BEA WebLogic Server™

WebLogic Security の管理

BEA WebLogic Server バージョン 7.0
マニュアルの改訂 : 2003 年 6 月 30 日

著作権

限定的権利条項

商標または登録商標

WebLogic Security の管理

パート番号	マニュアルの改訂	ソフトウェアのバージョン
なし	2003 年 6 月 30 日	BEA WebLogic Server バージョン 7.0

目次

3. セキュリティ プロバイダのコンフィグレーション

4. エンタープライズ情報システムでのシングル サインオン

5. 組み込み LDAP サーバの管理

6. SSL のコンフィグレーション

7. ユーザ アカウントの保護

このマニュアルの内容

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。構成は次のとおりです。

- 第 1 章「セキュリティ管理の概要」では、以前のリリースの WebLogic Server とこのリリースの WebLogic Server のセキュリティの違い、WebLogic Server のデフォルト セキュリティ コンフィグレーション、WebLogic Server のセキュリティのコンフィグレーション手順、および互換性セキュリティについて説明します。
- 第 2 章「デフォルト セキュリティ コンフィグレーションのカスタマイズ」では、デフォルト セキュリティ コンフィグレーションをカスタマイズする理由、新しいセキュリティ レルムのコンフィグレーション要件、およびセキュリティ レルムをデフォルト セキュリティ レルムとして設定する方法について説明します。
- 第 3 章「セキュリティ プロバイダのコンフィグレーション」では、WebLogic Server によって提供されるセキュリティ プロバイダをコンフィグレーションするときに設定しなければならない属性、およびカスタム セキュリティ プロバイダをコンフィグレーションする方法について説明します。
- 第 4 章「エンタープライズ情報システムでのシングル サインオン」では、EIS ユーザが WebLogic リソースにアクセスできるように、資格マップを作成する方法について説明します。
- 第 5 章「組み込み LDAP サーバの管理」では、WebLogic セキュリティ プロバイダが使用する LDAP サーバに関連する管理タスクについて説明します。
- 第 6 章「SSL のコンフィグレーション」では、WebLogic Server 用に SSL をコンフィグレーションする方法について説明します。
- 第 7 章「ユーザ アカウントの保護」では、ユーザ アカウントを保護するための属性の設定、およびユーザ アカウントのロックを解除する方法について説明します。
- 第 8 章「WebLogic ドメインのセキュリティのコンフィグレーション」では、WebLogic ドメインのセキュリティ属性を設定する方法について説明します。

- 第 9 章「互換性セキュリティの使い方」では、互換性セキュリティを使用する方法について説明します。

対象読者

これは、サーバ管理者とアプリケーション管理者向けのマニュアルです。

- **サーバ管理者**は、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セキュリティ コンフィグレーションの提案を行います。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 レルムの設定およびコンフィグレーション、サーバ リソースおよびアプリケーション 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**は、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たアプリケーション リソースへのアクセスを設定および管理したりします。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 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

WebLogic Server セキュリティ レルム内のリソースをセキュリティで保護する方法については、『WebLogic リソースのセキュリティ』を参照してください。WebLogic Server デプロイメントのセキュリティを完全にコンフィグレーションするには、このマニュアルと『WebLogic Security の管理』を合わせて利用してください。

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名とバージョン名、およびマニュアルのタイトルと作成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。

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 *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

表記法	適用
斜体の等幅テキスト	プレースホルダを示す。 例： `String *CustomerName*;`

大文字の等幅テキスト	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。
	例：
	`java utils.MulticastTest -n *name* -a *address*`
	`[-p *portnumber*] [-t *timeout*] [-s *send*]`

|	構文の中で相互に排他的な選択肢を区切る。
	例：
	`java weblogic.deploy [list|deploy|undeploy|update]`
	`password {application} {source}`

...	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
. . .	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。

1　セキュリティ管理の概要

以下の節では、6.x のリリースの WebLogic Server とこのリリースの WebLogic Server の違いを含む、WebLogic Server の新しいセキュリティ システムの概要について説明します。

注意： このマニュアルでは、6.x という用語は WebLogic Server 6.0 および 6.1、およびそれらに関連するサービス パックを表しています。

対象読者

『WebLogic Security の管理』は、サーバ管理者とアプリケーション管理者を対象にしています。

- **サーバ管理者**は、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セキュリティ コンフィグレーションの提案を行います。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 レルムの設定およびコンフィグレーション、サーバ リソースおよびアプリケーション 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**は、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たアプリケーション リソースへのアクセスを設定および管理したりします。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。

サーバ管理者とアプリケーション管理者は、このマニュアルだけでなく、『WebLogic リソースのセキュリティ』にも目を通してください。

WebLogic Server でのセキュリティの変更点

WebLogic Server には、セキュリティのコンフィグレーションと管理を簡素化しながらも、WebLogic Server デプロイメントを保護する堅牢な機能を提供するセキュリティ サービスが用意されています。この節では、セキュリティ サービスが以前のリリースの WebLogic Server からどのように変更されたかを説明します。

セキュリティ レルムのスコープの変更

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。6.x では、ファイル レルムか、または Lightweight Data Access Protocol (LDAP)、Windows NT、Unix、RDBMS レルムなどの代替レルムを選択しました。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した。セキュリティ レルムはドメインに適用され、1 つのドメインに複数のセキュリティ レルムを設定することはできませんでした。

このリリースの WebLogic Server 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、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。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、以下の 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。

- *myrealm* - デフォルトでコンフィグレーションされる WebLogic 裁決、認証、ID アサーション、認可、ロール マッピング、および資格マッピングの各プロバイダを備えています。
- *CompatibilityRealm* - 6.x のセキュリティ コンフィグレーションとの下位互換性を備えています。既存の 6.x セキュリティ コンフィグレーションには *CompatibilityRealm* を介してアクセスできます。

セキュリティ アプリケーション プログラミング インタフェース (API) を使用して記述されたカスタム セキュリティ レルムは、互換性セキュリティでのみサポートされます。このリリースの WebLogic Server では、必要なセキュリティ サービスを提供する新しいセキュリティ レルムをコンフィグレーションし、そのセキュリティ レルムをデフォルト セキュリティ レルムとして設定することで、認証および認可機能をカスタマイズできます。

WebLogic Server のデフォルト セキュリティのコンフィグレーションについては、1-9 ページの「WebLogic Server のデフォルト セキュリティ コンフィグレーション」を参照してください。

セキュリティ レルムをコンフィグレーションしてデフォルト セキュリティ レルムに設定する方法については、第 2 章「デフォルト セキュリティ コンフィグレーションのカスタマイズ」を参照してください。

互換性セキュリティの使い方については、第 9 章「互換性セキュリティの使い方」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、認証や認可など、セキュリティの特定の側面を処理するモジュール コンポーネントです。アプリケーションはデフォルトの WebLogic セキュリティ プロバイダを通じてサービスを利用することができますが、WebLogic Security サービスのインフラストラクチャは柔軟性が高いので、セキュリティ ベンダが WebLogic Server 用の独自のカスタム セキュリティ プロバイダを作成することもできます。WebLogic セキュリティ プロバイダとカスタム セキュリティ プロバイダを適宜組み合わせて独自のセキュリティ ソリューションを構築することができるので、ある分野では新たな技術の進歩を利用しつつ、それ以外の分野では実証済みの手法を堅持することができるようになります。WebLogic Server Administration Console を使用すると、統合された単一の管理インタフェースを通じてすべてのセキュリティ プロバイダを管理できます。

WebLogic Security サービスは、以下のセキュリティ プロバイダをサポートしています。

- **裁決** - セキュリティ レルムに複数の認可プロバイダがコンフィグレーションされる場合、特定のリソースに「アクセスできるか」という質問に対して、それぞれが異なる回答を返す可能性があります。複数の認可プロバイダの回答が一致しない場合にどうするかを決定するのが、裁決プロバイダの主な役割です。裁決プロバイダは、各認可プロバイダの回答に重みを割り当てることによって認可の衝突を解決し、最終決定を返します。
- **監査** - セキュリティ リクエスト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 監査プロバイダは、これらのサービスを提供します。
- **認証** - ユーザまたはシステム プロセスの身元を証明または確認するプロセスです。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。WebLogic Security サービスは、以下の認証をサポートしています。
  - ユーザ名とパスワードによる認証
  - WebLogic Server で直接行われる証明書ベースの認証
  - 外部 Web サーバを介してプロキシされる HTTP 証明書ベースの認証

  WebLogic 認証プロバイダは、これらのサービスを提供します。

- **ID アサーション** - 境界認証 (トークンを使用する特殊なタイプの認証) を行う認証プロバイダを、ID アサーション プロバイダと呼びます。ID アサーションでは、リクエストの外部に存在するクライアント提供のトークンを使用してクライアントの ID を確立します。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。WebLogic ID アサーション プロバイダは、これらのサービスを提供します。
- **認可** - ユーザとリソースとのやり取りを限定して、整合性、機密性、および可用性を確保するプロセス。すなわち、認可は、ユーザの身元などの情報に基づいて WebLogic リソースへのアクセスを制御します。WebLogic 認可プロバイダは、これらのサービスを提供します。
- **資格マッピング** - 資格マップは、WebLogic Server で使用する資格と、レガシー システムまたはリモート システムで使用する資格とのマップです。WebLogic Server は、このマップによって、そのシステム内の特定のリソースへの接続方法を知ります。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 プロバイダは、このようにして資格をマッピングします。WebLogic 資格マッピング プロバイダは、このサービスを提供します。
- **キーストア** - プライベート キーと信頼性のある認証局の証明書を格納する、パスワードで保護されたデータベースの作成と管理のためのメカニズムです。キーストアは、認証や署名を目的としてそれを必要とすることがあるアプリケーションから利用できます。WebLogic Server のセキュリティ アーキテクチャでは、WebLogic キーストア プロバイダを使用してキーストアにアクセスします。

  注意： カスタム キーストア プロバイダはこのリリースの WebLogic Server でサポートされていません。
- **ロール マッピング** - 指定されたリソースについて要求側に付与される一連のロールを取得します。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リソース (Web アプリケーションや Enterprise JavaBean (EJB) など) からの「アクセスできるか」という質問に答えることができます。

WebLogic セキュリティ プロバイダの機能の詳細については、第 3 章「セキュリティ 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。

デフォルト セキュリティ コンフィグレーションの詳細については、1-9 ページの「WebLogic Server のデフォルト セキュリティ コンフィグレーション」を参照してください。

カスタム セキュリティ プロバイダの作成については、『WebLogic Security サービスの開発』を参照してください。

ACL からセキュリティ ポリシーへ

WebLogic Server 6.x では、アクセス制御リスト (ACL) とパーミッションを使用して 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、ACL とパーミッションに代わってセキュリティ ポリシーを使用します。セキュリティ ポリシーは、WebLogic リソースへの「アクセス権は誰が持つか」という問いに答えます。セキュリティ ポリシーは、WebLogic リソースとユーザ、グループ、またはセキュリティ ロールの間の関連付けを定義するときに作成します。また、セキュリティ ポリシーに時間の制約を関連付けることもできます。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 ポリシーの作成は、多くのオプションを用いる複数の手順からなるプロセスです。このプロセスの詳細については、『WebLogic リソースのセキュリティ』を参照してください。WebLogic Server デプロイメントのセキュリティを完全にコンフィグレーションするには、このマニュアルと『WebLogic Security の管理』を合わせて利用してください。

互換性セキュリティでの ACL の使い方については、第 9 章「互換性セキュリティの使い方」を参照してください。

WebLogic リソース

WebLogic リソースは、権限のないアクセスから保護することができる WebLogic Server エンティティを表す構造化オブジェクトです。WebLogic Server では、以下のリソースが定義されます。

- WebLogic Server Administration Console や `weblogic.Admin` ツールなどの管理リソース

- エンタープライズ アプリケーションを表すアプリケーション リソース。このタイプのリソースには、EAR (エンタープライズ アプリケーション アーカイブ) ファイルと、EAR に含まれる EJB JAR ファイルのような個々のコンポーネントがある。
- Microsoft のフレームワークに従ってプログラム コンポーネント オブジェクトとして設計される Component Object Model (COM) リソース。このタイプのリソースには、BEA の双方向型 COM-Java (jCOM) ブリッジング ツールを介してアクセスする COM オブジェクトがある。
- コネクタとして設計されるエンタープライズ情報システム (EIS) リソース。Java アプリケーションと既存のエンタープライズ情報システムを統合できる。コネクタはリソース アダプタとも呼ばれる。
- EJB JAR ファイル、EJB JAR 内の個々の EJB、および EJB の個々のメソッドを含む Enterprise JavaBean (EJB) リソース
- 接続プールのグループ、個々の接続プール、およびマルチプールを含む Java DataBase Connectivity (JDBC)
- Java Naming and Directory Interface (JNDI) リソース
- Java Message Service (JMS) リソース
- WebLogic Server インスタンス (サーバ) に関連するサーバ リソース。このタイプのリソースには、サーバを起動、停止、ロック、またはロック解除する操作がある。
- Web アプリケーションに関連する URL リソース。このタイプのリソースには、WAR (Web アプリケーション アーカイブ) ファイル、または Web アプリケーションの個々のコンポーネント (サーブレットや JSP など) がある。

  注意： Web リソースは、このリリースの WebLogic Server では非推奨になっています。代わりに URL リソースを使用してください。
- Web ベースの分散アプリケーションのコンポーネント間で共有したり、コンポーネントとして使用したりできるサービスに関連する Web サービス リソース。このタイプのリソースには、Web サービス全体、または Web サービスの個々のコンポーネント (ステートレス セッション EJB、その EJB 内の特定のメソッド、`web-services.xml` ファイルを含む Web アプリケーションなど) がある。

デプロイメント記述子と WebLogic Server Administration Console

WebLogic Security サービスでは、デプロイメント記述子に定義された情報を利用してセキュリティ ロールを付与し、Web アプリケーションおよび EJB に対するセキュリティ ポリシーを定義できます。WebLogic Server が初めて起動すると、`weblogic.xml` および `weblogic-ejb-jar.xml` ファイルに格納されているセキュリティ ロールおよびセキュリティ ポリシー情報が、デフォルト セキュリティ レルムに定義されている認可プロバイダおよびロール マッピング プロバイダにロードされます。これらの情報の変更は、WebLogic Server Administration Console で行うことができます。

デプロイメント記述子内の情報を利用するには、セキュリティ レルム内の少なくとも 1 つの認可プロバイダおよびロール マッピング プロバイダがそれぞれ `DeployableAuthorizationProvider`、および `DeployableRoleProvider` セキュリティ サービス プロバイダ インタフェース (SSPI) を実装している必要があります。この SSPI を使用すると、プロバイダはデプロイメント記述子から情報を (検索ではなく) 格納できます。デフォルトでは、WebLogic 認可プロバイダおよびロール マッピング プロバイダがこの SSPI を実装します。

WebLogic Server Administration Console でデプロイメント記述子内のセキュリティ ロールおよびセキュリティ ポリシーを変更し、この情報を WebLogic Server Administration Console で引き続き変更する場合、セキュリティ レルムに対して属性を設定すると、WebLogic Server Administration Console で行った変更が WebLogic Server の再起動時にデプロイメント記述子内の古い情報によって上書きされなくなります。

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。

WebLogic Server のデフォルト セキュリティ コンフィグレーション

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、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。デフォルト セキュリティ コンフィグレーションでは、*myrealm* がデフォルト セキュリティ レルムとして設定され、WebLogic 裁決、認証、ID アサーション、認可、資格マッピング、およびロール マッピングの各プロバイダがセキュリティ プロバイダとして定義されています。デフォルト セキュリティ コンフィグレーションを使用するには、セキュリティ レルムのユーザ、グループ、およびセキュリティ ロールを定義し、ドメイン内の WebLogic リソースを保護するためのセキュリティ ポリシーを作成する必要があります。

WebLogic セキュリティ プロバイダの機能の詳細については、『WebLogic Security の紹介』を参照してください。これらの WebLogic セキュリティ プロバイダがセキュリティ要件を必ずしも完全に満たしていない場合には、それらを補うか、あるいは入れ替えることができます。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。

デフォルト セキュリティ コンフィグレーションでは自社の要件が満たされない場合、WebLogic セキュリティ プロバイダとカスタム セキュリティ プロバイダを自由に組み合わせて新しいセキュリティ レルムを作成し、そのセキュリティ レルムをデフォルト セキュリティ レルムとして設定できます。詳細については、第 2 章「デフォルト セキュリティ コンフィグレーションのカスタマイズ」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

セキュリティ機能は互いに関連しているので、セキュリティをコンフィグレーションする場合にどこから始めるべきか判断しにくいものです。実際、WebLogic Server デプロイメントのセキュリティをコンフィグレーションする場合には、同じ作業を繰り返すこともあります。コンフィグレーションの手順はいくつかありますが、次の手順を実行することをお勧めします。

1. 2-1 ページの「デフォルト セキュリティ コンフィグレーションのカスタマイズ」に目を通して、デフォルト セキュリティ コンフィグレーションを使用するかどうかを決定します。
   - デフォルト セキュリティ コンフィグレーションを使用する場合は、手順 3 に進みます。
   - デフォルト セキュリティ コンフィグレーションを使用しない場合は、手順 2 に進みます。
2. セキュリティ プロバイダのコンフィグレーションを変更する (WebLogic 認証プロバイダを使用する代わりに LDAP 認証プロバイダをコンフィグレーションする場合など) か、デフォルト セキュリティ レルムにカスタム セキュリティ プロバイダをコンフィグレーションします。この手順は省略可能です。デフォルトでは、WebLogic Server はデフォルト セキュリティ レルム (*myrealm*) の WebLogic セキュリティ プロバイダをコンフィグレーションします。デフォルト セキュリティ コンフィグレーションのカスタマイズが必要となる環境については、2-1 ページの「デフォルト セキュリティ コンフィグレーションをカスタマイズする理由」を参照してください。

   注意 : また、新しいセキュリティ レルムを作成し、そのセキュリティ レルムでセキュリティ プロバイダ (WebLogic またはカスタム) をコンフィグレーションし、そのセキュリティ レルムをデフォルト セキュリティ レルムとして設定することもできます。第 2 章「デフォルト セキュリティ コンフィグレーションのカスタマイズ」を参照してください。
3. 必要に応じて、組み込み LDAP サーバをコンフィグレーションします。デフォルトでは、組み込み LDAP サーバの属性がコンフィグレーションされます。ただし、これらの属性を変更して、自分の環境内で組み込み LDAP サーバの使用を最適化することができます。詳細については、第 5 章「組み込み LDAP サーバの管理」を参照してください。
4. ユーザ アカウントが適切に保護されていることを確認します。WebLogic Server には、ユーザ アカウントを保護するための属性セットが用意されています。デフォルトでは、これらの属性は最高のセキュリティ レベルに設定されています。ただし、WebLogic Server のデプロイ時に、ユーザ アカウントの制限を緩めなければならない場合があります。プロダクション環境に移行する前に、ユーザ アカウントに関する属性が最高の保護レベルに設定されて

いることを確認します。新しいセキュリティ レルムを作成する場合には、ユーザ ロックアウト属性を設定する必要があります。詳細については、第 7 章「ユーザ アカウントの保護」を参照してください。

5. セキュリティ ポリシーで WebLogic リソースを保護します。セキュリティ ポリシーの作成は、多くのオプションを用いる複数の手順からなるプロセスです。このプロセスの詳細については、『WebLogic リソースのセキュリティ』を参照してください。WebLogic Server デプロイメントのセキュリティを完全にコンフィグレーションするには、このマニュアルと『WebLogic Security の管理』を合わせて利用してください。
6. WebLogic Server に対して SSL を有効化します （この手順は省略可能です)。詳細については、第 6 章「SSL のコンフィグレーション」を参照してください。

また、以下のことを行うことができます。

- WebLogic Server の資格を EIS の資格 (Oracle データベースのユーザ名とパスワードなど) にマップする。第 4 章「エンタープライズ情報システムでのシングル サインオン」を参照してください。
- 接続フィルタをコンフィグレーションする。第 8 章「WebLogic ドメインのセキュリティのコンフィグレーション」を参照してください。
- WebLogic ドメイン間の相互運用性を有効化する。第 8 章「WebLogic ドメインのセキュリティのコンフィグレーション」を参照してください。

互換性セキュリティとは

互換性セキュリティとは、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です。互換性セキュリティでは、6.x のセキュリティ レルム、ユーザ、グループ、および ACL の管理、ユーザ アカウントの保護、レルム アダプタ監査プロバイダとレルム アダプタ認証プロバイダの ID アサーション プロバイダのコンフィグレーションを行うことができます。

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。互換性レルムのレルム アダプタ プロバイダ (監査、裁決、認可、および認証) を使用すると、6.x セキュリティ レルムの認証、認可、および監査サービスとの下

位互換性を保持できます。*CompatibilityRealm* 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。詳細については、第 9 章「互換性セキュリティの使い方」を参照してください。

互換性セキュリティで実行できるタスクの管理

互換性セキュリティでは WebLogic Server 6.x でサポートされる認証、認可、およびカスタム監査の実装にしかアクセスできないので、6.x のすべてのセキュリティ タスクが実行できるわけではありません。互換性セキュリティでは、以下のことを行います。

1. レルム アダプタ監査プロバイダをコンフィグレーションします。詳細については、9-5 ページの「レルム アダプタ監査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。
2. `weblogic.security.acl.CertAuthenticator` クラスの実装を使用できるように、レルム アダプタ認証プロバイダで ID アサーション プロバイダをコンフィグレーションします。詳細については、9-4 ページの「レルム アダプタ認証プロバイダでの ID アサーション 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。

   注意： レルム アダプタ裁決および認可プロバイダは、6.x の既存の `config.xml` ファイルの情報を利用して、*CompatibilityRealm* でデフォルトでコンフィグレーションされています。これらのプロバイダは、*CompatibilityRealm* でのみ使用できます。レルム アダプタ認証プロバイダも、*CompatibilityRealm* で自動的にコンフィグレーションされています。ただし、このプロバイダは、6.x セキュリティ レルムに格納されているユーザおよびグループへのアクセスを提供するために、他のレルムでもコンフィグレーションできます。詳細については、3-28 ページの「レルム アダプタ認証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。
3. `system` ユーザのパスワードを変更して、WebLogic Server のデプロイメントを保護します。

4. *CompatibilityRealm* のセキュリティ レルムを管理します。
5. *CompatibilityRealm* のセキュリティ レルムの追加ユーザを定義します。セキュリティ レルムにグループを実装して、ユーザをさらにまとめます。
6. WebLogic Server のデプロイメントのリソースの ACL およびパーミッションを管理します。
7. *CompatibilityRealm* に追加する WebLogic リソースに関するセキュリティ ロールおよびセキュリティ ポリシーを作成します。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。

SSL の使用、接続フィルタのコンフィグレーション、およびドメイン間の相互運用性の有効化も可能ですが、これらのタスクはこのリリースの WebLogic Server のセキュリティ機能を使用して実行します。詳細については、次を参照してください。

- 第 6 章「SSL のコンフィグレーション」
- 第 8 章「WebLogic ドメインのセキュリティのコンフィグレーション」

2 デフォルト セキュリティ コンフィグレーションのカスタマイズ

以下の節では、デフォルト セキュリティ レルムのカスタマイズおよび新しいセキュリティ レルムの作成について説明します。また、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方法についても説明します。

デフォルト セキュリティ コンフィグレーションをカスタマイズする理由

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、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。デフォルト セキュリティ コンフィグレーションでは、*myrealm* 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、WebLogic 裁決、認証、ID アサーション、認可、資格マッピング、およびロール マッピングの各プロバイダがセキュリティ プロバイダとして定義されています。デフォルト セキュリティ コンフィグレーションは、以下の場合にカスタマイズします。

- WebLogic セキュリティ プロバイダの 1 つをカスタム セキュリティ プロバイダに置き換える
- デフォルト セキュリティ レルムに別のセキュリティ プロバイダをコンフィグレーションする (たとえば、2 つの認証プロバイダを使用し、その 1 つで組み込み LDAP サーバを使用し、もう 1 つで既存のユーザおよびグループ ストアを使用する場合)
- 組み込み LDAP サーバ以外の LDAP サーバにアクセスする認証プロバイダを使用する
- WebLogic 認証プロバイダにユーザとグループを定義する代わりに、既存のユーザおよびグループ ストアを使用する
- デフォルト セキュリティ レルムに監査プロバイダを追加する
- デフォルト セキュリティ レルムに WebLogic キーストア プロバイダを追加する。この場合には、JKS キーストアを作成してから、そのキーストアにアクセスする WebLogic キーストア プロバイダをコンフィグレーションします。
- 互換性セキュリティからこのリリースの WebLogic Server セキュリティ プロバイダにアップグレードする
- セキュリティ プロバイダの属性のデフォルト設定を変更する。詳細については、第 3 章「セキュリティ 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。

デフォルト セキュリティ コンフィグレーションをカスタマイズする最も簡単な方法は、デフォルト セキュリティ レルム (*myrealm*) に目的のセキュリティ プロバイダを追加することです。セキュリティ レルム内の別のタイプのセキュリティ プロバイダのコンフィグレーションについては、第 3 章「セキュリティ 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。

しかし、新しいセキュリティ レルムを作成し、そのセキュリティ レルムでセキュリティ プロバイダをコンフィグレーションし、その新しいセキュリティ レルムをデフォルト セキュリティ レルムとして設定することもできます。セキュリティ コンフィグレーションをアップグレードする場合は、この方法をお勧めします。

以降の節では、新しいセキュリティ レルムを作成し、そのセキュリティ 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方法について説明します。

新しいセキュリティ レルムの作成

新しいセキュリティ レルムを作成するには、次の手順に従います。

1. [セキュリティ] ノードを展開します。
2. [レルム] ノードを展開します。

   WebLogic ドメインで使用可能なすべてのセキュリティ レルムが [レルム] テーブルに表示されます。
3. [新しい Realm のコンフィグレーション] リンクをクリックします。
4. [一般] タブの [名前] 属性に新しいセキュリティ レルムの名前を入力します。
5. パフォーマンスを制御するには、WebLogic Security サービスがセキュリティ チェックを実行する方法を指定します。

   [ロールとポリシーのチェック対象] の設定値は、以下のように指定します。

   - [デプロイメント記述子で保護された Web アプリケーションと EJB] オプションでは、WebLogic Security サービスが、関連のデプロイメント記述子 (`ejb-jar.xml`、`weblogic-ejb-jar.xml`、`web.xml`、および `weblogic.xml` ファイル) でセキュリティを指定されている URL (Web) および EJB リソースに対してのみセキュリティ チェックを実行することを指定する。これはデフォルト値です。
   - [すべての Web アプリケーションと EJB] オプションでは、WebLogic Security サービスがすべての URL (Web) および EJB リソースに対して、これらの WebLogic リソースのデプロイメント記述子にセキュリティ設定があるかどうかに関係なく、セキュリティ チェックを実行することを指定する。このオプションを選択すると、Web アプリケーションまたは EJB モジュールが再デプロイされたときに WebLogic Security サービスが実行する処理も指定する必要があります。詳細については、手順 6 を参照してください。

   注意： WebLogic Server 7.0 SP3 より前は、WebLogic Security サービスがセキュリティ チェックを実行する方法を、`fullyDelegateAuthorzation` コマンドライン引数を使用して指定する必要がありました。詳細については、『WebLogic リソースのセキュ

リティ』の「fullyDelegateAuthorization フラグについて」を参照してください。

6. URL (Web) および EJB リソースの保護に使用する方法を指定します。

   [デプロイメント記述子のセキュリティ動作] 設定の値は、以下のように指定します。

   - デプロイメント記述子 (`ejb-jar.xml`、`weblogic-ejb-jar.xml`、`web.xml`、および `weblogic.xml` ファイル) のみを使用して URL (Web) と EJB リソースを保護するには、[デプロイメント記述子からセキュリティを取得] オプションを選択する。『WebLogic Security プログラマーズ ガイド』の「Web アプリケーションでの宣言によるセキュリティの使用」および「EJB での宣言によるセキュリティの使用」を参照してください。
   - WebLogic Server Administration Console のみを使用して URL および EJB リソースを保護するには、[デプロイメント記述子内のセキュリティ データを無視] オプションを選択して、『WebLogic リソースの保護』で説明されている、Web アプリケーションおよび EJB を保護するための手順に従う。

   警告 : [デプロイメント記述子のセキュリティ動作] 設定値の切り替えは危険であり、セキュリティ コンフィグレーションが不適切になったり失われたりする場合があります。この設定の値を指定する前に、『WebLogic リソースのセキュリティ』の「URL リソースおよび EJB リソースを保護する方法」を必ず読んでください。

7. セキュリティ レルムの資格マッピング プロバイダが WebLogic Server Administration Console で作成された資格マップのみを使用することを指定するには、[デプロイの資格マッピングを無視] のデプロイメント記述子設定をチェックします。デフォルトでは、この属性はチェックされていません。つまり、資格マッピング プロバイダは、`weblogic-ra.xml` デプロイメント記述子ファイルに指定されている資格マップをロードします。

8. Web リソースは、このリリースの WebLogic Server では非推奨になっています。URL リソースの代わりに Web リソースを使用するカスタム認可プロバイダを新しいセキュリティ レルムでコンフィグレーションする場合は、[非推奨の Web リソースを使用] 属性を有効にします。この属性により、サーブレット コンテナの実行時の動作が、認可を処理するときに URL リソースではなく Web リソースを使用するように変更されます。

9. [作成] をクリックします。

10. セキュリティ レルムで必須のセキュリティ プロバイダをコンフィグレーションします。セキュリティ レルムが有効となるには、認証、認可、裁決、資格マッピング、およびロール マッピングの各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。コンフィグレーションしないと、新しいセキュリティ レルムをデフォルト セキュリティ レルムとして設定することができません。詳細については、第 3 章「セキュリティ 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。

11. 必要に応じて、ID アサーション、キーストア、および監査プロバイダを定義します。詳細については、第 3 章「セキュリティ 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。

12. 新しいセキュリティ レルムで WebLogic 認証、認可、資格マッピング、またはロール マッピングのいずれかのプロバイダをコンフィグレーションした場合は、組み込み LDAP サーバの属性のデフォルト設定を確認します。詳細については、第 5 章「組み込み LDAP サーバの管理」を参照してください。

13. セキュリティ ポリシーを使用して新しいセキュリティ レルム内の WebLogic リソースを保護します。セキュリティ ポリシーの作成は、多くのオプションを用いる複数の手順からなるプロセスです。このプロセスの詳細については、『WebLogic リソースのセキュリティ』を参照してください。WebLogic Server デプロイメントのセキュリティを完全にコンフィグレーションするには、このマニュアルと『WebLogic Security の管理』を合わせて利用してください。

14. 新しいセキュリティ レルム内のユーザ アカウントを保護します。詳細については、第 7 章「ユーザ アカウントの保護」を参照してください。

15. 新しいセキュリティ レルムを WebLogic ドメインのデフォルト セキュリティ レルムとして設定します。詳細については、2-6 ページの「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」を参照してください。

16. WebLogic Server を再起動します。

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

新しいセキュリティ レルムの属性を定義し、そのセキュリティ レルムのセキュリティ プロバイダをコンフィグレーションしたら、そのセキュリティ レルムをデフォルト (アクティブ) セキュリティ レルムとして設定します。

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には、次の手順に従います。

1. [ドメイン] ノード (examples など) を展開します。
2. [セキュリティ] タブをクリックします。
3. [一般] タブを選択します。

   [デフォルト レルム] 属性のプルダウン メニューに、WebLogic Server ドメインにコンフィグレーションされているセキュリティ レルムが表示されます。

   注意 : 新しいセキュリティ レルムを作成し、そのセキュリティ レルムに最小限の必須セキュリティ プロバイダをコンフィグレーションしていない場合、そのレルムはプルダウン メニューに表示されません。
4. デフォルト セキュリティ レルムに設定するセキュリティ レルムを選択します。
5. [適用] をクリックします。
6. WebLogic Server を再起動します。WebLogic Server を再起動しない場合、新しいレルムはデフォルト セキュリティ レルムに設定されません。

