「資格マッピング プロセス」

認証プロセス

図 4-2 に、ファットクライアント ログインの認証プロセスを示します。JAAS はサーバ上で動作してログインを実行します。シンクライアント ログイン (Web ブラウザ クライアント) の場合でも、JAAS はサーバ上で動作します。

図 4-2 認証プロセス



注意： JASS プロセスを直接扱うのは、カスタム認証プロバイダの開発者だけです。クライアント アプリケーションは、JNDI 初期コンテキストまたは JAAS のいずれかを使用してユーザ名とパスワードを受け渡します。

ユーザがユーザ名とパスワードの組み合わせを使用してシステムにログインしようとすると、WebLogic Server はそのユーザのユーザ名とパスワードを検証することによって信頼を確立し、JAAS の要件に従って、プリンシパルが格納されたサブジェクトを返します。図 4-2 で示されているとおり、このプロセスには LoginModule とプリンシパル検証プロバイダの使用が必要になります。プリンシパル検証プロバイダの詳細については、4-15 ページの「WebLogic プリンシパル検証プロバイダ」を参照してください。

呼び出し側の ID の確認に成功すると、認証コンテキストが確立され、ID が確認されたユーザまたはシステムに関しては、そのコンテキストを通じて他のエンティティに対する認証を行うことができます。認証コンテキストはまた、アプリケーション コンポーネントに委託することもでき、それによって、そのコンポーネントは別のアプリケーション コンポーネントを呼び出しつつ、元の呼び出し側として動作することができるようになります。

ID アサーション プロセス

ID アサーション プロバイダは、境界認証プロセスの一部として使用されます。境界認証が使用される場合 (図 4-3 を参照)、WebLogic Server ドメインの外部のトークンが、セキュリティ レルム内の、そのタイプのトークンの検証を担当し、「アクティブ」としてコンフィグレーションされている ID アサーション プロバイダに渡されます。そのトークンの有効性が問題なく検証されれば、ID アサーション プロバイダはそのトークンを WebLogic Server ユーザ名にマップし、そのユーザ名を WebLogic Server に送り返します。その後 WebLogic Server では、認証プロセスが続行されます。具体的には、ユーザ名が JAAS `CallbackHandler` を通じて送信され、コンフィグレーションされている各認証プロバイダの LoginModule に渡されるので、LoginModule はサブジェクト内に適切なプリンシパルを格納できるようになります。

注意 : X.501 および X.509 証明書に対して WebLogic ID アサーション プロバイダを使用するには、`weblogic.security.providers.authentication.UserNameMapper` インタフェースの実装を提供する必要があります。詳細については、『WebLogic Security サービスの開発』の「カスタム ID アサーション プロバイダを開発する必要があるか」を参照してください。

図 4-3　境界認証

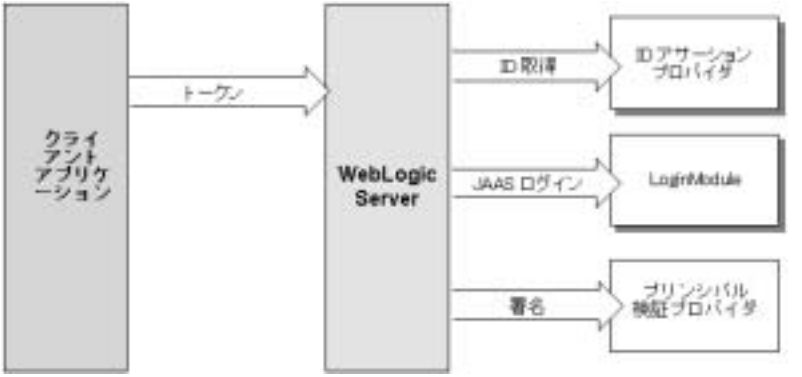
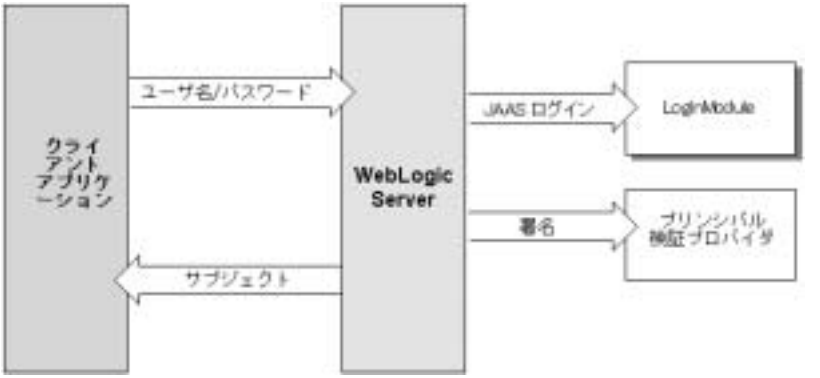


図 4-3 でも示されているように、境界認証では認証プロセスと同じコンポーネントが必要ですが、ID アサーション プロバイダも追加されます。

プリンシパル検証プロセス

図 4-4 に示すとおり、ユーザはユーザ名とパスワードの組み合わせを使ってシステムにログインしようとします。WebLogic Server は、コンフィグレーション済みの認証プロバイダの LoginModule を呼び出すことによって信頼を確立します。LoginModule は、ユーザのユーザ名とパスワードを検証し、JAAS の要件に従って、プリンシパルが格納されたサブジェクトを返します。

図 4-4　プリンシパル検証プロセス

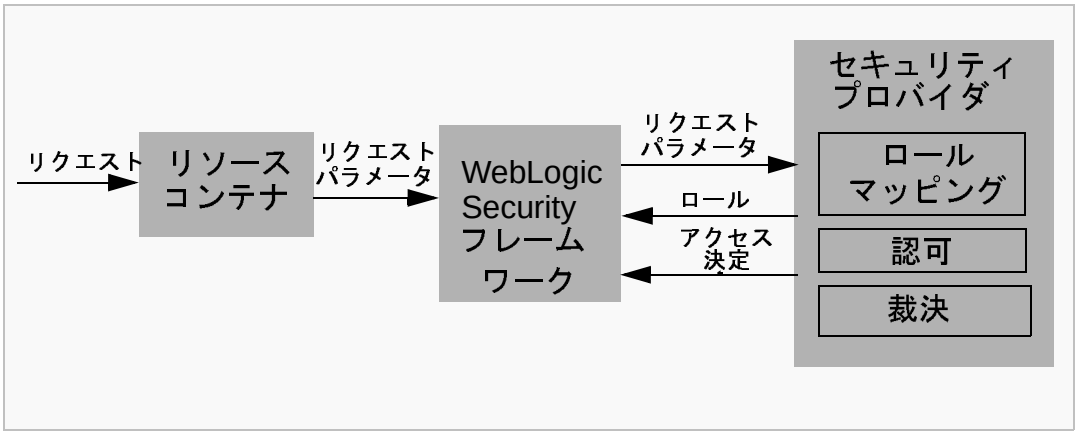


WebLogic Server は、指定されたプリンシパル検証プロバイダにサブジェクトを渡します。プリンシパル検証プロバイダは、そのプリンシパルに署名して、それらを WebLogic Server を通じてクライアント アプリケーションに返します。他のセキュリティ操作のためにサブジェクト内のプリンシパルが必要になった場合、同じプリンシパル検証プロバイダが、そのプリンシパルが署名時から変更されていないかどうかを検証します。

認可プロセス

図 4-5 に、認可プロセスにおける認可プロバイダ (および関連する裁決プロバイダとロール マッピング プロバイダ) と WebLogic Security フレームワークとの対話を示します。

図 4-5 認可プロセス



ユーザまたはシステム プロセスが WebLogic リソースを要求し、特定の操作を実行しようとすると、認可プロセスが開始されます。要求された WebLogic リソースのタイプを処理するリソース コンテナがリクエストを受け取ります (たとえば EJB コンテナは EJB リソースに対するリクエストを受け取ります)。リソース コンテナは、WebLogic Security フレームワークを呼び出して、リクエストのサブジェクト、要求されている WebLogic リソースなどの情報を含むリクエスト パラメータを渡します。WebLogic Security フレームワークは、コンフィグレーション済みのロール マッピング プロバイダを呼び出して、ロール マッピング プロバイダが使用できる形式でリクエスト パラメータを渡します。ロール マッピング プロバイダは、リクエスト パラメータを使用して、要求側のサブジェクト

が資格を有する一連のロールを計算し、該当するロールを WebLogic Security フレームワークに渡します。認可プロバイダは、そのサブジェクトが WebLogic リソースに対して要求されたアクションを実行できる資格があるかどうかを判別します (つまり、アクセス決定を行います)。複数の認可プロバイダがコンフィグレーションされている場合には、WebLogic Security フレームワークは、認可プロバイダから返されたアクセス決定に衝突があった場合、その調停を裁決プロバイダに委託します。裁決プロバイダでは、認可判定の最終結果を決定します。

裁決プロセス

複数の認可プロバイダがコンフィグレーションされている場合 (図 4-5 を参照) には、複数のアクセス決定を調停し、判定するための裁決プロバイダが必要になります。裁決プロバイダは TRUE か FALSE の判定を認可プロバイダに返し、認可プロバイダはその判定を WebLogic Security フレームワークを通してリソース コンテナに転送します。

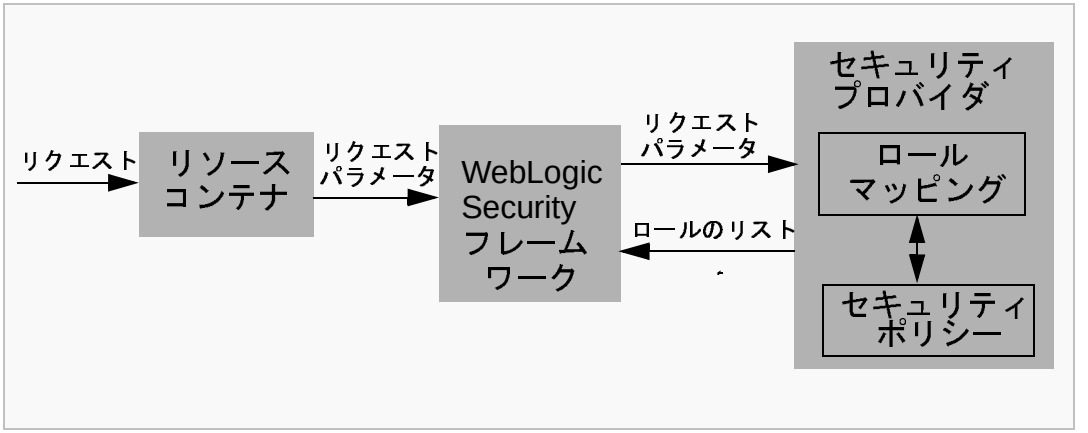
- 判定が TRUE の場合、リソース コンテナはリクエストを保護対象リソースにディスパッチします。
- 判定が FALSE の場合、リソース コンテナはセキュリティ例外を送出します。この例外は、要求側には要求したアクセスを保護された WebLogic リソースに対して行う権限がなかったことを示します。

ロール マッピング プロセス

WebLogic Security フレームワークは、認可判定の一環として、セキュリティ レルム用にコンフィグレーションされている各ロール マッピング プロバイダを呼び出します。関連情報については、4-6 ページの「認可プロセス」を参照してください。

ロール マッピング プロバイダと WebLogic Security フレームワークとの対話を図 4-6 に示します。

図 4-6 ロール マッピング プロセス



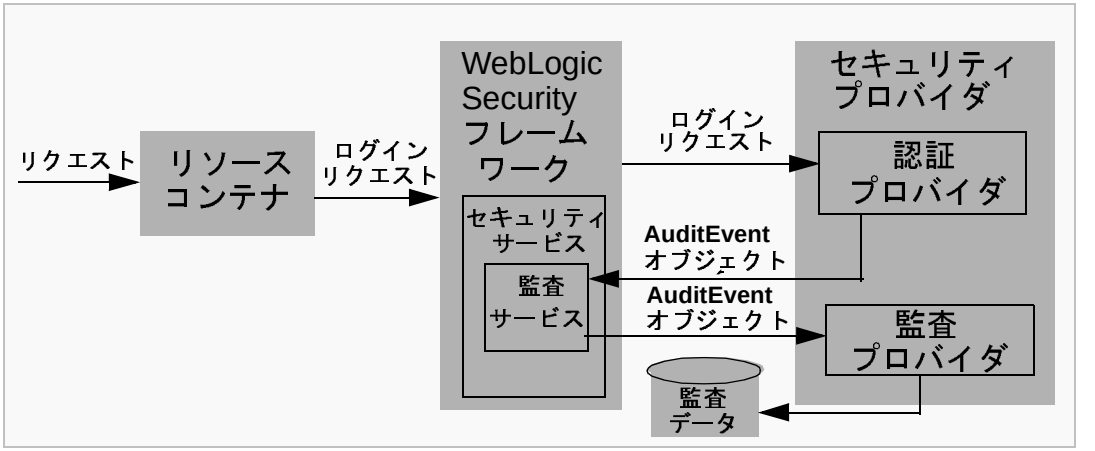
ユーザまたはシステム プロセスが WebLogic リソースを要求し、特定の操作を実行しようとすると、ロール マッピング プロセスが開始されます。要求された WebLogic リソースのタイプを処理するリソース コンテナがリクエストを受け取ります (たとえば EJB コンテナは EJB リソースに対するリクエストを受け取ります)。リソース コンテナは、WebLogic Security フレームワークを呼び出して、リクエストのサブジェクト、要求されている WebLogic リソースなどの情報を含むリクエスト パラメータを渡します。WebLogic Security フレームワークは、適用するロールのリストを取得するために、コンフィグレーション済みの各ロール マッピング プロバイダを呼び出します。要求側に特定のロールの資格があるとセキュリティ ポリシーに指定されている場合には、そのロールがサブジェクトに適用されるロールのリストに追加されます。このプロセスは、WebLogic リソースまたはリソース コンテナに適用されるセキュリティ ポリシーがすべて評価されるまで続行されます。ロールのリストは WebLogic Security フレームワークに返され、アクセス決定などの操作の一環として使用できるようになります。

ロール マッピング プロバイダによる動的ロール関連付けの結果、一連のロールが一度にサブジェクト内のプリンシパルに割り当てられます。その後、これらのロールを用いて、保護対象 WebLogic リソース、リソース コンテナ、およびアプリケーション コードについての認可判定が行われます。たとえば、エンタープライズ JavaBean (EJB) であれば、アクセスを許可するかどうかを決めるビジネス ポリシーを知らなくても、J2EE (Java 2 Enterprise Edition) の `isCallerInRole()` メソッドを使用してデータベース内のレコードからフィールドを取得することができます。

監査プロセス

図 4-7 に、監査プロバイダが WebLogic Security フレームワークと他のタイプのセキュリティ プロバイダ (ここでは認証プロバイダ) と対話する仕組みを示します。

図 4-7 監査プロセス

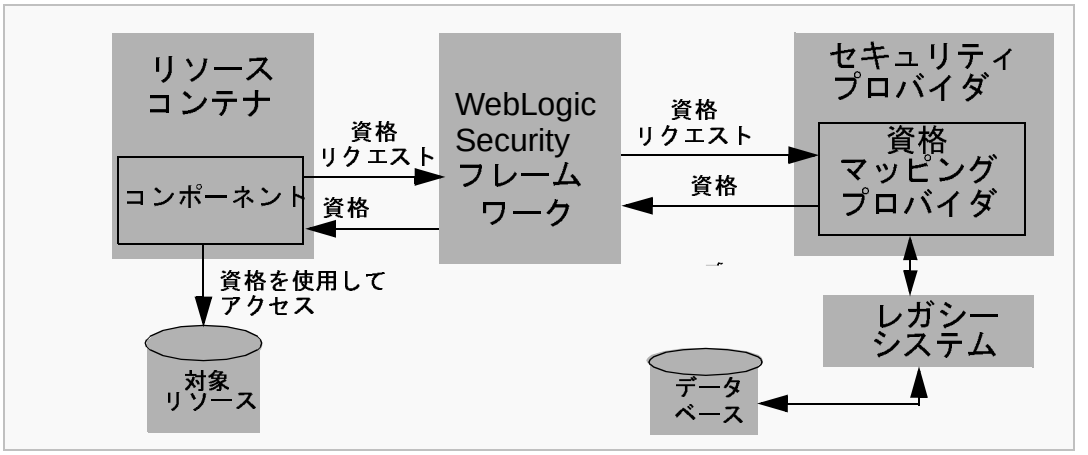


リソース コンテナが、ログイン リクエストの一部としてユーザの認証情報 (ユーザ名とパスワードの組み合わせなど) を WebLogic Security フレームワークに渡すと、監査プロセスが開始されます。WebLogic Security フレームワークは、ログイン リクエストに関連付けられている情報をコンフィグレーション済みの認証プロバイダに渡します。認証プロバイダが認証サービスを提供するほかに監査イベントをポストする場合、認証プロバイダは `AuditEvent` オブジェクトをインスタンス化します。`AuditEvent` オブジェクトには、監査対象のイベント タイプや監査の重大度レベルなどの情報が含まれます。その後、認証プロバイダは WebLogic Security フレームワーク内の監査サービスを呼び出して、`AuditEvent` オブジェクトを渡します。監査サービスは、`AuditEvent` オブジェクトをコンフィグレーション済みの監査プロバイダの実行時クラスに渡し、監査イベントの記録を有効にします。監査プロバイダの実行時クラスは、`AuditEvent` オブジェクトから取得した情報を使用して監査レコードの内容を管理します `AuditEvent` オブジェクト内に認証プロバイダによって指定された監査条件が満たされると、該当する監査プロバイダの実行時クラスは監査レコードを書き出します。監査プロバイダの実装によっては、監査記録をファイルやデータベースなどの永続ストレージ メディアに書き込むことができます。

資格マッピング プロセス

資格マッピング プロセスにおける資格マッピング プロバイダと WebLogic Security フレームワークとの対話を図 4-8 に示します。

図 4-8 資格マッピング プロセス



JavaServer Page (JSP)、サーブレット、エンタープライズ JavaBean (EJB)、リソース アダプタなどのアプリケーション コンポーネントが、エンタープライズ情報システム (EIS) (たとえば、Oracle、SQL Server などのリレーショナル データベース) にアクセスするために、適切なリソース コンテナを通じて WebLogic Security フレームワークを呼び出すと、資格マッピング プロセスが開始されます。呼び出しの中で、アプリケーション コンポーネントは、サブジェクト (「誰が」要求しているのか)、WebLogic リソース (「何を」要求しているのか)、および WebLogic リソースにアクセスするために必要な資格のタイプについての情報を渡します。WebLogic Security フレームワークは、資格に対するアプリケーション コンポーネントのリクエストを、アプリケーション コンポーネントが必要としている資格のタイプを処理するコンフィグレーション済み資格マッピング プロバイダに送信します。資格マッピング プロバイダは、データベースを照会してアプリケーション コンポーネントが要求しているものに一致する資格群を取得し、WebLogic Security フレームワークに資格を返します。WebLogic Security

フレームワークは、リソース コンテナを通じて要求側のアプリケーション コンポーネントに資格を返します。アプリケーション コンポーネントは、資格を使用して外部システムにアクセスします。

セキュリティ サービス プロバイダ インタフェース (SSPI)

WebLogic Server のこのリリースでのセキュリティは、一連のセキュリティ サービス プロバイダ インタフェース (SSPI) に基づいています。開発者およびサードパーティ ベンダは、SSPI を使用して WebLogic Server 環境向けのセキュリティ プロバイダを開発できます。認証、ID アサーション、認可、監査、裁決、ロール マッピング、および資格マッピング用の SSPI が利用可能です。

注意： キーストア プロバイダ用の SSPI はこのリリースの WebLogic Server 7.0 で非推奨となりました。代わりにキーストアを使用すること。キーストアの使用方法については、『WebLogic Security の管理』の「キーストアを作成してプライベート キーと信頼性のある認証局をそのキーストアにロードする」を参照してください。

SSPI を使用すると、カスタム セキュリティ プロバイダで WebLogic Server リソースを保護することができます。SSPI を使用してカスタム セキュリティ プロバイダを開発することも、サードパーティ ベンダからカスタム セキュリティ プロバイダを購入することもできます。

注意： カスタム セキュリティ プロバイダの開発を支援するために、サンプル カスタム セキュリティ プロバイダが用意されています。BEA dev2dev Web サイト (`http://dev2dev.bea.com/codelibrary/code/security_prov.jsp`) で入手できます。カスタム セキュリティ プロバイダの開発の詳細については、『WebLogic Security サービスの開発』を参照してください。

WebLogic セキュリティ プロバイダ

この節では、WebLogic Server 製品に付属している WebLogic セキュリティ プロバイダについて説明します。**セキュリティ プロバイダ**は、WebLogic Server のセキュリティ レルムに「プラグイン」してアプリケーションにセキュリティ サービスを提供するためのモジュールです。セキュリティ プロバイダは、アプリケーションの代わりに WebLogic Security フレームワークに働きかけを行います。

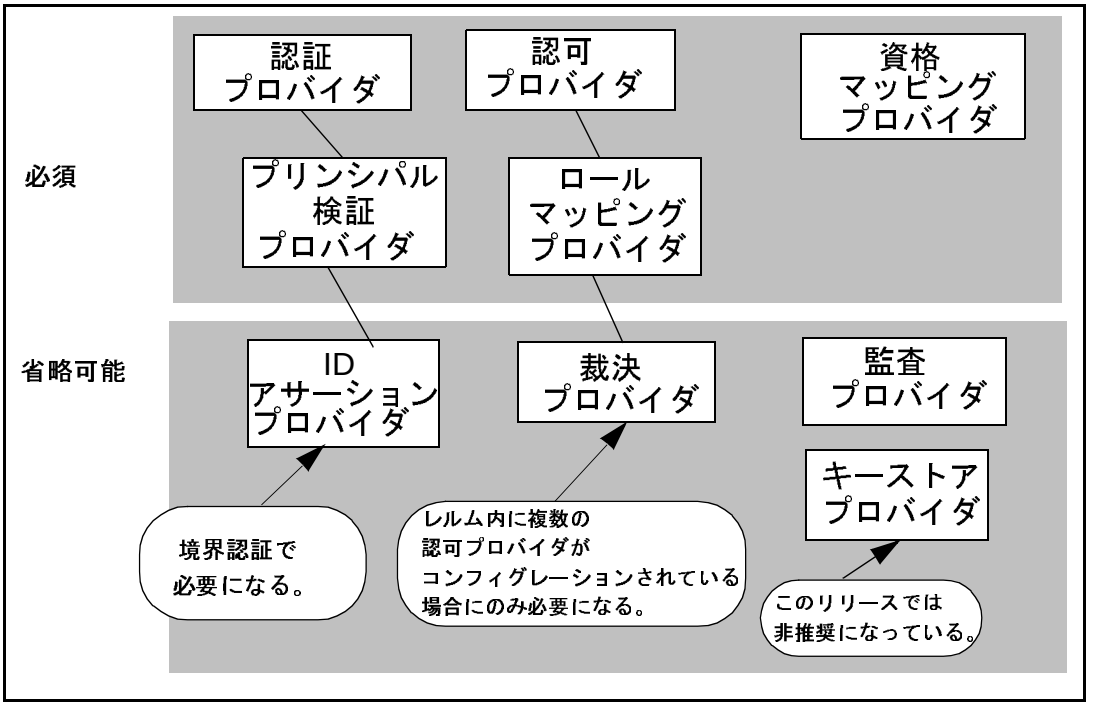
WebLogic Server 製品に付属している WebLogic セキュリティ プロバイダがセキュリティ要件を完全に満たしていない場合には、カスタム セキュリティ プロバイダを使用してそれらを補足するか、あるいはカスタム セキュリティ プロバイダに置き換えることができます。カスタム セキュリティ プロバイダは以下のようにして開発します。

- `weblogic.security.spi` パッケージから適切なセキュリティ サービス プロバイダ インタフェース (SSPI) を実装してセキュリティ プロバイダの実行時クラスを作成する。
- MBean 定義ファイル (MDF) を作成し、WebLogic MBeanMaker ユーティリティを使用して MBean タイプを生成する。MBean タイプは、セキュリティ プロバイダのコンフィグレーションと管理に使用します。