デフォルト セキュリティ レルムを設定したことを確認するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。

[レルム] テーブルに、WebLogic Server ドメインにコンフィグレーションされているすべてのレルムが表示されます。デフォルト (アクティブ) セキュリティ レルムには、[デフォルト レルム] 属性が `true` に設定されています。

セキュリティ レルムの削除

セキュリティ レルムを削除する場合、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、および資格マップ情報は組み込み LDAP サーバから削除されません。組み込み LDAP サーバから不要なエントリを削除するには、外部 LDAP ブラウザを使用します。詳細については、5-5 ページの「LDAP ブラウザによる組み込み LDAP サーバの内容の表示」を参照してください。

セキュリティ レルムを削除するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。

   [レルム] テーブルに、WebLogic ドメインにコンフィグレーションされているすべてのレルムが表示されます。

2. 削除するセキュリティ レルムの行にあるごみ箱アイコンをクリックします。

3. [はい] をクリックして次の質問に応答します。

   ドメイン コンフィグレーションから *`OldRealm`* を本当に削除しますか？

   セキュリティ レルムが削除されると、確認メッセージが表示されます。

以前のセキュリティ コンフィグレーションに戻す

新しいセキュリティ レルムやセキュリティ プロバイダのコンフィグレーションを間違えるのは珍しいことではありません。コンフィグレーションの間違いによって、サーバを起動できなくなったり、間違いを修正できなくなったりすることがあります。直前のセキュリティ コンフィグレーションに戻すには、次のコマンドライン引数を使用します。

```
-Dweblogic.safeCommoBoot=true
```

3 セキュリティ プロバイダのコンフィグレーション

以下の節では、WebLogic Server でサポートされているセキュリティ プロバイダとカスタム セキュリティ プロバイダをコンフィグレーションする方法について説明します。

- 3-45 ページの「WebLogic 資格マッピング プロバイダのコンフィグレーション」
- 3-47 ページの「WebLogic キーストア プロバイダのコンフィグレーション」
- 3-47 ページの「WebLogic ロール マッピング プロバイダのコンフィグレーション」
- 3-48 ページの「カスタム セキュリティ プロバイダのコンフィグレーション」
- 3-49 ページの「セキュリティ プロバイダの削除」

注意： レルム アダプタ監査、裁決、および認可の各プロバイダは、互換性セキュリティを使用した場合のみ利用できます。これらのプロバイダの詳細については、第 9 章「互換性セキュリティの使い方」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が必要になる場合

セキュリティ プロバイダのコンフィグレーション作業は、デフォルトでほとんど完了します。ただし、以下の場合には、セキュリティ プロバイダの属性をコンフィグレーションする必要があります。

- WebLogic ID アサーション プロバイダを使用する前に、アクティブ トークン タイプを定義する。詳細については、3-30 ページの「WebLogic ID アサーション 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。
- トークンをセキュリティ レルム内のユーザにマップするためのユーザ名マッパーを使用するために、WebLogic 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。詳細については、3-32 ページの「WebLogic ID アサーション プロバイダでのユーザ名マッパーの使用」を参照してください。
- デフォルト (アクティブ) セキュリティ レルムで監査を行うために、WebLogic 監査プロバイダまたはカスタム監査プロバイダをコンフィグレーションする。詳細については、3-5 ページの「WebLogic 監査プロバイダのコンフィグレーション」または 3-48 ページの「カスタム セキュリティ 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。

- プライベート キーおよび信頼性のある認証局をフラット ファイルではなくキーストアで保護するために、WebLogic キーストア プロバイダをコンフィグレーションする。詳細については、3-47 ページの「WebLogic キーストア 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。

  注意： カスタム キーストア プロバイダはこのリリースの WebLogic Server でサポートされていません。

- 組み込み LDAP サーバ以外の LDAP サーバを使用するために、LDAP 認証プロバイダのいずれかをコンフィグレーションする。LDAP 認証プロバイダは、WebLogic 認証プロバイダの代わりとして使用することも、WebLogic 認証プロバイダと併用することもできます。詳細については、3-11 ページの「LDAP 認証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。

- 6.x セキュリティ レルム (6.x Windows NT、UNIX、RDBMS セキュリティ レルムや 6.x カスタム セキュリティ レルムなど) 内の既存のユーザおよびグループを使用するために、レルム アダプタ認証プロバイダをコンフィグレーションする。レルム アダプタ認証プロバイダは、WebLogic 認証プロバイダの代わりとして使用することも、WebLogic 認証プロバイダと併用することもできます。詳細については、3-28 ページの「レルム アダプタ認証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。

  注意： このリリースの WebLogic Server には、6.x Windows NT、UNIX、RDBMS セキュリティ レルムに相当するものはありません。そのため、これらのセキュリティ レルム内のユーザおよびグループにアクセスする場合は、レルム アダプタ認証プロバイダを使用することをお勧めします。

- 新しいセキュリティ レルムを作成する場合に、そのレルムのセキュリティ プロバイダをコンフィグレーションする。デフォルト レルム (*myrealm*) を使用する場合、WebLogic 裁決、認証、ID アサーション、認可、資格マッピング、およびロール マッピングの各プロバイダはすでにコンフィグレーションされています。詳細については、2-3 ページの「新しいセキュリティ レルムの作成」を参照してください。

- カスタム セキュリティ プロバイダをセキュリティ レルムに追加したり、WebLogic セキュリティ プロバイダをカスタム セキュリティ プロバイダと置き換えたりする場合に、カスタム セキュリティ プロバイダの属性をコンフィグレーションする。カスタム セキュリティ プロバイダを作成する場合、WebLogic Server Administration Console を使用してコンフィグレーション可能な属性を実装することができます。ただし、これらの属性は実装に固有なので、このマニュアルでは取り上げていません。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 プロバイダ用のコンソール拡張の記述」を参照してください。

ここからは、各セキュリティ プロバイダに設定可能な属性について説明します。

WebLogic 裁決プロバイダのコンフィグレーション

セキュリティ レルムに複数の認可プロバイダがコンフィグレーションされる場合、特定のリソースに「アクセスできるか」という質問に対して、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回答が一致しない場合にどうするかを決定するのが、裁決プロバイダの主な役割です。裁決プロバイダは、各認可プロバイダの回答に重みを割り当てることによって認可の衝突を解決し、最終決定を返します。

各セキュリティ レルムには、裁決プロバイダがコンフィグレーションされている必要があります。セキュリティ レルムでは、WebLogic 裁決プロバイダまたはカスタム裁決プロバイダのいずれかを使用できます。この節では、WebLogic 裁決プロバイダをコンフィグレーションする方法について説明します。カスタム セキュリティ プロバイダ (カスタム裁決プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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 裁決プロバイダをコンフィグレーションするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ] ノードを展開します。
4. [裁決] をクリックします。

   [裁決] テーブルに、コンフィグレーションするレルムのデフォルト裁決プロバイダの名前が表示されます。

5. [新しい Default Adjudicator のコンフィグレーション] リンクをクリックします。

   既存のセキュリティ レルムで作業する場合は、[新しい Default Adjudicator と置換 ...] リンクをクリックします。

6. [一般] タブで、[完全一致の許可が必要] 属性を設定します (省略可能)。

   [完全一致の許可が必要] 属性によって、WebLogic 裁決プロバイダが認可プロバイダからの `PERMIT` および `ABSTAIN` 票をどのように処理するかが決まります。

   - この属性を有効にした場合、すべての認可プロバイダが `PERMIT` 票を投じなければ、裁決プロバイダは `true` を投じません。デフォルトでは、この属性は有効化されています。
   - この属性を無効にした場合、`ABSTAIN` 票は `PERMIT` 票としてカウントされます。[完全一致の許可が必要] 属性を無効にするには、対応するチェックボックスをクリックします。

7. [適用] をクリックして変更を保存します。

8. WebLogic Server を再起動します。

WebLogic 監査プロバイダのコンフィグレーション

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、監査プロバイダはコンピュータのアクティビティの電子的な記録を生成します。監査プロバイダのコンフィグレーションは任意です。デフォルト セキュリティ レルム (*myrealm*) には監査プロバイダはコンフィグレーションされていません。

セキュリティ レルムでは、WebLogic 監査プロバイダまたはカスタム監査プロバイダのいずれかを使用できます。この節では、WebLogic 監査プロバイダをコンフィグレーションする方法について説明します。カスタム セキュリティ プロバ

イダ (カスタム監査プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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。

警告 : 監査プロバイダを使用すると、数個のイベントのログが記録される場合でも WebLogic Server のパフォーマンスに影響が及びます。

WebLogic 監査プロバイダをコンフィグレーションするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ] ノードを展開します。
4. [監査] をクリックします。

   [監査] テーブルに、コンフィグレーションするレルムのデフォルト監査プロバイダの名前が表示されます。
5. [新しい Default Auditor のコンフィグレーション] リンクをクリックします。

   [一般] タブが表示されます。
6. WebLogic Server デプロイメントに適した重大度を選択します。

   監査プロバイダは、重大度属性で指定されたイベント レベルに基づいて特定のセキュリティ イベントを監査します。監査は、以下のレベルのセキュリティ イベントが発生したときに実行されます。

   - `INFORMATION`
   - `WARNING`
   - `ERROR`
   - `SUCCESS`
   - `FAILURE`
7. [作成] をクリックして変更を保存します。
8. WebLogic Server を再起動します。

WebLogic 監査プロバイダによって生成される監査イベントは、*`WL_HOME`*`\`*`yourdomain`*`\`*`yourserver`*`\DefaultAuditRecorder.log` に保存されます。監査プロバイダはセキュリティ レルムごとにコンフィグレーションされま

すが、各サーバはサーバ ディレクトリに存在する独自のログ ファイルに監査データを書き込みます。WebLogic 監査プロバイダは、以下のイベントのログを記録します。

表 3-1 WebLogic 監査プロバイダのイベント

監査イベント	意味
AUTHENTICATE	単純認証 (ユーザ名とパスワード) が発生した。
ASSERTIDENTITY	境界認証 (トークン ベース) が発生した。
USERLOCKED	無効なログイン試行によってユーザ アカウントがロックされた。
USERUNLOCKED	ユーザ アカウントのロックがクリアされた。
USERLOCKOUTEXPIRED	ユーザ アカウントのロックの期限が切れた。

認証プロバイダの選択

認証とは、ユーザまたはシステム プロセスの身元を証明または確認するプロセスのことです。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。

WebLogic Server のセキュリティ アーキテクチャでサポートされているのは、証明書に基づく認証 (WebLogic Server を直接用いる)、HTTP 証明書に基づく認証 (外部の Web サーバを介して行う)、境界に基づく認証 (Web サーバ、ファイアウォール、VPN)、および複数のセキュリティ トークン タイプ / プロトコルに基づく認証です。

認証は、認証プロバイダによって実行されます。WebLogic Server には、以下のタイプの認証プロバイダが用意されています。

- WebLogic 認証プロバイダのユーザおよびグループ メンバシップ情報は、組み込み LDAP サーバに格納されます。

- LDAP 認証プロバイダ。このプロバイダでは、外部 LDAP ストアにアクセスできます。WebLogic Server には、Open LDAP、Netscape iPlanet、Microsoft Active Directory、および Novell NDS ストアにアクセスできる LDAP 認証プロバイダが用意されています。LDAP 認証プロバイダは、他の LDAP ストアにアクセスする場合にも使用できます。ただし、事前に定義された LDAP プロバイダを選択し、カスタマイズする必要があります。
- レルム アダプタ認証プロバイダ。このプロバイダでは、6.x セキュリティ レルムに格納されているユーザおよびグループ情報にアクセスできます。
- WebLogic ID アサーション プロバイダ。このプロバイダでは、X.509 および IIOP-CSIv2 トークンが検証され、ユーザ名マッパーを使用してトークンが WebLogic Server セキュリティ レルム内のユーザ名にマップされます。

また、以下のプロバイダを使用することもできます。

- カスタム認証プロバイダ。このプロバイダは、さまざまなタイプの認証技術を備えています。
- カスタム ID アサーション プロバイダ。このプロバイダは、さまざまなタイプのトークンをサポートしています。

注意： WebLogic Server Administration Console は、WebLogic 認証プロバイダをデフォルト認証プロバイダとして、WebLogic ID アサーション プロバイダをデフォルト ID アサーション プロバイダとしてそれぞれ参照します。

各セキュリティ レルムには、少なくとも 1 つの認証プロバイダがコンフィグレーションされている必要があります。WebLogic Security フレームワークは、さまざまな要素から成る認証向けに、複数の認証プロバイダ (したがって、複数の LoginModule) をサポートするように設計されています。このため、セキュリティ レルムでは複数のタイプの認証プロバイダのほかに複数の認証プロバイダを使用できます。たとえば、網膜スキャンに基づく認証とユーザ名 / パスワードに基づく認証の両方を使用してシステムにアクセスする場合、2 つの認証プロバイダをコンフィグレーションします。

複数の認証プロバイダをコンフィグレーションする方法は、認証プロセスの全体的な結果に影響します。認証プロバイダは、コンフィグレーションされた順序で呼び出されます。したがって、認証プロバイダのコンフィグレーションには注意が必要です。認証プロバイダ間のログイン依存関係を設定し、プロバイダ間のシングル サイン オンを可能にするには、JAAS 制御フラグ属性を使用します。詳細については、3-10 ページの「JAAS 制御フラグ属性の設定」を参照してください。

認証プロバイダのコンフィグレーション 主な手順

認証プロバイダをコンフィグレーションするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ | 認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
4. 認証プロバイダまたは ID アサーション プロバイダ、あるいはその両方を選択します。
   - [新しい iPlanet Authenticator のコンフィグレーション]
   - [新しい Realm Adapter Authenticator のコンフィグレーション]
   - [新しい Active Directory Authenticator のコンフィグレーション]
   - [新しい Default Authenticator のコンフィグレーション]
   - [新しい Default Identity Asserter のコンフィグレーション]
   - [新しい OpenLDAP Authenticator のコンフィグレーション]
   - [新しい Novell Authenticator のコンフィグレーション]
5. 該当する節に移動して、認証プロバイダまたは ID アサーション プロバイダ、あるいはその両方をコンフィグレーションします。
   - 3-11 ページの「LDAP 認証プロバイダのコンフィグレーション」
   - 3-26 ページの「WebLogic 認証プロバイダのコンフィグレーション」
   - 3-28 ページの「レルム アダプタ認証プロバイダのコンフィグレーション」
   - 3-30 ページの「WebLogic ID アサーション プロバイダのコンフィグレーション」

- 3-33 ページの「LDAP X509 ID アサーション プロバイダのコンフィグレーション」

6. これらの手順を繰り返して、追加の認証プロバイダまたは ID アサーション プロバイダ、あるいはその両方をコンフィグレーションします。
7. 複数の認証プロバイダをコンフィグレーションする場合は、JAAS 制御フラグを設定します。詳細については、3-10 ページの「JAAS 制御フラグ属性の設定」を参照してください。
8. 認証プロバイダおよび ID アサーション プロバイダのコンフィグレーションが済んだら、WebLogic Server を再起動します。

JAAS 制御フラグ属性の設定

複数の認証プロバイダをコンフィグレーションする場合は、[認証プロバイダ | 一般] タブの JAAS 制御フラグ属性を使用して、ログイン シーケンスにおける認証プロバイダの使用方法を制御します。

JAAS 制御フラグの値の定義は次のとおりです。

- REQUIRED － 認証プロバイダは必ず呼び出され、ユーザは認証テストに合格しなければなりません。
- SUFFICIENT － ユーザが認証プロバイダの認証テストに合格した場合、そのユーザは認証を受けたことになるので、JAAS 制御フラグが REQUIRED に設定された認証プロバイダ以外の認証プロバイダは実行されません。
- REQUISITE － ユーザが認証プロバイダの認証テストに合格した場合、JAAS 制御フラグが REQUIRED に設定された認証プロバイダ以外の認証プロバイダが実行されますが、失敗する可能性があります。
- OPTIONAL － ユーザはこれらの認証プロバイダの認証テストをパスすることができます。ただし、セキュリティ レルム内のすべての認証プロバイダが JAAS 制御フラグを OPTIONAL に設定されている場合、ユーザはいずれかのプロバイダの認証テストに合格しなければなりません。

認証プロバイダが既存のセキュリティ レルムに追加されている場合、[制御フラグ] 属性は OPTIONAL にデフォルト設定されます。認証プロバイダが認証シーケンスで適切に動作するように、必要に応じて [制御フラグ] の設定を変更してください。

注意： WebLogic Server Administration Console は実際にはセキュリティ プロバイダ作成時に JAAS 制御フラグを OPTIONAL に設定します。セキュリティ プロバイダの MBean は実際にはデフォルトで REQUIRED に設定されます。

LDAP 認証プロバイダのコンフィグレーション

WebLogic Server は、特定の LDAP サーバをサポートおよび証明しません。LDAP v2 または v3 準拠のすべての LDAP サーバは WebLogic Server と正常に連係します。以下の LDAP ディレクトリ サーバについてはテスト済みです。

- Netscape iPlanet バージョン 4.1.3
- Windows 2000 に付属の Active Directory
- Open LDAP バージョン 2.0.7
- Novell NDS バージョン 8.5.1

詳細については、次を参照してください。

- 3-13 ページの「LDAP サーバとキャッシング情報の設定」
- 3-17 ページの「LDAP ディレクトリでのユーザの格納」
- 3-20 ページの「LDAP ディレクトリでのグループの格納」
- 3-21 ページの「LDAP ディレクトリでのグループの格納」

LDAP 認証プロバイダを使用するための要件

LDAP 認証プロバイダがセキュリティ レルムにコンフィグレーションされている唯一の認証プロバイダの場合、Admin ロールを持たなければ WebLogic Server を起動し、LDAP ディレクトリ内のユーザまたはグループを使用できません。LDAP ディレクトリで、次のいずれかを行います。

- WebLogic Server では、デフォルトによって Admin ロールに Administrators グループが含まれています。LDAP ディレクトリに Administrators グループを作成します。WebLogic Server を起動する LDAP ユーザがこのグループに含まれていることを確認します。

  Active Directory LDAP ディレクトリには、Administrators というデフォルト グループが存在します。WebLogic Server を起動するユーザを Administrators グループに追加し、[グループ ベース DN] 属性を定義して Administrators グループが見つかるようにします。

- LDAP ディレクトリに Administrators グループを作成しない場合 (LDAP ディレクトリが別の目的で Administrators グループを使用する場合など)、LDAP ディレクトリに新しいグループを作成 (または既存のグループを使用) して、WebLogic Server を起動するユーザをそのグループに追加します。次に、WebLogic Server Administration Console で、そのグループに Admin ロールを割り当てます。

LDAP 認証プロバイダのコンフィグレーション

LDAP 認証プロバイダをコンフィグレーションするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ | 認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
4. LDAP 認証プロバイダを選択します。

- [新しい iPlanet Authenticator のコンフィグレーション]
- [新しい Active Directory Authenticator のコンフィグレーション]
- [新しい OpenLDAP Authenticator のコンフィグレーション]
- [新しい Novell Authenticator のコンフィグレーション]

5. 複数の認証プロバイダを使用する場合、[一般] タブの [制御フラグ] 属性の値を定義します。[制御フラグ] 属性で、LDAP 認証プロバイダをどのように他の LDAP 認証プロバイダと共に使用するかを指定します。詳細については、3-11 ページの「LDAP 認証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。
6. [適用] をクリックして変更を保存します。
7. 3-13 ページの「LDAP サーバとキャッシング情報の設定」に進みます。

LDAP サーバとキャッシング情報の設定

LDAP サーバをコンフィグレーションするには、次の手順に従います。

1. 使用する LDAP 認証プロバイダの [コンフィグレーション] タブの下にある LDAP タブをクリックします。

   たとえば、iPlanet の [コンフィグレーション] タブの [iPlanet LDAP] タブをクリックします。

2. LDAP タブの属性値を定義して、WebLogic Server と LDAP サーバの通信を有効にします。

   次の表では、LDAP タブで設定する属性について説明します。

表 3-2 LDAP タブの属性

属性	説明
[ホスト]	LDAP サーバが稼働するコンピュータのホスト名。

表 3-2 LDAP タブの属性

属性	説明
[ポート]	LDAP サーバがリスンするポートの番号。SSL プロトコルを使用して WebLogic Server を LDAP サーバに接続する場合は、LDAP サーバの SSL ポートを指定する。
[SSL を有効化]	LDAP サーバと WebLogic Server との通信を保護するために SSL プロトコルを使用できるようにするためのオプション。LDAP サーバが SSL プロトコルを使用するようにコンフィグレーションされていない場合は、この属性を無効化する。
[プリンシパル]	WebLogic Server が LDAP サーバとの接続に使用する LDAP ユーザの識別名 (Distinguished Name: DN)。一般に、このユーザは LDAP ディレクトリ サーバのシステム管理者である。パスワードを変更する場合、この属性をシステム管理者に設定する必要がある。
[資格]	[プリンシパル] 属性で定義された LDAP ユーザを認証するパスワード。
[キャッシュを有効化]	Open LDAP サーバでのデータ キャッシュの使用を有効にする。
[キャッシュ サイズ]	キャッシュのルックアップの最大サイズ。デフォルトは 32 KB。
[キャッシュ TTL]	LDAP ルックアップの結果を保持する秒数。

3. 変更を保存するには、[適用] をクリックします。
4. [詳細] タブをクリックして、LDAP サーバの動作を制御するその他の属性をコンフィグレーションします。

次の表では、[詳細] タブで設定する属性について説明します。

表 3-3 [詳細] タブの属性

属性	説明
[グループ メンバシップ 検索]	グループ検索の対象範囲が制限されるのかどうかを指定する。この属性では、ネストされたグループでどのくらいの深さまで検索するのかを指定する。ネストされたグループ階層の最初のレベルのみを使用するコンフィグレーションの場合は、この属性で検索をグループの最初のレベルに制限するとパフォーマンスを向上させることができる。 ■ 検索の制限を指定した場合は、[最大グループ メンバシップ検索レベル] 属性を指定する必要がある ■ 検索を制限しない場合は、[最大グループ メンバシップ検索レベル] 属性は無視される

表 3-3 [詳細] タブの属性

属性	説明
[最大グループ メンバシップ検索レベル]	[グループ メンバシップ検索] 属性が指定された場合に、グループ メンバーシップ検索の深さを指定する。指定できる値は次のとおり。 ■ 0— 直接のメンバー グループのみ検索される。つまり、グループ A でメンバーシップを検索するときに、グループ A の直接のメンバーのみが検索される。グループ B がグループ A のメンバーである場合、そのメンバーは検索の対象にならない。 ■ 任意の正の数値 — 検索されるレベル数を示す。たとえば、この属性を 1 に設定すると、グループ A のメンバーシップの検索ではグループ A の直接のメンバーが返される。グループ B がグループ A のメンバーである場合は、グループ B のメンバーも検索の対象になる。ただし、グループ C がグループ B のメンバーである場合、グループ C のメンバーは検索の対象にならない。
[照会先に従う]	LDAP 認証プロバイダ内のユーザまたはグループの検索で、他の LDAP サーバまたは LDAP ディレクトリ内のブランチへの照会に従うことを指定する。デフォルトで、この属性は有効。
[照会先に匿名でバインドする]	デフォルトで、LDAP 認証プロバイダは、検索中に照会に従うときに同じ DN およびパスワードを使用して LDAP サーバに接続する。匿名ユーザとしてアクセスする場合は、この属性を有効にする。詳細については、LDAP システム管理者に問い合わせる。

表 3-3 [詳細] タブの属性

属性	説明
[結果までのタイム リミット]	LDAP サーバがタイムアウトまで結果を待機するミリ秒数。この属性を 0 に設定した場合、タイム リミットは無制限になる。デフォルトでは 0。
[接続タイムアウト]	LDAP サーバへの接続の確立まで待機する最大秒数。この属性を 0 に設定した場合、タイム リミットは無制限になる。デフォルトでは 0。
[並列接続遅延]	複数の LDAP サーバに同時に接続を試行する場合の遅延秒数。この属性を 0 に設定した場合、接続は連続して試行される。接続はリスト内の最初のサーバに対して試行される。ホストへの接続が失敗した場合にのみ、リスト内の次のエントリに対して接続が試行される。この属性を設定しない状態で、LDAP サーバが利用できない場合、アプリケーションが長時間ブロックされることがある。この属性が 0 より大きい場合、指定された時間が経過してから、別の接続が開始される。

5. 3-17 ページの「LDAP ディレクトリでのユーザの格納」に進みます。

デプロイメントをよりセキュアにするために、SSL プロトコルを使用して LDAP サーバと WebLogic Server 間の通信を保護することをお勧めします。詳細については、6-1 ページの「SSL のコンフィグレーション」を参照してください。

LDAP ディレクトリでのユーザの格納

LDAP ディレクトリにどのようにユーザを格納するかを指定するには、次の手順に従います。

1. 選択した LDAP サーバの [コンフィグレーション] タブの [ユーザ] タブをクリックします。

   たとえば、iPlanet の [コンフィグレーション] タブの下にある [ユーザ] タブをクリックします。

2. [ユーザ] タブの属性値を設定して、ユーザを LDAP ディレクトリに格納する方法を定義します。

   次の表では、[ユーザ] タブで設定する属性について説明します。

表 3-4 [ユーザ] タブの属性

属性	説明
[ユーザ オブジェクト クラス]	ユーザを格納する LDAP オブジェクト クラス。
[ユーザ名属性]	ユーザ名を指定する LDAP ユーザ オブジェクトの属性。
[ユーザ動的グループ DN 属性]	このユーザが所属する動的グループの DN を指定する LDAP ユーザ オブジェクトの属性。 Active Directory、Open LDAP、および Novell NDS ディレクトリ サーバでは動的グループはサポートされないため、これらのサーバに対してはこの属性を設定しないこと。 この属性が存在しない場合、WebLogic Server は [動的グループ オブジェクト クラス] 属性を検索して、このユーザが属するグループを調べる。 グループに他のグループが含まれる場合、WebLogic Server はそのグループの派生グループの URL を評価する。

表 3-4 [ユーザ] タブの属性

属性	説明
[ユーザ ベース DN]	ユーザが含まれる LDAP ディレクトリのツリーのベース DN。 WebLogic Server ユーザをディレクトリの複数のルートの下に格納する場合は、それらのルートを `user.dn.1`、`user.dn.2` などのように指定できる。LDAP 認証プロバイダは、一致するユーザが見つかるまでそれらの各ルートの下を検索する。 **注意：** `user.dn.n` エントリごとに、`user.filter.n` を指定する必要がある。
[ユーザ検索スコープ]	ユーザを検索するための LDAP ディレクトリ ツリーの深さを指定する。 有効値は [`subtree`] と [`onelevel`]。
[名前フィルタからのユーザ]	特定のユーザ名を検索するための LDAP 検索フィルタ。 検索フィルタが指定されていない場合 (null または空の場合)、ユーザ スキーマに基づいてデフォルトの検索フィルタが作成される。 LDAP 検索フィルタの作成については、お使いの LDAP サーバのマニュアルを参照。
[ユーザすべてのフィルタ]	ベース DN の下のすべてのユーザを検索するための LDAP 検索フィルタ。検索フィルタが指定されていない場合 (null または空の場合)、ユーザ スキーマに基づいてデフォルトの検索フィルタが作成される。 LDAP 検索フィルタの作成については、お使いの LDAP サーバのマニュアルを参照。

3. 変更を保存するには、[適用] をクリックします。
4. 3-20 ページの「LDAP ディレクトリでのグループの格納」に進みます。

LDAP ディレクトリでのグループの格納

LDAP ディレクトリにどのようにグループを格納するかを定義するには、次の手順に従います。

1. [コンフィグレーション] タブの [グループ] タブをクリックします。

   たとえば、iPlanet の [コンフィグレーション] タブの下にある [グループ] タブをクリックします。

2. [グループ] タブの属性値を設定して、グループを LDAP ディレクトリに格納する方法を定義します。

   次の表では、[グループ] タブで設定する属性について説明します。

表 3-5 [グループ] タブの属性

属性	説明
[グループ ベース DN]	グループが含まれる LDAP ディレクトリのツリーのベース DN。
[グループ検索スコープ]	グループを検索するための LDAP ディレクトリ ツリーの深さを指定する。 有効値は [subtree] と [onelevel]。
[名前フィルタからのグループ]	特定のグループ名を検索するための LDAP 検索フィルタ。 LDAP 検索フィルタの作成については、お使いの LDAP サーバのマニュアルを参照。

表 3-5 [グループ] タブの属性

属性	説明
[グループすべてのフィルタ]	ベース グループ DN の下のすべてのグループを検索するための LDAP 検索フィルタ。この属性が指定されていない場合 (null または空の場合)、グループ スキーマに基づいてデフォルトの検索フィルタが作成される。 LDAP 検索フィルタの作成については、お使いの LDAP サーバのマニュアルを参照。
[静的グループ オブジェクト クラス]	静的グループを格納する LDAP オブジェクト クラスの名前。
[静的グループ名属性]	グループ名を指定する静的 LDAP グループ オブジェクトの属性。

3. 変更を保存するには、[適用] をクリックします。
4. 3-21 ページの「LDAP ディレクトリでのグループの格納」に進みます。

LDAP ディレクトリでのグループの格納

注意： iPlanet 認証プロバイダは、動的グループをサポートしています。動的グループを使用するには [動的グループ オブジェクト クラス] と [動的グループ名属性]、および [動的メンバ URL 属性] を設定します。

LDAP ディレクトリにどのようにグループ メンバーを格納するかを定義するには、次の手順に従います。

1. [コンフィグレーション] タブの [メンバシップ] タブをクリックします。

   たとえば、iPlanet の [コンフィグレーション] タブの下にある [メンバシップ] タブをクリックします。

2. [メンバシップ] タブの属性値を設定して、グループ メンバーを LDAP ディレクトリに格納する方法を定義します。

   次の表では、[メンバシップ] タブで設定する属性について説明します。

表 3-6 [メンバシップ] タブの属性

属性	定義
[静的メンバ DN 属性]	グループ内のメンバーの DN を指定する LDAP グループ オブジェクトの属性。
[メンバ DN フィルタからの静的グループ DN]	グループ メンバーの DN が指定されている場合、そのメンバーを含む静的 LDAP グループの DN を返す LDAP 検索フィルタ。 この属性が指定されていない場合 (null または空の場合)、グループ スキーマに基づいてデフォルトの検索フィルタが作成される。 LDAP 検索フィルタの作成については、お使いの LDAP サーバのマニュアルを参照。
[動的グループ オブジェクト クラス]	動的グループを格納する LDAP オブジェクト クラスの名前。 Active Directory、Open LDAP、および Novell NDS ディレクトリ サーバでは動的グループは現在サポートされないため、これらのサーバを使用する場合はこの属性を設定しないこと。
[動的グループ名属性]	グループ名を指定する動的 LDAP グループ オブジェクトの属性。 Active Directory、Open LDAP、および Novell NDS ディレクトリ サーバでは動的グループは現在サポートされないため、これらのサーバを使用する場合はこの属性を設定しないこと。

表 3-6 [メンバシップ] タブの属性 (続き)

属性	定義
[動的メンバ URL 属性]	動的グループのメンバーの URL を指定する動的 LDAP グループ オブジェクトの属性。 **注意 :** この属性の値が文字列で予約値が含まれている場合、不正な URL 例外が送出される。この例外が送出されないようにするには、この属性で :、,、?、および / を使用しない。 Active Directory、Open LDAP、および Novell NDS ディレクトリ サーバでは動的グループは現在サポートされないため、これらのサーバを使用する場合はこの属性を設定しないこと。

3. 変更を保存するには、[適用] をクリックします。
4. 必要な場合、追加の認証プロバイダと ID アサーション プロバイダをコンフィグレーションします。
5. WebLogic Server を再起動します。