詳細については、『WebLogic Security サービスの開発』を参照してください。

図 4-9 に、WebLogic セキュリティ レルムにおいて必須および省略可能なセキュリティ プロバイダを示します。

図 4-9 WebLogic セキュリティ プロバイダ



以下の節では、WebLogic セキュリティ プロバイダについて説明します。

- 4-14 ページの「WebLogic 認証プロバイダ」
- 4-14 ページの「WebLogic ID アサーション プロバイダ」
- 4-15 ページの「WebLogic プリンシパル検証プロバイダ」
- 4-16 ページの「WebLogic 認可プロバイダ」
- 4-16 ページの「WebLogic 裁決プロバイダ」
- 4-17 ページの「WebLogic ロール マッピング プロバイダ」
- 4-18 ページの「WebLogic 監査プロバイダ」
- 4-18 ページの「WebLogic 資格マッピング プロバイダ」

- 4-18 ページの「WebLogic キーストア プロバイダ」
- 4-19 ページの「WebLogic レルム アダプタ プロバイダ」

WebLogic 認証プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic 認証プロバイダが含まれています。

注意： WebLogic 認証プロバイダを WebLogic 認可プロバイダと組み合わせれば、WebLogic Server リリース 6.x で利用できたファイル レルムの機能の代わりになります。

WebLogic 認証プロバイダは、ユーザ名とパスワードの委託認証をサポートし、組み込み LDAP サーバを利用してユーザとグループの情報を格納します。このプロバイダを使用すると、ユーザとグループ メンバーシップを編集、表示、および管理できます。

また、WebLogic Server は、Open LDAP、Netscape iPlanet、Microsoft Active Directory、および Novell NDS といった外部 LDAP ストアにアクセスする一連の LDAP 認証プロバイダも提供します。

注意： デフォルトでは、LDAP 認証プロバイダは WebLogic デフォルト セキュリティ レルムではコンフィグレーションされていません。

WebLogic ID アサーション プロバイダ

WebLogic ID アサーション プロバイダでは、X.509 証明書を使用した証明書認証および CSIv2 (CORBA Common Secure Interoperability version 2) ID アサーションがサポートされています。

WebLogic ID アサーション プロバイダはトークン タイプを検証し、X.509 デジタル証明書と X.501 識別名を WebLogic ユーザ名にマップします。また、CSIv2 ID アサーションに使用する信頼性のあるクライアント プリンシパルのリストも指定します。ワイルドカード文字 (*) を使用すると、すべてのプリンシパルに信頼性があると指定できます。クライアントが信頼性のあるクライアント プリンシパルのリストにない場合は、CSIv2 ID アサーションが失敗し、呼び出しが拒否されます。

WebLogic ID アサーション プロバイダでは、以下のトークン タイプがサポートされています。

- AU_TYPE — WebLogic AuthenticatedUser がトークンとして使用される場合に使用します。
- X509_TYPE — X.509 クライアント証明書がトークンとして使用される場合に使用します。
- CSI_PRINCIPAL_TYPE — CSIv2 プリンシパル名 ID がトークンとして使用される場合に使用します。
- CSI_ANONYMOUS_TYPE — CSIv2 匿名 ID がトークンとして使用される場合に使用します。
- CSI_X509_CERTCHAIN_TYPE — CSIv2 X.509 証明書チェーン ID がトークンとして使用される場合に使用します。
- CSI_DISTINGUISHED_NAME_TYPE — CSIv2 識別名 ID がトークンとして使用される場合に使用します。

WebLogic プリンシパル検証プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic プリンシパル検証プロバイダが含まれています。このプロバイダは、WebLogic Server プリンシパルの署名と検証を行います。言い換えると、WebLogic Server ユーザまたは WebLogic Server グループを表すプリンシパルに対して署名と検証を行います。

注意： weblogic.security パッケージの WLSPrincipals クラスを使用すると、プリンシパル (ユーザまたはグループ) が WebLogic Server に対して特別な意味を持っているかどうか (それが定義済みの WebLogic Server ユーザまたは WebLogic Server グループであるかどうか) が分かります。さらに、WebLogic Server ユーザまたはグループを表すプリンシパルは、weblogic.security.spi パッケージの WLSUser インタフェースと WLSGroup インタフェースを実装する必要があります。

WebLogic プリンシパル検証プロバイダには、WLSUserImpl および WLSGroupImpl という名前の WLSUser インタフェースおよび WLSGroup インタフェースの実装が含まれています。それらは、weblogic.security.principal パッケージに定義されています。また、PrincipalValidatorImpl という名前

の PrincipalValidator SSPI の実装も含まれています。PrincipalValidator SSPI の詳細については、『WebLogic Security サービスの開発』の「PrincipalValidator SSPI を実装する」を参照してください。

ID アサーション プロバイダが特定のタイプのトークンをサポートするのと同じように、プリンシパル検証プロバイダは特定のタイプのプリンシパルに対する署名と信頼性の確認を行います。そのため、WebLogic プリンシパル検証プロバイダを使用して、WebLogic Server ユーザまたは WebLogic Server グループを表すプリンシパルに対して署名と検証を行うことができます。

WebLogic 認可プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic 認可プロバイダが含まれています。このプロバイダは、WebLogic Server のこのバージョンの認可機能をデフォルトで適用するものです。WebLogic 認可プロバイダは、特定のユーザが保護対象の WebLogic リソースへのアクセスを許可されているかどうかを判定するため、ポリシーベースの認可エンジンを使用してアクセス決定を返します。また、WebLogic 認可プロバイダは、システム内のセキュリティ ポリシーのデプロイメントとアンデプロイメントもサポートしています。

WebLogic 裁決プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic 裁決プロバイダが含まれています。このプロバイダは通常、複数の認可プロバイダのアクセス決定から異なる結果が返された場合に裁決を行い、WebLogic リソースへのアクセスを許可するかどうかを最終的に判定します。ただし、デフォルトのセキュリティ レルムには認可プロバイダが 1 つしかなく、生成されるアクセス決定は 1 つのみなので、WebLogic 裁決プロバイダは使用されません。

注意： 互換性レルムには認可プロバイダが 2 つあるので、そのレルムでは WebLogic 裁決プロバイダが使用されます。

WebLogic 裁決プロバイダには、動作を制御する [完全一致の許可が必要] という属性があります。[完全一致の許可が必要] 属性は、デフォルトでは TRUE に設定されており、WebLogic 裁決プロバイダはその設定に従って次のように動作します。

- すべての認可プロバイダのアクセス決定が PERMIT を返した場合、最終的な判定として TRUE を返します (つまり WebLogic リソースへのアクセスは許可されます)。
- 一部の認可プロバイダのアクセス決定が PERMIT を返し、その他が ABSTAIN を返した場合、最終的な判定として FALSE を返します (つまり WebLogic リソースへのアクセスは拒否されます)。
- いずれかの認可プロバイダのアクセス決定が ABSTAIN または DENY を返した場合、最終的な判定として FALSE を返します (つまり WebLogic リソースへのアクセスは拒否されます)。

[完全一致の許可が必要] 属性を FALSE に変更した場合、WebLogic 裁決プロバイダは次のように動作します。

- すべての認可プロバイダのアクセス決定が PERMIT を返した場合、最終的な判定として TRUE を返します (つまり WebLogic リソースへのアクセスは許可されます)。
- 一部の認可プロバイダのアクセス決定が PERMIT を返し、その他が ABSTAIN を返した場合、最終的な判定として TRUE を返します (つまり WebLogic リソースへのアクセスは許可されます)。
- いずれかの認可プロバイダのアクセス決定が DENY を返した場合、最終的な判定として FALSE を返します (つまり WebLogic リソースへのアクセスは拒否されます)。

注意： [完全一致の許可が必要] 属性は、WebLogic 裁決プロバイダをコンフィグレーションするときに設定します。裁決プロバイダのコンフィグレーションの詳細については、『WebLogic Security の管理』の「WebLogic 裁決プロバイダのコンフィグレーション」を参照してください。

WebLogic ロール マッピング プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic ロール マッピング プロバイダが含まれています。このプロバイダは、デフォルト ユーザと WebLogic リソースのそれぞれについて、保護されている特定の WebLogic リソースに関する特定のユーザ (サブジェクト) の動的ロールを決定します。また、WebLogic ロール マッピング プロバイダは、システム内のロー

ルのデプロイメントとアンデプロイメントをサポートしています。WebLogic ロール マッピング プロバイダは、WebLogic 認可プロバイダと同じセキュリティ ポリシー エンジンを使用します。

WebLogic 監査プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic 監査プロバイダが含まれています。このプロバイダは、WebLogic Security フレームワークで内部的に決定された、複数のセキュリティ リクエストの情報を記録します。また、WebLogic 監査プロバイダはこれらのセキュリティ リクエストに関連付けられているイベント データとリクエストの結果も記録します。

WebLogic 資格マッピング プロバイダ

WebLogic Server のデフォルト (アクティブ) セキュリティ レルムには WebLogic 資格マッピング プロバイダが含まれています。WebLogic 資格マッピング プロバイダを使用して、WebLogic Server ユーザを、エンタープライズ情報システム (EIS) (たとえば、Oracle、SQL Server などのリレーショナル データベース) にアクセスするためにリソース アダプタで使用すべき適切な資格に関連付ける (つまりマップする) ことができます。プロバイダは、ユーザの認証資格 (ユーザ名とパスワード) をレガシー アプリケーションで必要になる認証資格にマップするので、レガシー アプリケーションは必要な資格情報を取得できます。たとえば、EIS はメインフレーム トランザクション処理システムでも、データベース システムでも、Java プログラミング言語で記述されていないレガシー アプリケーションでもかまいません。

WebLogic Server のユーザおよびグループと他のシステムのユーザ名 / パスワードとの間のマッピングであれば、WebLogic 資格マッピング プロバイダで十分です。

WebLogic キーストア プロバイダ

WebLogic キーストア プロバイダでは、Sun Microsystems から Java Software Development Kit (SDK) の形で提供されるキーストアの参照実装を使用します。そこでは標準の「JKS」キーストア タイプが利用されますが、これはキーストア

をファイル (マシンごとに 1 つ) として実装したものです。各プライベート キーはそれぞれのパスワードで保護されます。WebLogic キーストア プロバイダに関連付けられているキーストア ファイルは 2 つあります。

- 一方のキーストア ファイルには信頼性のある認証局 (CA) の証明書が記載されています。WebLogic Server には信頼性のある認証局キーストア ファイルが付属しており、WebLogic Server はデフォルトではそれを使用して、SSL で使用される信頼性のある CA の証明書を見つけ、クライアント証明書が正しいかどうかを確認します。

- もう一方のキーストア ファイルには、サーバのプライベート キーが記載されています。WebLogic Server はこのファイルからプライベート キーを取得して SSL を初期化します。Sun Microsystems SDK の `keytool` ユーティリティか WebLogic Server の `ImportPrivateKey` ユーティリティを利用すれば、このファイルにプライベート キーを追加することができます。なお、WebLogic Server では、このキーストア ファイルからプライベート キーだけを取得し証明書は取得しないことに注意してください。

注意： キーストア プロバイダは、WebLogic Server のこのリリースでは非推奨になりました。カスタム キーストア プロバイダの開発はサポートされません。代わりに Java キーストア (JKS) を使用してください。Java キーストアの使用方法については、『WebLogic Security の管理』の「キーストアを作成してプライベート キーと信頼性のある認証局をそのキーストアにロードする」を参照してください。

WebLogic レルム アダプタ プロバイダ

WebLogic レルム アダプタ プロバイダは、既存のバージョン 6.x セキュリティ レルムを WebLogic Server のこのリリースのセキュリティ機能と併用できるようにすることで、バージョン 6.x WebLogic セキュリティ レルムとの下位互換性を提供するものです。WebLogic レルム アダプタ プロバイダは、WebLogic Server 6.x で使用されるレルム API (`weblogic.security.acl`) を WebLogic Server のこのリリースで使用される API にマップします。以下の WebLogic レルム アダプタ プロバイダが提供されています。

- 認証 (ID アサーション プロバイダを含む)

  レルム アダプタ認証プロバイダを利用すれば、バージョン 6.x のセキュリティ レルムとそれらのデータ ストアを WebLogic Server の WebLogic セ

キュリティ プロバイダと併用できるようになります。さらに、`weblogic.security.acl.CertAuthenticator` クラスの実装を WebLogic Server で使用することも可能になります。レルム アダプタ認証プロバイダには、X.509 トークンに基づいた ID アサーションを提供するコンポーネントが含まれています。

- 認可

  互換性セキュリティでは、WebLogic 認可プロバイダとレルム アダプタ認可プロバイダの 2 種類の認可プロバイダが使用されます。WebLogic 認可プロバイダが新しいセキュリティ ポリシーに使用されるのに対して、レルム アダプタ認可プロバイダは WebLogic Server 6.1 のアクセス制御リスト (ACL) へのマッピングに使用されます。

- 監査

  レルム アダプタ監査プロバイダを利用すれば、互換性セキュリティを使用する WebLogic Server デプロイメントで `weblogic.security.audit` インタフェースの実装を使用できるようになります。

- 裁決

  互換性セキュリティを使用するセキュリティ レルム向けに WebLogic 認可プロバイダとレルム アダプタ認可プロバイダの両方を併用できるようにする裁決プロバイダ。

これらのセキュリティ プロバイダは WebLogic Server Administration Console を使ってコンフィグレーションされますが、既存の 6.x セキュリティ レルムでは WebLogic Server 6.1 に存在するのと同じ MBean とユーザ インタフェースを引き続き使用することになります。

注意： WebLogic レルム アダプタ プロバイダは非推奨になったため、WebLogic Server 7.0 セキュリティ モデルにアップグレードする間だけ使用してください。

アーキテクチャによってユーザにもたらされるメリット

WebLogic Security サービス アーキテクチャは、以下のカテゴリのユーザに対して特にメリットをもたらします。

- 4-21 ページの「アプリケーション開発者」
- 4-22 ページの「サーバ管理者 / アプリケーション管理者」
- 4-22 ページの「サードパーティ セキュリティ サービス プロバイダ」

アプリケーション開発者

Web アプリケーションおよび EJB に対するほとんどのセキュリティはシステム管理者によって実装されるので、アプリケーション開発者は、コードで対処すべき特別な考慮事項がない限り、アプリケーションの保護に関する詳細に注意を払う必要はありません。アプリケーション内でのカスタム セキュリティのプログラミングについても、WebLogic Server アプリケーション開発者は、WebLogic Server で使用されるサブジェクトやプリンシパルに関する情報を取得する (ユーザの情報を特定する) のに BEA が提供するアプリケーション プログラミング インタフェース (API) を利用できます。これらの API は、`weblogic.security` パッケージに入っています。

WebLogic Server の Java 標準の包括的なサポートにより、WebLogic Server 用のアプリケーションの開発者は、J2EE で定義されたセキュリティ固有のメソッドだけでなく、JAAS、JSSE などの Java プラットフォーム セキュリティ パッケージの API も使用できます。

サーバ管理者 / アプリケーション管理者

管理者は、WebLogic Server セキュリティ プロバイダをそのまま使用して、包括的なセキュリティ ソリューションを実装できます。また、管理者は Administration Console を使用して、セキュリティ ロールの定義や WebLogic リソースに対するセキュリティ ポリシーの割り当てをし、企業独自のビジネス ルールを実現する認可方式を作成できます。

サードパーティ セキュリティ サービス プロバイダ

業界をリードするセキュリティ サービス プロバイダのほとんどが、BEA WebLogic Server 7.0 をサポートする計画を発表しています。これらのサードパーティ プロバイダは、セキュリティ サービス プロバイダ インタフェース (SSPI) を使用して、自社の製品を WebLogic Server 環境に統合する予定です。WebLogic セキュリティ プロバイダの基底の統合メカニズムとして SSPI を使用し、WebLogic Server 環境向けのカスタマイズされたセキュリティ プロバイダを開発できます。認証、ID アサーション、認可、監査、裁決、ロール マッピング、および資格マッピング用の SSPI が利用可能です。

注意： キーストア プロバイダ用の SSPI はこのリリースの WebLogic Server 7.0 で非推奨となりました。代わりにキーストアを使用すること。キーストアの使用方法については、『WebLogic Security の管理』の「キーストアを作成してプライベート キーと信頼性のある認証局をそのキーストアにロードする」を参照してください。

このアーキテクチャを使用することで、セキュリティ開発者は、実装が容易な、密接に統合されたソリューションを提供できます。その結果、開発要件が緩和され、企業のセキュリティ管理ソリューションの実装時における投資に対する回収率が向上します。

BEA 準拠のセキュリティ ベンダについては、http://www.bea.com/framework.jsp?CNT=isv.htm&FP=/content/partners/strategic/ の BEA Security Center を参照してください。

5 用語

WebLogic Server のセキュリティに関するドキュメントで扱う主な用語は以下のとおりです。

アクセス制御リスト (ACL)

WebLogic Server 6.x で、コンピュータ リソースに対するアクセスの制御に使用されるデータ構造。アクセス制御リスト (ACL) の各エントリには、個々のユーザやユーザのグループを表す特定のプリンシパルに関連付けられた一連のパーミッションが含まれます。エントリは、肯定的なものでも否定的なものでもかまいません。パーミッションを付与するエントリが肯定的となり、パーミッションを拒否するエントリが否定的となります。WebLogic Server 7.0 以降では、ACL は非推奨になり、代わりにセキュリティ ポリシーが使用されます。引き続き ACL を使用して WebLogic リソースを保護する場合は、互換性セキュリティを使用します。「互換性セキュリティ」、「グループ」、「プリンシパル」、「セキュリティ ポリシー」、「ユーザ」、「WebLogic リソース」も参照してください。

アクセス決定

指定された操作を WebLogic リソースに対して実行するパーミッションがサブジェクトにあるかどうかを判別するコード。アクセス決定の結果は、許可、拒否、または決定の放棄のいずれかです。アクセス決定は、認可プロバイダのコンポーネントです。「認可プロバイダ」、「サブジェクト」、「WebLogic リソース」も参照してください。

ACL

アクセス制御リスト (ACL) を参照。

裁決プロバイダと裁決

複数のアクセス決定から返される結果を調停し、アクセス決定間の衝突を解決して、`PERMIT` または `DENY` の最終判定を行う WebLogic セキュリティ プロバイダ。裁決は裁決プロバイダのコンポーネントです。「アクセス決定」、「セキュリティ プロバイダ」も参照してください。

非対称鍵暗号化方式
: データの暗号化と解読にプライベート キーと公開鍵という異なる鍵を使用する暗号化アルゴリズムを採用した、鍵に基づく暗号化方式。公開鍵で暗号化されたデータは、プライベート キーでしか解読できません。反対に、プライベート キーで暗号化されたデータは公開鍵でしか解読できません。この非対称性によって、公開鍵暗号化方式が非常に実用的になりました。非対称鍵暗号化方式は、公開鍵暗号化方式とも呼ばれます。「プライベート キー」、「公開鍵」、「対称鍵暗号方式」も参照してください。

監査
: リクエストの操作とそれらのリクエストの結果に関する情報を、否認防止を目的として収集、格納、および配布するプロセス。監査は、コンピュータのアクティビティの電子的な記録を提供します。「監査プロバイダ」も参照してください。

監査プロバイダ
: 監査サービスを提供するセキュリティ プロバイダ。「監査」、「セキュリティ プロバイダ」も参照してください。

認証
: ユーザまたはシステム プロセスの ID を証明または確認するプロセス。認証にはまた、必要に応じて、身元情報を記憶したり、転送したり、またさまざまなシステム コンポーネントの利用に供することも必要になります。通常、認証はユーザ名とパスワードの組み合わせを使用して行われますが、トークンを使用することもできます。「認証プロバイダ」、「ID アサーション」、「LoginModule」、「境界認証」、「トークン」、「ユーザ」も参照してください。

認証プロバイダ
: ユーザを検証することで WebLogic Server が信頼を確立できるようにするセキュリティ プロバイダ。WebLogic Security サービス アーキテクチャは、ユーザ名とパスワードの認証、証明書に基づく認証 (WebLogic Server を直接用いる)、および HTTP 証明書に基づく認証 (外部の Web サーバを介して行う) を実行する認証プロバイダをサポートします。「認証」、「デジタル証明書」、「セキュリティ プロバイダ」、「ユーザ」も参照してください。

認可
: ユーザのセキュリティ ロールと要求された WebLogic リソースのセキュリティ ポリシーに基づいて WebLogic リソースに対するユーザのアクセ

スを許可または拒否するプロセス。「認可プロバイダ」、「セキュリティ ポリシー」、「ユーザ」、「WebLogic リソース」も参照してください。

認可プロバイダ

ユーザのセキュリティ ロールと要求された WebLogic リソースのセキュリティ ポリシーに基づいて WebLogic リソースへのアクセスを制御するセキュリティ プロバイダ。「セキュリティ プロバイダ」、「ユーザ」、「WebLogic リソース」も参照してください。

キャッシング レルム

互換性セキュリティを使用する場合にのみ、WebLogic Server 7.0 以降に適用される WebLogic Server 6.x の機能。キャッシング レルムはメモリ内の一時的な場所であり、プライマリ レルムから頻繁に呼び出される ACL、ユーザ、グループなどが格納されます。WebLogic Server 6.x では、ユーザ、グループ、および ACL のオブジェクトは `filerealm.properties` ファイルに格納されます。ファイルからの読み込みは非常に時間がかかる場合があります。デフォルトでは、キャッシング レルムはプライマリ レルムの上の通信層であり、ルックアップに使用されます。キャッシング レルムでのルックアップが失敗した場合、ルックアップはプライマリ レルムで実行されます。「アクセス制御リスト (ACL)」、「互換性セキュリティ」、「グループ」、「ユーザ」も参照してください。

証明書

デジタル証明書を参照。

証明書認証

デジタル証明書を使用して、信頼できる身元証明をサーバからクライアントに提供するための手法。証明書認証は、ユーザが持つプライベート キーと、ユーザが知っているパスワード (プライベート キーを保護するためのもの) に基づいて認証を行うことから、一般的にパスワード認証よりも好まれます。「認証」、「証明書」、「認証局」も参照してください。

認証局

公開鍵証明書を発行する信頼性のあるエンティティ。認証局はユーザの現実世界の身元を証明します。公証人のようなものです。「証明書チェーン」、「デジタル証明書」、「エンティティ」、「プライベート キー」、「公開鍵」、「信頼性のある (ルート) 認証局」も参照してください。

証明書チェーン

信頼性のある認証局のプライベート キー、それに対応する公開鍵、およびデジタル証明書のチェーンを含む配列。各認証局は前のデジタル証明書の発行元です。サーバの証明書、権限、権限 2、権限 3 でチェーンが構成されている場合、サーバの証明書は権限によって署名され、権限の証明書は権限 2 によって署名され、権限 2 の証明書は権限 3 によって署名されています。これらの権限のいずれかの認証局がクライアントによって認識されると、クライアントはサーバを認証します。「信頼性のある (ルート) 認証局」も参照してください。

互換性レルム

互換性セキュリティを使用している場合のデフォルト (アクティブ) セキュリティ レルムとなるセキュリティ レルム。互換性レルムでは、既存の WebLogic Server 6.x の認証プロバイダおよび認可プロバイダが使用されます。そのため、WebLogic Server 7.0 以降でそれらのプロバイダを使用できます。互換性セキュリティで使用可能なセキュリティ レルムは、互換性レルムだけです。「互換性セキュリティ」、「デフォルト レルム」、「セキュリティ プロバイダ」、「セキュリティ レルム」、「WebLogic セキュリティ プロバイダ」も参照してください。

互換性セキュリティ

WebLogic Server 6.x のセキュリティ コンフィグレーションをこのリリースの WebLogic Server で実行するための機能。WebLogic Server 7.0 以降では互換性セキュリティを使用して、6.x のセキュリティ レルムのコンフィグレーション、ユーザ、グループ、および ACL の定義、ユーザ アカウント保護の管理、カスタム監査プロバイダのインストールを行います。互換性セキュリティで使用可能なセキュリティ レルムは、互換性レルムだけです。互換性レルムのレルム アダプタ プロバイダを使用すると、6.x セキュリティ レルムの認証および認可サービスとの下位互換性を保持できます。「アクセス制御リスト (ACL)」、「監査プロバイダ」、「互換性レルム」、「グループ」、「レルム アダプタ認証プロバイダ」、「レルム アダプタ認可プロバイダ」、「セキュリティ レルム」、「ユーザ」も参照してください。

接続フィルタ

サーバがネットワーク クライアントからの受信時接続を許可すべきかどうかを決定するために WebLogic Server で使用されるプログラム可能なフィルタ。ユーザの特性に基づいて WebLogic リソースを保護するセキュリティ ポリシーに加えて、ネットワーク接続に基づいてフィルタ処