LDAP 認証プロバイダのフェイルオーバのコンフィグレーション

WebLogic Server 7.0 SP2 以降では、複数の LDAP サーバを備える外部 LDAP プロバイダをコンフィグレーションし、LDAP サーバの 1 つが利用できない場合にはフェイルオーバを有効にすることができます。

LDAP 認証プロバイダ用にコンフィグレーションされた LDAP サーバのフェイルオーバをコンフィグレーションするには、次の手順に従います。

1. フェイルオーバをコンフィグレーションする LDAP 認証プロバイダの [コンフィグレーション] タブの下にある LDAP タブをクリックします。

   たとえば、iPlanet の [コンフィグレーション] タブの [iPlanet LDAP] タブをクリックします。

2. LDAP タブをクリックします。
3. LDAP タブ上の [Host] 属性で複数の LDAP サーバ名を指定します。属性には、スペースで区切られたホスト名リストを含める必要があります。各ホスト名の末尾にはコロンとポート番号を含めることができます。次に例を示します。

   ```
   directory.knowledge.com:1050 people.catalog.com 199.254.1.2
   ```

4. [適用] をクリックします。
5. [詳細] タブをクリックします。
6. [並列接続遅延] 属性を設定します。

   [並列接続遅延] 属性では、複数のサーバに同時接続しようとしているときの遅延の秒数を指定します。接続はリスト内の最初のサーバに対して試行されます。ホストへの接続が失敗した場合にのみ、リスト内の次のエントリに対して接続が試行されます。この設定により、ホストが停止している場合に、アプリケーションにおいて非常に長い時間にわたりブロックが行われる可能性があります。属性が 0 より大きい値に設定されていると、指定した遅延秒数の経過後に、別の接続設定スレッドが開始されます。属性を 0 に設定した場合、接続は連続して試行されます。

7. [接続タイムアウト] 属性を設定します。

   [接続タイムアウト] 属性では、LDAP サーバへの接続が確立されるまでに待機する最大秒数を指定します。属性を 0 に設定した場合には最長時間の制限はなく、WebLogic Server は TCP/IP レイヤがタイムアウトして接続障害を返すまで待機します。この属性は、TCP/IP のコンフィグレーションに応じて、60 秒を超える値に設定できます。

8. [適用] をクリックします。
9. WebLogic Server を再起動します。

次の例では、LDAP 属性が LDAP フェイルオーバ用に設定されている場合に発生する使い方のシナリオを提示します。

例 1

LDAP 属性が次のように設定されている場合。

LDAP 属性	値
[ホスト]	directory.knowledge.com:1050 people.catalog.com 199.254.1.2 LDAP サーバの動作状況は次のとおり。 directory.knowledge.com:1050 は停止している people.catalog.com は起動している 199.254.1.2 は起動している
[並列接続遅延]	0
[接続タイムアウト]	10

このシナリオでは、WebLogic Server は directory.knowledge.com に接続しようとします。10 秒後、接続の試行はタイムアウトし、WebLogic Server は [ホスト] 属性で指定されている次のホスト (people.catalog.com) への接続を試行します。その後は、WebLogic Server はこの接続のための LDAP サーバとして people.catalog.com を使用します。

例 2

LDAP 属性が次のように設定されている場合。

LDAP 属性	値
[ホスト]	directory.knowledge.com:1050 people.catalog.com 199.254.1.2 LDAP サーバの動作状況は次のとおり。 directory.knowledge.com:1050 は停止している people.catalog.com は起動している 199.254.1.2 は起動している
[並列接続遅延]	1
[接続タイムアウト]	10

このシナリオでは、WebLogic Server は directory.knowledge.com に接続しようとします。10 秒後、接続の試行はタイムアウトし、WebLogic Server は [ホスト] 属性で指定されている次のホスト (people.catalog.com) を試行し、並行して people.catalog.com へ接続しようとします。その後は、WebLogic Server はこの接続のための LDAP サーバとして people.catalog.com を使用します。people.catalog.com への接続が成功した後は、directory.knowledge.com への接続をキャンセルします。

WebLogic 認証プロバイダのコンフィグレーション

注意： WebLogic Server Administration Console は、WebLogic 認証プロバイダをデフォルト認証プロバイダとして参照します。

WebLogic 認証プロバイダでは、大文字と小文字は区別されません。一意のユーザ名を指定してください。

WebLogic 認証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。WebLogic 認証プロバイダのユーザおよびグループ メンバシップ情報は、組み込み LDAP サーバに格納されます。

WebLogic 認証プロバイダをコンフィグレーションするには、次の手順に従います。

1. 第 5 章「組み込み LDAP サーバの管理」の説明に従って組み込み LDAP サーバをコンフィグレーションします。
2. [セキュリティ | レルム] ノードを展開します。
3. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
4. [プロバイダ | 認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
5. [新しい Default Authenticator のコンフィグレーション] リンクをクリックします。

6. [一般] タブの属性値を定義します。
   - [最小パスワード文字数] 属性は、WebLogic 認証プロバイダでユーザを定義するときに指定するパスワードに適用されます。
   - [制御フラグ] 属性で、WebLogic 認証プロバイダをどのように他の LDAP 認証プロバイダと共に使用するかを指定します。詳細については、3-10 ページの「JAAS 制御フラグ属性の設定」を参照してください。
7. [適用] をクリックして変更を保存します。
8. [詳細] タブの属性値を定義します。

   [グループ メンバシップ検索]— グループ検索の対象範囲が制限されるかどうかを指定します。この属性では、ネストされたグループでどのくらいの深さまで検索するのかを指定します。ネストされたグループ階層の最初のレベルのみを使用するコンフィグレーションの場合は、この属性で検索をグループの最初のレベルに制限するとパフォーマンスを向上させることができます。
   - 検索の制限を指定した場合は、[最大グループ メンバシップ検索レベル] 属性を指定する必要がある
   - 検索を制限しない場合は、[最大グループ メンバシップ検索レベル] 属性は無視される

   [最大グループ メンバシップ検索レベル]—[グループ メンバシップ検索] 属性が指定された場合に、グループ メンバーシップ検索の深さを指定します。指定できる値は次のとおりです。
   - 0— 直接のメンバー グループのみ検索されます。つまり、グループ A でメンバーシップを検索するときに、グループ A の直接のメンバーのみが検索されます。グループ B がグループ A のメンバーである場合、そのメンバーは検索の対象になりません。
   - 任意の正の数値 — 検索されるレベル数を示します。たとえば、この属性を 1 に設定すると、グループ A のメンバーシップの検索ではグループ A の直接のメンバーが返されます。グループ B がグループ A のメンバーである場合は、グループ B のメンバーも検索の対象になります。ただし、グループ C がグループ B のメンバーである場合、グループ C のメンバーは検索の対象になりません。
9. [適用] をクリックして変更を保存します。

10. 必要な場合、追加の認証プロバイダと ID アサーション プロバイダをコンフィグレーションします。

11. WebLogic Server を再起動します。

レルム アダプタ認証プロバイダのコンフィグレーション

レルム アダプタ認証プロバイダを使用すると、バージョン 6.x のセキュリティ レルムのユーザとグループをこのリリースの WebLogic Server で使用できるようになります。レルム アダプタ認証プロバイダは、ユーザとグループを 6.x の Windows NT、UNIX、RDBMS セキュリティ レルム、または 6.x のカスタム セキュリティ レルムに格納している場合に使用します (このリリースの WebLogic Server には 6.x の Windows NT、UNIX、RDBMS セキュリティ レルムに相当するものは存在しません)。レルム アダプタ認証プロバイダは、WebLogic 認証プロバイダの代わりとしてコンフィグレーションすることも、WebLogic 認証プロバイダ以外に追加コンフィグレーションすることもできます。

互換性セキュリティを使用する場合、デフォルトによって *CompatibilityRealm* にレルム アダプタ認証プロバイダがコンフィグレーションされます。ただし、レルム アダプタ認証プロバイダは、どのセキュリティ レルムでもコンフィグレーションできます。*CompatibilityRealm* でレルム アダプタ認証プロバイダを使用する方法については、9-2 ページの「CompatibilityRealm のデフォルト セキュリティ コンフィグレーション」を参照してください。

また、`weblogic.security.acl.CertAuthenticator` クラスの実装をこのリリースの WebLogic Server で使用することもできます。レルム アダプタ認証プロバイダには、X.509 トークンに基づく ID アサーションを提供する ID アサーション プロバイダが含まれています。WebLogic Server で CertAuthenticator を使用する方法については、9-4 ページの「レルム アダプタ認証プロバイダでの ID アサーション 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。

認証プロバイダがすでにコンフィグレーションされているセキュリティ レルムにレルム アダプタ認証プロバイダを追加する場合、WebLogic Server Administration Console はレルム アダプタ認証プロバイダの [制御フラグ] 属性を OPTIONAL に設定し、ドメイン ディレクトリ内で `fileRealm.properties` ファ

イルの存在をチェックします。`fileRealm.properties` が存在しない場合、WebLogic Server Administration Console はレルム アダプタ認証プロバイダをセキュリティ レルムに追加しません。

注意： レルム アダプタ認証プロバイダによって生成されるサブジェクトには、ユーザが所属するグループのプリンシパルは含まれません。`weblogic.security.SubjectUtils.isUserInGroup()` メソッドを使用して、ユーザがグループに属するかどうかを調べます。レルム アダプタ認証プロバイダによって生成されるサブジェクトを使用する場合、ユーザが所属するすべてのグループを繰り返す方法はありません。

レルム アダプタ認証プロバイダの属性を定義するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ | 認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
4. [新しい Realm Adapter Authenticator のコンフィグレーション] リンクをクリックします。
5. [一般] タブの [制御フラグ] 属性を設定します。[制御フラグ] 属性で、レルム アダプタ認証プロバイダをどのように他の LDAP 認証プロバイダと共に使用するかを指定します。3-10 ページの「JAAS 制御フラグ属性の設定」を参照してください。
6. [適用] をクリックして変更を保存します。
7. WebLogic Server を再起動します。
8. 必要に応じて、`weblogic.security.acl.CertAuthenticator` クラスの実装をこのリリースの WebLogic Server で使用できるように、レルム アダプタ認証プロバイダで ID アサーション プロバイダをコンフィグレーションします。ID アサーション プロバイダは、X.509 トークンを使用して ID アサーションを実行します。

   [アクティブ タイプ] リスト ボックスに X.509 と入力します。
9. [適用] をクリックして変更を保存します。

10. WebLogic Server を再起動します。
11. 必要な場合、追加の認証プロバイダと ID アサーション プロバイダをコンフィグレーションします。
12. WebLogic Server を再起動します。

WebLogic ID アサーション プロバイダのコンフィグレーション

注意： WebLogic Server Administration Console は、WebLogic ID アサーション プロバイダをデフォルト ID アサーション プロバイダとして参照します。

境界認証を使用する場合、ID アサーション プロバイダを使用する必要があります。境界認証では、WebLogic Server の外部にあるシステムがトークンを通じて信頼を確立します (これは単純認証とは対照的です。単純認証では、WebLogic Server がユーザ名とパスワードを通じて信頼を確立します)。ID アサーション プロバイダは、トークンを検証し、そのトークンの妥当性と信頼性を確立するために必要なアクションを実行します。各 ID アサーション プロバイダは、1 つまたは複数のトークン フォーマットをサポートします。

セキュリティ レルムでは、WebLogic ID アサーション プロバイダまたはカスタム ID アサーション プロバイダのいずれかを使用できます。この節では、WebLogic 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方法について説明します。カスタム セキュリティ プロバイダ (カスタム ID アサーション プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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。

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。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 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。WebLogic ID アサーション プロバイダを使用する場合は、アクティブ トークン タイプをコンフィグレーションします。WebLogic ID アサーション プロバイダは、X509 証明書と CORBA Common Secure Interoperability バージョン 2 (CSI v2) を使用する ID アサーションをサポートしています。

セキュリティ レルムで複数の ID アサーション プロバイダがコンフィグレーションされている場合、すべての ID アサーション プロバイダが同じトークン タイプをサポートできます。ただし、アクティブなトークンを持つのはセキュリティ レルム内で 1 つのプロバイダだけです。

セキュリティ レルムで WebLogic ID アサーション プロバイダを使用する場合、ID アサーション プロバイダによって認証されたトークンをセキュリティ レルム内のユーザにマップするユーザ名マッパーを使用することもできます。ユーザ名マッパーのコンフィグレーションの詳細については、3-32 ページの「WebLogic ID アサーション プロバイダでのユーザ名マッパーの使用」を参照してください。

WebLogic ID アサーション プロバイダの属性を定義するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ | 認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
4. [認証プロバイダ] タブから、[新しい Default Identity Asserter のコンフィグレーション] リンクをクリックします。

   [一般] タブが表示されます。
5. ユーザ名マッパーをコンフィグレーションします。詳細については、3-32 ページの「WebLogic ID アサーション プロバイダでのユーザ名マッパーの使用」を参照してください。
6. [信頼されたクライアント プリンシパル] 属性で、CSIv2 ID アサーションを使用できるクライアント プリンシパルのリストを定義します。アスタリスク (*) を使用すると、すべてのクライアント プリンシパルを指定できます。この属性は、CSI v2 ID アサーションを使用する場合にのみ必要です。
7. WebLogic ID アサーション プロバイダのアクティブ トークン タイプを定義します。ID アサーション プロバイダによってサポートされているトークン タイプのリストが、[サポートタイプ] 属性に表示されます。サポートされているトークン タイプの名前を [アクティブ タイプ] 属性に入力します。
8. [適用] をクリックして変更を保存します。

9. [詳細] タブをクリックします。
10. [Base64 デコーティングが必要] 属性の設定を確認します。

    Web アプリケーションで認証タイプが CLIENT-CERT に設定されている場合、WebLogic Server の Web アプリケーション コンテナは、リクエスト ヘッダおよびクッキーの値に関して ID アサーションを実行します。ヘッダ名またはクッキー名が ID アサーション プロバイダでコンフィグレーションされているアクティブ トークン タイプと一致した場合、値はプロバイダに渡されます。

    リクエスト ヘッダの値またはクッキーの値が、ID アサーション プロバイダに送られる前に Base64 デコードされるかどうかは、[Base64 デコーティングが必要] 属性によって決まります。下位互換性のため、この属性の設定はデフォルトで有効化されますが、ほとんどの ID アサーション プロバイダはこの属性を無効化します。
11. [適用] をクリックします。
12. 必要な場合、追加の認証プロバイダと ID アサーション プロバイダをコンフィグレーションします。
13. WebLogic Server を再起動します。

WebLogic ID アサーション プロバイダでのユーザ名マッパーの使用

双方向 SSL を使用する場合、WebLogic Server は、SSL 接続を確立するときに Web ブラウザまたは Java クライアントのデジタル証明書を検証します。ただし、デジタル証明書は Web ブラウザまたは Java クライアントを WebLogic Server セキュリティ レルムのユーザとしては認識しません。Web ブラウザまたは Java クライアントがセキュリティ ポリシーで保護された WebLogic Server リソースをリクエストする場合、WebLogic Server は Web ブラウザまたは Java クライアントにユーザ名とパスワードを指定するように要求します。WebLogic ID アサーション プロバイダでは、Web ブラウザまたは Java クライアントのデジタル証明書を WebLogic Server セキュリティ レルム内のユーザにマップするユーザ名マッパーを使用できます。

ユーザ名マッパーは、`weblogic.security.providers.authentication.UserNameMapper` インタフェースの実装でなければなりません。このインタフェースは、ニーズに適したスキーマに基づいてトークンを WebLogic Server ユーザ名にマップします。また、このインタフェースを使用して X.501 識別名をユーザ名にマップすることもできます。

WebLogic ID アサーション プロバイダでユーザ名マッパーを使用するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ | 認証プロバイダ] ノードを展開します。
4. デフォルト ID アサーション プロバイダを選択します。

   [一般] タブが表示されます。
5. [ユーザ名マッパーのクラス名] 属性に、`weblogic.security.providers.authentication.UserNameMapper` インタフェースの実装名を入力します。

   `weblogic.security.providers.authentication.UserNameMapper` インタフェースの実装は CLASSPATH に指定されている必要があります。
6. [適用] をクリックします。
7. WebLogic Server を再起動します。

LDAP X509 ID アサーション プロバイダのコンフィグレーション

注意： LDAP X509 ID アサーション プロバイダで提供される機能のコンフィグレーションは、WebLogic Server の将来のリリースで変更されます。このバージョンの LDAP X509 ID アサーション プロバイダは、WebLogic Server の将来のリリースとの上位互換性はありません。また、このプロ

バイダは Sun One LDAP サーバでしかテストしていません。LDAP X509 ID アサーション プロバイダの有用性については、WebLogic Server の『リリース ノート』を参照してください。

LDAP X509 ID アサーション プロバイダは、X509 証明書を受信し、その証明書と関連付けられたユーザを LDAP オブジェクトでルックアップし、LDAP オブジェクト内の証明書が提示された証明書と一致することを確認した上で、LDAP オブジェクトからユーザの名前を取得します。

LDAP X509 ID アサーション プロバイダは、次のように機能します。

1. アプリケーションは、境界認証を使用するように設定する必要があります (つまり、ユーザまたはシステム プロセスがトークンを使用して ID のアサーションを行う)。SSL ハンドシェークの過程で、アプリケーションは証明書を提示します。証明書のサブジェクト DN は、LDAP サーバでユーザを表すオブジェクトを特定するために使用できます。そのオブジェクトには、ユーザの証明書と名前が格納されています。
2. LDAP X509 ID アサーション プロバイダは、サブジェクト DN の証明書を使用して、LDAP サーバでユーザの LDAP オブジェクトを検索する LDAP 検索を作成します。LDAP X509 ID アサーション プロバイダはオブジェクトから証明書を取得し、保持している証明書と一致することを確認して、ユーザの名前を取得します。
3. ユーザ名は、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認証プロバイダに渡されます。認証プロバイダは、ユーザが存在することを確認し、そのユーザが属するグループを特定します。

通常、LDAP X509 ID アサーション プロバイダを使用する場合は、LDAP サーバを使用する LDAP 認証プロバイダをコンフィグレーションすることも必要です。認証プロバイダは、ユーザが存在することを確認し、そのユーザが属するグループを特定します。必ず、両方のプロバイダが同じ LDAP サーバと通信するように正しくコンフィグレーションされているようにしてください。

LDAP X509 ID アサーション プロバイダは、以下のようにして使用します。

1. ユーザの証明書を取得し、LDAP サーバに配置します。証明書のサブジェクト DN と LDAP サーバにおけるそのユーザのオブジェクトの位置には、相関関係がなければなりません。ユーザの LDAP オブジェクトには、証明書の属性とサブジェクトで使用されるユーザ名も含まれている必要があります。

2. 証明書のサブジェクト DN が渡された場合に LDAP ディレクトリでユーザの LDAP オブジェクトを検索するように、LDAP X509 ID アサーション プロバイダをコンフィグレーションします。基本的に、LDAP におけるユーザの DN とユーザの LDAP オブジェクトには、証明書のサブジェクト DN の値と一致する属性がなければなりません。

   例 **1 :** LDAP オブジェクトがサブジェクト DN の属性と一致

   証明書のサブジェクト **DN :**
   `CN=`**`fred`**`, ou=Acme, c=US.`

   LDAP DN :
   `ou=people, cn=flintstone`

   LDAP オブジェクト **:**
   `uid=`**`fred`**`, CN=flintstone`(username), `usercert=cert`

   例 **2 :** LDAP DN の属性がサブジェクト DN の構成要素と一致

   証明書のサブジェクト **DN :**
   `DN: CN=`**`fred`**`, ou=Acme, c=US.`

   LDAP DN :
   `ou=people, uid=`**`fred`**

   LDAP オブジェクト :
   `SN=flintstone`(username), `uid=fred,usercert=cert`

3. LDAP サーバを検索してユーザの LDAP オブジェクトを特定するように、LDAP X509 ID アサーション プロバイダをコンフィグレーションします。そのためには、以下のデータが必要です。

   - 検索の起点となるベース LDAP DN。LDAP X509 ID アサーション プロバイダの [Certificate Mapping] 属性は、証明書のサブジェクト DN からベース LDAP DN を作成する方法を ID アサーション プロバイダに通知します。LDAP オブジェクトには、証明書を保持する属性が必要です。
   - 定義済みの属性セットと一致する LDAP オブジェクトのみを返す検索フィルタ。このフィルタは、LDAP の検索を制限します。証明書のサブジェクト DN から検索フィルタを作成するように、[User Filter Attributes] 属性をコンフィグレーションします。
   - ベース LDAP DN を検索する LDAP ディレクトリ内の位置。LDAP X509 ID アサーション プロバイダは、再帰的に検索を行います (1 レベル下へ)。この属性は、証明書のサブジェクト DN の属性値と同じである必要があります。

4. LDAP X509 ID アサーション プロバイダの [Certificate Attribute] をコンフィグレーションして、ユーザの LDAP オブジェクトのどの属性が証明書を保持するのかを指定します。LDAP オブジェクトには、証明書を保持する属性が必要です。
5. LDAP X509 ID アサーション プロバイダの [Username Attribute] をコンフィグレーションして、サブジェクト DN で表されるユーザ名を保持する LDAP オブジェクトの属性を指定します。
6. LDAP X509 ID アサーション プロバイダの LDAP サーバ接続をコンフィグレーションします。LDAP サーバの属性情報は、このセキュリティ レルムでコンフィグレーションされている LDAP 認証プロバイダに定義されている情報と同じでなければなりません。
7. LDAP X509 ID アサーション プロバイダと一緒に使用する LDAP 認証プロバイダをコンフィグレーションします。LDAP サーバの属性情報は、手順 6 でコンフィグレーションした LDAP X509 ID アサーション プロバイダに定義されている情報と同じでなければなりません。

LDAP X509 ID アサーション プロバイダの属性を定義するには、次の手順に従います。

1. [セキュリティ｜レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) を選択します。
3. [プロバイダ｜認証プロバイダ] ノードを展開します。

   [認証プロバイダ] テーブルに、デフォルトの認証プロバイダと ID アサーション プロバイダの名前が表示されます。
4. [Authenticators] タブで、[新しい LDAPX509Identity Asserter のコンフィグレーション] リンクをクリックします。

   [一般] タブが表示されます。
5. LDAP X509 ID アサーション プロバイダの名前とトークンの情報を定義します。

   次の表では、[一般] タブで設定する属性について説明します。

表 3-7 [一般] タブの属性

属性	説明
[名前]	この LDAP X509 ID アサーション プロバイダの名前。
[記述]	この LDAP X509 ID アサーション プロバイダの簡単な説明。
[バージョン]	この LDAP X509 ID アサーション プロバイダのバージョン番号。
[サポート タイプ]	この LDAP X509 ID アサーション プロバイダでサポートされているトークン タイプ。この属性は常に X509 に設定される。
[アクティブなタイプ]	この LDAP X509 ID アサーション プロバイダが認証に使用するトークン タイプ。このトークン タイプは常に X509 に設定される。 同じセキュリティ レルムでコンフィグレーションされている他の ID アサーション プロバイダでこの属性が X509 に設定されていないようにする。

6. [適用] をクリックして変更を保存します。
7. [詳細] タブを選択します。
8. LDAP ディレクトリのユーザ名を証明書のユーザ名にマップするための属性、および LDAP X509 ID アサーション プロバイダで使用される LDAP サーバについての情報を定義します。

   次の表では、[詳細] タブで設定する属性について説明します。

   注意： `$subj` は、証明書のサブジェクト属性を示します (`CN=meyer.beasys.com, ou=CCE, o=BEASYS, L=SFO, C=US` など)。

表 3-8 [詳細] タブの属性

属性	定義
[Certificate Mapping]	ユーザの LDAP オブジェクトを特定するために使用するベース LDAP DN の作成方法を指定する。この属性は、証明書のサブジェクト DN からオブジェクトを検索する方法を定義する。 通常、この値は LDAP 認証プロバイダの [User Base DN] 属性と同じ。サブジェクト DN のフィールドをこのベース DN に含めることができる。 たとえば、証明書のサブジェクトが `CN=meyer.beasys.com, ou=fred, o=BEASYS, L=SFO, C=US` で、マッピングが `ou=people, ou=$subj.ou` の場合、WebLogic Server はユーザを特定する時に `ou=people, ou=fred, o=BEASYS, c=US` を DN として使用する。

表 3-8 [詳細] タブの属性

属性	定義
[User Filter Attributes]	[Certificate Mapping] 属性で定義されているベース LDAP DN の LDAP オブジェクトの中からユーザの LDAP オブジェクトを選択する方法を指定する。この属性は、証明書のサブジェクト DN から LDAP オブジェクトを検索する方法を定義する。 LDAP オブジェクトのクラスは person でなければならない。この属性には、文字列の配列を設定する。その各文字列は、LDAP オブジェクトが一致しなければならない属性。 通常、この属性の値は証明書のサブジェクト DN の属性値と一致する LDAP オブジェクト。 次に例を示す。 構文が次のようである場合、LDAP ユーザ オブジェクトの `uid` 属性はサブジェクト DN の属性と一致する。 `LDAPATTRNAME=$subj.SUBJECDNATTRNAME` 例：`uid=$subj.DN` この属性は、ユーザ名を検索フィルタにマップする LDAP 認証プロバイダの [User Name Filter] 属性とよく似ている。違いは以下のとおり。 ■ この属性は証明書のサブジェクト DN をフィルタにマップし、LDAP 認証プロバイダは単一の文字列を使用して、システム管理者がフィルタを完全に管理できるようにする。 ■ LDAP X509 認証プロバイダは `objectclass=person` をフィルタに追加し、結合された文字列の配列を使用する。

表 3-8 [詳細] タブの属性

属性	定義

[Certificate Attribute]	ユーザの証明書を格納する、ユーザの LDAP オブジェクトの属性を指定する。この属性は、証明書を検索する方法を定義する。有効な値は、`userCertificate` と *u*`serCertificate;binary`。デフォルトは、`userCertificate`。
	■ LDAP ブラウザを使用して証明書を LDAP ディレクトリにロードすると、バイナリ型の属性 `userCertificate` が作成される。証明書にアクセスするには、[Certificate Attribute] を `userCertificate` として定義する。
	■ `ldapmodify` を使用して新しい属性を作成すると (たとえば次のコマンドを使用して)、
	`ldapmodify -p 1155 -D Principal -w Password`
	`dn: cn=support@bea.com, ou=Certs, dc=bea, dc=com`
	`changetype: modify`
	`add: UserCertificate`
	`userCertificate;binary::`
	`MIICxDCCAi2gAwIBAgIDIDANbgkqn...`
	証明書のデータが LDAP ディレクトリにロードされたときに、属性 `userCertificate;binary` が作成されます。証明書にアクセスするには、[Certificate Attribute] を `userCertificate;binary` として定義する。

[Username Attribute]	ユーザの名前を格納する、ユーザの LDAP オブジェクトの属性を指定する。ユーザの名前は、サブジェクトで使用される。この属性は、ユーザの名前を検索する方法を定義する。 通常、この属性は LDAP 認証プロバイダの [User Name] 属性と一致する。

表 3-8 [詳細] タブの属性

属性	定義
[Base64 デコーディングが必要]	リクエスト ヘッダの値またはクッキーの値が、ID アサーション プロバイダに送られる前に Base64 デコードされるかどうかを指定する。この設定はデフォルトでは下位互換性のために有効になっているが、ほとんどの ID アサーション プロバイダではこの属性は無効化される。
[ホスト]	LDAP サーバが稼働するコンピュータのホスト名。
[ポート]	LDAP サーバがリスンするポートの番号。SSL プロトコルを使用して WebLogic Server を LDAP サーバに接続する場合は、LDAP サーバの SSL ポートを指定する。
[SSL を有効化]	LDAP サーバと WebLogic Server との通信を保護するために SSL プロトコルを使用できるようにするためのオプション。LDAP サーバが SSL プロトコルを使用するようにコンフィグレーションされていない場合は、この属性を無効化する。
[プリンシパル]	WebLogic Server が LDAP サーバとの接続に使用する LDAP ユーザの識別名 (Distinguished Name: DN)。一般に、このユーザは LDAP ディレクトリ サーバのシステム管理者である。パスワードを変更する場合、この属性をシステム管理者に設定する必要がある。
[資格]	[プリンシパル] 属性で定義された LDAP ユーザを認証するパスワード。
[キャッシュを有効化]	Open LDAP サーバでのデータ キャッシュの使用を有効にする。
[キャッシュ サイズ]	キャッシュのルックアップの最大サイズ。デフォルトは 32 KB。

表 3-8 [詳細] タブの属性

属性	定義
[キャッシュ TTL]	LDAP ルックアップの結果を保持する秒数。
[照会先に従う]	LDAP X509 ID アサーション プロバイダ内のユーザまたはグループの検索で、他の LDAP サーバまたは LDAP ディレクトリ内のブランチへの照会に従うことを指定する。デフォルトで、この属性は有効。
[照会先に匿名でバインドする]	デフォルトでは、検索時に照会先に従う場合、LDAP X509 ID アサーション プロバイダは LDAP サーバへの接続で使用するものと同じ DN およびパスワードを使用する。匿名ユーザとして接続する場合は、この属性を有効にする。詳細については、LDAP システム管理者に問い合わせること。
[結果までのタイム リミット]	タイムアウトするまで LDAP サーバが結果を待機する最大時間 (ミリ秒単位)。この属性が 0 に設定されている場合、最大時間の制限はない。デフォルトでは 0。
[接続タイムアウト]	LDAP サーバへの接続が確立されるまでに待機する最大時間 (秒単位)。この属性が 0 に設定されている場合、最大時間の制限はない。デフォルトでは 0。
[並列接続遅延]	複数の LDAP サーバに同時に接続しようとしたときの遅延 (秒単位)。この属性が 0 に設定されている場合、接続の試行はシリアライズされる。接続の試行はリストの最初のサーバに対して行われる。ホストへの接続が失敗した場合にのみ、リスト内の次のエントリに対して接続が試行される。この属性が設定されず、LDAP サーバが使用できない場合、アプリケーションが長時間ブロックされることがある。この属性が 0 より大きい場合は、指定された時間の経過後に別の接続が開始される。

9. [適用] をクリックします。
10. 必要に応じて、他の認証プロバイダまたは ID アサーション プロバイダ、あるいはその両方をコンフィグレーションします。
11. WebLogic Server を再起動します。

サーブレットの ID アサーションの順序付け

HTTP リクエストが送信されるときに、ID アサーションに使用できる一致が複数あることもあります。WebLogic Server の ID アサーションは、次の順序付けを使用します。

1. X.509 デジタル証明書 (クライアントまたはクライアントと Web サーバの間で双方向 SSL を行うプロキシ プラグインに双方向 SSL を通知する)。X.509 がデフォルト セキュリティ レルムの ID アサーション プロバイダでコンフィグレーションされているアクティブなトークン タイプの 1 つである場合。
2. 名前が `WL-Proxy-Client-<TOKEN>` 形式のヘッダ。*<TOKEN>* は、デフォルト セキュリティ レルムの ID アサーション プロバイダでコンフィグレーションされているアクティブなトークン タイプの 1 つです。

   注意 : この手法は非推奨であり、下位互換性を目的としてのみ使用します。
3. 名前が *<TOKEN>* 形式のヘッダ。*<TOKEN>* は、デフォルト セキュリティ レルムの ID アサーション プロバイダでコンフィグレーションされているアクティブなトークン タイプの 1 つです。
4. 名前が *<TOKEN>* 形式のクッキー。*<TOKEN>* は、デフォルト セキュリティ レルムの ID アサーション プロバイダでコンフィグレーションされているアクティブなトークン タイプの 1 つです。

たとえば、デフォルト セキュリティ レルムの ID アサーション プロバイダが、`FOO` および `BAR` をアクティブなトークン タイプとしてコンフィグレーションされている場合 (次の例では、HTTP リクエストに、アクティブ トークン タイプ以外の ID アサーションに関連する情報が何もないと想定する)、ID アサーションは次のようにして行われます。

- リクエストが双方向 SSL 接続経由で `FOO` ヘッダを伴って送られてきた場合は、X.509 が ID アサーションに使用される

- リクエストが FOO ヘッダと WL-Proxy-Client-BAR ヘッダを使用して送られてきた場合は、BAR トークンが ID アサーションに使用される
- リクエストが FOO ヘッダと BAR クッキーを伴って送られてきた場合は、FOO トークンが ID アサーションに使用される

同じレベルの複数のトークン間の順序付けは定義されていないので、次のように処理されます。

- リクエストが FOO ヘッダと BAR ヘッダを伴って送られてきた場合は、FOO または BAR トークンのどちらかが ID アサーションに使用される。ただし、どちらが使用されるかは不明
- リクエストが FOO クッキーと BAR クッキーを伴って送られてきた場合は、FOO または BAR トークンのどちらかが ID アサーションに使用される。ただし、どちらが使用されるかは不明

WebLogic 認可プロバイダのコンフィグレーション

認可とは、ユーザとリソースとのやり取りを限定して、整合性、機密性、および可用性を保証するプロセスのことです。すなわち、認可では、ユーザの身元などの情報に基づいてリソースへのアクセスを制御します。

セキュリティ レルムでは、WebLogic 認可プロバイダまたはカスタム認可プロバイダのいずれかを使用できます。この節では、WebLogic 認可プロバイダをコンフィグレーションする方法について説明します。カスタム セキュリティ プロバイダ (カスタム認可プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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 認可プロバイダをコンフィグレーションするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。

3. [プロバイダ] ノードを展開します。
4. [認可] をクリックします。

   [認可] テーブルに、コンフィグレーションするレルムのデフォルト認可プロバイダの名前が表示されます。
5. [新しい Default Authorizer のコンフィグレーション] リンクをクリックします。
6. [一般] タブの属性値を定義します。

   [ポリシー デプロイメントを有効化] 属性で、この認可プロバイダがセキュリティ レルムのロール情報を (検索ではなく) 格納するかどうかを指定します。[ポリシー デプロイメントを有効化] 属性をサポートするには、認可プロバイダは `DeployableAuthorizationProvider` セキュリティ サービス プロバイダ インタフェース (SSPI) を実装する必要があります。デフォルトで、この属性は有効になっています。ポリシー情報は、組み込み LDAP サーバに格納されます。
7. [適用] をクリックして変更を保存します。
8. WebLogic Server を再起動します。

WebLogic 資格マッピング プロバイダのコンフィグレーション

資格マッピングとは、リモート システム (既存のシステムやアプリケーションなど) の認証および認可メカニズムによって適切な資格セットを取得して、WebLogic リソースにアクセスしようとするユーザを認証するプロセスです。

資格マップの作成については、第 4 章「エンタープライズ情報システムでのシングル サインオン」と『WebLogic J2EE コネクタ アーキテクチャ』のセキュリティに関するトピックを参照してください。

セキュリティ レルムでは、WebLogic 資格マッピング プロバイダまたはカスタム資格マッピング プロバイダのいずれかを使用できます。この節では、WebLogic 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方法について

説明します。カスタム セキュリティ プロバイダ (カスタム資格マッピング プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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 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には、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ] ノードを展開します。
4. [資格マッパー] をクリックします。
5. [新しい Default Credential Mapper のコンフィグレーション] リンクをクリックします。
6. [一般] タブで、[資格マッピング デプロイメントを有効化] 属性を設定します。

   [資格マッピング デプロイメントを有効化] 属性で、この資格マッピング プロバイダが資格マップをデプロイメント記述子 (`weblogic-ra.xml` ファイル) からセキュリティ レルムにインポートするかどうかを指定します。[資格マッピング デプロイメントを有効化] 属性をサポートするには、資格マッピング プロバイダは `DeployableCredentialProvider` SSPI を実装する必要があります。デフォルトでは、この属性は有効になっています。資格マッピング情報は、組み込み LDAP サーバに格納されます。

   詳細については、『WebLogic Security サービスの開発』の「DeployableCredentialMappingProvider SSPI を実装する」を参照してください。
7. [適用] をクリックして変更を保存します。
8. WebLogic Server を再起動します。

WebLogic キーストア プロバイダのコンフィグレーション

キーストアは、プライベート キーと信頼性のある認証局 (CA) を格納するファイルを作成および管理するためのメカニズムです。WebLogic キーストア プロバイダは、キーストアのインスタンスを検索します。WebLogic キーストア プロバイダのコンフィグレーションは、SSL プロトコルの設定時のオプションの 1 つです。詳細については、6-15 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の格納」を参照してください。WebLogic キーストアのコンフィグレーションは、セキュリティ レルムをカスタマイズするとき、または新しいセキュリティ レルムを作成するときのオプションの手順です。

WebLogic ロール マッピング プロバイダのコンフィグレーション

ロール マッピング プロバイダは、特定のリソースのサブジェクトに付与されるロール セットを計算します。ロール マッピング プロバイダは、このロール情報を認可プロバイダに提供します。このため、認可プロバイダは WebLogic 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 レルムでは、WebLogic ロール マッピング プロバイダまたはカスタム ロール マッピング プロバイダのいずれかを使用できます。この節では、WebLogic 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方法について説明します。カスタム セキュリティ プロバイダ (カスタム ロール マッピング プロバイダを含む) のコンフィグレーションについては、3-48 ページの「カスタム セキュリティ 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。

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には、次の手順に従います。

1. [セキュリティ] ノードを展開します。
2. [レルム] ノードを展開します。

3. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
4. [プロバイダ] ノードをクリックします。
5. [ロール マッパー] をクリックします。

   [ロール マッパー] テーブルが表示されます。このテーブルには、コンフィグレーションするレルムのデフォルト ロール マッピング プロバイダの名前が表示されます。
6. [新しい Default Role Mapper のコンフィグレーション] リンクをクリックします。

   [一般] タブが表示されます。
7. [一般] タブの属性値を定義します。

   [ロール デプロイメントを有効化] 属性で、このロール マッピング プロバイダが Web アプリケーションおよび EJB 用のデプロイメント記述子の情報をセキュリティ レルムにロードするかどうかを指定します。[ロール デプロイメントを有効化] 属性をサポートするには、ロール マッピング プロバイダは `DeployableRoleProvider` SSPI を実装する必要があります。デフォルトでは、この属性は有効になっています。ロール情報は組み込み LDAP サーバに格納されます。

   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。
8. [適用] をクリックして変更を保存します。
9. WebLogic Server を再起動します。

カスタム セキュリティ プロバイダのコンフィグレーション

カスタム セキュリティ プロバイダをコンフィグレーションするには、次の手順に従います。

1. カスタム セキュリティ プロバイダを作成します。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。

   プロバイダ用の MBean JAR ファイルを `WL_HOME\lib\mbeantypes` ディレクトリに置きます。

2. WebLogic Server Administration Console を起動します。
3. [セキュリティ | レルム] ノードを展開します。
4. コンフィグレーションするレルムの名前 (TestRealm など) をクリックします。
5. [プロバイダ] ノードを展開します。
6. コンフィグレーションするプロバイダ タイプのノードを展開します。たとえば、カスタム認証プロバイダをコンフィグレーションするには認証プロバイダ ノードを展開します。

   そのプロバイダ用のタブが表示されます。

7. [新しいカスタム *Security_Provider_Type* のコンフィグレーション] リンクをクリックします。

   `Security_Provider_Type` は、カスタム セキュリティ プロバイダの名前です。この名前は、MBean 定義ファイル (MDF) の *`MBeanType`* タグに含まれる `DisplayName` 属性から読み込まれます。

8. [一般] タブが表示されます。

   [名前] 属性に、カスタム セキュリティ プロバイダの名前が表示されます。

9. 必要な場合、カスタム セキュリティ プロバイダの属性値を変更します。
10. [適用] をクリックして変更を保存します。
11. WebLogic Server を再起動します。

セキュリティ プロバイダの削除

セキュリティ プロバイダを削除するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。

2. 削除するプロバイダがコンフィグレーションされているレルムの名前 (TestRealm など) をクリックします。
3. [プロバイダ] ノードを展開します。
4. 削除するプロバイダのタイプ ([TestRealm ｜認可] など) をクリックします。
5. そのプロバイダ用のテーブル ([認可] テーブルなど) が表示されます。そのテーブルには、コンフィグレーションされているすべてのプロバイダの名前が表示されます。
6. プロバイダを削除するには、そのプロバイダ テーブルのごみ箱アイコンをクリックします。
7. WebLogic Server を再起動します。

注意 : コンフィグレーション済みのセキュリティ プロバイダを WebLogic Server Administration Console で削除および修正するには、セキュリティ プロバイダ データベースの手動クリーンアップが必要となる場合があります。

4　エンタープライズ情報システムでのシングル サインオン

この節では、エンタープライズ情報システム (EIS) ユーザが保護された WebLogic リソースにアクセスできるようにするための資格マップを作成する方法について説明します。

注意： この章は、このリリースの WebLogic Server のセキュリティ機能を使用する WebLogic Server デプロイメントと互換性セキュリティを使用するデプロイメントに適用されます。

概要

J2EE コネクタ アーキテクチャによって定義されるリソース アダプタは、EIS で定義されたユーザが保護されている WebLogic リソースへのアクセスをリクエストした場合に、ユーザを認証するために必要な資格を取得できます。リソース アダプタをホストする WebLogic Server のコンテナは、資格マップを使用して、WebLogic リソース用の資格セットを検索できます。資格マップでは、WebLogic Server セキュリティ レルムのユーザと、EIS (Oracle データベース、SQL サーバ、SAP アプリケーションなど) のユーザの認証に使用される ID (ユーザ名とパスワードの組み合わせ) が関連付けられます。

資格マップは以下の 2 つの手順で作成します。

1. EIS ユーザの WebLogic Server ユーザまたはグループを作成します。ユーザまたはグループは、コンフィグレーションされている認証プロバイダに定義される必要があります。複数の WebLogic Server ユーザ間またはグループを同じリモート ユーザまたはグループにマップできます。管理を効率化するために、グループを使用して資格マップを作成するようにしてください。
2. EIS ユーザの資格マップを作成します。EIS に対する認証を受けるユーザのユーザ名とパスワードか、または EIS ユーザが属するグループ名を使用してユーザを定義します。資格マップは組み込み LDAP サーバに格納されます。

リソース アダプタでのセキュリティの使用方法については、『WebLogic J2EE コネクタ アーキテクチャ』を参照してください。

WebLogic Server で資格マップを作成するには、デプロイメント記述子を使用する方法と WebLogic Server Administration Console を使用する方法があります。以降の節では、それぞれの方法について説明します。

デプロイメント記述子を使用した資格マップの作成

資格マップは、`weblogic-ra.xml` デプロイメント記述子ファイルの `<security-principal-map>` 要素で指定できます。`<security-principal-map>` 要素では、EIS にログインするための資格と WebLogic リソースに対して認証するための資格との関連付けを指定します。コード リスト 4-1 に、`weblogic-ra.xml` デプロイメント記述子ファイルに指定された資格マップの一例を示します。

コード リスト 4-1　サンプル資格マップ

```
<security-principal-map>
    -<map-entry>
        <initiating-principal>raruser</initiating-principal>
        <initiating-principal>javajoe</initiating-principal>
        -<resource-principal>
           <resource-username>scott</resource-username>
            <resource-password>tiger</resource-password>
        </resource-principal>
    </map-entry>
```

デプロイメント記述子を使用して資格マップを作成する方法は、このリリースの WebLogic Server では非推奨となりました。代わりに、WebLogic Server Administration Console を使用して資格マップを作成します。詳細については、4-4 ページの「WebLogic Administration Console を使用した資格マップの作成」を参照してください。

`<security-principal-map>` 要素が定義された `weblogic-ra.xml` デプロイメント記述子ファイルを持つリソース アダプタをデプロイする場合、このファイルのデータを組み込み LDAP サーバにインポートすると、WebLogic 資格マッピング プロバイダで使用できるようになります。

`weblogic-ra.xml` デプロイメント記述子ファイルの情報を組み込み LDAP サーバにインポートするには、デフォルト (アクティブ) セキュリティ レルムの資格マッピング プロバイダの [資格マッピング デプロイメントを有効化] 属性を有効にします。リソース アダプタをデプロイすると、資格マップ情報が資格マッピング プロバイダにロードされます。

[資格マッピング デプロイメントを有効化] 属性をサポートするには、資格マッピング プロバイダは DeployableCredentialProvider SSPI を実装する必要があります。デフォルトでは、この属性は有効になっています。したがって、`weblogic-ra.xml` デプロイメント記述子ファイルの情報は、リソース アダプタをデプロイすると、自動的に WebLogic 資格マッピング プロバイダにロードされます。

ただし、`weblogic-ra.xml` デプロイメント記述子ファイルの情報が組み込み LDAP サーバにロードされても、元のリソース アダプタは変更されません。したがって、リソース アダプタを WebLogic Server Administration Console を使用して再デプロイしたり、ディスク上で変更したり、WebLogic Server を再起動したりする場合などに、元のリソース アダプタが再デプロイされると、`weblogic-ra.xml` デプロイメント記述子ファイルの情報が再度ロードされるので、資格マッピング情報が失われる可能性があります。

新しい資格マッピング情報が `weblogic-ra.xml` デプロイメント記述子ファイル内の古い情報によって上書きされないようにするには、[デプロイの資格マッピングを無視] のデプロイメント記述子設定を指定します。

1. [セキュリティ] ノードを展開します。
2. [レルム] ノードを展開します。

WebLogic ドメインで使用可能なすべてのセキュリティ レルムが [レルム] テーブルに表示されます。

3. 使用しているレルムの名前をクリックします。
4. [一般] タブをクリックします。
5. [デプロイの資格マッピングを無視] のデプロイメント記述子設定をチェックします。この設定では、セキュリティ レルムの資格マッピング プロバイダが WebLogic Server Administration Console によって作成された資格マップのみを使用することを指定します。デフォルトでは、この属性はチェックされていません。つまり、資格マッピング プロバイダは、`weblogic-ra.xml` デプロイメント記述子ファイルに指定されている資格マップをロードします。
6. [適用] をクリックします。
7. WebLogic Server を再起動します。

また、`weblogic-ra.xml` デプロイメント記述子ファイルを変更して、`<security-principal-map>` 要素を削除することをお勧めします。

`weblogic-ra.xml` デプロイメント記述子ファイルを使用して資格マップを指定する方法については、『WebLogic J2EE コネクタ アーキテクチャ』を参照してください。

WebLogic Administration Console を使用した資格マップの作成

資格マップ間のマッピングは、WebLogic Server Administration Console を使用して行えるようになりました。WebLogic 資格マッピング プロバイダを使用する場合、資格マップは組み込み LDAP サーバに格納されます。

資格マップを作成するには、次の手順に従います。

1. [デプロイメント記述子内のセキュリティ データを無視] 属性がデフォルト (アクティブ) セキュリティ レルムで有効になっていることを確認します。この属性が有効になっていないと、資格マップが `weblogic-ra.xml` デプロイメント記述子ファイル内の古い情報で上書きされる恐れがあります。

2. EIS ユーザに対応するユーザまたはグループを定義します。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。
3. [コネクタ] ノードを展開します。
4. 目的のリソース アダプタを右クリックします。
5. [資格マップを定義] オプションをクリックします。

   [資格マッピング] テーブルには、コンフィグレーション済み資格マッパーに定義されているすべての資格マップが表示されます。
6. [新しい Cred Map のコンフィグレーション] リンクをクリックします。
7. EIS ユーザの名前を [リモート ユーザ資格マップ] フィールドに入力します。たとえば、scott のように入力します。
8. EIS ユーザのパスワードを [リモート パスワード] フィールドに入力します。たとえば、tiger のように入力します。
9. [適用] をクリックします。
10. 目的のリソース アダプタを右クリックします。
11. [ロールのマップ ...] オプションをクリックします。
12. 手順 2 で EIS ユーザ用として定義した WebLogic Server ユーザまたはグループの名前を [WLS ユーザ] フィールドに入力します。
13. EIS ユーザの名前を [リモート ユーザ] フィールドに入力します。
14. [適用] をクリックします。

5　組み込み LDAP サーバの管理

デフォルトでは、WebLogic 認証プロバイダ、認可プロバイダ、資格マッピング プロバイダ、およびロール マッピングのセキュリティ プロバイダ データベースとして、組み込み LDAP サーバが使用されます。これらのプロバイダを使用する場合、組み込み LDAP サーバの管理が必要になります。以下の節では、組み込み LDAP サーバを管理する方法について説明します。

組み込み LDAP サーバのコンフィグレーション

組み込み LDAP サーバには、ユーザ、グループ、グループ メンバシップ、セキュリティ ロール、セキュリティ ポリシー、および資格マップ情報が格納されます。デフォルトでは、WebLogic Server ドメインごとに組み込み LDAP サーバが各属性のデフォルト値を使用してコンフィグレーションされます。WebLogic 認証プロバイダ、認可プロバイダ、資格マッピング プロバイダ、およびロール マッピング プロバイダは、組み込み LDAP サーバをデータベースとして使用します。新しいセキュリティ レルムでこれらのプロバイダを使用する場合、組み込み LDAP サーバのデフォルト値を変更して、環境に合わせて動作を最適化することができます。

組み込み LDAP サーバをコンフィグレーションするには、次の手順に従います。

1. ドメイン ノード (examples など) を展開します。
2. [セキュリティ | 組み込み LDAP] タブを選択します。
3. [組み込み LDAP] タブで属性を設定します。次の表では、[組み込み LDAP] タブの各属性について説明します。

表 5-1　[組み込み LDAP] タブの属性

属性	説明
[資格]	組み込み LDAP サーバへの接続に使用される資格 (通常はパスワード)。このパスワードが未設定の場合、WebLogic Server は起動時にパスワードを生成し、属性を初期化し、コンフィグレーションを `config.xml` ファイルに保存する。外部 LDAP ブラウザと組み込み LDAP 管理者アカウント (`cn=Admin`) を使用して組み込み LDAP サーバに接続するには、この属性を生成された値から変更すること。
[バックアップ時間 (時間)]	組み込み LDAP サーバのデータ ファイルをバックアップする時刻 (時)。この属性は [バックアップ時間 (分)] 属性と組み合わせて使用され、組み込み LDAP サーバのデータ ファイルをバックアップする時刻を決定する。指定された時刻になると、WebLogic Server は組み込み LDAP サーバへの書き込みを一時停止し、データ ファイルを `ldap/backup` ディレクトリの zip ファイルにバックアップしてから、書き込みを再開する。デフォルトでは 23。
[バックアップ時間 (分)]	組み込み LDAP サーバのデータ ファイルをバックアップする時刻 (分)。この属性は [バックアップ時間 (時間)] 属性と組み合わせて使用され、組み込み LDAP サーバのデータ ファイルをバックアップする時刻を決定する。デフォルトでは 5 分。

表 5-1 [組み込み LDAP] タブの属性

属性	説明
[バックアップ コピー数]	組み込み LDAP サーバのデータ ファイルのバックアップ コピー数。この値によって、`ldap/backup` ディレクトリ内の zip ファイルの数が制限される。デフォルトでは 7。
[キャッシュを有効化]	組み込み LDAP サーバでキャッシュを使用するかどうかを指定する。このキャッシュは、管理サーバ上で稼働するマスター組み込み LDAP サーバに対して管理対象サーバが読み込みと書き込みを実行するときに使用される。
[キャッシュ サイズ]	組み込み LDAP サーバで使用されるキャッシュのサイズ (単位は K)。デフォルトは 32K。
[キャッシュ TTL]	キャッシュの存続期間 (TTL) の秒数。デフォルトでは 60 秒。
[起動時にレプリカを更新]	管理対象サーバがレプリケートされたデータを起動時にすべてリフレッシュするかどうかを指定する。この属性は、管理対象サーバがアクティブでない間に大量の変更を行ったため、それらの変更を管理サーバから管理対象サーバに送らずにレプリカ全体をダウンロードしたいときに役立つ。 この属性を使用すると、新しいシステム パスワードをドメイン内の管理対象サーバと管理サーバに伝播できる。 デフォルトは false。

表 5-1　[組み込み LDAP] タブの属性

属性	説明
[マスターを優先]	ローカルのレプリケート対象組み込み LDAP サーバの代わりに常にマスター LDAP サーバ (管理サーバ上で稼働) に接続するよう指定する。これにより、管理対象サーバは、管理サーバの情報のレプリカを持つローカルの組み込み LDAP サーバではなく、管理サーバ内の組み込み LDAP サーバからセキュリティ データを取り出す。

4. [適用] をクリックして変更を保存します。

5. WebLogic Server を再起動します。

組み込み LDAP サーバを WebLogic Server ドメインで使用する場合、更新はマスター LDAP サーバに送信されます。マスター LDAP サーバは、すべての変更のログを保持します。また、マスター LDAP サーバは、レプリケート対象サーバのリストと各サーバの現在の変更ステータスも保持します。マスター LDAP サーバは、各レプリケート対象サーバに適切な変更を送信して、各サーバの変更ステータスを更新します。この処理は、マスター LDAP サーバが更新されたときに行われます。ただし、更新数によっては、変更が管理対象サーバにレプリケートされるまでに数秒かかることがあります。マスター LDAP サーバは、管理サーバ上の組み込み LDAP サーバです。レプリケート対象サーバは、WebLogic Server ドメイン内のすべての管理対象サーバです。

注意 : コンフィグレーション済みのセキュリティ プロバイダを WebLogic Server Administration Console で削除および修正するには、組み込み LDAP サーバの手動クリーンアップが必要となる場合があります。不要な情報を削除するには、外部 LDAP ブラウザを使用します。

組み込み LDAP サーバのバックアップのコンフィグレーション

組み込み LDAP サーバのバックアップをコンフィグレーションするには、次の手順に従います。

1. ドメイン ノード (examples など) を展開します。
2. [セキュリティ ｜組み込み LDAP] タブを選択します。
3. [組み込み LDAP] タブで、[バックアップ時間 (時間)]、[バックアップ時間 (分)]、および [バックアップ コピー数] 属性を設定します。
4. [適用] をクリックして変更を保存します。
5. WebLogic Server を再起動します。

LDAP ブラウザによる組み込み LDAP サーバの内容の表示

LDAP ブラウザを使用して組み込み LDAP サーバの内容を表示するには、次の手順に従います。

1. 組み込み LDAP の資格を変更します。
   a. ドメイン ノード (examples など) を展開します。
   b. [Security ｜組み込み LDAP] タブを選択します。
   c. [資格] 属性を変更します。詳細については、表 5-1 を参照してください。
   d. [適用] をクリックします。
   e. WebLogic Server を再起動します。
2. コマンド プロンプトで次のコマンドを入力して、LDAP ブラウザを起動します。

```
lbe.sh
```

3. LDAP ブラウザで新しい接続をコンフィグレーションします。
   a. [Host] フィールドを *localhost* に設定します。
   b. [Port] フィールドを 7001 (SSL を使用する場合は 7002) に設定します。
   c. [Base DN] フィールドを *dc=mydomain* に設定します。*mydomain* は使用する WebLogic Server ドメインの名前を表します。
   d. [Anonymous Bind] オプションのチェックをはずします。
   e. [User DN] フィールドを cn=Admin に設定します。
   f. [Password] フィールドを手順 1 で指定したパスワードに設定します。
4. 新しい接続をクリックします。

   LDAP ブラウザを使用して、組み込み LDAP サーバの階層をナビゲートします。

組み込み LDAP サーバの情報のエクスポートとインポート

LDAP ブラウザを使用すると、組み込み LDAP サーバに格納されている情報をエクスポートおよびインポートできます。表 5-2 には、データが組み込み LDAP サーバの階層のどこに格納されているかをまとめてあります。

表 5-2 組み込み LDAP サーバ内のセキュリティ データの場所

セキュリティ データ	組み込み LDAP サーバ DN
ユーザ	ou=people,ou=*myrealm*,dc=*mydomain*
グループ	ou=groups,ou=*myrealm*,dc=*mydomain*
セキュリティ ロール	ou=ERole,ou=*myrealm*,dc=*mydomain*

表 5-2 組み込み LDAP サーバ内のセキュリティ データの場所

セキュリティ データ	組み込み LDAP サーバ DN
セキュリティ ポリシー	ou=EResource,ou=*myrealm*,dc=*mydomain*

組み込み LDAP サーバのセキュリティ データをエクスポートするには、次の手順に従います。

1. コマンド プロンプトで次のコマンドを入力して、LDAP ブラウザを起動します。

   ```
   lbe.sh
   ```

2. エクスポートするデータを指定します。たとえば、ユーザをエクスポートするには ou=people,ou=*myrealm*,dc=*mydomain* を指定します。
3. [LDIF | Export] オプションを選択します。
4. [Export all children] を選択します。
5. データのエクスポート先となるファイルの名前を指定します。

組み込み LDAP サーバにセキュリティ データをインポートするには、次の手順に従います。

1. コマンド プロンプトで次のコマンドを入力して、LDAP ブラウザを起動します。

   ```
   lbe.sh
   ```

2. インポートするデータを指定します。たとえば、ユーザをインポートするには ou=people,ou=*myrealm*,dc=*mydomain* を指定します。
3. [LDIF | Import] オプションを選択します。
4. [Update/Add] を選択します。
5. データのインポート元となるファイルの名前を指定します。

アクセス制御の構文

組み込み LDAP サーバは、IETF の「LDAP Access Control Model for LDAPv3」(2001 年 3 月 2 日、ドラフト) をサポートしています。この節では、これらのルールが組み込み LDAP サーバでどのように実装されているかを説明します。これらのルールは、標準で想定しているようにディレクトリ内のエントリに適用することも、アクセス制御ファイル (acls.prop) を編集することでコンフィグレーションおよび管理することもできます。

アクセス制御ファイル

注意 : デフォルトでは、組み込み LDAP サーバには、WebLogic Server の管理者アカウントからしかアクセスできません。外部 LDAP ブラウザだけを使用する場合は、acls.prop ファイルを編集する必要はありません。

組み込み LDAP サーバが管理しているアクセス制御ファイル (acls.prop) には、LDAP ディレクトリ全体のアクセス制御リスト (ACL) の全リストが入っています。アクセス制御ファイルの各行には、1 つのアクセス制御ルールが入っています。アクセス制御ルールは、以下の要素で構成されています。

- ルールを適用する LDAP ディレクトリ内の場所
- ルールを適用する場所の中のスコープ
- アクセス権 (許可または拒否)
- パーミッション (許可または拒否)
- ルールを適用する属性
- アクセスの許可または拒否の対象となるサブジェクト

コード リスト 5-1 はサンプルのアクセス制御ファイルです。

コード リスト 5-1　acl.props ファイルのサンプル

```
[root]|entry#grant:r,b,t#[all]#public
```

```
ou=Employees,dc=octetstring,dc=com|subtree#grant:r,c#[all]#public:
ou=Employees,dc=octetstring,dc=com|subtree#grant:b,t#[entry]#public:
ou=Employees,dc=octetstring,dc=com|subtree#deny:r,c#userpassword#public:
ou=Employees,dc=octetstring,dc=com|subtree#grant:r#userpassword#this:
ou=Employees,dc=octetstring,dc=com|subtree#grant:w,o#userpassword,title,
description,
postaladdress,telephonenumber#this:
cn=schema|entry#grant:r#[all]#public:
```

アクセス制御の場所

各アクセス制御ルールは、LDAP ディレクトリ内の指定した場所に適用されます。通常、場所は識別名 (DN) ですが、ディレクトリ全体にアクセス制御ルールを適用する場合は、特殊な場所 `[root]` を `acls.prop` ファイルに指定できます。

アクセスまたは変更の対象となる LDAP サーバのエントリがアクセス制御ルールの場所またはその下の場所にない場合、指定されたアクセス制御ルールはそれ以上評価されません。

アクセス制御のスコープ

アクセス制御のスコープは次のように定義されます。

- Entry − Entry スコープは、LDAP ディレクトリ内のエントリがアクセス制御ルールの場所と同じ DN を持つ場合にのみ評価されます。こうしたルールは、並列しているエントリやサブツリーのエントリよりも 1 つのエントリに問題の情報が含まれている場合に便利です。
- Subtree − Subtree スコープは、LDAP ディレクトリ内のエントリがアクセス制御ルールの場所と同じか、またはその場所で終わっている場合に評価されます。このスコープは、場所エントリとそのサブツリー全体を保護します。

ディレクトリ内のエントリが相互に矛盾するアクセス制御ルールの対象となっている場合 (あるルールが Entry ルールで、他方が Subtree ルールのような場合)、Entry ルールが Subtree ルールよりも優先されます。

アクセス権とパーミッション

アクセス権は、オブジェクト全体またはオブジェクトの属性に適用されます。アクセスは許可または拒否のいずれかです。アクセス制御ルールを作成または更新する場合、grant または deny のいずれのアクションも指定できます。

LDAP アクセス パーミッションはそれぞれ独立しています。あるパーミッションが別のパーミッションを意味するわけではありません。パーミッションでは、実行可能な LDAP の処理のタイプを指定します。

属性のパーミッション

次のパーミッションは、属性に伴うアクションに適用されます。

表 5-3 属性のパーミッション

パーミッション	説明
r Read	読み取り属性。付与した場合、読み取りまたは検索処理で属性または値を返すことが許可される。
w Write	変更または追加属性。付与した場合、変更処理で属性または値を追加することが許可される。
o Obliterate	変更および削除属性。付与した場合、変更処理で属性または値を削除することが許可される。
s Search	指定した属性を持つエントリの検索。付与した場合、検索処理で属性または値を含めることが許可される。
c Compare	属性値の比較。付与した場合、比較処理で属性または値を含めることが許可される。
m Make	このエントリの下の新しいエントリに属性を作成する。

m パーミッションは、オブジェクトの作成時に指定したすべての属性に対して必要です。変更処理で w および o パーミッションが使用されるように、追加処理では m パーミッションが使用されます。w および o パーミッションは追加処理と関係がなく、m は変更処理と関係がありません。新しいオブジェクトはまだ存在していないので、作成時に必要な a および m パーミッションは新しいオブジェクトの親に対して付与しなければなりません。この要件は、変更対象のオブジェクトに対して付与しなければならない w および o パーミッションとは異なります。m パーミッションは w および o パーミッションとは別個のものなので、新しい子をエントリに追加するために必要なパーミッションと、そのエントリの既存の子を変更するために必要なパーミッションとの間で、衝突は発生しません。変更処理で値を置換するためには、ユーザは w および o パーミッションを持っていなければなりません。

エントリのパーミッション

以下のパーミッションは LDAP エントリ全体に適用されます。

表 5-4 エントリのパーミッション

パーミッション	説明
a Add	このエントリの下にエントリを追加する。付与した場合、DIT サブジェクトにエントリを作成することが許可され、新しいエントリの作成時に指定するすべての属性および値を制御できるようになる。エントリを追加するためには、少なくとも必須属性を追加するためのパーミッションも付与しなければならない。
d Delete	このエントリを削除する。付与した場合、エントリ内の属性に対する制御に関係なく、DIT からエントリを削除することが許可される。

表 5-4 エントリのパーミッション

パーミッション	説明
e Export	エントリとすべてのサブエントリを新しい場所にエクスポートする。 付与した場合、エントリとそのサブエントリをエクスポート、つまり現在の場所から削除し、新しい目的の場所のパーミッションに従ってその場所に配置することが許可される。 最後の RDN を変更する場合、現在の場所には Rename パーミッションも必要となる。 エントリまたはそのサブエントリをエクスポートするために、指定されている属性 (RDN 属性を含む) に対する事前のパーミッションは必要ない。このことは、RDN を変更した結果、新しい属性値が追加または削除されるような処理を行う場合でも同様である。
i Import	指定した場所からエントリとそのサブエントリをインポートする。 付与した場合、エントリとそのサブエントリをインポート、つまり他の場所から削除し、指定した場所に配置することが許可される (新しい場所に適切なパーミッションが付与されている場合)。 エントリまたはそのサブエントリをインポートする場合、指定されている属性 (RDN 属性を含む) に対する事前のパーミッションは必要ない。このことは、RDN を変更した結果、新しい属性値が追加または削除されるような処理を行う場合でも同様である。