理を行うことでセキュリティの層を追加できます。「セキュリティ ポリシー」、「ユーザ」、「WebLogic リソース」も参照してください。

コネクタ
: リソース アダプタを参照。

コンテキスト ハンドラ
: リソース コンテナから追加のコンテキストとコンテナ固有の情報を取得し、アクセスやロール マッピングを決定するセキュリティ プロバイダにこれらの情報を提供する高性能の WebLogic クラス。
`ContextHandler` インタフェースを使用すると、内部 WebLogic リソース コンテナで WebLogic Security フレームワーク呼び出しに追加情報を渡すことができます。その結果、セキュリティ プロバイダは、特定のメソッドの引数で提供される以上のコンテキスト情報を取得できるようになります。ContextHandler は本質的には名前と値のリストなので、セキュリティ プロバイダは検索する名前を認識しておく必要があります。つまり、ContextHandler の使用には、WebLogic リソース コンテナとセキュリティ プロバイダの間の緊密な連携が不可欠です。「セキュリティ プロバイダ」、「WebLogic コンテナ」、「WebLogic Security フレームワーク」も参照してください。

資格
: 新しいサービスにアクセスするサブジェクトを認証するための情報が格納される、サブジェクトのセキュリティ関連の属性。資格のタイプには、ユーザ名とパスワードの組み合わせ、Kerberos チケット、および公開鍵証明書などがあります。「資格マッピング」、「資格マッピング プロバイダ」、「デジタル証明書」、「Kerberos チケット」、「公開鍵」、「サブジェクト」も参照してください。

資格マッピング
: 対象リソースにアクセスするユーザを認証するための適切な資格群を、レガシー システムのデータベースを使用して取得するプロセス。
WebLogic Server では、資格マッピングを使用して、WebLogic Server で使用される資格を、レガシー システム (またはリモート システム) で使用される資格にマップします。その後、WebLogic Server は資格マップを使用して、認証済みサブジェクトに代わってリモート システムにログインします。「資格」、「資格マッピング プロバイダ」、「リソース」も参照してください。

資格マッピング プロバイダ

資格マッピング サービスを提供して、新しいタイプの資格を WebLogic Server 環境に追加するために使用されるセキュリティ プロバイダ。「資格」、「資格マッピング」、「セキュリティ プロバイダ」も参照してください。

ドメイン間シングル サインオン

ユーザが一度認証を行えば、複数のアプリケーションが異なる DNS ドメインにある場合でもこれらのアプリケーションにアクセスできるようになる WebLogic Server のセキュリティ機能。この機能を使用すると、シングル サインオン ドメインに参加する関連会社やパートナのネットワークを構築できます。「シングル サインオン」も参照してください。

注意： ドメイン間シングル サインオンは Java クライアント (つまり、Java 仮想マシン (JVM) を実行しているクライアント) に対してのみサポートされており、Web ブラウザ クライアントに対してはサポートされていません。

CSIv2 プロトコル

IIOP (GIOP 1.2) と CORBA CSIv2 (Common Secure Interoperability version 2) の CORBA 仕様に基づいたプロトコル。EJB 2.0 およびその他の J2EE 1.3 コンテナの安全相互運用要件は、CSIv2 仕様の準拠レベル 0 に対応しています。CORBA Security Attribute Service (SAS) は、iCSIv2 で使用されるプロトコルです。詳細については、`http://www.omg.org/technology/documents/formal/omg_security.htm` を参照してください。

カスタム セキュリティ プロバイダ

サードパーティ ベンダまたはセキュリティ アプリケーション開発者によって記述された、WebLogic Security サービスに統合可能なセキュリティ プロバイダ。カスタム セキュリティ プロバイダは、SSPI (Security Service Provider Interfaces) の実装で、WebLogic Server 製品には付属していません。「セキュリティ プロバイダ」、「セキュリティ レルム」、「セキュリティ サービス プロバイダ インタフェース (SSPI)」、「WebLogic セキュリティ プロバイダ」、「WebLogic Security サービス」も参照してください。

カスタム セキュリティ レルム

WebLogic Server 7.0 以降では、互換性セキュリティでのみサポートされるセキュリティ レルム。WebLogic Server 6.x では、独自のセキュリ

ティ レルムを作成して WebLogic Server 環境に統合することで認証をカスタマイズします。「互換性セキュリティ」も参照してください。

データベース デリゲータ
: セキュリティ プロバイダとそのセキュリティ プロバイダ データベースの間で行われる初期化呼び出しを仲介する中間クラス。「セキュリティ プロバイダ データベース」も参照してください。

宣言によるセキュリティ
: アプリケーション デプロイメント記述子を使用して定義 (宣言) されるセキュリティ。Web アプリケーションの場合、`web.xml` ファイルおよび `weblogic.xml` ファイルでデプロイメント記述子を定義します。EJB の場合、`ejb-jar.xml` ファイルおよび `weblogic-ejb-jar.xml` ファイルでデプロイメント記述子を定義します。

デフォルト レルム
: アクティブなセキュリティ レルム。WebLogic Server 7.0 以降では、WebLogic Server ドメインに複数のセキュリティ レルムをコンフィグレーションできますが、デフォルト (アクティブ) セキュリティ レルムに指定できるのはそのうちの 1 つだけです。「カスタム セキュリティ レルム」、「セキュリティ レルム」、「WebLogic Server ドメイン」も参照してください。

デジタル証明書
: 特定の公開鍵と名前などの属性を関連付けるデジタル ステートメント。ステートメントは認証局によってデジタル署名されます。認証局が本物のステートメントだけに署名すると信頼することにより、公開鍵が、証明書にその名前が記された人物のものであると信頼することができます。「認証局」、「デジタル署名」、「公開鍵」、「信頼性のある (ルート) 認証局」も参照してください。

デジタル署名
: 2 つのエンティティの身元を検証することで、それらのエンティティ間でやり取りされるデータのセキュリティを保護する場合に使用されるビットからなる文字列。特に、レコードの送信元エンティティからのデータが途中で変更されていないことを検証する場合に使用されます。デジタル署名は、エンティティの署名されたデータとプライベートキーから計算されます。デジタル署名による信頼は、信頼性を確認するために使用される公開鍵の範囲にとどまります。「エンティティ」、「プライベート キー」、「公開鍵」も参照してください。

ドメイン コンフィグレーション ウィザード

新しい WebLogic Server ドメインの作成を容易にする、対話形式のグラフィカル ユーザ インタフェース (GUI)。このウィザードでは、複数のスタンドアロン サーバ、ノード マネージャを使用する管理サーバと管理対象サーバ、クラスタ化されたサーバの WebLogic Server ドメイン コンフィグレーションを作成できます。コンフィグレーション ウィザードを使用して、WebLogic Server ドメインの適切なディレクトリ構造、基本的な `config.xml` ファイル、およびドメイン内のサーバの起動に使用できるスクリプトを作成できます。

組み込み LDAP サーバ

ユーザ、グループ、セキュリティ ロール、セキュリティ ポリシー、および資格情報が格納されるサーバ。WebLogic 認証プロバイダ、認可プロバイダ、ロール マッピング プロバイダ、および資格マッピング プロバイダは、組み込み LDAP サーバをセキュリティ プロバイダ データベースとして使用します。「資格」、「グループ」、「セキュリティ ポリシー」、「セキュリティ ロール」も参照してください。

エンティティ

特定の個別の単位として独立して存在する、人物、企業、オブジェクトなど。

ファイル レルム

WebLogic Server 6.x で、ユーザ、グループ、暗号化パスワード、および ACL をファイルに格納するレルム。WebLogic Server 7.0 以降では、ファイル レルムは互換性セキュリティでのみ使用します。「互換性セキュリティ」も参照してください。

ファイアウォール

企業内のネットワークとインターネットとの間のトラフィックをモニタしたり、企業内のネットワークに出入できるネットワーク トラフィックのタイプを規制したりするソフトウェア。ファイアウォールをインターネットに接続したり企業内のネットワーク内に設定したりすることで、ネットワークへの不正アクセスを防ぐことができます。ファイアウォールは、コンピュータに関する情報やネットワークでやり取りされる情報を保護します。ファイアウォールは、許可されるプロトコルのタイプの制限や、IP アドレスおよび DNS ノード名によるネットワーク ノードからのアクセスの制限など、さまざまなタイプのフィルタを使用してアクセスを防ぎます。

グローバル ロール

セキュリティ レルム内のすべての WebLogic リソースに適用されるセキュリティ ロール。たとえば、WebLogic ロール マッピング プロバイダがデフォルト セキュリティ レルムで使用されている場合は、ユーザ、グループ、アクセス時間に関してグローバル ロールを定義できます。「ロール マッピング プロバイダ」、「スコープ ロール」、「セキュリティ レルム」、「セキュリティ ロール」、「WebLogic リソース」も参照してください。

グループ

部署、職務、肩書きなどの特性を共有するユーザの集合。グループは、サーバ管理者によって割り当てられる静的な ID であり、セキュリティ ロールに関連付けられます。グループにパーミッションを与えると、そのグループのメンバーである各ユーザにパーミッションを与えることになります。「ユーザ」も参照してください。

ホスト名検証 (Host Name Verification)

SSL 接続先のホスト名が予定していた通信先、または許可された通信先であることを確認するプロセス。「ホスト名検証 (Host Name Verifier)」、「セキュア ソケット レイヤ (SSL)」も参照してください。

ホスト名検証 (Host Name Verifier)

SSL 接続先のホストが予定していた通信先、または許可された通信先であることを確認するコード。WebLogic クライアントまたは WebLogic Server インスタンスが別のアプリケーション サーバの SSL クライアントとして機能している場合には、介在者の攻撃を防ぐのに役立ちます。WebLogic Server の SSL ハンドシェーク機能としてのデフォルトの動作は、SSL サーバのデジタル証明書のサブジェクト識別名 (DN) にある一般名と、SSL 接続の開始に使用する SSL サーバのホスト名を比較することです。サブジェクト DN とホスト名が一致しない場合、SSL 接続は中断されます。「デジタル証明書」、「ホスト名検証 (Host Name Verification)」、「セキュア ソケット レイヤ (SSL)」、「サブジェクト」も参照してください。

ID アサーション

クライアントの ID が、外部のソースから生成されるクライアント提供のトークンを使用して確立される、特殊なタイプの認証。ID は、トークンがユーザ名にマッピングされるときに断定されます。たとえば、クライアントの ID はデジタル証明書を使用して確立することができ、その証明書をシステム内で回すことができるため、ユーザは何度もサイン

オンを求められることはありません。このため、ID アサーションを使用すると、シングル サインオンが可能になります。「認証」、「デジタル証明書」、「ID アサーション プロバイダ」、「シングル サインオン」、「SSL トンネリング」、「トークン」も参照してください。

ID アサーション プロバイダ

境界認証 (トークンを使用する特別なタイプの認証) を実行するセキュリティ プロバイダ。ID アサーション プロバイダを使用しても、WebLogic Server ではユーザを検証して信頼を確立できます。したがって、ID アサーション プロバイダの機能は、トークンを検証してユーザ名にマップすることになります。「境界認証」、「セキュリティ プロバイダ」、「トークン」、「ユーザ」も参照してください。

JAAS 制御フラグ

セキュリティ レルムに複数の認証プロバイダがコンフィグレーションされている場合に、ログイン シーケンスによる認証プロバイダの使用方法を決定する制御フラグ。「認証プロバイダ」も参照してください。

JAAS LoginModule

セキュリティ レルム内のユーザの認証と、サブジェクト内への必要なプリンシパル (ユーザ / グループ) の格納を担当するモジュール。LoginModule は、認証プロバイダの必須コンポーネントであり、境界認証用に別個の LoginModule を開発する必要がある場合は ID アサーション プロバイダのコンポーネントにもなります。境界認証に使用されない LoginModule も、提示された証明データ (たとえば、ユーザのパスワード) が正しいかどうかを確認します。「認証」、「グループ」、「ID アサーション プロバイダ」、「境界認証」、「プリンシパル」、「セキュリティ レルム」、「サブジェクト」も参照してください。

JAAS (Java Authentication and Authorization Service)

認証と、ユーザのアクセス制御を可能にする Java パッケージのセット。JASS は、標準 Pluggable Authentication Module (PAM) フレームワークの Java バージョンを実装し、ユーザベース認証をサポートします。WebLogic Server では、JAAS の認証部分のみを実装します。「認証」、「認可」、「ユーザ」も参照してください。