表 5-4 エントリのパーミッション

パーミッション	説明
n RenameDN	LDAP エントリの DN を変更する。エントリの名前を新しい RDN に変更するには、変更によってサブエントリの DN が変更されることを考慮して、Rename パーミッションを付与する必要がある。上位エントリの名前が変更されない場合は、そのパーミッションを付与するだけでよい。 エントリの名前を変更する場合、指定されている属性 (RDN 属性を含む) に対する事前のパーミッションは必要ない。このことは、RDN を変更した結果、新しい属性値が追加または削除されるような処理を行う場合でも同様である。
b BrowseDN	エントリの DN を参照する。付与した場合、エントリの名前を明示的に指定しないでディレクトリを操作しながらエントリにアクセスすることが許可される。
t ReturnDN	処理の結果にエントリの DN を表示する。付与した場合、エントリの識別名を処理の結果に表示することが許可される。

属性のタイプ

アクセス制御ルールが適用される属性のタイプは、必要に応じて表示される必要があります。以下のキーワードを使用できます。

- [entry] は、パーミッションがオブジェクト全体に適用されることを示す。例として、オブジェクトの削除や子オブジェクトの追加などのアクションがあります。
- [all] は、パーミッションがエントリのすべての属性に適用されることを示す。

キーワード [all] と別の属性の両方を ACL に指定した場合、属性に関してより具体的なパーミッションが、[all] キーワードによって指定されたより具体的でないパーミッションをオーバライドします。

サブジェクトのタイプ

アクセス制御ルールは、さまざまなサブジェクトのタイプに関連付けることができます。アクセス制御ルールが接続中のセッションに適用されるかどうかは、アクセス制御ルールのサブジェクトによって決まります。

以下のサブジェクトのタイプが定義されています。

- `AuthzID` ー サブジェクトの定義の一部として指定できる単一のユーザに適用されます。LDAP ディレクトリでのユーザの ID は通常、DN として定義されます。
- `Group` ー 以下のオブジェクト クラスのいずれかによって指定されるユーザ グループに適用されます。
  - `groupOfUniqueNames`
  - `groupOfNames`
  - `groupOfUniqueURLs`

  最初の 2 つのタイプのグループにはユーザのリストが入りますが、3 つ目のタイプでは定義した条件に基づいてユーザが自動的にグループに含められます。
- `Subtree` ー LDAP ディレクトリ ツリー内のサブジェクトとすべてのサブエントリの一部として定義される DN に適用されます。
- `IP Address` ー 特定のインターネット アドレスに適用されます。このサブジェクト タイプは、すべてのアクセスがプロキシまたはその他のサーバを経由する場合に便利です。適用対象は、ある範囲 (サブネット) ではなく、特定のホストです。
- `Public` ー 認証されているかどうかにかかわらず、ディレクトリに接続しているすべてのユーザに適用されます。
- `This` ー アクセス対象のエントリと一致する DN を持つユーザに適用されます。

許可 / 拒否の評価ルール

エントリ内の情報へのクライアント アクセスを許可するか拒否するかは、アクセス制御ルールと保護対象のエントリに関連するさまざまな要因によって決定されます。決定過程には目安となるいくつかの原則があります。

- より具体的なルールがそうでないルールをオーバライドする。たとえば、ACL の個々のユーザ エントリはグループ エントリに優先します。
- ACL の値どうしが衝突した場合、拒否が許可に優先する
- アクセス制御情報がない場合、拒否がデフォルトとなる。また、エントリ スコープはサブジェクト スコープに優先します。

ルールの具体性を調べても衝突が解決しない場合、どちらのルールを適用するかはルールのサブジェクトによって決定されます。`IP Address` サブジェクトに基づくルールは最も優先度が高く、特定の `AuthzID` または `This` サブジェクトに適用されるルールが続きます。さらに `Group` サブジェクトに適用されるルールが続き、最後は `Subtree` および `Public` サブジェクトに適用されるルールとなります。

6 SSL のコンフィグレーション

以下の節では、WebLogic Server に SSL をコンフィグレーションする方法について説明します。

- 6-55 ページの「SSL プロトコルの使用による weblogic.Admin から WebLogic Server への接続」
- 6-57 ページの「BEA Tuxedo クライアントおよび WebLogic Server での SSL プロトコルの使用」

注意： この章は、このリリースの WebLogic Server のセキュリティ機能を使用する WebLogic Server デプロイメントと互換性セキュリティを使用するデプロイメントに適用されます。

SSL のコンフィグレーションは任意ですが、プロダクション環境では SSL を使用することをお勧めします。

SSL の概要

セキュア ソケット レイヤ (Secure Sockets Layer: SSL) では、ネットワーク接続している 2 つのアプリケーションが互いの ID を認証できるようにするとともに、アプリケーション間でやりとりされるデータを暗号化することで、セキュアな接続を実現します。認証を使用すると、サーバとクライアント (任意) はネットワーク接続の相手側アプリケーションの ID を検証できます。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。

WebLogic Server の SSL は、SSL 3.0 および TLS 1.0 仕様を実装しています。

注意： このリリースの WebLogic Server のプロキシ プラグインは、SSL 3.0 だけをサポートしています。

WebLogic Server は、専用のリスン ポート (デフォルトは 7002) で SSL をサポートします。SSL 接続を確立するには、接続 URL に SSL リスン ポートと HTTPS スキーマを指定して (`https://myserver:7002` など)、Web ブラウザから WebLogic Server に接続します。

SSL を使用すると、計算処理による負荷が大きくなり、接続のオーバーヘッドが増大します。必要のない場合には、SSL を使用しないでください。ただし、プロダクション環境では、常に SSL を使用してください。

プライベート キー、デジタル証明書、信頼性のある認証局

プライベート キー、デジタル証明書、および信頼性のある認証局 (CA) は、サーバの ID を確立および検証します。

SSL では、公開鍵暗号化技術を使用して認証を行います。公開鍵暗号化では、サーバ用の公開鍵とプライベート キー (秘密鍵) が生成されます。これらのキーは関連付けられているので、公開鍵で暗号化されたデータは、対応するプライベート キーを使用することによってのみ復号化できます。プライベート キーは注意深く保護されているので、その所有者だけが公開鍵で暗号化されたメッセージを復号化できます。

公開鍵は、デジタル証明書に組み込まれます。デジタル証明書には、公開鍵の所有者に関する情報 (名前、番地、電子メール アドレスなど) も組み込まれます。プライベート キーとデジタル証明書は、サーバの ID を提供します。

デジタル証明書に組み込まれたデータは認証局 (信頼性のある CA) によって検証され、その認証局のデジタル証明書でデジタル署名されます。有名な認証局には、Verisign や Entrust.net などがあります。信頼性のある認証局は、サーバの信頼を確立します。

SSL 接続に参加するアプリケーションは、相手側がデジタル署名を評価および承認するときに認証を受けます。Web ブラウザ、サーバ、およびその他の SSL 対応アプリケーションは、信頼性のある認証局によって署名され、他の点でも有効なデジタル証明書を真正として承認します。たとえば、デジタル証明書が期限切れの場合や、署名に使用された認証局のデジタル証明書が期限切れの場合、デジタル証明書は無効になります。サーバのデジタル証明書内のホスト名がクライアントによって指定されたホスト名と一致しない場合、サーバの証明書は無効になります。

一方向および双方向 SSL

SSL は、一方向または双方向としてコンフィグレーションできます。

- 一方向 SSL では、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りますが、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ありません。SSL 接続を成立させるには、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。一方向 SSL は、インターネット上で顧客が個人データを共有する前にセキュアな接続を実現したい場合によく使用されます。
- 双方向 SSL では、クライアントもサーバに対して証明書を提示します。WebLogic Server をコンフィグレーションすることで、クライアントが信頼性のある有効な証明書を提示しなければ SSL 接続を確立できないようにすることができます。

SSL の設定：主な手順

SSL を設定するには、次の手順に従います。

1. WebLogic Server 用の ID (プライベート キーとデジタル証明書) および信頼 (信頼性のある認証局の証明書) を取得します。この手順を行うには、WebLogic Server キットのデジタル証明書、プライベート キー、信頼性のある CA の証明書を使用するか、CertGen ユーティリティ、Certificate Request Generator サーブレット、Sun Microsystem の `keytool` ユーティリティ、または Entrust や Verisign などの信頼性のあるベンダを使用します。
2. プライベート キー、デジタル証明書、信頼性のある CA の証明書を格納します。デジタル証明書は、常に WebLogic Server のドメイン ディレクトリ内のファイルに格納します。プライベート キーと信頼性のある CA の証明書は、WebLogic キーストア プロバイダがアクセスするキーストアか、またはドメイン ディレクトリ内のファイルのいずれかに格納します。
3. SSL ポートを有効にします。デフォルトでは、SSL ポートはパフォーマンスを低下させるため有効になっていません。
4. WebLogic Server Administration Console またはサーバ起動スクリプトで SSL 属性を設定します。SSL 属性では、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 の証明書の格納場所を定義します。また、必要な場合、クライアント証明書の提示を必須とする属性を設定します (双方向 SSL の場合)。

詳細については、以下を参照してください。

6-5 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の取得」

6-15 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の格納」

6-22 ページの「SSL ポートの有効化」

6-23 ページの「一方向 SSL の属性の設定」

6-24 ページの「双方向 SSL の属性の設定」

プライベート キー、デジタル証明書、信頼性のある認証局の取得

SSL を使用するには、プライベート キー、対となる公開鍵が組み込まれたデジタル証明書、および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が必要となります。WebLogic Server は、以下のソースのプライベート キー、デジタル証明書、信頼性のある認証局をサポートしています。

- *WL_HOME*\server\lib ディレクトリ内のデモ用デジタル証明書、プライベート キー、および信頼性のある認証局。

  コンフィグレーション ウィザードでドメインを作成すると、デモ用の証明書とプライベート キーがドメイン ディレクトリにコピーされます。デモ用のデジタル証明書、プライベート キー、および信頼性のある認証局は、プロダクション環境用ではなくデモまたはテスト目的で使用してください。

- CertGen ユーティリティ。このユーティリティは、プロダクション環境用ではなくデモまたはテスト目的専用のデジタル証明書とプライベート キーを生成します。デジタル証明書に有効期限を設定したり、デジタル証明書に正しいホスト名を指定してホスト名検証を使用できるようにしたりする場合は、Cert Gen ユーティリティを使用します。Cert Gen ユーティリティを使用してプライベート キーとデジタル証明書を取得する方法については、6-7 ページの「Cert Gen ユーティリティの使い方」を参照してください。

- Certificate Request Generator サーブレット。このサーブレットは、プライベート キー ファイルと証明書署名リクエスト (CSR) を生成します。信頼性のある CA (VeriSign や Entrust.net など) からデジタル証明書を取得するには、CSR と呼ばれる特定のフォーマットでリクエストを提出する必要があります。CA から受領したデジタル証明書と Certificate Request Generator サーブレットから受領したプライベート キーは、プロダクション環境に適しています。詳細については、6-9 ページの「Certificate Request Generator サーブレットの使い方」を参照してください。
- Sun Microsystem の `keytool` ユーティリティ。このユーティリティを使用すると、プライベート キー、WebLogic Server 用の自己署名デジタル証明書、および CSR を取得できます。CSR を認証局に提出すると、WebLogic Server 用のデジタル証明書を取得できます。`keytool` は、自己署名デジタル証明書を新しいデジタル証明書に更新するために使用します。Sun の `keytool` ユーティリティの詳細については、『keytool － Key and Certificate Management Tool』を参照してください。

WebLogic Server は、`.pem` または `.der` フォーマットのデジタル証明書を使用できます。

`.pem` (privacy-enhanced mail) フォーマットのファイルは次の行で始まります。

```
----BEGIN CERTIFICATE----
```

また、次の行で終わります。

```
----END CERTIFICATE----
```

`.pem` フォーマット ファイルは、複数のデジタル証明書をサポートしています (たとえば証明書チェーンを含めることができます)。ただし、ファイル内のデジタル証明書の順序は重要です。たとえば、認証局 A、認証局 B (認証局 A の発行元)、認証局 C (認証局 B の発行元) のようにルート認証局まで並んでいる必要があります。

`.der` フォーマットのファイルにはバイナリ データが含まれます。`.der` ファイルは単一の証明書用にしか使用できませんが、`.pem` ファイルは複数の証明書用に使用できます。

Microsoft は認証局としてよく利用されます。Microsoft は、信頼性のある CA の証明書を p7b フォーマットで発行します。この証明書は PEM に変換してからでないと、WebLogic Server では使用できません。詳細については、6-13 ページの「Microsoft p7b フォーマットから PEM フォーマットへの変換」を参照してください。

プライベート キー ファイル (キーストア内に格納されていないプライベート ファイル) は PKCS#5/PKCS#8 PEM フォーマットでなければなりません。

Cert Gen ユーティリティの使い方

注意： CertGen ユーティリティは、プロダクション環境用ではなくデモまたはテスト目的専用のデジタル証明書とプライベート キーを生成します。

CertGen ユーティリティは、プライベート キーおよびデモ用認証局 (`CertGenCA`) によって署名されたデジタル証明書を作成します。CertGen ユーティリティによって生成されたデジタル証明書は、対象マシンのホスト名を共通名として持ちます。ホスト名検証を使用する場合、デジタル証明書は SSL を使用するマシンごとに生成する必要があります。

CertGen ユーティリティは、2 つの `.pem` ファイルと 2 つの `.der` ファイルを生成します。Web ブラウザで `.der` ファイルを参照して、生成された証明書の詳細を確認します。`.pem` ファイルは、WebLogic Server を起動するときか、またはクライアントでデジタル証明書を使用するときに使用します。

証明書を生成するには、以下の操作を行います。

1. CertGen ユーティリティを実行しているディレクトリに、次のファイルをコピーします。
   - `WL_HOME\server\lib\CertgenCA.der` - WebLogic Server によって信頼されている認証局のデジタル証明書
   - `WL_HOME\server\lib\CertGenCAKey.der` － WebLogic Server によって信頼されている認証局のプライベート キー
2. コマンド プロンプトに対して、次のコマンドを入力します。

   prompt> `java utils.CertGen` *password certfile keyfile* `[export] [`*`hostname`*`]`

   各要素の説明は次のとおりです。
   - *`password`* はプライベート キーのパスワード。
   - *`certfile`* はデジタル証明書ファイルの名前。このファイルはドメイン ディレクトリに置きます。
   - *`keyfile`* は生成されたプライベート キー ファイルの名前。このファイルはドメイン ディレクトリに置きます。

- *hostname* はデジタル証明書の取得対象のマシン名。このオプションを指定すると、ホスト名検証を使用することができます。

デフォルトでは、CertGen ツールは国内向けレベルの証明書を生成します。ツールで国外向けレベルの証明書を生成する場合は、[export] オプションを指定します。ホスト名を使用する国内向けレベルのデジタル証明書を輸出するには、[export] を " " と指定します。

CertGen ツールは、JDK バージョン 1.3 の `InetAddress.getLocalHost().getHostname()` メソッドを使用して、サブジェクトの共通名に格納するホスト名を取得します。`getHostName()` メソッドの動作は、プラットフォームによって異なります。Solaris などのプラットフォームでは完全修飾ドメイン名 (FQDN) を返し、Windows NT などでは短いホスト名を返します。WebLogic Server がクライアントとして動作する場合 (およびデフォルトでホスト名検証が有効化されている場合)、URL に指定されるホスト名がサブジェクトの共通名フィールドと一致する必要があります。一致しない場合、接続は失敗します。

Solaris では、コマンド ラインで `hostname` と入力すると、サーバは `/etc/hosts` ファイルを検索して短いホスト名を取得します。`java.net.InetAddress.getHostName()` を呼び出すと、ホストは `/etc/nsswitch.conf` ファイルを検索し、そのホストのコンフィグレーションに応じて FQDN または短いホスト名を返します。

一方、ホスト エントリが次のようにコンフィグレーションされているとします。

```
hosts:   dns nis [NOTFOUND=return]
```

この場合、ホストはまずネーム サービス ルックアップを実行し、DNS を利用できなかった場合にのみ `/etc/hosts` ファイルを使用します。このケースでは、FQDN が返されます。

一方、ホスト エントリが次のようにコンフィグレーションされているとします。

```
hosts:   files dns nis [NOTFOUND=return]
```

この場合、ホストはまず `/etc/hosts` ファイルを検索し、次に DNS を利用します。このケースでは、短いホスト名が返されます。

Certificate Request Generator サーブレットの使い方

WebLogic Server デプロイメントをプロダクション環境で使用する前に、プライベート キーと証明書を信頼性のある認証局 (VeriSign や Entrust.net など) から取得しておく必要があります。認証局 (CA) からデジタル証明書を取得するには、CSR と呼ばれる特定のフォーマットでリクエストを提出する必要があります。Certificate Request Generator サーブレットは、ユーザから情報を収集して、プライベート キー ファイルと CSR を生成します。生成された CSR を、認証局に提出します。

CSR を生成するには、次の手順に従います。

1. `certificate.war` ファイルを applications ディレクトリにコピーします (このファイルはサーバの起動前かサーバの実行中にコピーします)。この手順は、コンフィグレーション ウィザードによって実行されます。
2. Web ブラウザで、Certificate Request Generator サーブレットの URL を次のように入力します。

   `http` (または `https`)`://`*`hostname:port`*`/certificate/`

   URL の各要素は次のように定義します。

   - *`hostname`* は、WebLogic Server を実行しているマシンの DNS 名。
   - *`port`* は、WebLogic Server が SSL 接続をリスンするポートの番号。デフォルトでは 7002。7002 は、WebLogic Server が SSL 通信をリスンするデフォルト ポートです。WebLogic Server が通信をリスンするポートを任意に指定できます。

     たとえば、WebLogic Server が `ogre` というマシン上で動作しており、Certificate Request Generator サーブレットを実行するために SSL 通信をデフォルト ポートの `7002` でリスンするようコンフィグレーションされている場合は、Web ブラウザに次の URL を入力しなければなりません。

     `https://ogre:7002/certificate/`
3. Certificate Request Generator サーブレットによって、Web ブラウザからフォームがロードされます。次の表の情報を参照して、ブラウザに表示されたフォームに必要な情報を入力します。

表 6-1 Certificate Request Generator フォームのフィールド

フィールド	説明
[Country code]	国ごとの 2 文字の ISO コード。アメリカのコードは US。
[Organizational unit name]	組織の事業部、部、またはその他の運営単位の名前。
[Organization name]	組織の名前。認証局が、この組織に登録されているドメインに所属するホスト名をこの属性に入力するよう要求する場合がある。
[Email address]	管理者の E メール アドレス。このアドレスがデジタル証明書の送信先になる。
[Full host name]	デジタル証明書のインストール先となる WebLogic Server の完全修飾名。この名前は、WebLogic Server の DNS ルックアップ用の名前 (たとえば `node.com`) である。Web ブラウザでは、URL のホスト名とデジタル証明書の名前を比較する。ホスト名を後で変更した場合は、新しいデジタル証明書を要求しなければならない。
[Locality name (city)]	市または町の名前。市で付与されたライセンスを使用して運用する場合は、この属性は必須、つまり、ライセンスを付与された市の名前を入力しなければならない。
[State name]	組織の所在地がアメリカまたはカナダの場合に、組織が業務を行っている州の名前。短縮してはならない。
[Private Key Password]	プライベート キーの暗号化に使用するパスワード。 WebLogic Server で保護されたキーを使用するために、このフィールドにパスワードを入力する。キーが使用される場合は常にパスワードが要求される。サーブレットで、PKCS-8 で暗号化されたプライベート キーが作成される。

表 6-1 Certificate Request Generator フォームのフィールド

フィールド	説明
[Strength]	生成するキーの長さ (ビット単位)。キーが長いほど、暗号の解読はより困難になる。 国内バージョンの WebLogic Server では、512 ビット、1024 ビット、または 2048 ビットのキーを選択できる。1024 ビットのキーが望ましい。

4. [Generate Request] ボタンをクリックします。

   必須属性が空白の場合、または属性に無効な値が指定されている場合は、Certificate Request Generator サーブレットによってメッセージが表示されます。メッセージが表示された場合は、ブラウザの [戻る] ボタンをクリックして、エラーを修正します。

   すべての属性が受け付けられると、Certificate Request Generator サーブレットは次のファイルを WebLogic Server のスタート ディレクトリに作成します。

   - *hostname*-key.der ー プライベート キー ファイル。
   - *hostname*-request.dem ー バイナリ フォーマットの証明書リクエスト ファイル。
   - *hostname*-request.pem ー 認証局に提出する CSR ファイル。このファイルの内容は .dem ファイルと同じデータですが、E メールにコピーしたり、Web フォームに貼り付けたりできるように、ASCII でエンコードされています。

5. 認証局を選択し、その認証局の Web サイトの指示に従って、デジタル証明書を購入します。

   - VeriSign, Inc. では、WebLogic Server 用に 2 つのオプションを用意しています。1 つは、国内および国外用の Web ブラウザ向けの強力な 128 ビット暗号化を特徴とする Global Site Services、もう 1 つは、国内用 Web ブラウザに 128 ビットの暗号を、国外用 Web ブラウザには 40 ビットの暗号を提供する Secure Site Services です。
   - Entrust.net の証明書は、国内用 Web ブラウザに 128 ビットの暗号を、国外用 Web ブラウザに 40 ビットの暗号を提供します。

証明書チェーンの使い方

WebLogic Server では、証明書チェーンを使用することができます。

WebLogic Server で証明書チェーンを使用するには、次の手順に従います。

1. すべてのデジタル証明書が PEM フォーマットであることを確認します。デジタル証明書が DER フォーマットの場合、der2pem ユーティリティを使用してフォーマットを変換することができます。Microsoft から発行されたデジタル証明書を使用する場合は、6-13 ページの「Microsoft p7b フォーマットから PEM フォーマットへの変換」を参照してください。ここでは、その他のタイプのデジタル証明書を変換する手順について説明します。必要な作業は、デジタル証明書を Base 64 フォーマットで保存することだけです。

2. テキスト エディタを起動して、すべてのデジタル証明書ファイルを 1 つのファイルに含めます。順序は重要です (ファイルは信頼の順序に従って含めます)。サーバのデジタル証明書はファイル内の最初のデジタル証明書でなければなりません。そのデジタル証明書の発行元が次のファイルとなるようにして続けて、最後をルート認証局の自己署名デジタル証明書にします。このデジタル証明書はファイル内の最後のデジタル証明書でなければなりません。

   デジタル証明書とデジタル証明書の間には空白行を入れないでください。

3. WebLogic Server Administration Console の [SSL Attributes] タブの [サーバ証明書ファイル] にファイルを指定します。

コード リスト 6-1 は証明書チェーンのサンプルです。

コード リスト 6-1 証明書チェーンのサンプル ファイル

```
-----BEGIN CERTIFICATE-----
MIICyzCCAjSgAwIBAgIBLDANBgkqhkiG9w0BAQQFADCBtjELMAkGA1UEBhMCVVMxEzARBgNVBAgTCk
NhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiBGcmFuY2lzY28xFTATBgNVBAoTDEJFQSBXZWJMb2dpYzERMA
8GA1UECxMIU2VjdXJpdHkxLzAtBgNVBAMTJkRlbW8gQ2VydGlmaWNhdGUgQXV0aG9yaXR5IENvbnN0cm
FpbnRzMR8wHQYJKoZIhvcNAQkBFhBzZWN1cml0eUBiZWEuY29tMB4XDTAyMTEwMTIwMDIxMloXDTA2MTA
xNTIwMDIxMlowgZ8xCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRUwEwYDVQQKEwxCRUEgV2ViTG9naWMxETAPBgNVBAsTCFNlY3VyaXR5MRkwFwYDVQQDE
xB3ZWJsb2dpYy5iZWEuY29tMR4wHAYJKoZIhvcNAQkBFg9zdXBwb3J0QGJlYS5jb20wgZ8wDQYJKoZIhv
cNAQEBBQADgY0AMIGJAoGBAMJX8nKUgsFej8pEu/1IVcHUkwY0c2JbBzOryu3sce4QjX+rGxiCjoPm2MY
yts2BvonuJ6CztdZf8B/LBEWCz+qRrtdFn9mKSZWGvrAkmMPz2RhXEOThpoRo5kZz2FQ9XF/PxIJXTYCM
7yooRBwXoKYjquRwiZNtUiU9kYi6Z3prAgMBAAEwDQYJKoZIhvcNAQEEBQADgYEAh2eqQGxEMUnNTwEUD
0tBq+7YuAkjecEocGXvi2G4YSoWVLgnVzJoJuds3c35KE6sxBe1luJQuQkE9SzALG/6lDIJ5ctPsHFmZz
ZxY7scLl6hWj5ON8oN2YTh5Jo/ryqjvnZvqiNIWe/gqr2GLIkajC0mz4un1LiYORPig3fBMH0=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
```

```
MIIC+jCCAmOgAwIBAgIBADANBgkqhkiG9w0BAQQFADCBtjELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhb
Glmb3JuaWExFjAUBgNVBAcTDVNhbiBGcmFuY2lzY28xFTATBgNVBAoTDEJFQSBXZWJMb2dpYzERMA8GA1
UECxMIU2VjdXJpdHkxLzAtBgNVBAMTJkRlbW8gQ2VydGlmaWNhdGUgQXV0aG9yaXR5IENvbnN0cmFpbnR
zMR8wHQYJKoZIhvcNAQkBFhBzZWN1cml0eUBiZWEuY29tMB4XDTAyMTEwMTIwMDIxMVoXDTA2MTAxNjIw
MDIxMVowgbYxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4gRnJhb
mNpc2NvMRUwEwYDVQQKEwxCRUEgV2ViTG9naWMxETAPBgNVBAsTCFNlY3VyaXR5MS8wLQYDVQQDEyZEZW
1vIENlcnRpZmljYXRlIEF1dGhvcml0eSBDb25zdHJhaW50czEfMB0GCSqGSIb3DQEJARYQc2VjdXJpdHl
AYmVhLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA3ynD8l5JfLob4g6d94dNtI0Eep6QNl9b
blmswnrjIYz1BVjjRjNVal9fRs+8jvm85kIWlerKzIMJgiNsj50WlXzNX6orszggSsW15pqV0aYE9Re9K
CNNnORlsLjmRhuVxg9rJFEtjHMjrSYr2IDFhcdwPgIt0meWEVnKNObSFYcCAwEAAaMWMBQwEgYDVR0TAQ
H/BAgwBgEB/wIBATANBgkqhkiG9w0BAQQFAAOBgQBS+0oqWxGyqbZO028zf9tQT2RKojfuwywrDoGW96U
n5IqpFnBHIu5atliJo3OUpiH18KkwLN8DVP/3t3K3O3kXdIuLbqAL0i5xyBlAhr7gE5eVhIyeMg7ETBPL
yGO2BF13Y24LlsO+MX9jW7fxMraPN608QeJXkZw0E0cGwrw2AQ==
-----END CERTIFICATE-----
```

Microsoft p7b フォーマットから PEM フォーマットへの変換

Microsoft は認証局としてよく利用されます。Microsoft から発行されるデジタル証明書のフォーマット (p7b) は、WebLogic Server では使用できません。p7b フォーマットのデジタル証明書を PEM フォーマットに変換するには、次の手順に従います。

1. Windows 2000 の Windows エクスプローラで、変換するファイル (*filename*.p7b) をクリックします。

   証明書ウィンドウが表示されます。

2. 証明書ウィンドウの左ペインで、変換するファイルを展開します。
3. 証明書オプションを選択します。

   p7b ファイル内の証明書のリストが表示されます。

4. PEM フォーマットに変換する証明書を選択します。

   [Certificate Export] ウィザードが表示されます。

5. [次へ] をクリックします。
6. `Base64 Encoded Cert` オプションを選択します (PEM フォーマットをエクスポートします)。
7. [次へ] をクリックします。
8. 変換したデジタル証明書の名前を入力します。

9. [完了] をクリックします。

注意 : 証明書チェーンが含まれている p7b 証明書ファイルの場合、発行者の PEM デジタル証明書と証明書ファイルを連結する必要があります。連結されたデジタル証明書は、WebLogic Server で使用できます。

独自の認証局の使い方

企業の多くは、独自の認証局にもなっています。こうした信頼性のある認証局を WebLogic Server で使用するには、次の手順に従います。

1. 信頼性のある CA の証明書が PEM フォーマットであることを確認します。デジタル証明書が DER フォーマットの場合、der2pem ユーティリティを使用してフォーマットを変換することができます。Microsoft から発行されたデジタル証明書を使用する場合は、6-13 ページの「Microsoft p7b フォーマットから PEM フォーマットへの変換」を参照してください。
2. 信頼性のある CA の証明書を、ファイルまたは WebLogic キーストア プロバイダがアクセスする JKS キーストアに格納します。詳細については、6-15 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の格納」を参照してください。
3. その信頼性のある認証局を使用するよう、WebLogic Server をコンフィグレーションします。詳細については、6-23 ページの「一方向 SSL の属性の設定」を参照してください。

Web ブラウザ用のデジタル証明書の取得

低レベルのセキュリティのブラウザ証明書は、Web ブラウザを使って簡単に取得できます (通常は、[Option] または [Preferences] の [Security] メニューを選択します)。[個人用証明書] の項目に移動して、新しいデジタル証明書を要求します。要求者の個人情報が求められます。

受け取ったデジタル証明書には、名前、公開鍵、および第三者による認証を受けたいその他の情報 (たとえば電子メール アドレス) などの公開情報が入っています。デジタル証明書は、認証が要求されたときに提示します。

デジタル証明書の取得プロセスの一部として、Web ブラウザは公開鍵 / プライベート キーのペアを生成します。プライベート キーは秘密にしておく必要があります。デジタル証明書の取得プロセス自体が安全に行われるように、プライベート キーはローカル ファイル システムに格納し、Web ブラウザを操作するマシン上に置かないようにしてください。一部のブラウザでは、保存されていないパスワードを使用してプライベート キーを暗号化することができます。プライベート キーを暗号化する場合、Web ブラウザではセッションごとにパスワードの入力を少なくとも 1 回求められます。

デジタル証明書が複数のブラウザ (または同じブラウザで異なるバージョン) に関して常に有効とはならないことに注意してください。

プライベート キー、デジタル証明書、信頼性のある認証局の格納

プライベート キー、デジタル証明書、信頼性のある CA を取得したら、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。デジタル証明書はファイルだけに格納できます。プライベート キーは、ファイルまたは WebLogic キーストア プロバイダがアクセスするキーストアに格納できます。信頼性のある CA の証明書はファイルまたは JKS キーストアに格納できます。キーストアは、WebLogic キーストア プロバイダからアクセスしたり、コマンドラインで指定したりできます。

- 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 の証明書をファイルに格納する場合、WebLogic Server Administration Console で SSL の [サーバ キー ファイル名]、[サーバ証明書ファイル名]、および [信頼性のある CA ファイル名] 属性を設定する必要があります。SSL 属性の設定については、6-23 ページの「一方向 SSL の属性の設定」を参照してください。
- プライベート キーと信頼性のある CA の証明書をキーストアに格納する場合、キーストアを作成し、そのキーストアにプライベート キーと信頼性のある CA の証明書をロードする必要があります。このリリースの WebLogic Server では JKS キーストアのみサポートされています。

  プライベート キーに関しては、WebLogic キーストア プロバイダがそのキーストアを指し示すようにコンフィグレーションし、WebLogic Server

Administration Console で SSL の [サーバ プライベート キーのエイリアス] および [サーバ プライベート キーの Pass Phrase] 属性を設定する必要があります。

信頼性のある認証局に関しては、WebLogic キーストア プロバイダがキーストアを指し示すようにコンフィグレーションし、WebLogic Server Administration Console で [信頼性のある CA ファイル名] 属性を設定するか、またはサーバ起動スクリプトで `-Dweblogic.security.SSL.trustedCAkeystore` コマンドライン引数を使用してキーストアを指定します。

詳細については、次を参照してください。

- 6-16 ページの「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」
- 6-19 ページの「WebLogic Server キーストア プロバイダがキーストアを指し示すようにコンフィグレーションする」
- 6-23 ページの「一方向 SSL の属性の設定」

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

キーストアは、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 の証明書を格納するファイルを作成および管理するためのメカニズムです。このリリースの WebLogic Server では JKS キーストアのみサポートされています。WebLogic キーストア プロバイダは、キーストアのインスタンスを検索します。次のユーティリティを使用して、キーストアを作成し、プライベート キーと信頼性のある CA の証明書をそのキーストアにロードします。

- WebLogic `ImportPrivateKey` ユーティリティ。`ImportPrivateKey` ユーティリティを使用すると、プライベート キー ファイルをキーストアにロードすることができます。詳細については、『管理者ガイド』の「ImportPrivateKey」を参照してください。
- Sun Microsystem の keytool ユーティリティ。keytool ユーティリティを使用すると、プライベート キー / デジタル証明書のペアを生成してから、署名済みのプライベート キーをキーストアにインポートすることができます。詳細

については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。

注意： keytool ユーティリティでは、プライベート キーをキーストアにインポートするときに各プライベート キーのデジタル証明書が必要になります。このデジタル証明書は、WebLogic Server では使用されません。

keytool ユーティリティでは新しいプライベート キーと信頼性のある CA の証明書をキーストアにロードできますが、既存のプライベート キーをファイルからキーストアにインポートすることはできません。代わりに WebLogic ImportPrivateKey ユーティリティを使用してください。

プライベート キーと信頼性のある CA の証明書用に 1 つのキーストアを作成するか、またはプライベート キー用に 1 つ、信頼性のある CA の証明書用に 1 つ、合わせて 2 つのキーストアを作成します。ID と信頼の両方で 1 つのキー ストアを使用することもできますが、以下の理由により、ID と信頼で別々のキー ストアを使用することをお勧めします。

ID キー ストア (プライベート キーとデジタル証明書のペア) と信頼キー ストア (信頼性のある CA 証明書) は、セキュリティ要件が異なる場合があります。次に例を示します。

- 信頼キー ストアはネットワーク経由で配布できるが、ID キー ストアは会社の方針によってネットワークに乗せられない場合がある
- ID キー ストアはオペレーティング システムによって非認可ユーザの読み込みと書き込みの両方から保護される場合があるが、信頼キー ストアで必要なのは書き込みの保護のみ
- 信頼キー ストアのパスワードは通常、ID キー ストアのパスワードよりも多くの人に知られる

ID については、証明書 (非機密データ) をキー ストアに格納するだけで十分ですが、信頼については、証明書とプライベート キー (機密データ) をキー ストアに格納する必要があります。

マシンはドメイン全体で同じ信頼ルールを使用する傾向がありますが (つまり、同じ信頼性のある CA のセットを使用する)、ID はサーバごとになる傾向があります。ID はプライベート キーを必要とし、プライベート キーはマシン間でコピーを行うべきではありません。したがって、そのマシンに必要なサーバ ID のみ含まれるキー ストアがマシンごとに別個に作成されます。ただし、信頼キー ストアはマシン間でコピーできるので、信頼ルールの標準化が容易になります。

ID は、多くの場合 nCipher などのハードウェア キー ストアに格納されます。信頼は証明書だけでプライベート キーがないので、ファイルベースの JDK キー ストアにセキュリティ上の問題なく格納することができます。

デフォルトでは、WebLogic Server はドメイン ディレクトリから `wlDefaultKeyStore.jks` というプライベート キー キーストアを検索します。信頼性のある CA の証明書用のキーストアは、デモ用の信頼性のある認証局 (*`WL_HOME`*`\server\lib\cacerts`) または JDK の信頼性のある認証局 (`...\jre\lib\security\cacerts`) のいずれかを使用できます。

注意： *`WL_HOME`*`\server\lib\cacerts` ファイルは、プロダクション環境用ではなくデモまたはテスト目的で使用してください。このファイルには、`\jre\lib\security\cacerts` 内の信頼性のある認証局が入っています。

WebLogic Server は、固有のエイリアスを介してキーストアのすべてのプライベート キー エントリにアクセスします。エイリアスは、プライベート キーをキーストアにロードするときに指定します。

信頼性のある認証局は、エイリアスによって WebLogic Server に対して個々に識別されません。WebLogic Server によって信頼性があると識別されたキーストア内の認証局はすべて信頼されます。WebLogic Server は信頼性のある CA の証明書にアクセスする場合にエイリアスを使用しませんが、キーストアでは、信頼性のある CA の証明書をロードするときにエイリアスが必要です。

エイリアスの大文字 / 小文字は区別されません。このため、`Hugo` と `hugo` は同じキーストア エントリを指します。プライベート キーのエイリアスは、SSL をコンフィグレーションするときに [サーバ プライベート キーのエイリアス] 属性で指定します。

よく使う Keytool のコマンド

表 6-2 では、WebLogic Server で JKS キーストアを作成および使用する場合の `keytool` のコマンドについて説明します。

注意： `keytool` ユーティリティは Sun Microsytems の製品です。したがって、BEA Systems ではこのユーティリティに関する詳細なマニュアルを提供していません。詳細については、『keytool - Key and Certificate Management Tool』を参照してください。

表 6-2 よく使われる Keytool のコマンド

コマンド	説明
`keytool -genkey -keystore` *`keystorename`* `-storepass` *`keystorepassword`*	キーストアを作成する。
`keytool -alias` *`aliasforprivatkey`* `-import -file` *`privatekeyfile.pem`* `-keypass` *`privatekeypassword`* `-keystore` *`keystorename`* `-storepass` *`keystorepassword`*	プライベート キーをキーストアにロードする。
`keytool -alias` *`aliasfortrustedca`* `-trustcacerts -import -file` *`trustedcafilename.pem`* `-keystore` *`keystorename`* `-storepass` *`keystorepassword`*	信頼性のある CA の証明書をキーストアにロードする。
`keytool -list -keystore` *`keystorename`*	キーストアの内容を表示する。
`keytool -keystore` *`keystorename`* `-storepass` *`keystorepassword`* `-delete -alias` *`privatekeyalias`*	指定したエイリアスに対応するプライベート キーをキーストアから削除する。
`keytool -help`	keytool のオンライン ヘルプを表示する。

WebLogic Server キーストア プロバイダがキーストアを指し示すようにコンフィグレーションする

WebLogic キーストア プロバイダを使用する場合、プライベート キーのパスワードを含む `config.xml` ファイルはハッシュ化されます。これによって、プライベート キーと信頼性のある CA の証明書をファイルに格納するときには得られないセキュリティが実現されます。

WebLogic キーストア プロバイダのコンフィグレーションは任意です。プライベート キーとデジタル証明書をファイルに格納する場合、WebLogic キーストア プロバイダをコンフィグレーションする必要はありません。JKS キーストアは、WebLogic キーストア プロバイダを使用しなくても利用できます。詳細については、6-21 ページの「JKS キーストアの使い方」を参照してください。

WebLogic キーストア プロバイダをコンフィグレーションするには、キーストアを作成し、WebLogic Server Administration Console で WebLogic キーストア プロバイダがそのキーストアを指し示すように属性を設定する必要があります。

注意： WebLogic キーストア プロバイダは、ドメイン単位でコンフィグレーションします。しかし、キーストアはマシン単位でコンフィグレーションします。同じマシン上で複数のセキュリティ レルムまたは複数のサーバが動作している場合、それらはすべて同じキーストアを使用できます。

WebLogic キーストア プロバイダをコンフィグレーションするには、次の手順に従います。

1. キーストアを作成します。詳細については、6-16 ページの「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。
2. WebLogic Server Administration Console で [セキュリティ | レルム] ノードを展開します。
3. コンフィグレーションするレルムの名前 (*myrealm* など) をクリックします。
4. [プロバイダ] ノードを展開します。
5. [キー ストア] をクリックします。

   [キー ストア] タブが表示されます。このタブには、このセキュリティ レルムにコンフィグレーションされているキーストアの名前が表示されます。デフォルトでは、WebLogic キーストア プロバイダがコンフィグレーションされます。

   注意： WebLogic Server Administration Console は、WebLogic キーストア プロバイダを DefaultKeystore として参照します。

6. [DefaultKeystore] をクリックします。
7. [一般] タブの [プライベート キー ストアの場所] 属性に、キーストアのディレクトリ位置を入力します。デフォルトは `wlDefaultKeyStore.jks` です。

この属性には、ディレクトリとファイルの場所を、絶対パスまたはサーバのルート ディレクトリ (つまりドメイン ディレクトリ) を基準にした相対パスで指定する必要があります。

警告： サポートされているすべてのオペレーティング システムで、キーストアはドメイン内のすべてのサーバに対して同じ場所でなければなりません。同じ場所にないと、WebLogic Server はキーストアを見つけられません。

8. [ルート CA キー ストアの場所] 属性に、信頼性のある CA の証明書が格納されるキーストアのディレクトリ位置を入力します。

   この属性には、ディレクトリとファイルの場所を、絶対パスまたはサーバのルート ディレクトリ (つまりドメイン ディレクトリ) を基準にした相対パスで指定する必要があります。この手順は、プライベート キーと信頼性のある CA の証明書が同じキーストアに格納される場合は必要ありません。

   サーバが信頼する認証局を格納するキーストアの場所を `-Dweblogic.security.SSL.trustedCAKeyStore` コマンドライン引数を使用して指定する場合は、[ルート CA キー ストアの場所] 属性は空白にしておきます。

9. [プライベート キー ストアの Pass Phrase] 属性に、信頼性のある CA に作成したときに指定したパスワードを入力します。

10. [適用] をクリックして変更を保存します。

11. WebLogic Server を再起動します。

JKS キーストアの使い方

WebLogic キーストア プロバイダを使用して信頼性のある CA の証明書とプライベート キーを格納しない場合、WebLogic Server の起動時に JKS キーストアの場所をコマンド ライン引数として指定できます。

`-Dweblogic.security.SSL.trustedCAKeyStore` コマンド ライン引数を使用して、サーバまたはクライアントが信頼する認証局が格納されているキーストアの保存場所を指定します。パスの値は、サーバが起動するディレクトリを基準とする相対パスまたは絶対パスでなければなりません。

Weblogic Server は、次の順序で信頼性のある認証局を検索します。

- `-Dweblogic.security.SSL.trustedCAkeystore` コマンドライン引数に指定された JKS キーストアを使用する。
- WebLogic キーストア プロバイダの [ルート CA キー ストアの場所] 属性に指定されているキーストアを使用する。
- 6.x の `config.xml` ファイル (ドメイン ディレクトリ内に存在する場合) 内の信頼性のある CA の証明書ファイルを使用する。
- *`WL_HOME`*`\server\lib\cacerts` を使用する。

SSL ポートの有効化

この手順では、個々のサーバと SSL クライアントの間で明示的に SSL を使用するよう指定する方法を説明します。WebLogic Server は内部で SSL を使用して、管理サーバ、管理対象サーバ、およびノード マネージャ間、WebLogic Server と LDAP 認証プロバイダが使用する LDAP サーバ間、WebLogic Server と SSL クライアント コード間、および管理ポートを介した通信を保護します。

SSL ポートを有効にするには、次の手順を実行します。

1. [サーバ] ノードを展開します。
2. [接続 | SSL ポート] タブを選択して以下の属性を設定します。
   - [有効化] 属性を選択すると、SSL が有効化されます。
   - [SSL リスン ポート] 属性は、WebLogic Server が SSL 接続をリスンする専用ポートの番号です。デフォルトでは 7002。
3. [適用] をクリックして変更を保存します。
4. 一方向または双方向 SSL の属性を設定します。詳細については、6-23 ページの「一方向 SSL の属性の設定」または 6-24 ページの「双方向 SSL の属性の設定」を参照してください。
5. WebLogic Server を再起動します。

一方向 SSL の属性の設定

一方向 SSL の属性を定義するには、次の手順に従います。

1. SSL ポートを有効にします。詳細については、6-22 ページの「SSL ポートの有効化」を参照してください。
2. [接続 | SSL] タブを選択し、表 6-3 に示す属性を設定します。

表 6-3 SSL 属性

SSL 属性	説明
[サーバ プライベート キーのエイリアス]	WebLogic Server のプライベート キーをキーストアにロードするときに指定されるエイリアス。この属性は、WebLogic Server のプライベート キーをキーストアに格納した場合にのみ定義する。この属性を使用するには、WebLogic キーストア プロバイダをコンフィグレーションしておく必要がある。 すべてのプライベート キー キーストア エントリには、固有のエイリアスを介してアクセスできる。エイリアスは、プライベート キーをキーストアにロードするときに指定する。エイリアスの大文字 / 小文字は区別されない。このため、Hugo と hugo は同じキーストア エントリを指す。
[サーバ プライベート キーの Pass Phrase]	WebLogic Server のプライベート キーをキーストアにロードするときに指定されるパスワード。この属性は、WebLogic Server のプライベート キーをキーストアに格納した場合にのみ定義する。この属性を使用するには、WebLogic キーストア プロバイダをコンフィグレーションしておく必要がある。

表 6-3 SSL 属性

SSL 属性	説明
[サーバ証明書ファイル名]	WebLogic Server 用のデジタル証明書のディレクトリ パスと名前。 2 つの証明書以上の深さの証明書チェーンを使用する場合、チェーン全体を PEM フォーマットで証明書ファイルに含める必要がある。
[サーバ キー ファイル名]	WebLogic Server 用のプライベート キーのディレクトリ パスと名前。この属性は、WebLogic Server のプライベート キーをキーストアに格納した場合にのみ指定する。 プライベート キー ファイルをパスワードで保護している場合、サーバの起動時に `weblogic.management.pkpassword` コマンドライン引数を指定する。
[信頼性のある CA ファイル名]	PEM エンコードされた信頼性のある認証局の証明書を格納したファイルの名前。

3. [適用] をクリックして変更を保存します。
4. WebLogic Server を再起動します。

双方向 SSL の属性の設定

双方向 SSL では、SSL 接続を確立するためにクライアントは WebLogic Server にデジタル証明書を提示する必要があります。最初にクライアントがサーバのデジタル証明書を認証し、次にサーバがクライアントのデジタル証明書を認証します。

双方向 SSL を使用する前に、SSL クライアントとサーバには ID が必要です。また、クライアントとサーバは互いの証明書の発行元を信頼している必要があります。

双方向 SSL の属性を定義するには、次の手順に従います。

1. SSL を有効化します。
2. [接続 | SSL] タブを選択し、6-23 ページの「一方向 SSL の属性の設定」に示した一方向 SSL の属性を設定します。
3. [SSL] タブで、以下の双方向 SSL 属性のいずれかを有効にします。
   - [クライアント証明書を強制] 属性をチェックして、クライアントからの証明書の提示を強制します。有効かつ信頼性のある証明書チェーンが提示されない場合は SSL 接続が終了します。
   - [クライアント証明書を要求 (強制しない)] 属性をチェックして、クライアントからの証明書の提示を要求します。この属性を有効にした場合、クライアントが証明書を提示しなくても接続は継続します。
4. [適用] をクリックします。
5. WebLogic Server を再起動します。

SSL のコマンドライン引数

ファイル内のプライベート キーを WebLogic Server で使用するには、次のコマンド ライン引数を使用します。

```
-Dweblogic.management.pkpassword=pkpassword
```

password は、プライベート キーのパスワードです。

注意 : パスワードはクリア テキストで扱われるので、このコマンドライン引数を使用する際にはセキュリティ上のリスクがあります。

WebLogic キーストア プロバイダの [ルート CA キー ストアの場所] 属性に指定されているキーストア以外の信頼性のある CA キーストアを指定するには、次のコマンド ライン引数を使用します。

```
-Dweblogic.security.SSL.trustedCAkeystore=path
```

パスの値は、サーバの起動ディレクトリを基準とした JKS キーストア ファイルの相対パス名または絶対パス名でなければなりません。このコマンドライン引数は、WebLogic Server Administration Console で指定された情報をオーバライドします。WebLogic Server が信頼を取得する方法の詳細については、6-21 ページの「JKS キーストアの使い方」を参照してください。

Basic Constraints が含まれた証明書の検証を WebLogic Server で制御するには、次のコマンド ライン引数を使用します。

`-Dweblogic.security.SSL.enforceConstraints=`*`option`*

デフォルトは `strong` です。詳細については、6-48 ページの「SSL 証明書検証」を参照してください。

注意： `weblogic.security.SSL.sessionCache.size` および `weblogic.security.SSL.sessionCache.ttl` コマンドライン引数は、このリリースの WebLogic Server ではサポートされていません。詳細については、6-28 ページの「SSL セッションの動作」を参照してください。

SSL のデバッグの有効化

SSL デバッグを有効にすると、SSL ハンドシェーク中に発生した SSL イベントについての詳細な情報を入手できます。SSL デバッグ トレースには以下の情報が表示されます。

- 信頼性のある認証局
- SSL サーバ コンフィグレーション情報
- サーバ ID (プライベート キーとデジタル証明書)
- 使用中のライセンスで許可されている強度
- 有効化されている暗号
- SSL ハンドシェークで渡された SSL レコード
- WebLogic Server が検出した SSL エラー (信頼および有効性チェックおよびデフォルト ホスト名検証など)

- I/O 関連情報

SSL のデバッグを有効にするには、次のコマンドライン プロパティを指定します。

```
-Dssl.debug=true -Dweblogic.StdoutDebugEnabled=true
```

SSL デバッグのプロパティは、SSL サーバ、SSL クライアント、およびノード マネージャの起動スクリプトに含めることができます。ノード マネージャが起動する管理対象サーバの場合、このコマンドライン引数を管理対象サーバの [リモート スタート] タブで指定します。

SSL デバッグでは、SSL プロセスで ALERT が生成されるたびにスタック トレースがダンプされます。ALERT のタイプと重大度は、TLS 仕様で定義されている一般的なものです。

スタック トレースによって、ALERT が生成された場合に情報がログ ファイルにダンプされます。したがって、SSL の問題をトラッキングする場合、SSL 接続の両側 (SSL クライアント側と SSL サーバ側) でデバッグを有効にしておく必要があります。ログ ファイルには、エラーの発生場所に関する詳細な情報が格納されます。ALERT の発生場所を特定するには、ALERT の発生後にトレース メッセージがないか調べます。トレース メッセージの後の ALERT は、エラーがピアで発生したことを示します。問題を特定するには、SSL 接続のピアで SSL デバッグを有効にする必要があります。

SSL の問題をトラッキングする場合は、ログ ファイルの情報を調べて以下の点を確認します。

- 正しい `config.xml` ファイルがロードされたかどうか
- ライセンスが正しいかどうか (国内向けまたは国外向け)
- 信頼性のある認証局がこのサーバに関して有効かつ正しいかどうか
- ホスト名検証が成功したかどうか
- 証明書検証が成功したかどうか

注意： 重大度 1 タイプ 0 は正常なクローズ ALERT であり、問題ではありません。

SSL セッションの動作

WebLogic Server では、SSL セッションをキャッシュできます。これらのセッションは、サーバの稼働時間にわたって存続します。この動作はこのリリースの WebLogic Server で変更されました。`weblogic.security.SSL.sessionCache.size` および `weblogic.security.SSL.sessionCache.ttl` コマンドライン引数は無視されます。

デフォルトでは、HTTPS URL を使用するクライアントは、URL ごとに新しい SSL セッションを取得します。これは、各 URL が異なる SSL コンテキストを使用するため、SSL セッションを共有または再利用できないからです。SSL セッションを取得するには、`weblogic.net.http.HttpsClient` クラスまたは `weblogic.net.http.HttpsURLConnection` クラスを使用します。クライアントは、SSL ソケット ファクトリをクライアント間で共有することによって、URL を再開することもできます。

SSL ソケットを直接使用するクライアントは、SSL セッション キャッシュの動作を制御できます。SSL セッション キャッシュは、各 SSL コンテキストに固有のものです。特定の SSL コンテキストによって返された SSL ソケット ファクトリ インスタンスによって作成されたすべての SSL ソケットは、SSL セッションを共有します。

クライアントは、デフォルトによって同じ IP アドレスとポートでセッションを再開します。デフォルトでは、複数の SSL ソケットが同じホストとポートを使用して SSL セッションを共有し、SSL ソケットが共通の SSL コンテキストを使用していると見なします。

SSL セッションをまったく使用しないクライアントは、SSL の `setEnableSessionCreation(false)` を明示的に呼び出して、SSL セッションがキャッシュされないようにする必要があります。この設定は、SSL セッションがキャッシュに追加されるかどうかだけを制御するもので、SSL ソケットがキャッシュ済みの SSL セッションを検索することを停止するものではありません。たとえば、SSL ソケット 1 がセッションをキャッシュし、SSL ソケット 2 が `setEnableSessionCreation` を `false` に設定した場合でも、SSL ソケット 1 の SSL セッションはキャッシュに存在するので、SSL ソケット 2 はそのセッションを再利用できます。

SSL セッションは SSL コンテキストの存続期間にわたって存在し、SSL ソケットの存続期間によって制御されません。このため、新しい SSL ソケットを作成し、同じホストとポートに接続した場合、その SSL ソケットがキャッシュ内に前の SSL セッションを持つ SSL コンテキストの SSL ソケット ファクトリを使用して作成される限り、前のセッションを再開できます。

WebLogic Server 6.x では、実行スレッド内でキャッシュされる SSL セッションが 1 つ存在します。このリリースの WebLogic Server では、セッション キャッシングはスレッドによって共有可能な SSL コンテキストによって維持されます。1 つのスレッドは 1 つの SSL セッションではなくセッション キャッシュ全体にアクセスできるので、複数の SSL セッションを 1 つ (または複数の) スレッドで使用および共有できます。

ホスト名検証の使い方

ホスト名検証を行うと、クライアントの接続先のホスト名 URL と、SSL 接続の過程でサーバが送り返すデジタル証明書内のホスト名が一致することが保証されます。ホスト名検証は、SSL クライアントまたはクライアントの役割を果たしている SSL サーバがリモート ホストのアプリケーション サーバに接続する場合に役立ちます。ホスト名検証を行うことは介在者の攻撃を防ぐことにつながります。

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 にある共通名と、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。名前が一致しない場合に SSL 接続を実際に中断するのは、SSL クライアントです。

デフォルト以外の動作が必要な場合は、ホスト名検証を無効にするか、カスタム ホスト名検証を登録します。ホスト名検証を無効にすると、WebLogic Server は介在者の攻撃に対して無防備な状態になります。プロダクション環境では、ホスト名検証を有効にすることをお勧めします。

ホスト名検証は、以下のいずれかの方法で無効にできます。

- WebLogic Server Administration Console で、[サーバ] ノードの [SSL] タブで [ホスト名検証を無視] 属性をチェックします。
- SSL クライアントのコマンドラインで、次の引数を入力します。

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

Java クライアントを使用する場合、ホスト名検証はコマンドラインで設定する必要があります。

また、カスタム ホスト名検証を記述できます。詳細については、『WebLogic Security プログラマーズ ガイド』を参照してください。カスタム ホスト名検証を記述する場合、ホスト名検証を実装するクラスの名前を WebLogic Server の CLASSPATH に指定する必要があります。

カスタム ホスト名検証を使用するには、次の手順に従います。

1. [サーバ] ノードを展開します。
2. [SSL] タブの [ホスト名の検証] 属性に、カスタム ホスト名検証をロードするために使用する Java クラスの名前を入力します。
3. [適用] をクリックします。
4. WebLogic Server を再起動します。

Java クライアントを使用する場合、カスタム ホスト名検証は次の引数を使用してコマンドラインで指定する必要があります。

```
-D weblogic.security.SSL.hostnameVerifier=classname
```

classname には、`weblogic.security.SSL.HostnameVerifier` インタフェースの実装を指定します。

ノード マネージャに関する SSL のコンフィグレーション

ノード マネージャは、管理サーバおよび管理対象サーバとの通信の保護に双方向 SSL を使用します。SSL のコンフィグレーションには、ノード マネージャと、ノード マネージャが通信する管理サーバおよび管理対象サーバの各々の ID と信頼を取得してから、適切な ID と信頼を使って、ノード マネージャ、管理サーバ、および任意の管理対象サーバをコンフィグレーションすることが含まれます。また、ホスト名検証と管理ポートの使用を考慮に入れる必要があります。こ

の節では、WebLogic Server 付属のデモ用 ID および信頼を使用する場合と、プロダクション レベルの ID および信頼を使用する場合の 2 つのシナリオについて説明します。

詳細については、次を参照してください。

6-31 ページの「管理サーバに関する SSL の要件」

6-33 ページの「管理対象サーバに関する SSL の要件」

6-34 ページの「ノード マネージャに関する SSL の要件」

6-35 ページの「ID と信頼 : デモとプロダクション環境の違い」

6-36 ページの「ホスト名検証の要件」

6-40 ページの「ノード マネージャのプロダクション環境用 SSL コンフィグレーション : 主な手順」

6-42 ページの「SSL を使用するための管理サーバのコンフィグレーション」

6-43 ページの「SSL を使用するための管理対象サーバのコンフィグレーション」

6-45 ページの「SSL を使用するためのノード マネージャのコンフィグレーション」

注意 : WebLogic Server では、ID と信頼を格納および指定するためのさまざまな手法を提供していますが、この節では、推奨される使用の例について説明します。

管理サーバに関する SSL の要件

SSL を使用するには、管理サーバに以下のものが必要です。

- ID— 管理サーバは、WebLogic キーストア プロバイダを通じてアクセスされる JKS キーストア内に格納されたプライベート キーを使用します。

  管理サーバのデジタル証明書はファイルに格納する必要があります。また、デジタル証明書には IP アドレスではなくホスト名を指定しなければなりません。デジタル証明書の場所は、WebLogic Server Administration Console で指定されます。

デフォルトでは、WLS ドメイン テンプレートのデモ用デジタル証明書およびプライベート キーは、ドメインの作成時にコンフィグレーション ウィザードによってドメインにコピーされます。管理サーバは、デジタル証明書とプライベート キーを ID として使用します。ただし、デモ用デジタル証明書とプライベート キーはプロダクション環境で使用しないでください。

- 信頼 — 管理サーバは、JKS キーストアに格納されている信頼性のある CA の証明書を使用できます。JKS キーストアは、管理サーバの起動スクリプトの `-Dweblogic.security.SSL.trustedCAkeystore` コマンドライン引数で指定されます。

注意： デフォルトでは、管理サーバの起動スクリプトは `-Dweblogic.security.SSL.trustedCAkeystore` コマンドライン引数で信頼を指定するため、WebLogic Server の管理サーバを通じて指定される信頼設定はすべて無視されます。

デフォルトでは、管理サーバは、*`WL_HOME`*`\server\lib` の `cacerts` ファイル内のすべての認証局を信頼します。

プロダクション環境では、管理サーバはノード マネージャと管理対象サーバの両方の認証局を信頼する、つまり、信頼性のある CA の証明書が管理サーバの信頼性のあるキーストアに入っている必要があります。

- SSL ポート ー デフォルトでは、SSL ポートは管理サーバ上で有効になっています。
- ホスト名検証 ー デフォルトでは、ホスト名検証は管理サーバ上で有効になっていません。ホスト名検証は、プロダクション環境でのみ必要です。

  ホスト名検証は、管理サーバの起動スクリプトの `-Dweblogic.security.SSL.ignoreHostnameVerification` コマンドライン引数で指定されます。

- 管理ポート ー 管理ポートを使用するかどうかは、SSL をコンフィグレーションするときに指定できます。管理ポートは、内部で SSL を使用してすべての管理通信を保護します。デフォルトでは、管理ポートは有効になっていません。

  プロダクション環境では管理ポートを使用することをお勧めします。管理ポートを使用する場合、管理サーバとすべての管理対象サーバが管理ポートを使用するようにコンフィグレーションしなければなりません。管理ポートを使用するようコンフィグレーションされていない管理対象サーバは、起動時に管理サーバに接続できません。

管理対象サーバに関する SSL の要件

SSL を使用するには、管理対象サーバに以下のものが必要です。

- ID— 管理対象サーバは、WebLogic キーストア プロバイダを通じてアクセスされるキーストア内に格納されたプライベート キーを使用します。

  管理サーバのデジタル証明書はファイルに格納する必要があります。また、デジタル証明書には IP アドレスではなくホスト名を指定しなければなりません。デジタル証明書の場所は、WebLogic Server Administration Console で指定されます。

  デフォルトでは、管理対象サーバには ID がコンフィグレーションされていません。コンフィグレーション ウィザードによって提供されるデモ用デジタル証明書とプライベート キーは、管理対象サーバの ID として使用することができます。ただし、このデジタル証明書とプライベート キーはプロダクション環境で使用しないでください。

- 信頼 — 管理対象サーバは、JKS キーストアに格納されている信頼性のある CA の証明書を使用します。JKS キーストアは、管理対象サーバのリモート起動引数における `weblogic.security.SSL.trustedCAkeystore` コマンドライン引数によって指定されます。

  デフォルトでは、管理対象サーバは、*`WL_HOME`*`\server\lib` の `cacerts` ファイル内のすべての認証局を信頼します。

  プロダクション環境では、管理対象サーバはノード マネージャと管理サーバの両方の認証局を信頼する、つまり、信頼性のある CA の証明書が管理対象サーバの信頼性のあるキーストアに入っている必要があります。

- ホスト名検証 ー デフォルトでは、ホスト名検証は管理対象サーバ上で有効になっていません。ホスト名検証は、プロダクション環境でのみ必要です。

  ホスト名は、管理対象サーバのリモート起動引数における `-Dweblogic.security.SSL.ignoreHostnameVerification` コマンドライン引数で指定されます。

- 管理ポート ー 管理ポートを使用するかどうかは、SSL をコンフィグレーションするときに指定できます。管理ポートは、内部で SSL を使用してすべての管理通信を保護します。デフォルトでは、管理ポートは有効になっていません。

  プロダクション環境では管理ポートを使用することをお勧めします。管理ポートを使用する場合、管理サーバとすべての管理対象サーバが管理ポート

を使用するようにコンフィグレーションしなければなりません。管理ポートを使用するようコンフィグレーションされていない管理対象サーバは、起動時に管理サーバに接続できません。

ノード マネージャに関する SSL の要件

注意： SSL 使用時にノード マネージャを NT サービスとしては実行しないことをお勧めします。

SSL を使用するには、ノード マネージャに以下のものが必要です。

- ID ― ノード マネージャはファイルに格納されているデジタル証明書とプライベート キーのみを使用できます。ID は、次のコマンドライン引数を使用してノード マネージャの起動スクリプトに指定します。

  `weblogic.nodemanager.keyFile=`*`filename`*

  `weblogic.nodemanager.keyPassword=`*`password_for_the_private_key`*

  `weblogic.nodemanager.certificateFile=`*`filename`*

  デフォルトでは、WebLogic Server インストーラは、デモ用デジタル証明書とプライベート キーを *`WL_HOME`*`\common\nodemanager\config` ディレクトリにコピーします。ただし、このデジタル証明書とプライベート キーはプロダクション環境で使用しないでください。

- 信頼 ― ノード マネージャは、JKS キーストアに格納されている信頼性のある CA の証明書を使用して信頼を確立します。信頼性のあるキーストアは、次のコマンドライン引数を使用してノード マネージャの起動スクリプトに指定します。

  `weblogic.security.SSL.trustedCAkeystore=`*`keystorename`*

  双方向 SSL が使用されるので、ノード マネージャも管理サーバと管理対象サーバが使用する認証局を信頼する、つまり、信頼性のある CA の証明書がノード マネージャの信頼性のあるキーストアに入っている必要があります。

  デフォルトでは、ノード マネージャは、*`WL_HOME`*`\server\lib` の `cacerts` ファイル内のすべての認証局を信頼します。

- ホスト名検証 ― デフォルトでは、ホスト名検証はノード マネージャ上で有効になっていません。ホスト名検証は、プロダクション環境でのみ必要です。ホスト名検証は、次のコマンドライン引数を使用してノード マネージャの起動スクリプトに有効にします。

```
weblogic.security.SSL.ignoreHostnameVerification=false
```