Java Cryptography Architecture

Java プラットフォーム向けの暗号機能へのアクセスと暗号機能の開発に使用するフレームワーク。Sun Microsystems, Inc. が提供する Java

Cryptography Architecture の説明については、`http://java.sun.com/j2se/1.3/docs/guide/security/CryptoSpec.html#Introduction` を参照してください。「Java Cryptography Extensions (JCE)」も参照してください。

Java Cryptography Extensions (JCE)

暗号、鍵交換、および Message Authentication Code (MAC) アルゴリズム用の API を含む Java Cryptography Architecture API を拡張した Java パッケージのセット。Sun Microsystems, Inc. が提供する JCE の説明については、`http://java.sun.com/j2se/1.4/docs/guide/security/jce/JCERefGuide.html` を参照してください。

「JAAS (Java Authentication and Authorization Service)」も参照してください。

Java Naming and Directory Interface (JNDI)

Java アプリケーションにネーミング サービスを提供するアプリケーション プログラミング インタフェース (API)。JNDI は、Sun Microsystems の J2EE 技術の不可欠なコンポーネントで、特定のネーミング サービスまたはディレクトリ サービスの実装とは無関係に定義されています。JNDI では、単一の方法で、さまざまな新しいサービスや既存のサービスにアクセスできます。このサポートでは、標準サービス プロバイダ インタフェース (SPI) 規約を使用して JNDI フレームワークに任意のサービス プロバイダ実装をプラグインできます。さらに、適切なサービス プロバイダをプラグインすることで、WebLogic Server の Java アプリケーションから LDAP などの外部ディレクトリ サービスに標準化された方法でアクセスできるようになります。

Java セキュリティ マネージャ

Java 仮想マシン (JVM) のセキュリティ マネージャ。Java セキュリティ マネージャは Java API と連携し、`java.lang.SecurityManager` クラスを通じてセキュリティ境界を定義することにより、プログラマが各自の Java アプリケーション用のカスタム セキュリティ ポリシーを確立することを可能にしています。

WebLogic Server では、Java 2 の Java セキュリティ マネージャ の使用がサポートされており、信頼性のないコードが Java セキュリティ ポリシー ファイルで制限されたアクションを実行できないようにすることができます。Java セキュリティ マネージャは、Java セキュリティ ポリ

シー ファイルを使用して一連のパーミッションをクラスに強制的に付与します。これらのパーミッションを使用すると、JVM のインスタンスで実行される指定されたクラスに特定の実行時処理を許可するか拒否するかを設定できます。「Java セキュリティ ポリシー ファイル」、「ポリシー条件」も参照してください。

Java セキュリティ ポリシー ファイル

WebLogic Server でサポートされている Java 仮想マシン (JVM) のインスタンスで実行される指定したクラスに、一連のパーミッションを強制的に付与するために Java セキュリティ マネージャによって使用されるファイル。JVM のインスタンスで実行されるクラスは、パーミッションを使用して特定の実行時処理を許可または拒否します。「Java セキュリティ マネージャ」、「ポリシー条件」も参照してください。

JNDI

Java Naming and Directory Interface (JNDI) を参照。

Kerberos チケット

物理的にセキュアでないネットワークに対するアクセスの制御に使用される、数百バイトの長さのシーケンス。Kerberos チケットは Kerberos プロトコルに基づいています。Kerberos はネットワーク認証プロトコルであり、このプロトコルを使用することで、ネットワークを越えて通信を行うエンティティ (ユーザおよびサービス) は盗聴、反復などの攻撃を防ぎながら相互に身元を証明できます。このプロトコルは、秘密鍵暗号化方式を使用してクライアント / サーバ アプリケーションに強力な認証を提供するために設計されました。詳細については、`http://web.mit.edu/kerberos/www/` を参照してください。「プライベート キー」も参照してください。

キーストア

プライベート キーと信頼性のある認証局の組み合わせのメモリ内集合。情報は、パスワード、クレジット カード番号、暗証番号などの個人を識別する情報のパスフレーズによって保護されます。Administration Console では、キーストアは信頼キーストアと呼ばれます。詳細については、Sun Microsystems, Inc. による SDK 1.3 Javadoc (http://java.sun.com/j2se/1.3/docs/api/index.html) を参照してください。「プライベート キー」、「信頼性のある (ルート) 認証局」も参照してください。

LDAP 認証プロバイダ

Lightweight Data Access Protocol (LDAP) サーバ (iPlanet、Active Directory、Novell、OpenLDAP など) を使用してユーザおよびグループ情報にアクセスする認証プロバイダ。「グループ」、「ユーザ」も参照してください。

LDAP セキュリティ レルム

WebLogic Server 6.x のセキュリティ レルム。WebLogic Server 6.x では、セキュリティ レルムは認証および認可サービスを提供します。LDAP セキュリティ レルムは、LDAP サーバを使用した認証を提供します。このサーバを使用すると、組織内のすべてのユーザを LDAP ディレクトリだけで管理できます。LDAP セキュリティ レルムは、Open LDAP、Netscape iPlanet、Microsoft Site Server、および Novell NDS をサポートしています。WebLogic Server 7.0 以降では、LDAP セキュリティ レルムは互換性セキュリティを使用している場合にのみ使用できます。「認証」、「認可」、「互換性セキュリティ」、「ファイル レルム」、「セキュリティ レルム」、「ユーザ」も参照してください。

LoginModule

JAAS LoginModule を参照。

MBean

管理対象 Bean (managed bean) の略で、JMX (Java Management eXtensions) で管理可能なリソースを表す Java オブジェクトのこと。MBean は、MBean タイプのインスタンスです。MBean は、セキュリティ プロバイダをコンフィグレーションおよび管理するために使用します。「MBean タイプ」、「セキュリティ プロバイダ」も参照してください。

MBean 定義ファイル (MDF)

WebLogic MBeanMaker で MBean タイプのファイルを生成するために使用される XML ファイル。「MBean タイプ」、「WebLogic MBeanMaker」も参照してください。

MBean 実装ファイル

カスタム セキュリティ プロバイダ用の MBean タイプを作成するために WebLogic MBeanMaker ユーティリティによって生成される複数の中間 Java ファイルの 1 つ。このファイルを編集して、特定のメソッド実装を提供します。「MBean 情報ファイル」、「MBean インタフェース ファイ

ル」、「MBean タイプ」、「WebLogic MBeanMaker」も参照してください。

MBean 情報ファイル
: カスタム セキュリティ プロバイダ用の MBean タイプを作成するために WebLogic MBeanMaker ユーティリティによって生成される複数の中間 Java ファイルの 1 つ。このファイルの大部分はメタデータなので、編集は不要です。「MBean 実装ファイル」、「MBean インタフェース ファイル」、「MBean タイプ」、「WebLogic MBeanMaker」も参照してください。

MBean インタフェース ファイル
: カスタム セキュリティ プロバイダ用の MBean タイプを作成するために WebLogic MBeanMaker ユーティリティによって生成される複数の中間 Java ファイルの 1 つ。このファイルは実行時クラスまたは MBean 実装がコンフィグレーション データを取得するために使用する MBean のクライアントサイド API であり、編集は不要です。「MBean 実装ファイル」、「MBean 情報ファイル」、「MBean タイプ」、「実行時クラス」、「WebLogic MBeanMaker」も参照してください。

MBean JAR ファイル (MJF)
: セキュリティ プロバイダの実行時クラスと MBean タイプが格納される JAR ファイル。MJF は、WebLogic MBeanMaker によって作成されます。「MBean タイプ」、「実行時クラス」、「セキュリティ プロバイダ」、「WebLogic MBeanMaker」も参照してください。

MBean タイプ
: セキュリティ プロバイダのコンフィグレーションと管理に使用する MBean を作成するためのファクトリ。MBean タイプは、WebLogic MBeanMaker によって作成されます。「MBean」、「セキュリティ プロバイダ」、「WebLogic MBeanMaker」も参照してください。

メッセージ ダイジェスト
: プレーン テキストからデジタル作成されたハッシュ (フィンガープリント)。完全なメッセージがハッシュの作成に使用されても、ハッシュからメッセージを再作成することはできません。メッセージ ダイジェストは、介在者の攻撃を防ぐのに役立ちます。特定のプレーン テキストのダイジェストは 1 つしかないので、ダイジェストをメッセージの信頼性の確認に使用できます。このため、このプロセスはメッセージのデジタル署名を生成することになります。デジタル署名を使用して否認防止サー

ビスおよび整合性サービスを提供できます。「メッセージ ダイジェスト アルゴリズム」も参照してください。

メッセージ ダイジェスト アルゴリズム
: プレーン テキストからのメッセージ ダイジェストの作成に使用される計算手順。メッセージ ダイジェストが作成されたら、他のセキュリティ メカニズムを使用してそのダイジェストを暗号化し伝達します。「メッセージ ダイジェスト」も参照してください。

相互認証
: サーバとクライアントの両方に身元証明の提示を要求する認証。双方向 SSL 認証は、サーバとクライアントの両方がデジタル証明書を提示して身元を証明する相互認証の一形態です。ただし双方向 SSL 認証では SSL レベルで認証が行われる一方で、他の形態の相互認証ではプロトコル スタックのより上位のレベルで認証が実行されます。「認証」、「デジタル証明書」、「セキュア ソケット レイヤ (SSL)」、「双方向 SSL 認証」、「信頼性のある (ルート) 認証局」も参照してください。

否認防止
: セキュリティ イベントが発生したという決定的な証拠。

一方向 SSL 認証
: サーバはクライアントに対して証明書を提示する必要があるが、クライアントはサーバに対して証明書を提示する必要がないという SSL 認証のタイプ。クライアントはサーバを認証する必要がありますが、サーバはどのクライアントの接続も受け入れます。WebLogic Server ではデフォルトで有効になっています。「相互認証」、「双方向 SSL 認証」も参照してください。

境界認証
: アプリケーション サーバ ドメインの外側で発生する認証。境界認証は通常、リモート ユーザが断定された ID と、検証の実行に使用される特定の形態の対応する証明データ (一般的にはパスフレーズの形態、具体的にはパスワード、クレジット カード番号、暗証番号などの個人を識別する情報) を認証サーバ (一般的には Web サーバ) に指定することによって行われます。認証サーバは、検証を実行し、アーティファクトつまりトークンをアプリケーション サーバ ドメイン (WebLogic Server ドメインなど) に渡します。アプリケーション サーバはそのトークンをドメイン内のシステムで回すことができるため、ユーザは何度もサインオンを求められることはありません。

実際に ID を保証するエンティティである、**認証エージェント**は、さまざまな形態を取ることができます。たとえば、仮想プライベート ネットワーク (VPN)、ファイアウォール、企業認証サービス (Web サーバ) といったグローバルな ID サービスの形態を取ることができます。

WebLogic Server のセキュリティ アーキテクチャは、境界に基づく認証 (Web サーバ、ファイアウォール、VPN) を実行し、複数のセキュリティ トークン タイプとプロトコル (SOAP、IIOP-CSIv2) を処理する ID アサーション プロバイダをサポートします。「認証」、「ID アサーション」も参照してください。

ポリシー条件
: セキュリティ ポリシーを作成する際の条件。ポリシー条件に対して特定の情報 (実際のユーザ名、グループ、セキュリティ ロール、開始 / 終了時間など) を指定したものは式と呼ばれます。「ポリシー文」を参照してください。

ポリシー式
: 「ポリシー文」を参照。

ポリシー文
: 誰に WebLogic リソースへのアクセス権が付与されるかを定義する式の集合。したがって、作成するセキュリティ ポリシーの主要部分となります。ポリシー文は、ポリシー式とも呼ばれます。「ポリシー条件」を参照してください。

プリンシパル
: 認証の結果としてユーザ、グループ、またはシステム プロセスに割り当てられる ID。プリンシパルは、任意の数のユーザまたはグループで構成できます。通常、プリンシパルはサブジェクトに格納されます。「認証」、「グループ」、「サブジェクト」、「ユーザ」も参照してください。

プリンシパル検証
: 署名し、その署名後にプリンシパルが変更されていないことを確認する行為。プリンシパル検証によってプリンシパルの信頼性が確立されます。「プリンシパル」も参照してください。

プライベート キー
: 秘密のメッセージを交換する当事者だけが知っている暗号化 / 復号化鍵。そのオーナ以外には非公開のまま保管する必要があるので、プライベート キーと呼ばれます。「公開鍵」も参照してください。

プライベート キー アルゴリズム
: 暗号文をエンコード (または暗号化) するために使用される計算手順。プライベート キーで暗号化されたデータは、公開鍵を使用することによってのみ解読できます。「プライベート キー」、「公開鍵」、「RDBMS セキュリティ レルム」も参照してください。