- 信頼性のあるホスト ー ノード マネージャは、信頼性のあるホスト上で動作する管理サーバからのみコマンドを受け入れます。ノード マネージャ プロセスに関して信頼性のあるホストは、ファイル内の IP アドレスまたは DNS 名で識別されます。デフォルトでは、このファイルは `nodemanager.hosts` という名前で、*`WL_HOME`*`\common\nodemanager\config` ディレクトリにインストールされます。

  ノード マネージャはマシンごとに 1 つありますが、ノード マネージャが管理するドメインでは複数のマシンが存在することがあります。このマシンから実行するすべてのドメインの管理サーバのマシンが `nodemanager.hosts` ファイルに入っていることを確認してください。デフォルトでは、`nodemanger.hosts` ファイルは常に `localhost` に設定されています。

  ホストは、IP アドレスでも名前でも指定できます。ホスト名を使用する場合は、ノード マネージャ上で逆引き参照を有効にします。

警告 : ホスト名を使用すると、WebLogic Server デプロイメントは介在者の攻撃に対して無防備な状態になる場合があります。

詳細については、「ノード マネージャ ホスト ファイルの設定」を参照してください。

ID と信頼 : デモとプロダクション環境の違い

WebLogic Server では、管理サーバと管理対象サーバ、およびノード マネージャが使用できるデモ用 ID (プライベート キーとデジタル証明書) と信頼 (信頼性のある認証局) が提供されます。

- コンフィグレーション ウィザードはドメインを作成するときに、WLS ドメイン テンプレートのデモ用の信頼と ID を管理サーバ用としてドメインにコピーします。このデモ用プライベート キーとデジタル証明書はテストまたはデモ用として使用できますが、プロダクション環境では使用しないでください。
- デフォルトでは、管理対象サーバには ID と信頼がコンフィグレーションされていません。ただし、管理対象サーバもコンフィグレーション ウィザードによって提供された ID と信頼を使用することができます。

- ノード マネージャには、デモ用プライベート キーとデジタル証明書がインストーラによって提供されます。このデモ用プライベート キーとデジタル証明書はテストまたはデモ用として使用できますが、プロダクション環境では使用しないでください。

プロダクション環境では、ノード マネージャ、管理サーバ、および管理対象サーバ用のプライベート キーとデジタル証明書を、Verisign, Inc. や Entrust.net などの有名な認証局から取得してください。詳細については、6-5 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の取得」を参照してください。

ホスト名検証の要件

ホスト名検証を無効にした状態で運用することはデモ用環境では問題ありませんが、プロダクション環境に移行する場合、接続先のホストが目的のホストであることを保証するために、ホスト名検証を使用することをお勧めします。ホスト名検証を使用するには、以下の点についてチェックしてください。

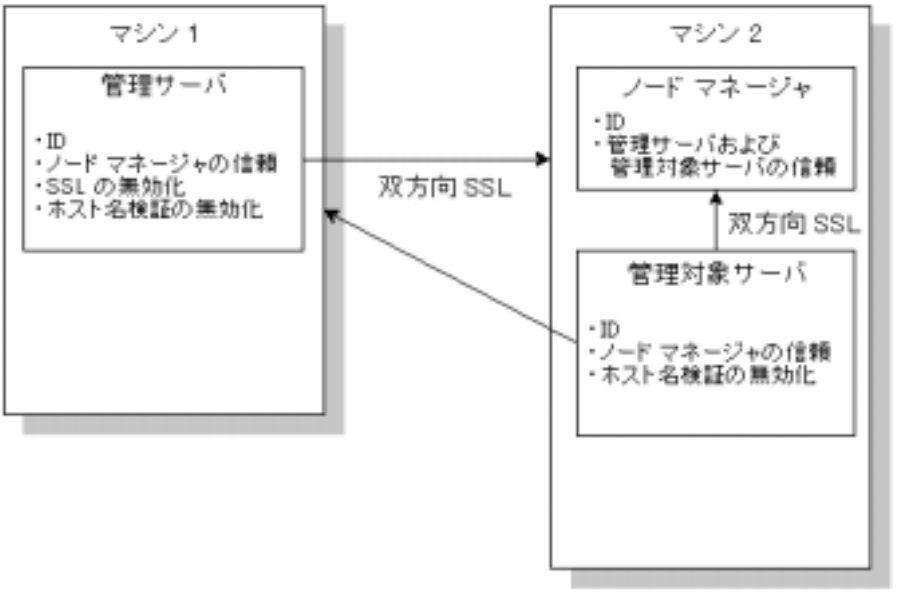
- すべてのデジタル証明書に IP アドレスではなくホスト名が指定されている
- すべてのデジタル証明書に正しいホスト名が指定されている
- すべての URL に IP アドレスではなくホスト名が使用されている
- ホスト名検証がノード マネージャ、管理サーバ、および管理対象サーバで有効になっている

ノード マネージャのデモ用 SSL コンフィグレーション：主な手順

WebLogic Server によって提供されるデモ用 ID と信頼を使用したノード マネージャに対する SSL のコンフィグレーションでは、SSL 属性のデフォルト設定を確認し、管理サーバと管理対象サーバが異なるポートで SSL 通信をリスンしていることを確認します。デモ用 ID と信頼の場合にはデフォルト設定が使用されるので、詳しいコンフィグレーション手順は説明しません。

図 6-1 に、SSL のデモ用コンフィグレーションを示します。

図 6-1 ノード マネージャに関する SSL のデモ用コンフィグレーション



SSL とデモ用 ID および信頼を使用するようノード マネージャをコンフィグレーションするには、次の手順に従います。

1. コンフィグレーション ウィザードを起動し、新しい WebLogic ドメインを作成します。
   a. [Admin Server with Managed Server(s)] オプションを選択します。
   b. 新しい管理対象サーバをドメインに追加します。管理サーバ上のリスン ポート以外のリスン ポートを管理対象サーバ用に指定します。管理対象サーバの [リスン アドレス] を `localhost` に設定します。

   詳細については、『WebLogic Server ドメイン管理』を参照してください。
2. 管理サーバを起動します。管理ポートは有効にしないでください。
3. 管理対象サーバ用のマシンを作成します。[リスン アドレス] を `localhost` に設定します。管理対象サーバをマシンに追加します。詳細については、「マシンのコンフィグレーション」を参照してください。

4. 管理サーバで SSL が有効になっていることを確認します。管理ポートは有効になっていないので、SSL を有効にする必要があります。SSL を有効にしないと、管理サーバはノード マネージャと通信できません。

   [接続 | SSL ポート] タブをクリックし、[有効化] 属性がチェックされていることを確認します。詳細については、6-22 ページの「SSL ポートの有効化」を参照してください。

5. [接続 | SSL] タブで、管理サーバに対する SSL 属性が以下のように設定されていることを確認します。

   - [サーバ キー ファイル名] ー `demokey.pem`
   - [サーバ証明書ファイル名] ー `democert.pem`
   - ホスト名検証 ー 無効

   詳細については、6-23 ページの「一方向 SSL の属性の設定」を参照してください。

6. 管理サーバの起動スクリプトで、信頼が *`WL_HOME`*`\server\lib\cacerts` として設定されていることを確認します。

7. 管理対象サーバで SSL が有効になっていることを確認します。SSL ポートには、管理サーバが使用する SSL ポート以外のポートが指定されていることを確認します。

   [接続 | SSL ポート] タブをクリックし、[有効化] 属性がチェックされていることを確認します。詳細については、6-22 ページの「SSL ポートの有効化」を参照してください。

8. [接続 | SSL] タブで、管理対象サーバに対する SSL 属性を以下のように設定します。

   - [サーバ キー ファイル名] ー 管理対象サーバが動作するドメインのデモ用プライベート キー (`.../demokey.pem`) の絶対パス名
   - [サーバ証明書ファイル名] ー 管理対象サーバが動作するドメインのデモ用デジタル証明書 (`.../democert.pem`) の絶対パス名
   - ホスト名検証 ー 無効

   詳細については、6-23 ページの「一方向 SSL の属性の設定」を参照してください。

9. [サーバ | コンフィグレーション | リモート スタート] タブで管理対象サーバの起動コマンドライン引数をコンフィグレーションします。

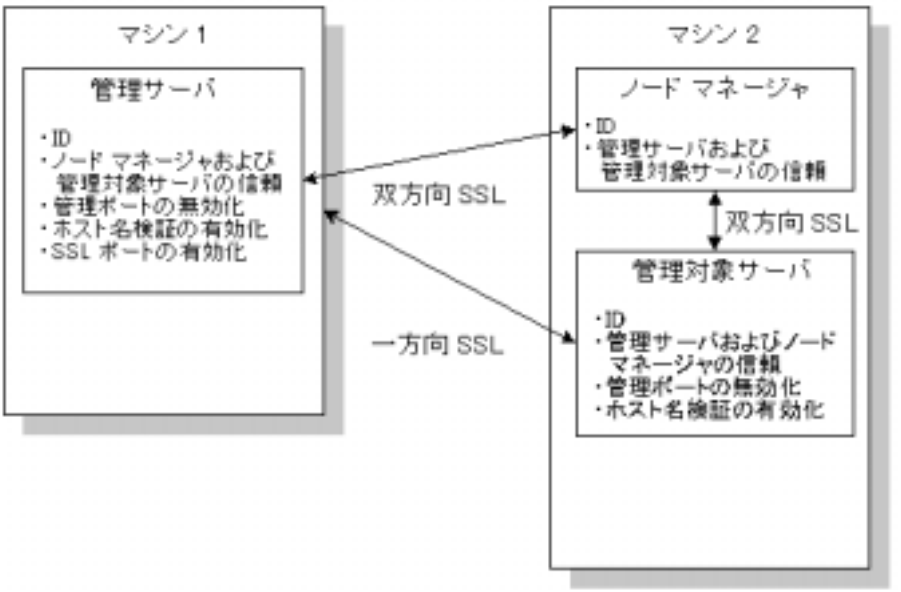
- [ユーザ名] (必須) ー この管理対象サーバを起動する特権を持つユーザの名前 (`weblogic` など) を入力します。
- [パスワード] (必須) ー [変更 ...] をクリックして、サーバを起動するためのパスワードを変更します (weblogic など)。新しいパスワードを [新しいパスワード] および [再入力] テキスト ボックスに入力し、[適用] をクリックして変更を適用します。

10. 管理サーバを停止します。
11. 管理サーバを起動します。

ノード マネージャのプロダクション環境用 SSL コンフィグレーション : 主な手順

図 6-2 に、プロダクション環境用 SSL コンフィグレーションを示します。

図 6-2　ノード マネージャに関する SSL のプロダクション用コンフィグレーション



ノード マネージャに関して SSL をコンフィグレーションするには、次の手順に従います。

1. ノード マネージャ、管理サーバ、およびすべての管理対象サーバ用の ID と信頼を取得します。

注意 : 管理サーバ、すべての管理対象サーバ、およびノード マネージャ用の ID と信頼を取得する際には、(デジタル証明書が URL と一致するように) デジタル証明書にホスト名が含まれていることを確認してください。

2. コンフィグレーション ウィザードを起動し、新しい WebLogic ドメインを作成します。
3. 管理サーバを起動します。
4. WebLogic ドメインの管理ポートをコンフィグレーションします。
5. すべての管理対象サーバ用のマシンを作成します。[リスン アドレス] を管理対象サーバが動作するマシンのホスト名に設定します。管理対象サーバをマシンに追加します。詳細については、「マシンのコンフィグレーション」を参照してください。
6. 管理サーバ用の SSL を無効にします。管理ポートが有効になっているので、ノード マネージャはこのポートをすべての管理トラフィック用に使用します。管理ポートは SSL を使用してすべての通信を保護するので、SSL ポートを有効にしておく必要はありません。SSL ポートを介してアプリケーション トラフィックを実行する場合にのみ、SSL ポートを有効にしておきます。
7. 管理サーバ用の ID をコンフィグレーションします。
8. 管理サーバのホスト名検証を無効にします。
9. 管理サーバの信頼性のある認証局を指定します。
10. すべての管理対象サーバの管理ポートを有効にします。
11. 管理対象サーバ用の SSL を無効にします。
12. リモート起動引数を使用して ID、信頼をコンフィグレーションし、すべての管理対象サーバのホスト名検証を有効化します。
13. ノード マネージャの `nodemanager.hosts` ファイルを編集します。ノード マネージャの通信先となる管理サーバが動作するマシンの名前を入力します。IP アドレスではなくホスト名でマシンを指定する場合は、DNS の逆引き参照を有効にします。
14. ノード マネージャの起動スクリプトを編集します。ノード マネージャの ID と信頼の場所を指定し、ホスト名検証を有効にします。
15. ノード マネージャを起動します。
16. 管理サーバを停止します。
17. 管理サーバを起動します。

SSL を使用するための管理サーバのコンフィグレーション

デフォルトでは、管理サーバの ID と信頼は、コンフィグレーション ウィザードによって提供されるデモ用デジタル証明書とプライベート キーを使用してコンフィグレーションされます。また、SSL はデフォルトでコンフィグレーションされます。詳細については、6-36 ページの「ノード マネージャのデモ用 SSL コンフィグレーション：主な手順」を参照してください。この節では、管理ポートとプロダクション レベルの ID および信頼を使用して、SSL を使用するよう管理サーバをコンフィグレーションする方法について説明します。

SSL を使用するよう管理サーバをコンフィグレーションするには、次の手順に従います。

1. 管理サーバ用の ID と信頼を取得します。管理サーバのデジタル証明書にホスト名が含まれていることを確認してください。

   詳細については、6-5 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の取得」を参照してください。

2. コンフィグレーション ウィザードを起動し、新しい WebLogic ドメインを作成します。

   a. [Admin Server with Managed Server(s)] オプションを選択します。

   b. 管理対象サーバをドメインに追加します。管理サーバ上のリスン ポート以外のリスン ポートを管理対象サーバ用に指定します。[リスン アドレス] を管理対象サーバが動作するマシンのホスト名に設定します。

   詳細については、『WebLogic Server ドメイン管理』を参照してください。

3. 管理サーバを起動します。

4. WebLogic Server ドメインの管理ポートを有効にします。詳細については、「ドメイン全体の管理ポートのコンフィグレーション」を参照してください。

5. WebLogic Server Administration Console の [接続 | SSL ポート] タブの [有効化] 属性のチェックをはずして、SSL を無効にします。

6. WebLogic Server Administration Console の [接続 | SSL] タブの [SSL] 属性上の [サーバの証明書ファイル名] 属性でデジタル証明書の場所を指定します。

7. JKS キーストアを作成し、そのキーストア内に管理サーバのプライベート キーを格納します。詳細については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。
8. Configure the WebLogic Keystore provider to access the JKS keystore. 詳細については、6-19 ページの「WebLogic Server キーストア プロバイダがキーストアを指し示すようにコンフィグレーションする」を参照してください。
9. 管理サーバが信頼している認証局用の JKS キーストアを作成します。信頼性のある CA 証明書を JKS キーストアにロードします。詳細については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。
10. 管理サーバの起動スクリプトを編集します。
    - `-Dweblogic.security.SSL.trustedCAkeystore=`*`path_to_keystore`* コマンドライン引数を使用して、管理サーバ用の信頼性のある認証局が格納されている JKS キーストアの場所を指定します。
    - `-weblogic.security.SSL.ignoreHostnameVerification=false` コマンドライン引数を使用して、ホスト名検証を有効にします。
11. 管理対象サーバをコンフィグレーションします。6-43 ページの「SSL を使用するための管理対象サーバのコンフィグレーション」を参照してください。
12. ノード マネージャをコンフィグレーションします。6-45 ページの「SSL を使用するためのノード マネージャのコンフィグレーション」を参照してください。
13. ノード マネージャを起動します。
14. 管理サーバを停止します。
15. 管理サーバを起動します。

SSL を使用するための管理対象サーバのコンフィグレーション

デフォルトでは、管理対象サーバには ID がコンフィグレーションされていません。コンフィグレーション ウィザードによって提供されるデモ用プライベート キーとデジタル証明書を使用すると、管理対象サーバの ID を確立できますが、このプライベート キーとデジタル証明書はプロダクション環境で使用しないで

ください。また、管理対象サーバは、*WL_HOME*\server\lib ディレクトリの cacerts ファイル内のすべての認証局をデフォルトで信頼します。詳細については、6-36 ページの「ノード マネージャのデモ用 SSL コンフィグレーション：主な手順」を参照してください。

この節では、管理ポートとプロダクション レベルの ID および信頼を使用して、SSL を使用するよう管理対象サーバをコンフィグレーションする方法について説明します。

SSL を使用するよう管理対象サーバをコンフィグレーションするには、次の手順に従います。

1. 管理対象サーバ用の ID と信頼を取得します。管理対象サーバのデジタル証明書にホスト名が含まれていることを確認してください。
2. SSL を使用するようドメインの管理サーバをコンフィグレーションします。詳細については、6-42 ページの「SSL を使用するための管理サーバのコンフィグレーション」を参照してください。
3. 管理対象サーバの管理ポートをコンフィグレーションします。管理サーバが使用する以外の管理ポートを指定してください。管理サーバが同じマシン上で管理対象サーバとして動作する場合は、[SSL] タブの [Local Port Override] 属性を設定して管理ポート番号をオーバライドします。

   管理ポートを有効にしたので、管理対象サーバの [リモート スタート] タブのコマンドライン引数を使用して、信頼とホスト名検証を指定する必要があります。
4. WebLogic Server Administration Console の [接続｜ SSL] タブの [SSL] 属性上の [サーバの証明書ファイル名] 属性でデジタル証明書の場所を指定します。
5. JKS キーストアを作成し、そのキーストア内に管理対象サーバのプライベート キーを格納します。詳細については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。
6. JKS キーストアにアクセスするように WebLogic キーストア プロバイダをコンフィグレーションします。詳細については、6-19 ページの「WebLogic Server キーストア プロバイダがキーストアを指し示すようにコンフィグレーションする」を参照してください。
7. 管理対象サーバが信頼している認証局用の JKS キーストアを作成します。信頼性のある CA 証明書を JKS キーストアにロードします。詳細については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。

8. [サーバ | コンフィグレーション | リモート スタート] タブで管理対象サーバの起動コマンドライン引数をコンフィグレーションします。
   - [ユーザ名] (必須) ー この管理対象サーバを起動する特権を持つユーザの名前 (`weblogic` など) を入力します。
   - [パスワード] (必須) ー [変更 ...] をクリックして、サーバを起動するためのパスワードを変更します (weblogic など)。新しいパスワードを [新しいパスワード] および [再入力] テキスト ボックスに入力し、[適用] をクリックして変更を適用します。
   - 信頼 ー `weblogic.security.SSL.trustedCAKeyStore` コマンドライン引数を使用して、管理対象サーバが信頼する認証局を格納しているキーストアを指定します。キーストアの絶対パス名を使用してください。
   - ホスト名検証 ー `-Dweblogic.security.SSL.ignoreHostnameVerification=false` コマンドライン引数を指定して、ホスト名検証を有効にします。
9. ノード マネージャをコンフィグレーションします。6-45 ページの「SSL を使用するためのノード マネージャのコンフィグレーション」を参照してください。
10. ノード マネージャを起動します。
11. 管理サーバを停止します。
12. 管理サーバを起動します。

SSL を使用するためのノード マネージャのコンフィグレーション

SSL を使用するようノード マネージャをコンフィグレーションするには、次の手順に従います。

1. SSL を使用するようドメインの管理サーバをコンフィグレーションします。詳細については、6-42 ページの「SSL を使用するための管理サーバのコンフィグレーション」を参照してください。
2. SSL を使用するようドメインの管理対象サーバをコンフィグレーションします。詳細については、6-43 ページの「SSL を使用するための管理対象サーバのコンフィグレーション」を参照してください。

3. ノード マネージャ用の ID と信頼を取得します。詳細については、6-5 ページの「プライベート キー、デジタル証明書、信頼性のある認証局の取得」を参照してください。

   ノード マネージャはファイルに格納されているデジタル証明書とプライベート キーのみを使用できます。

4. JKS キーストアを作成し、信頼性のある認証局をキーストアにロードします。詳細については、6-18 ページの「よく使う Keytool のコマンド」を参照してください。

   双方向 SSL が使用されるので、ノード マネージャも管理サーバと管理対象サーバが使用する認証局を信頼する必要があります。

5. ノード マネージャはマシンごとに 1 つありますが、ノード マネージャが管理するドメインでは複数のマシンが存在することがあります。ノード マネージャが管理するドメイン内のすべてのマシンで手順 4 および 5 を実行してください。

6. ノード マネージャの起動スクリプトを編集します。

   起動スクリプトは、*WL_HOME*\server\bin ディレクトリにあります。標準のノード マネージャ起動スクリプトの名前は、Windows システムでは `startNodeManager.cmd`、UNIX システムでは `startNodeManager.sh` です。

   次のコマンドライン引数を使用して、ID と信頼を指定します。

   - `weblogic.nodemanager.keyFile=`*`filename`* を使用して、プライベート キー ファイルの場所を指定する。
   - プライベート キー ファイルをパスワードで保護している場合は、`weblogic.nodemanager.keyPassword=`*`password`* を使用してパスワードを指定する。
   - `weblogic.nodemanager.certificateFile=`*`filename`* を使用して、ノード マネージャのデジタル証明書の場所を指定する。
   - `weblogic.security.SSL.trustedCAkeystore=`*`keystorename`* を使用して、信頼性のある JKS キーストアの場所を指定する。
   - `weblogic.security.SSL.ignoreHostnameVerification=false` を使用して、ホスト名検証を有効にする。
   - `weblogic.nodemanager.trustedHosts=`*`pathtofile`* を使用して、ノード マネージャの `nodemanager.hosts` ファイルの場所を指定する。

- [リスン アドレス] プロパティを管理対象サーバが動作するマシンのホスト名に設定します。
- `ReverseDNSEnabled` プロパティを IP アドレスではなく管理サーバのホスト名を使用するように設定します。

7. *`WL_HOME`*`\common\nodemanager\config` の `nodemanager.hosts` ファイルを編集して、ノード マネージャが信頼するマシン上のすべての管理サーバを含めます。DNS の逆引き参照が使用されている場合、管理サーバは IP アドレスまたはホスト名で指定されます。

   ドメイン内の各マシンの `nodemanager.hosts` ファイルが更新されていることを確認します。デフォルトでは、`nodemanger.hosts` ファイルは常に `localhost` に設定されています。

8. ノード マネージャを起動します。
9. 管理サーバを停止します。
10. 管理サーバを起動します。

SSL を使用した RMI over IIOP のコンフィグレーション

SSL を使用すると、RMI リモート オブジェクトへの IIOP 接続を保護できます。SSL は、認証を通じて接続を保護し、オブジェクト間のデータ交換を暗号化します。

SSL を使用して RMI over IIOP 接続を保護するには、次の手順に従います。

1. SSL を使用するよう WebLogic Server をコンフィグレーションします。
2. SSL を使用するよう Object Request Broker (ORB) をコンフィグレーションします。SSL のコンフィグレーションの詳細については、クライアント ORB の製品マニュアルを参照してください。
3. `host2ior` ユーティリティを使用して、WebLogic Server IOR をコンソールに出力します。`host2ior` ユーティリティでは、SSL 接続用と非 SSL 用に 2 種類のインターオペラブル オブジェクト参照 (IOR) が出力されます。IOR のヘッダは、IOR が SSL 接続で使用できるかどうかを示します。

4. SSL IOR は、WebLogic Server JNDI ツリーにアクセスする CosNaming サービスへの初期参照を取得するときに使用します。

RMI over IIOP の使い方の詳細については、『WebLogic RMI プログラマーズ ガイド』と『WebLogic RMI over IIOP プログラマーズ ガイド』を参照してください。

SSL 証明書検証

旧リリースでは、WebLogic Server は証明書チェーンの各証明書が認証局によって発行されたことを保証しませんでした。この問題は、誰かが信頼性のある認証局から個人用証明書を取得し、その証明書を使用して他の証明書を発行しても、WebLogic Server は無効な証明書を検出できないことを意味しました。このリリースでは、WebLogic Server で使用するすべての X509 V3 CA 証明書は CA として定義される Basic Constraint 拡張を備えている必要があります。このため、証明書チェーンのすべての証明書が認証局によって発行されたことが保証されます。デフォルトでは、この条件を満たしていない認証局の証明書は拒否されます。この節では、証明書検証レベルを制御するコマンドライン引数について説明します。

証明書検証に合格しない証明書チェーンを使用して WebLogic Server が起動された場合、クライアントが証明書検証を拒否できることを示す情報メッセージがログに記録されます。

証明書検証レベルの制御

WebLogic Server はデフォルトで、CA として定義された Basic Constraint 拡張を持たない証明書チェーンのデジタル証明書を拒絶します。ただし、この要件を満たさない証明書を使用することも、IETF RFC 2459 標準に準拠するようにセキュリティ レベルを上げることもできます。次のコマンドライン引数を使用すると、WebLogic Server によって実行される証明書検証のレベルを制御できます。

```
-Dweblogic.security.SSL.enforceConstraints=option
```

表 6-4 では、このコマンドライン引数のオプションについて説明します。

表 6-4 -Dweblogic.security.SSL.enforceConstraints のオプション

オプション	説明

`strong` または `true`	CA の証明書の Basic Constraints 拡張が CA として定義されていることをチェックする場合は、このオプションを使用する。
	次に例を示す。
	`-Dweblogic.security.SSL.enforceConstraints=strong`
	または
	`-Dweblogic.security.SSL.enforceConstraints=true`
	WebLogic Server はデフォルトで、このレベルの証明書検証を実行する。

`strict`	CA の証明書の Basic Constraints 拡張が CA として定義されていることをチェックし、critical に設定する場合は、このオプションを使用する。このオプションは IETF RFC 2459 標準を強制する。 次に例を示す。 `-Dweblogic.security.SSL.enforceConstraints=strict` 市販の CA の証明書の多くは IETF RFC 2459 標準に準拠していないので、このオプションはデフォルトになっていない。
`off`	証明書検証を無効にするにはこのオプションを使用する。このオプションは注意して使用すること。たとえば、有名な認証局から購入した CA 証明書が新しい検証に合格しない場合は、このオプションを使用する。ただし、商用ベースの認証局の CA 証明書のほとんどは、デフォルトの `strong` オプションで機能する。 次に例を示す。 `-Dweblogic.security.SSL.enforceConstraints=off` このオプションはプロダクション環境で使用しないで、IETF RFC 2459 標準に準拠した CA 証明書を新たに購入することが望ましい。

証明書チェーンのチェック

WebLogic Server では、既存の証明書チェーンが WebLogic Server によって拒絶されるかどうかをチェックするための ValidateCertChain コマンドライン ユーティリティを提供しています。このユーティリティは、PEM ファイル、PKCS-12 ファイル、PKCS-12 キーストア、および JKS キーストアの証明書チェーンを使用します。このユーティリティでは、証明書チェーン全体を使用する必要があります。ValidateCertChain コマンドライン ユーティリティの構文は次のとおりです。

```
java utils.ValidateCertChain -file pemcertificatefilename
java utils.ValidateCertChain -pem pemcertificatefilename
java utils.ValidateCertChain -pkcs12store pkcs12storefilename
java utils.ValidateCertChain -pkcs12file pkcs12filename password
java utils.ValidateCertChain -jks alias storefilename [storePass]
```

有効な証明書チェーンの例を示します。

```
java utils.ValidateCertChain -pem zippychain.pem

Cert[0]: CN=zippy,OU=FOR TESTING
ONLY,O=MyOrganization,L=MyTown,ST=MyState,C=US

Cert[1]: CN=CertGenCAB,OU=FOR TESTING
ONLY,O=MyOrganization,L=MyTown,ST=MyState,C=US

Certificate chain appears valid
```

無効な証明書チェーンの例を示します。

```
java utils.ValidateCertChain -jks mykey mykeystore

Cert[0]: CN=corba1,OU=FOR TESTING ONLY,
O=MyOrganization,L=MyTown,ST=MyState,C=US

CA cert not marked with critical BasicConstraint indicating it is
a CA
Cert[1]: CN=CACERT,OU=FOR TESTING ONLY,
O=MyOrganization,L=MyTown,ST=MyState,C=US

Certificate chain is invalid
```

証明書に関する問題のトラブルシューティング

以前のリリースの WebLogic Server で、SSL 通信が正常に動作していて予期しないエラーが発生するようになった場合は、WebLogic Server が使用している証明書チェーンが検証に失敗したことが原因で問題が発生している可能性があります。

証明書チェーンが拒否された場所を特定し、受け入れ可能なもので証明書チェーンを更新するか、または `-Dweblogic.security.SSL.enforceConstraints` コマンドライン引数の設定を変更するかを決定してください。

証明書に関する問題に対処するには、次のいずれかの解決策を使用します。

- SSL 通信を使用するプロセスの証明書の場所がわかる場合は、ValidateCertChain コマンドライン ユーティリティを使用して、証明書チェーンが受け入れられるかどうかをチェックする。
- SSL 通信を使用するプロセスに関して SSL デバッグを有効にする。SSL デバッグ トレースの構文は次のとおりです。

  ```
  -Dssl.debug=true -Dweblogic.StdoutDebugEnabled=true
  ```

  次のメッセージは、SSL エラーの原因が証明書チェーン内の問題にあることを示しています。

  ```
  <CA certificate rejected. The basic constraints for a CA
  certificate were not marked for being a CA, or were not marked
  as critical>
  ```

  一方向 SSL を使用している場合は、クライアント ログでこのエラーを探してください。双方向 SSL を使用している場合は、クライアント ログとサーバ ログでこのエラーを探してください。

WebLogic Server での nCipher JCE プロバイダの使い方

注意： JCE プロバイダは、JDK 1.3 からダウンロード可能な Java Cryptography Extension (JCE) のアプリケーション プログラミング インタフェース (API) を使用して記述されています。このタイプのプロバイダは、

WebLogic セキュリティ サービス プロバイダ インタフェース (SSPI) を使用して記述されたプロバイダとは異なります。WebLogic Server はデフォルトでは JCE プロバイダを提供していません。JDK 1.3 のデフォルト JCE プロバイダ (SunJCE) は、このリリースの WebLogic Server でテストされていません。

SSL は、Web サーバで使用可能なリソースを保護するための主要コンポーネントです。ただし、SSL トラフィックは重いので、Web サーバのパフォーマンスに影響するボトルネックが発生する可能性があります。ハードウェア アクセラレータは、Web サーバから SSL 処理の負荷を取り除き、サーバがより多くのトランザクションを処理できるようにします。また、ハードウェア アクセラレータは、強力な暗号と暗号化プロセスを提供して、鍵の整合性と機密性を保持します。

WebLogic Server は、nCipher JCE プロバイダの使用をサポートしています。nCipher JCE プロバイダの詳細については、http://www.ncipher.com/solutions/webservers.html を参照してください。

1. 製品マニュアルに従って、nCipher JCE プロバイダ用のハードウェアをインストールおよびコンフィグレーションします。
2. nCipher JCE プロバイダ用のファイルをインストールします。以下のファイルが必要です。
   - JCE 1.2.1 フレームワーク JAR
   - 管轄ポリシー ファイル
   - JCE プロバイダ
   - JAR ファイルに署名した証明書

   注意： この手順は、nCipher JCE プロバイダ用のハードウェアのインストール中に実行済みかもしれません。その場合は、ファイルが正しくインストールされていることを確認してください。

   ファイルは以下のいずれかの方法でインストールされます。

   - エクステンションとしてインストールされる。次のいずれかの場所にファイルをコピーします。

     Windows NT

     JAVA_HOME\lib\ext

     次に例を示します。

     WL_HOME\jdk131\jre\lib\ext

UNIX

JAVA_HOME/lib/ext

次に例を示します。

WL_HOME/jdk131/jre/libext

- サーバの CLASSPATH にインストールされる。

3. セキュリティ プロパティ ファイル (java.security) を編集して、WebLogic Server の承認済み JCE プロバイダのリストに nCipher JCE プロバイダを追加します。セキュリティ プロパティ ファイルの場所は次のとおりです。