プログラムによるセキュリティ
: Java メソッドを使用してサーブレットと EJB で定義されるアプリケーション セキュリティ。

公開鍵
: メッセージおよびデジタル署名を効果的に暗号化するために使用でき、プライベート キーと結合される、暗号化鍵として認証局から提供される値。誰でも使用できるので、公開鍵と呼ばれます。公開鍵暗号化方式は、データの暗号化と解読に異なる鍵を使用するので、非対称鍵暗号化方式とも呼ばれます。「非対称鍵暗号化方式 プライベート キー」も参照してください。

公開鍵アルゴリズム
: プレーン テキストをエンコード (または暗号化) するために使用される計算手順。公開鍵で暗号化されたデータは、プライベート キーを使用することによってのみ解読できます。「プライベート キー」、「プライベート キー アルゴリズム」、「公開鍵」も参照してください。

公開鍵暗号化方式
: 非対称鍵暗号化方式を参照。

RDBMS セキュリティ レルム
: WebLogic Server 6.x のセキュリティ レルム。WebLogic Server 6.x では、セキュリティ レルムで認証および認可サービスを提供していました。RDBMS セキュリティ レルムは、ユーザ、グループ、および ACL をリレーショナル データベースに保存します。WebLogic Server 7.0 以降では、RDBMS セキュリティ レルムは互換性セキュリティを使用している場合にのみ使用できます。「アクセス制御リスト (ACL)」、「認証」、「認可」、「互換性セキュリティ」、「グループ」、「セキュリティ レルム」、「ユーザ」も参照してください。

レルム アダプタ裁決プロバイダ
: 互換性セキュリティを使用するセキュリティ レルム向けに WebLogic 認可プロバイダとレルム アダプタ認可プロバイダの両方を併用できるよう

にする裁決プロバイダ。「互換性セキュリティ」、「互換性レルム」も参照してください。

レルム アダプタ監査プロバイダ
: 互換性セキュリティを使用する WebLogic Server デプロイメントで `weblogic.security.audit` インタフェースの実装を使用できるようにする、互換性レルムの監査プロバイダ。互換性レルムとレルム アダプタ プロバイダにアクセスするには、WebLogic Server Administration Console で互換性セキュリティを実行する必要があります。「互換性セキュリティ」、「互換性レルム」も参照してください。

レルム アダプタ認証プロバイダ
: 6.x セキュリティ レルムの認証サービスとの下位互換性を保持できるようにする、互換性レルムの認証プロバイダ。互換性レルムとレルム アダプタ プロバイダにアクセスするには、WebLogic Server Administration Console で互換性セキュリティを実行する必要があります。「互換性セキュリティ」、「互換性レルム」も参照してください。

レルム アダプタ認可プロバイダ
: 6.x セキュリティ レルムの認可サービスとの下位互換性を保持できるようにする、互換性レルムの認可プロバイダ。互換性レルムとレルム アダプタ プロバイダにアクセスするには、WebLogic Server Administration Console で互換性セキュリティを実行する必要があります。「互換性セキュリティ」、「互換性レルム」も参照してください。

レルム アダプタ プロバイダ
: WebLogic Server 7.0 以降で互換性セキュリティを使用する場合に、WebLogic Server 6.x のセキュリティ サービスにアクセスするために使用されるセキュリティ プロバイダのタイプ。これらのプロバイダを使用することで 6.x セキュリティ プロバイダを使用できます。つまり、WebLogic Server 7.0 以降で 6.x セキュリティ プロバイダを使用できます。互換性レルムとレルム アダプタ プロバイダにアクセスするには、WebLogic Server Administration Console で互換性セキュリティを実行する必要があります。「互換性セキュリティ」、「互換性レルム」も参照してください。

リソース
: WebLogic リソースを参照。

リソース アダプタ

アプリケーション サーバ (WebLogic Server など) やアプリケーション クライアントでエンタープライズ情報システム (EIS) に接続するために使用されるシステムレベルのソフトウェア ドライバ (コネクタともいう)。リソース アダプタには、Java コンポーネントと、必要に応じて EIS との対話に必要なネイティブ コンポーネントが含まれます。

WebLogic J2EE コネクタ アーキテクチャは、EIS ベンダおよびサードパーティ アプリケーション開発者が開発し、Sun Microsystems の J2EE プラットフォーム仕様、バージョン 1.4 に準拠しているアプリケーション サーバにデプロイ可能なリソース アダプタをサポートしています。

ロール条件

セキュリティ ロール (グローバルまたはスコープ) をユーザまたはグループに付与する条件。ロール条件に対して特定の情報 (実際のユーザ名、グループ、開始 / 終了時間など) を指定したものは式と呼ばれます。「セキュリティ ポリシー」、「ロール マッピング」 を参照してください。

ロール式

ロール条件の作成時に指定する特定の情報。「ロール条件」を参照してください。

ロール マッピング

WebLogic Security サービスがセキュリティ ロール条件とユーザまたはグループを比較して、そのユーザまたはグループにセキュリティ ロールを動的に付与するかどうかを決定するプロセス。ロール マッピングは実行時に、保護対象の WebLogic リソースに対するアクセス決定を下す直前に発生します。「アクセス決定」、「グループ」、「プリンシパル」、「ロール条件」、「セキュリティ ロール」、「ユーザ」、「WebLogic リソース」、「WebLogic Security サービス」も参照してください。

ロール マッピング プロバイダ

WebLogic リソースを操作しようとしているサブジェクト内のプリンシパルに適用されるロールを調べるセキュリティ プロバイダ。通常、この操作では WebLogic リソースへのアクセスを取得する必要があるので、ロール マッピング プロバイダは認可プロバイダとともに使用するのが一般的です。「認可プロバイダ」、「プリンシパル」、「セキュリティ ロール」、「サブジェクト」、「WebLogic リソース」 も参照してください。

ロール文
: セキュリティ ロールが付与される条件を定義した式の集合。したがって、作成するセキュリティ ロールの主要部分となります。「ロール式」を参照してください。

実行時クラス
: セキュリティ サービス プロバイダ インタフェース (SSPI) を実装し、セキュリティ プロバイダの実際のセキュリティ関連の動作を格納する Java クラス。「セキュリティ プロバイダ」、「セキュリティ サービス プロバイダ インタフェース (SSPI)」も参照してください。

スコープ ロール
: セキュリティ レルム内の特定の WebLogic リソースに適用されるセキュリティ ロール。「グローバル ロール」、「ロール マッピング プロバイダ」、「セキュリティ ロール」、「セキュリティ レルム」も参照してください。

秘密鍵暗号化方式
: 対称鍵暗号方式を参照。

セキュア ソケット レイヤ (SSL)
: Netscape 社がアプリケーション間のデータ プライバシを提供するために開発したインターネット トランスポート レベル技術。通常、セキュア ソケット レイヤ (SSL) は、(1) 互いの ID を認証するためにアプリケーションが利用するメカニズム、および (2) アプリケーション間でやり取りするデータの暗号化を提供します。SSL では、認証を提供する公開鍵暗号化方式、およびプライバシとデータの整合性を提供する秘密鍵暗号化方式とデジタル署名の使用がサポートされています。「認証」、「デジタル署名」、「公開鍵暗号化方式」、「対称鍵暗号方式」も参照してください。

Security Assertion Markup Language (SAML)
: セキュリティ情報を交換するために使用する XML ベースのフレームワーク。SAML 実装は、相互運用性のある XML ベースのセキュリティ ソリューションを提供します。このソリューションにより、認証および認可情報のセキュアなやり取りが可能になります。SAML は、Web サービスに対してシングル サインオン機能を有効にするための主要な要素です。詳細については、`http://xml.coverpages.org/saml.html` を参照してください。

SAML などの異なるタイプのトークンをサポートするカスタム ID アサーション プロバイダを WebLogic Server 向けに開発できます。「認証」、「認可」、「ID アサーション」、「境界認証」、「ドメイン間シングル サインオン」、「ユーザ」も参照してください。

セキュリティ ポリシー

権限のないアクセスから WebLogic リソースを保護するための、WebLogic リソースとユーザ、グループ、セキュリティ ロールとの関連付け。WebLogic リソースは、セキュリティ ポリシーが割り当てられるまでは保護されません。セキュリティ ポリシーは、個々の WebLogic リソースか、または WebLogic リソースのコンポーネントに割り当てることができます。

WebLogic Server 7.0 以降では、アクセス制御リスト (ACL) に代わってセキュリティ ポリシーが使用されます (互換性セキュリティを使用する場合を除く)。「アクセス制御リスト (ACL)」、「グループ」、「セキュリティ ロール」、「ユーザ」、「WebLogic リソース」も参照してください。

セキュリティ プロバイダ

WebLogic Server 7.0 以降で、アプリケーションにセキュリティ サービス (認証、認可、監査、資格マッピングなど) を提供するために、WebLogic Server セキュリティ レルムに「プラグイン」できるソフトウェア モジュール。セキュリティ プロバイダは、実行時クラスと MBean (それぞれ SSPI および MBean タイプから作成される) で構成されます。セキュリティ プロバイダは、WebLogic Server に付属している WebLogic セキュリティ プロバイダか、またはカスタム セキュリティ プロバイダです。「カスタム セキュリティ プロバイダ」、「MBean」、「MBean タイプ」、「実行時クラス」、「セキュリティ サービス プロバイダ インタフェース (SSPI)」、「WebLogic セキュリティ プロバイダ」も参照してください。

セキュリティ プロバイダ データベース

一部のセキュリティ プロバイダでセキュリティ サービスを提供するために使用される、ユーザ、グループ、セキュリティ ポリシー、ロール、および資格が格納されたデータベース。セキュリティ プロバイダ データベースとしては、(WebLogic セキュリティ プロバイダで使用されるような) 組み込み LDAP サーバ、(サンプル セキュリティ プロバイダで使用されるような) properties ファイル、またはすでに使用しているプロダクション レベルのデータベースを使用できます。「資格」、「組み込み

LDAP サーバ」、「グループ」、「セキュリティ ロール」、「セキュリティ ポリシー」、「WebLogic セキュリティ プロバイダ」も参照してください。

セキュリティ レルム

WebLogic Server 6.x では、セキュリティ レルムは認証および認可サービスを提供します。ファイル レルムか、または Lightweight Data Access Protocol (LDAP)、Windows NT、UNIX、RDBMS レルムなどの代替セキュリティ レルムを使用します。認証をカスタマイズする場合は、独自のセキュリティ レルムを記述して WebLogic Server 環境に統合することができます。WebLogic Server 6.x では、ドメインに複数のセキュリティ レルムをコンフィグレーションできません。「ファイル レルム」も参照してください。

WebLogic Server 7.0 以降では、セキュリティ レルムはスコーピング (有効範囲の設定) メカニズムとして機能します。各セキュリティ レルムは、コンフィグレーション済みのセキュリティ プロバイダ、ユーザ、グループ、ロール、およびセキュリティ ポリシーで構成されます。単一のドメインに複数のセキュリティ レルムをコンフィグレーションできますが、デフォルト (アクティブ) セキュリティ レルムに指定できるのはそのうちの 1 つだけです。WebLogic Server には、myrealm と互換性レルムという 2 つのデフォルト セキュリティ レルムが用意されています。既存の 6.x セキュリティ コンフィグレーションには互換性レルムを介してアクセスできます。WebLogic Server 6.x では可能でしたが、アプリケーション プログラミング インタフェースを使用して独自のセキュリティ レルムを記述することはできなくなりました。代わりに、新しいセキュリティ レルム (デフォルトでは myrealm という) をコンフィグレーションして必要なセキュリティ サービスを提供し、そのセキュリティ レルムをデフォルト セキュリティ レルムに設定します。「互換性レルム」、「カスタム セキュリティ レルム」、「デフォルト レルム」、「ドメイン コンフィグレーション ウィザード」、「セキュリティ プロバイダ」、「WebLogic リソース」も参照してください。

セキュリティ ロール

特定の条件に基づいてユーザまたはグループに付与される、動的に計算される特権。グループとロールの違いは、グループがサーバ管理者によって割り当てられる静的な ID であるのに対し、ロールのメンバーシップはユーザ名、グループ メンバーシップ、時刻などのデータに基づいて動的に計算されることです。セキュリティ ロールは個々のユーザま

たはグループに付与されます。また、複数のロールを使用して WebLogic リソース用のセキュリティ ポリシーを作成できます。セキュリティ ロールを作成したら、そのロールと WebLogic リソースとの関連付けを定義します。この関連付け (セキュリティ ポリシー) により、WebLogic リソースに誰がどのようにアクセスできるかが指定されます。「グローバル ロール」、「グループ」、「ロール マッピング」、「スコープ ロール」、「セキュリティ ポリシー」、「ユーザ」、「WebLogic リソース」も参照してください。

セキュリティ サービス プロバイダ インタフェース (SSPI)
: カスタム セキュリティ プロバイダを開発して WebLogic Server Security サービスに統合するために使用する WebLogic パッケージのセット。これらのインタフェースは、WebLogic セキュリティ プロバイダおよびカスタム セキュリティ プロバイダによって実装されます。WebLogic Security フレームワークは、これらのインタフェースでメソッドを呼び出してセキュリティ操作を実行します。「セキュリティ プロバイダ」、「WebLogic Security フレームワーク」も参照してください。

シングル サインオン
: ユーザが一度アプリケーション コンポーネントにサインオンすれば、他のさまざまなアプリケーション コンポーネントに (それらが独自の認証方式を使用している場合でも) アクセスできる機能。シングル サインオンは、ID アサーション、LoginModule、およびトークンを使用することで実現されます。「認証」、「ドメイン間シングル サインオン」、「ID アサーション」、「JAAS LoginModule」、「トークン」、「ユーザ」も参照してください。

SSL ハードウェア アクセラレータ
: クライアントに対する SSL のパフォーマンスを向上させるために Web スイッチに接続する、セキュア ソケット レイヤ (SSL) の周辺プラットフォーム。たとえば Alteon SSL アクセラレータを WebLogic Server で使用できます。このアクセラレータは、Web スイッチを介してクライアント (この場合は WebLogic Server) と TCP ハンドシェークを行い、セッションのすべての SSL 暗号化と解読を行います。

SSL トンネリング
: セキュア ソケット レイヤ (SSL) の IP ベースのプロトコル上のトンネリング。トンネリングとは、各 SSL レコードをカプセル化し、別のプロト

コル上でレコードを送信するために必要なヘッダと一緒にパッケージ化することです。

SSPI MBean

WebLogic セキュリティ プロバイダの MBean タイプを生成するために BEA で使用するインタフェース。そのインタフェースからカスタム セキュリティ プロバイダの MBean タイプを生成できます。SSPI MBean は、コンフィグレーション上は必須であり、管理上はオプションです。「カスタム セキュリティ プロバイダ」、「MBean タイプ」、「WebLogic セキュリティ プロバイダ」も参照してください。

サブジェクト

単一のエンティティ (たとえば人物) に関連する情報を、JAAS (Java Authentication and Authorization Service) に従ってグループ化したもの。関連する情報には、サブジェクトの ID、プリンシパル、セキュリティ関連の属性 (たとえば、パスワード、暗号鍵) などが含まれます。サブジェクトには任意の数のプリンシパルを格納できます。WebLogic Server のようなアプリケーション サーバでは、ユーザもグループもプリンシパルとして使用することができます。WebLogic セキュリティ プロバイダ (WebLogic Server 製品に付属のセキュリティ プロバイダ) では、ユーザのプリンシパル (`WLSUser Principal`) およびそのユーザが属す各グループのプリンシパル (`WLSGroups Principals`) がサブジェクトに格納されています。カスタム セキュリティ プロバイダに格納される ID は、上記とは異なる場合があります。「認証」、「カスタム セキュリティ プロバイダ」、「グループ」、「JAAS 制御フラグ」、「プリンシパル」、「ユーザ」も参照してください。

対称鍵暗号方式

データの暗号化と解読に同一の鍵を使用する暗号化アルゴリズムを採用した鍵ベースの暗号化方式。対称鍵暗号化方式は、秘密鍵暗号化方式とも呼ばれます。「非対称鍵暗号化方式」も参照してください。

トークン

ユーザの認証プロセスまたはシステム プロセスの一部として生成されるアーティファクト。ID アサーションを使用する場合は、認証を受けたユーザかどうかを判別するためにトークンを提示します。トークンには、Kerberos や SAML (Security Assertion Markup Language) など、さまざまなタイプがあります。「認証」、「Security Assertion Markup Language (SAML)」、「セキュア ソケット レイヤ (SSL)」、「ID アサーション」、「SSL トンネリング」、「ユーザ」も参照してください。

トラスト マネージャ

ピアのデジタル証明書内での検証エラーをオーバーライドし、SSL ハンドシェークを継続できるようにするインタフェース。また、サーバのデジタル証明書チェーンで付加的な検証を実行することで、SSL ハンドシェークを中止することもできます。

信頼性のある (ルート) 認証局

デジタル署名および公開鍵 / プライベート キーの組み合わせの作成に使用されるデジタル証明書を発行する有名かつ信頼された第三者組織または企業。信頼性のある認証局の機能は、証明書を提示する個人または組織の身元を保守する公証人の機能に似ています。信頼性のある認証局は、他の証明書に署名するために使用される証明書を発行します。認証局はルート認証局とも呼ばれます。その権限が認識されているため、その身元の検証が不要だからです。信頼性のある (ルート) 認証局 (CA) の証明書は、証明書を認証するアプリケーションにインストールされます。たとえば、Web ブラウザは通常、あらかじめインストールされた、いくつかの信頼性のある (ルート) CA の証明書とともに配布されます。サーバの証明書に署名した証明局が不明な場合に、サーバの証明書がクライアントによって必ず認証されるようにするには、周知の認証局によって署名された証明書で終了する証明書チェーンを発行することをお勧めします。「証明書チェーン」、「プライベート キー」、「公開鍵」も参照してください。

双方向 SSL 認証

クライアントとサーバの間で接続スレッドを有効にする前に、両方に証明書の提示を要求する認証。双方向の SSL 認証では、WebLogic Server はクライアントに対して自身を認証するだけでなく (証明書認証の最低限の要件)、要求側のクライアントにも認証を要求します。クライアントは、信頼された認証局が発行したデジタル証明書を提出するよう要求されます。このタイプの認証は、アクセスを許可する対象を信頼されたクライアントに制限する場合に便利です。双方向の SSL 認証は、相互認証の一形態です。「認証」、「デジタル証明書」、「相互認証」、「セキュア ソケット レイヤ (SSL)」、「信頼性のある (ルート) 認証局」も参照してください。

UNIX セキュリティ レルム

WebLogic Server 6.x のセキュリティ レルム。UNIX セキュリティ レルムは小さなネイティブ プログラム (`wlauth`) を実行して、ユーザとグループを検索し、UNIX ログイン名とパスワードに基づいてユーザを認証します。`wlauth` プログラムは PAM (Pluggable Authentication

Modules) を使用します。これにより、オペレーティング システムの認証サービスを、このサービスを使用するアプリケーションを変更することなくコンフィグレーションできます。WebLogic Server 7.0 以降では、UNIX セキュリティ レルムは互換性セキュリティを使用している場合にのみ使用できます。「認証」、「認可」、「互換性セキュリティ」、「グループ」、「セキュリティ レルム」も参照してください。

ユーザ

認証が可能なエンティティ。ユーザは、個人または Java クライアントなどのソフトウェア エンティティでもかまいません。各ユーザには、セキュリティ レルムの中で固有の ID が与えられます。セキュリティ管理を効率化するために、ユーザをグループに追加するようにしてください。グループは、通常、企業の同じ部門に所属しているなどの共通点を持つユーザの集合です。ユーザはセキュリティ ロールと関連付けられているグループに入れるか、またはセキュリティ ロールと直に関連付けることができます。「エンティティ」、「グループ」、「セキュリティ ロール」、「WebLogic リソース」も参照してください。

WebLogic コンポーネント

WebLogic Server は J2EE コンポーネント技術 (サーブレット、JSP ページ、およびエンタープライズ JavaBean (EJB) など) を実装します。WebLogic Server アプリケーションを構築するには、必要に応じてこれらのサービス API を使用して、コンポーネントを作成し組み立てる必要があります。コンポーネントは、WebLogic Server の Web コンテナまたは EJB コンテナ内で実行されます。Web コンポーネントは、ブラウザベースの J2EE アプリケーションに対してプレゼンテーション ロジックを提供します。EJB コンポーネントは、ビジネスのオブジェクトやプロセスをカプセル化します。「WebLogic コンテナ」、「Windows NT セキュリティ レルム」も参照してください。

WebLogic コンテナ

開発の短縮化と移植性を促進するために、J2EE ではコンポーネントで必要になる共通のサービスを識別し、それらをコンポーネントをホストするコンテナに実装します。コンテナでは、J2EE 仕様で定義されたライフサイクルのサポートやサービスを提供しているため、構築するコンポーネントでは、下層部の詳細を扱う必要はありません。コンポーネントには作成するオブジェクトやプロセスの記述に必要なコードのみが含まれます。実行環境や、トランザクション管理、アクセス制御、ネットワーク通信、永続性メカニズムなどのサービスにアクセスするためのコードは含まれません。これらのサービスは、WebLogic Server に実装

されるコンテナによって提供されます。さらに、WebLogic コンテナはアプリケーションに J2EE アプリケーション プログラミング インタフェース (API) へのアクセスを提供します。WebLogic コンテナはサーバが起動したら使用できます。このコンポーネント / コンテナの抽象化概念により、開発者は専門分野の仕事ができます。WebLogic Server では、Web コンテナ、EJB コンテナという 2 種類のコンテナが用意されています。「WebLogic コンポーネント」、「Windows NT セキュリティ レルム」も参照してください。

WebLogic J2EE サービス

WebLogic Server は、J2EE サービス (標準のネットワーク プロトコル、データベース システム、メッセージング システムへのアクセスなど) を実装します。WebLogic Server アプリケーションを構築するには、必要に応じてこれらのサービス API を使用して、コンポーネントを作成し組み立てる必要があります。Web アプリケーションと EJB は、JDBC、JMS (Java Messaging Service)、および JTA (Java Transaction API) などの J2EE アプリケーション サービス上に構築されています。「WebLogic コンポーネント」も参照してください。

WebLogic MBeanMaker

MBean 定義ファイル (MDF) を入力とし、MBean タイプのファイルを出力するコマンドライン ユーティリティ。「MBean 定義ファイル (MDF)」、「MBean タイプ」も参照してください。

WebLogic リソース

イベント、サーブレット、JDBC 接続プール、JMS 送り先、JNDI コンテキスト、接続、ソケット、ファイル、企業のアプリケーションやリソース (データベースなど) といった、WebLogic Server からアクセス可能なエンティティ。「エンティティ」も参照してください。

WebLogic Security フレームワーク

セキュリティの施行を統合し、他の WebLogic Server コンポーネントにセキュリティをサービスとして提供する `weblogic.security.service` パッケージのインタフェース。セキュリティ プロバイダは、セキュリティ サービスを必要とするアプリケーションの代わりに WebLogic Security フレームワークに働きかけを行います。「セキュリティ プロバイダ」も参照してください。

WebLogic セキュリティ プロバイダ

BEA が WebLogic Server 製品の一部として提供するセキュリティ プロバイダ。WebLogic セキュリティ プロバイダは、WebLogic Server の SSPI (Security Service Provider Interface) を使用して開発されています。「カスタム セキュリティ プロバイダ」、「セキュリティ プロバイダ」、「セキュリティ サービス プロバイダ インタフェース (SSPI)」も参照してください。

WebLogic Security サービス

セキュリティ アーキテクチャを実装する WebLogic Server サブシステム。このサブシステムは、WebLogic Security フレームワーク、セキュリティ サービス プロバイダ インタフェース (SSPI)、および WebLogic セキュリティ プロバイダという 3 つの主要なコンポーネントで構成されています。

WebLogic Server ドメイン

1 つのコンフィグレーション ファイルで定義された、サーバ、サービス、インタフェース、マシン、および関連する WebLogic リソース マネージャの集合。「WebLogic リソース」も参照してください。

Windows NT セキュリティ レルム

WebLogic Server 6.x のセキュリティ レルム。Windows NT セキュリティ レルムでは、Windows NT ドメイン向けに定義されたアカウント情報を使用して、ユーザとグループを認証します。WebLogic Server 7.0 以降では、Windows NT セキュリティ レルムは互換性セキュリティを使用している場合にのみ使用できます。「認証」、「認可」、「互換性セキュリティ」、「グループ」、「セキュリティ レルム」、「ユーザ」も参照してください。