   Windows NT

   JAVA_HOME\lib\security\java.security

   UNIX

   JAVA_HOME/lib/security/java.security

   nCipher JCE プロバイダを次のように指定します。

   security.provider.*n*=com.ncipher.provider.km.mCipherKM

   要素の説明は次のとおりです。

   n には、特定のプロバイダが要求されていない場合に、要求されたアルゴリズムを検索するプロバイダの順序を決める優先順位を指定します。順序は 1 を基準にしており、1 が最も優先順位が高く、次に 2 が高いというように続きます。

   nCipher JCE プロバイダは、セキュリティ プロパティ ファイルの RSA JCA プロバイダの後に続く必要があります。次に例を示します。

   ```
   security.provider.1=sun.security.provider.Sun
   security.provider.2=com.sun.rsajca.Provider
   security.provider.3=com.ncipher.provider.km.mCipherKM
   ```

   注意： nCipher JCE プロバイダを正しく動作させるには、JCE プロバイダをこの順序で指定する必要があります。

4. WebLogic Server を起動します。

5. nCipher JCE プロバイダが正しく動作するには、nCipher の製品マニュアルに従って、デバッグを有効にする必要があります。

SSL プロトコルのバージョンの指定

WebLogic Server では、SSL V3.0 プロトコルと TLS V1.0 プロトコルの両方をサポートしています。デフォルトでは、WebLogic Server は SSL V3.0 プロトコルを使用します。ほとんどの場合、SSL V3.0 プロトコルをそのまま使用できますが、TLS V1.0 プロトコルが必要となる状況 (互換性、SSL のパフォーマンス、およびセキュリティ要件が最大限の環境) もあります。

`weblogic.security.SSL.protocolVersion` コマンドライン引数により、SSL 接続に使用するプロトコルを指定できます。

WebLogic Server がクライアントとして動作する場合、WebLogic Server から の SSL V2.0 hello で、SSL ハンドシェークが開始されます。ピアは SSL V3.0 または TLS V1.0 メッセージで応答する必要があります。それ以外の場合、SSL 接続は中断されます。この動作はデフォルト設定です。

注意 : SSL V3.0 および TLS V1.0 プロトコルは、入れ替えがききません。TLS V1.0 プロトコルを使用するのは、必要な SSL クライアントがすべて確実にこのプロトコルを使用可能である場合のみとしてください。

次のコマンドライン引数は、WebLogic Server が SSL V3.0 接続または TLS V1.0 接続のみをサポートするように指定できます。

- `-Dweblogic.security.SSL.protocolVersion=SSL3`—SSL V3.0 メッセージのみを送信および受け入れ。
- `-Dweblogic.security.SSL.protocolVersion=TLS1`—TLS V1.0 メッセージのみを送信および受け入れ。
- `-Dweblogic.security.SSL.protocolVersion=ALL`— これがデフォルトの動作。

SSL プロトコルの使用による weblogic.Admin から WebLogic Server への接続

SSL プロトコルを使用しての `weblogic.Admin` から WebLogic Server への接続には、サーバ上で双方向 SSL を無効にし、クライアントの URL でセキュアなサーバ ポートを使用し、クライアントの信頼を指定し、クライアントがどのようにホスト名検証を使用するかをコンフィグレーションする必要があります。次の節では、これらの手順について詳しく説明します。

SSL サーバ上で双方向 SSL が無効になっていることを確認

`weblogic.Admin` を使用している場合、SSL の ID を指定する方法はありません。SSL サーバが双方向 SSL 用にコンフィグレーションされている場合は、ID (プライベート キーおよびデジタル証明書または証明書チェーン) が必要です。したがって、双方向 SSL は、`weblogic.Admin` 使用時には有効化できません。`weblogic.Admin` から SSL サーバへの SSL 接続を確立する前に、SSL サーバが双方向 SSL を使用するようにコンフィグレーションされていないことを確認します。SSL サーバ上で双方向 SSL が有効になっていると、SSL 接続は失敗します。

WebLogic Server 使用時に双方向 SSL を無効化するには、次の手順に従います。

1. [サーバ] ノードを展開します。
2. SSL サーバとして動作するサーバを選択します。
3. [接続 | SSL] タブを選択します。
4. [クライアント証明書を強制] 属性のチェックがはずされていることを確認します。
5. [適用] をクリックします。

6. WebLogic Server を再起動します。

URL でのセキュアなポートの使用

接続を行うために SSL プロトコルを使用するには、`weblogic.Admin` の URL でセキュアなプロトコルおよびポートを指定します。次に例を示します。

```
weblogic.Admin -url t3s://localhost:9002
```

weblogic.Admin の信頼の指定

SSL クライアントはすべて、信頼を指定する必要があります。信頼とは、信頼性のある認証局のうちどれがクライアントによって信頼されているかを指定する、一連の CA 証明書です。SSL 接続を確立するには、クライアントがサーバのデジタル証明書を発行した認証局を信頼している必要があります。

`weblogic.Admin` を使用している場合、信頼性のある CA 証明書が、キーストアに格納されている必要があります。デフォルトでは、JDK (`...\jre\lib\security\cacerts`) から利用可能なすべての信頼性のある認証局が、`weblogic.Admin` によって信頼されています。しかし、別の信頼キーストアを指定するオプションもあります。JDK 信頼キーストア以外の信頼キーストアを指定するには、次のコマンドライン引数を使用します。

```
-Dweblogic.security.SSL.trustedCAkeystore=pathtokeystore
```

`pathtokeystore` は、信頼性のある CA 証明書を含むキーストア ファイルです。

weblogic.Admin のホスト名検証の指定

デフォルトでは、`weblogic.Admin` はホスト名検証チェックを実行します。サーバから受け取られたデジタル証明書内の CN フィールドと、サーバへの接続にクライアントが使用した URL 内のサーバ名を比較します。ホスト名検証チェックに合格するためには、CN フィールドとサーバ名が一致している必要があります。このチェックは、介在者の攻撃を防ぐために実行されます。

次のコマンドライン引数を指定することによって、チェックを無効化することが可能です。

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

注意： SSL サーバの URL で IP アドレスが指定されている場合は、ホスト名検証チェックを無効にしてください。

カスタム ホスト名検証を指定するには、次のコマンドライン引数を使用します。

```
-Dweblogic.security.SSL.hostnameVerifier=classname
```

classname には、`weblogic.security.SSL.HostnameVerifier` インタフェースの実装を指定します。

BEA Tuxedo クライアントおよび WebLogic Server での SSL プロトコルの使用

BEA Tuxedo クライアントおよび WebLogic Server 間での通信の保護には、SSL プロトコルを使用できます。この節では、次のコード サンプルのクライアントを使い、SSL プロトコルの使用を可能とするために必要な手順を詳細に説明します。

```
BEA_HOME/sample/examples/iiop/ejb/stateless/tuxclient
```

WebLogic Server と BEA Tuxedo クライアントの間で SSL プロトコルおよび一方向認証を使用するには、次の手順に従います。

1. 一方向 SSL を使用するよう WebLogic Server をコンフィグレーションします。詳細については、6-23 ページの「一方向 SSL の属性の設定」を参照してください。
2. WebLogic Server の `config.xml` ファイル内のサーバ証明書ファイル名属性で、WebLogic Server の証明書を含むファイルの名前を指定します。このファイルにおける証明書の順序は重要です。最初にサーバのデジタル証明書を指定し、次に WebLogic Server の証明書を発行した認証局の証明書、その次に認証局の証明書の発行者を指定します。

3. WebLogic Server を再起動します。
4. BEA Tuxedo クライアントの SSL ライセンスを取得します。
5. BEA Tuxedo の信頼性のある認証局ファイル (`$TUXDIR/udataobj/security/certs/trust_ca.cer`) に、WebLogic Server の信頼性のある認証局をロードします。

   注意： このファイル内の証明書は、PEM フォーマットでなければなりません。
6. WebLogic Server を再起動します。
7. 以下の処理を行うコマンドライン オプションを指定して、BEA Tuxedo クライアントを起動します。
   - SSL プロトコルの使用を有効にする。
   - SSL ネットワーク接続が受け入れられるセキュア ポートを指定する。
   - WebLogic Server が動作しているネットワーク アドレスが、サーバのデジタル証明書内のドメイン名によって指定されているものと同一でない場合に、ピア検証を無効化する。

   次に例を示します。

```
./Client.exe -ORBid BEA_IIOP -ORBpeerValidate warn\
-ORBInitRef NameService=corbalocs:iiop:localhost:7002
/NameService
```

7　ユーザ アカウントの保護

以下の節では、ユーザ アカウントを保護する方法とユーザ アカウントのロックを解除する方法について説明します。

- 7-2 ページの「ユーザ アカウントのロックアウト属性の設定」
- 7-5 ページの「ユーザ アカウントのロック解除」

注意： 互換性セキュリティでユーザ アカウントを保護する方法については、9-5 ページの「互換性セキュリティでのユーザ アカウントの保護」を参照してください。

パスワードの保護

WebLogic Server ドメイン内のリソースにアクセスするためのパスワードを保護することは重要です。ユーザ名とパスワードは以前、WebLogic セキュリティ レルムにクリア テキストで保存されていました。現在、WebLogic Server ドメイン内のすべてのパスワードはハッシュ化されます。`SerializedSystemIni.dat` ファイルには、パスワード用のハッシュが入っています。このファイルは特定の WebLogic Server ドメインに関連付けられているので、ドメイン間で移動することはできません。

`SerializedSystemIni.dat` ファイルが破損した場合は、WebLogic Server ドメインを再コンフィグレーションしなければなりません。したがって、以下の予防措置をとってください。

- `SerializedSystemIni.dat` ファイルのバックアップを作成し、安全な場所に保管する。
- WebLogic Server デプロイメントのシステム管理者は読み書き特権を持ち、その他のユーザは何の特権も持たないように、`SerializedSystemIni.dat` ファイルのパーミッションを設定する。

ユーザ アカウントのロックアウト属性の設定

WebLogic Server には、ユーザ アカウントを侵入者から保護するための属性セットが定義されています。デフォルト セキュリティ コンフィグレーションでは、これらの属性は最高の保護レベルに設定されています。新しいセキュリティ レルムを作成する場合は、これらの属性を定義する必要があります。

システム管理者は、すべての属性を無効にしたり、アカウントがロックされるまでの無効なログイン試行回数を増やしたり、ユーザ アカウントがロックされるまでの無効なログイン試行期間を延ばしたり、ユーザ アカウントのロック時間を変更したりできます。これらの属性を変更すると、セキュリティ レベルが低下して攻撃を受けやすくなることに注意してください。

ユーザ ロックアウトの属性を設定するには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. コンフィグレーションするレルムの名前 (*myrealm* など) をクリックします。
3. [ユーザ ロックアウト] タブをクリックします。
4. 指示に従って値を入力したり、必要なチェックボックスをチェックしたりすることで、このタブで必要な属性を定義します (詳細については次の表を参照してください)。

   ユーザ アカウントがこれらの属性値を超えた場合、そのアカウントはロックされ、[ユーザ] タブのテーブルのそのアカウントの行に [詳細] という文字列が表示されます。詳細については、7-5 ページの「ユーザ アカウントのロック解除」を参照してください。
5. 変更を保存するには、[適用] をクリックします。
6. WebLogic Server を再起動します。

次の表では、[ユーザ ロックアウト] タブの各属性について説明します。

表 7-1 ユーザ ロックアウトの属性

属性	説明
[ロックアウト有効化]	ユーザ アカウントへの無効なログインが指定された [ロックアウトしきい値] を超えたときにそのユーザ アカウントのロックを要求する。デフォルトで、この属性は有効。
[ロックアウトしきい値]	アカウントがロックされるまでに許容されるユーザ パスワードの入力回数。この回数を超えてログインを試みると、(ユーザ名 / パスワードの組み合わせが正しい場合でも) セキュリティ例外が発生して、アカウントがロックアウトされる。システム管理者が明示的にロックを解除するか、またはロックアウト遅延時間が終了するまで、アカウントはロックアウトされたままとなる。ただし、無効なログインが [ロックアウト リセット遅延] 属性で定義された時間内に繰り返された場合。デフォルトでは 5。
[ロックアウト遅延]	[ロックアウト リセット遅延] 属性で定義された時間内に無効なログインが一定回数以上繰り返されたためにユーザ アカウントがロックされた後、ユーザ アカウントにアクセスできるようになるまでの時間 (分単位)。デフォルトでは 30 分。

表 7-1 ユーザ ロックアウトの属性

属性	説明
[ロックアウト リセット遅延]	ここで指定した分単位の時間内に一定回数以上の無効なログインが試みられた場合に、ユーザのアカウントをロックする。 Lockout Threshold 属性に定義された無効なログインの試行回数が、この属性で定義された時間内に行われた場合、アカウントはロックされる。たとえば、[ロックアウト リセット遅延] の値が 5 分、[ロックアウト遅延] の値が 3 で、6 分間に 3 回の無効なログインが試行された場合、アカウントはロックされない。しかし、5 分以内に 5 回の無効なログインが繰り返された場合、アカウントはロックされる。 デフォルトでは 5 分。
[ロックアウト キャッシュ サイズ]	試行しなかったログインと試行した無効なログインのキャッシュ サイズを指定する。 デフォルトでは 5。
[ロックアウト GC しきい値]	サーバがメモリ内に保存する無効ログイン レコードの最大数。現在の無効ログイン レコードの数がこの属性値と同じかそれを超過すると、期限切れとなったレコードがサーバのガベージ コレクションによって削除される。レコードは、それに関連付けられているユーザがロック アウトされたときに期限切れとなる。デフォルトは 400 レコード。

注意： [ユーザ ロックアウト] 属性は、セキュリティ レルムとそのすべてのセキュリティ プロバイダに適用されます。ユーザ アカウントを保護する独自のメカニズムを備えた認証プロバイダを使用する場合は、[ロックアウト有効化] 属性を無効化します。

ユーザ アカウントがロックされたためそのユーザ アカウントを削除して、同じ名前とパスワードを持つ別のユーザ アカウントを追加しても、UserLockout 属性がリセットされません。

ユーザ アカウントのロック解除

ユーザ アカウントのロックを解除するには、次の手順に従います。

1. [ユーザ] タブのテーブルの [詳細] リンクをクリックします。

   [詳細] タブには、ユーザがロック アウトされたときに発生したイベントの説明が表示されます。

2. [ロック解除] をクリックします。

8 WebLogic ドメインのセキュリティのコンフィグレーション

以下の節では、WebLogic ドメインのセキュリティ属性の設定方法について説明します。

- 8-1 ページの「WebLogic ドメイン間の信頼関係の有効化」
- 8-2 ページの「接続フィルタのコンフィグレーション」

注意： この章は、このリリースの WebLogic Server のセキュリティ機能を使用する WebLogic Server デプロイメントと互換性セキュリティを使用するデプロイメントに適用されます。

WebLogic ドメイン間の信頼関係の有効化

信頼関係は、ある WebLogic Server ドメイン (以下「ドメイン」) のサブジェクト内のプリンシパルがローカル ドメインのプリンシパルとして受け付けられたときに確立されます。

このリリースの WebLogic Server では、ドメイン間の信頼関係により厳しい制約が設定されています。あるドメインの資格属性が別のドメインの資格属性と一致したときに信頼関係が確立されるようになりました。

デフォルトでは、管理サーバを最初に起動したときには資格属性は定義されません。管理サーバは、起動時に資格属性が定義されていないことを認識し、ランダム資格を生成します。管理サーバは、この資格を使用してそのドメインに作成されたサブジェクト内のプリンシパルに署名します。資格を格納する `config.xml` ファイルは、資格が生成されてから保存する必要があります。このドメインの管理対象サーバは、起動時に管理サーバから資格を取得します。

WebLogic Server は、新しいサブジェクトの作成が要求されると検証 (プリンシパルの署名方法とローカル プリンシパルの署名方法の比較) を実行します。

注意： テキスト形式の資格は、次に `config.xml` ファイルがディスクに保存されるときに暗号化されます。

WebLogic Server 6.x ドメインと WebLogic Server 7.0 ドメインを相互運用する場合、WebLogic Server 7.0 ドメインの資格属性を、WebLogic Server 6.0 の `system` ユーザのパスワードに変更します。

2 つの 7.0 ドメインを相互運用する場合、両ドメインで次の手順を実行します。

WebLogic Server ドメイン間の信頼関係を確立するには、次の手順に従います。

1. [ドメイン] ノードを展開します。
2. [セキュリティ | 詳細設定] タブを選択します。
3. [生成された資格を有効化] 属性のチェックをはずします。
4. [資格] の横の [変更 ...] リンクをクリックします。
5. ドメインのパスワードを入力します。パスワードは慎重に選択してください。大文字、小文字、および数字の組み合わせを使用することをお勧めします。
6. [パスワードの確認] フィールドに新規パスワードをもう一度入力します。
7. [適用] をクリックします。
8. WebLogic Server を再起動します。

接続フィルタのコンフィグレーション

接続フィルタを使用すると、ネットワーク レベルでアクセスを拒否することができます。接続フィルタによって、個々のサーバ、サーバ クラスタ、または内部ネットワーク (イントラネット) のサーバ リソースを保護することができます。たとえば、ユーザの企業のネットワーク外部からの非 SSL 接続を拒否できます。ネットワーク接続フィルタは、プロトコル、IP アドレス、および DNS ノード名に基づいてフィルタ処理するようコンフィグレーションできる点において一種のファイアウォールです。

WebLogic Server では、ConnectionFilterImpl というデフォルト接続フィルタが用意されています。この接続フィルタは、すべての着信接続を受け入れます。また、サーバは、このクラスが提供する静的ファクトリ メソッドを使うことで、現在の接続フィルタを取得できます。アクセスを拒否するよう接続フィルタをコンフィグレーションするには、WebLogic Server Administration Console で接続フィルタ ルールを入力するだけです。

また、weblogic.security.net パッケージのクラスを実装することで、カスタム接続フィルタを使用することもできます。接続フィルタの記述については、『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の使い方」を参照してください。デフォルト接続フィルタと同様に、カスタム接続フィルタも WebLogic Server Administration Console でコンフィグレーションします。

接続フィルタをコンフィグレーションするには、次の手順に従います。

1. [ドメイン] ノードを展開します。
2. [セキュリティ | フィルタ] タブを選択します。
3. ドメインで使用する接続フィルタを指定します。
   - デフォルト接続フィルタをコンフィグレーションするには、[接続フィルタ] 属性フィールドに weblogic.security.net.ConnectionFilterImpl を指定します。
   - カスタム接続フィルタをコンフィグレーションするには、[接続フィルタ] 属性フィールドにネットワーク接続フィルタを実装するクラスを指定します。このクラス名は、WebLogic Server の CLASSPATH にも指定する必要があります。
4. 接続フィルタのルールの構文を入力します。接続フィルタ ルールの詳細については、「ネットワーク接続フィルタの使い方」を参照してください。
5. [適用] をクリックします。
6. WebLogic Server を再起動します。
7. [ドメイン] ノードを展開します。
8. [セキュリティ] タブをクリックします。
9. [詳細設定] タブをクリックします。

10. [接続ログを有効化] 属性をクリックして、受け付けたメッセージのログの記録を有効にします。この属性は、正常に実行された接続およびサーバ内の接続データをログに記録します。この情報は、サーバ接続に関連する問題のデバッグに使用できます。
11. [適用] をクリックします。

9　互換性セキュリティの使い方

次の節では、互換性セキュリティをコンフィグレーションする方法について説明します。

注意： 互換性セキュリティは、このリリースの WebLogic Server では非推奨になっています。互換性セキュリティは、WebLogic Server デプロイメントをこのリリースの WebLogic Server のセキュリティ機能にアップグレードするときにのみ使用してください。

互換性セキュリティの実行：主な手順

互換性セキュリティを設定するには、次の手順に従います。

1. 互換性セキュリティを使用する前に、6.x の WebLogic ドメイン (`config.xml` ファイルを含む) をバックアップします。互換性セキュリティを起動するためのサンプル `config.xml` ファイルについては、『BEA WebLogic Server 7.0 へのアップグレード』の「互換性セキュリティでの WebLogic Server の起動」を参照してください。
2. 6.x config.xml ファイルに以下を追加します。

```
<Security Name="mydomain" Realm="mysecurity"/>
<Realm Name="mysecurity" FileRealm="myrealm"/>
<FileRealm Name="myrealm"/>
```

3. WebLogic Server 7.0 を新しいディレクトリにインストールします。既存の 6.x インストール ディレクトリに上書きしないでください。詳細については、『インストール ガイド』を参照してください。
4. 6.x サーバの起動スクリプトを、WebLogic Server 7.0 を指すよう修正します。具体的には、次のとおり修正します。
   - クラスパスが WebLogic Server 7.0 の `weblogic.jar` ファイルを指すようにする
   - `JAVA_HOME` 変数が WebLogic Server 7.0 を指すようにする
5. 6.x サーバの起動スクリプトを使用して WebLogic Server を起動します。

互換性セキュリティを適切に実行しているかどうかを検証するには、次の手順に従います。

1. WebLogic Server Administration Console で [ドメイン] ノードを開きます。
2. WebLogic Server ドメイン (以下「ドメイン」) をクリックします。
3. [ドメイン ログの表示] リンクをクリックします。

   次のメッセージがログに表示されます。

   ```
   Security initializing using realm CompatibilityRealm
   ```

また、[CompatibilitySecurity] ノードが WebLogic Server Administration Console に表示されます。

CompatibilityRealm のデフォルト セキュリティ コンフィグレーション

デフォルトでは、*CompatibilityRealm* は、レルム アダプタ裁決プロバイダ、レルム アダプタ認証プロバイダ、WebLogic 認可プロバイダ、レルム アダプタ認可プロバイダ、WebLogic 資格マッピング プロバイダ、および WebLogic ロール マッピング プロバイダでコンフィグレーションされています。

- *CompatibilityRealm* では、レルム アダプタ認証プロバイダが、`config.xml` ファイルに定義されている 6.x セキュリティ レルムからユーザとグループを取得する。
  - 6.x セキュリティのコンフィグレーションでファイル レルムを使用していた場合、WebLogic Server Administration Console の互換性セキュリティ オンライン ヘルプにおける「CompatibilityRealm でのユーザの定義」および「CompatibilityRealm でのグループの定義」に記載の手順を実行することで、レルム アダプタ認証プロバイダ内のユーザおよびグループを管理できる。
  - 代替セキュリティ レルム (LDAP、Windows NT、RDBMS、またはカスタム) を使用している場合、ユーザとグループを管理するには、そのレルムで用意されている管理ツールを使用する必要がある。

  Windows NT、RDBMS、UNIX、またはカスタム セキュリティ レルムに格納されているユーザおよびグループの数が多く、WebLogic、LDAP、またはカスタム認証プロバイダにアップグレードしたい場合、新しいセキュリティ レルムでレルム アダプタ認証プロバイダをコンフィグレーションすると、既存の 6.x のストアにアクセスできます。

  注意： レルム アダプタ認証プロバイダは、CompatibiltyRealm 以外のレルムでコンフィグレーション可能な唯一のレルム アダプタ プロバイダです。

  レルム アダプタ認証プロバイダのコンフィグレーションの詳細については、3-28 ページの「レルム アダプタ認証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。

  レルム アダプタ認証プロバイダで 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と、互換性セキュリティで `weblogic.security.acl.CertAuthenticator` クラスの実装を使用できます。詳細については、9-4 ページの「レルム アダプタ認証プロバイダでの ID アサーション 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。

- 6.x セキュリティ レルムのアクセス制御リスト (ACL) は、レルム アダプタ認可プロバイダに格納される。

- レルム アダプタ認可プロバイダを使用すると、互換性セキュリティで ACL を使用することも、セキュリティ ロールおよびセキュリティ ポリシーを使用することもできる。レルム アダプタ裁決プロバイダは、ACL と、WebLogic Server Administration Console で設定した新しいセキュリティ ポリシーとのアクセス決定の衝突を解決します。

- WebLogic 資格マッピング プロバイダを使用すると、互換性セキュリティで資格マップを使用できる。
- レルム アダプタ監査プロバイダを追加すると、CompatibilityRealm から `weblogic.security.audit.AuditProvider` クラスの実装にアクセスできる。

レルム アダプタ認証プロバイダでの ID アサーション プロバイダのコンフィグレーション

レルム アダプタ認証プロバイダには、ID アサーション プロバイダが含まれています。ID アサーション プロバイダは、`weblogic.security.acl.CertAuthenticator` クラスの実装に下位互換性を提供します。ID アサーションは X.509 トークンに関して実行されます。デフォルトでは、ID アサーション プロバイダはレルム アダプタ認証プロバイダで有効になっていません。

レルム アダプタ認証プロバイダで ID アサーションを有効にするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. [CompatibilityRealm] をクリックします。
3. [プロバイダ] ノードを展開します。
4. [認証プロバイダ] をクリックします。
5. [レルム] テーブルの [Realm Adapter Authenticator] リンクをクリックします。

   [一般] タブが表示されます。
6. [アクティブ タイプ] リスト ボックスに X.509 と入力します。

   この手順によって、6.x 証明書認証プロバイダを使用できるようになります。
7. [適用] をクリックします。

8. WebLogic Server を再起動します。

レルム アダプタ監査プロバイダのコンフィグレーション

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場合に `weblogic.security.audit.AuditProvider` クラスの実装を使用できるようになります。レルム アダプタ監査プロバイダが正しく動作するためには、`weblogic.security.audit.AuditProvider` クラスの実装が、[ドメイン | セキュリティ | 一般] タブの [監査プロバイダ クラス] 属性に定義されている必要があります。

レルム アダプタ監査プロバイダをコンフィグレーションするには、次の手順に従います。

1. [以前のセキュリティ | レルム] ノードを展開します。
2. [プロバイダ] ノードを展開します。
3. [監査] をクリックします。
4. [Realm Adapter Auditor のコンフィグレーション] リンクをクリックします。

   [一般] タブが表示されます。
5. [作成] をクリックして変更を保存します。
6. WebLogic Server を再起動します。

互換性セキュリティでのユーザ アカウントの保護

WebLogic Server には、ユーザ アカウントを侵入者から保護するための属性セットが用意されています。デフォルトでは、これらの属性は最高の保護レベルに設定されています。システム管理者は、すべての属性を無効にしたり、アカウント

がロックされるまでの無効なログイン試行回数を増やしたり、ユーザ アカウントがロックされるまでの無効なログイン試行期間を延ばしたり、ユーザ アカウントのロック時間を変更したりできます。これらの属性を変更すると、セキュリティ レベルが低下して攻撃を受けやすくなることに注意してください。

WebLogic Server ドメインのユーザ アカウントを保護するには、次の手順に従います。

1. [ドメイン] ノードをクリックします。
2. [セキュリティ | パスワード] タブを選択します。
3. 指示に従って値を入力したり、必要なチェックボックスをチェックしたりすることで、このタブで必要な属性を定義します (詳細については次の表を参照してください)。
4. [適用] をクリックして選択を保存します。
5. WebLogic Server を再起動します。

次の表では、[パスワード] タブの各属性について説明します。

表 9-1 パスワード保護の属性

属性	説明
[最小パスワード文字数]	パスワードに必要な文字数。パスワードは 8 文字以上でなければならない。デフォルトでは 8。
[ロックアウト有効化]	ユーザ アカウントへの無効なログインが指定された [ロックアウトしきい値] を超えたときにそのユーザ アカウントのロックを要求する。デフォルトで、この属性は有効。

表 9-1 パスワード保護の属性 (続き)

属性	説明
[ロックアウトしきい値]	アカウントにログインしようとする場合に、アカウントがロックアウトされるまでにユーザが間違ったパスワードを入力してもよい回数。この回数を超えてログインを試みると、(ユーザ名 / パスワードの組み合わせが正しい場合でも) セキュリティ例外が発生して、アカウントがロックアウトされる。システム管理者が明示的にロックを解除するか、またはロックアウト遅延時間が終了するまで、アカウントはロックアウトされたままとなる。ただし、無効なログインが [ロックアウト リセット遅延] 属性で定義された時間内に繰り返された場合。デフォルトでは 5。
[ロックアウト遅延]	[ロックアウト リセット遅延] 属性で定義された時間内に無効なログインが一定回数以上繰り返されたためにユーザ アカウントがロックされた後、ユーザ アカウントにアクセスできるようになるまでの時間 (分単位)。ユーザ アカウントをロック解除するには、`weblogic.passwordpolicy` の `unlockuser` パーミッションが必要。デフォルトでは 30 分。
[ロックアウト リセット遅延]	ここで指定した分単位の時間内に一定回数以上の無効なログインが試みられた場合に、ユーザのアカウントをロックする。 [ロックアウトしきい値] 属性で定義された無効なログインの試行回数が、この属性に定義された時間内に行われた場合、アカウントはロックアウトされる。たとえば、この属性の値が 5 分で、6 分間に 3 回ログインが失敗した場合、アカウントはロックされない。しかし、5 分以内に 5 回の無効なログインが繰り返された場合、アカウントはロックされる。 デフォルトでは 5 分。

表 9-1 パスワード保護の属性 (続き)

属性	説明
[ロックアウト キャッシュ サイズ]	試行しなかったログインと試行した無効なログインのキャッシュ サイズを指定する。デフォルトでは 5。

ユーザ アカウントを保護するための属性セットには、ドメインで設定するものとセキュリティ レルムで設定するものがあります。いずれかの属性セット (たとえばセキュリティ レルムの属性) を設定し、一方の値が他方の値を超えた場合、ユーザ アカウントはロックされないことに注意してください。これは、ドメインで設定するユーザ アカウントの属性がセキュリティ レルムで設定するユーザ アカウントの属性をオーバライドするからです。こうした状況を避けるには、セキュリティ レルムで設定するユーザ アカウントの属性を無効にします。

セキュリティ レルムで設定するユーザ アカウントの属性を無効にするには、次の手順に従います。

1. [セキュリティ | レルム] ノードを展開します。
2. [CompatibilityRealm] ノードを展開します。
3. [ユーザ ロックアウト] タブを選択します。
4. [ロックアウト有効化] 属性のチェックをはずします。
5. [適用] をクリックします。
6. WebLogic Server を再起動します。

警告 : セキュリティ レルムのユーザ アカウントの属性を無効にする場合は、ドメインでユーザ アカウントの属性を設定しないと、ユーザ アカウントが保護されません。

互換性セキュリティから 6.x セキュリティへのアクセス

互換性セキュリティを使用しているときには、ユーザおよびグループを定義するセキュリティ レルムと、WebLogic Server ドメイン内のリソースを保護する ACL を備えた、既存の `config.xml` ファイルがあることが前提となっています。セキュリティ レルムのコンフィグレーションや ACL の定義などの 6.x セキュリティ管理タスクは必要とされないため、この章では説明しません。ただし、既存の 6.x セキュリティ レルムが破損し、それを復元する以外に方法がない場合に備えて、WebLogic Server Administration Console のオンライン ヘルプにおける互換性セキュリティの節で、以下の 6.x セキュリティ管理タスクが説明されています。

- ファイル レルムのコンフィグレーション
- キャッシング レルムのコンフィグレーション
- LDAP V1 セキュリティ レルムのコンフィグレーション
- LDAP V2 セキュリティ レルムのコンフィグレーション
- Windows NT セキュリティ レルムのコンフィグレーション
- UNIX セキュリティ レルムのコンフィグレーション
- RDBMS セキュリティ レルムのコンフィグレーション
- カスタム セキュリティ レルムのインストール
- ユーザの定義
- ユーザの削除
- ユーザ パスワードの変更
- ユーザ アカウントのロック解除
- ゲスト ユーザの無効化
- グループの定義
- グループの削除

- ACL の定義

注意： 互換性セキュリティは下位互換性を提供するだけなので、長期にわたるセキュリティ対策と考えないでください。