

Oracle® Enterprise Performance Management System セキュリティ構成ガイド



リリース 11.2

F28849-24

2023 年 10 月

ORACLE®

Copyright © 2005, 2023, Oracle and/or its affiliates.

著者: EPM Information Development Team

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

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

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

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

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

目次

ドキュメントのアクセシビリティについて

ドキュメントのフィードバック

1 EPM System セキュリティについて

EPM System について	1-1
必要な知識	1-1
セキュリティ・インフラストラクチャのコンポーネント	1-2
ユーザー認証	1-2
プロビジョニング(役割ベースの認証)	1-6
Shared Services Console の起動	1-9

2 EPM System コンポーネントの SSL 使用可能化

前提条件	2-1
情報ソース	2-1
場所の参照	2-2
EPM System 製品の SSL 使用可能化について	2-2
サポートされている SSL シナリオ	2-3
必須の証明書	2-4
SSL オフローダでの SSL 停止	2-4
EPM System の完全な SSL デプロイメント	2-6
デプロイメント・アーキテクチャ	2-6
前提条件	2-7
完全 SSL 用 EPM System の構成	2-8
EPM System の共通設定の再構成	2-9
オプション: WebLogic Server に対するルート CA 証明書のインストール	2-10
WebLogic Server に対する証明書のインストール	2-11
WebLogic Server の構成	2-12
SSL 対応 Oracle Database との HFM サーバーの接続の有効化	2-14

Oracle HTTP Server に関する手順	2-20
WebLogic Server に配置された EPM System Web コンポーネントの構成	2-24
ドメイン構成の更新	2-25
サーバーおよび EPM System の再起動	2-27
デプロイメントのテスト	2-27
SSL 使用可能な外部ユーザー・ディレクトリの構成	2-27
Web サーバーでの SSL の停止	2-28
Essbase 11.1.2.4 用 SSL	2-30
Essbase コンポーネントのインストールとデプロイ	2-32
信頼できるサードパーティ CA 証明書の Essbase への使用	2-33
セッションごとの SSL 接続の確立	2-40
Essbase 21c 用 SSL	2-41
Essbase コンポーネントのインストールとデプロイ	2-44
信頼できるサードパーティ CA 証明書の Essbase への使用	2-44
セッションごとの SSL 接続の確立	2-50

3 セキュリティ・エージェントでの SSO の使用可能

サポートされている SSO の方法	3-1
Oracle Access Manager からのシングル・サインオン	3-4
OracleAS シングル・サインオン	3-5
デプロイメントのテスト	3-7
EPM System 向けの OSSO の使用可能化	3-7
SSO 用の EPM System 製品の保護	3-11
アイデンティティ管理製品を使用したヘッダーベース SSO	3-16
Oracle Identity Cloud Services を使用したヘッダーベース SSO のための EPM System の構成	3-18
前提条件およびサンプル URL	3-18
EPM System のヘッダーベース認証の使用可能化	3-19
Oracle Identity Cloud Services への EPM System アプリケーションおよびゲートウェイの追加	3-19
アプリケーション・ゲートウェイの構成	3-25
認証のためのユーザー・ディレクトリの構成	3-25
EPM System における SSO の使用可能	3-25
EPM Workspace 設定の更新	3-26
SiteMinder SSO	3-26
Kerberos シングル・サインオン	3-29
SSO 用の EPM System の構成	3-43
Smart View に対するシングル・サインオンのオプション	3-45

4 ユーザー・ディレクトリの構成

ユーザー・ディレクトリおよび EPM System セキュリティ	4-1
ユーザー・ディレクトリ構成に関連する操作	4-2
Oracle Identity Manager と EPM System	4-2
Active Directory の情報	4-3
OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成	4-4
リレーショナル・データベースをユーザー・ディレクトリとして構成する	4-18
ユーザー・ディレクトリの接続のテスト	4-21
ユーザー・ディレクトリ設定の編集	4-21
ユーザー・ディレクトリ構成の削除	4-22
ユーザー・ディレクトリの検索順の管理	4-23
セキュリティ・オプションの設定	4-24
暗号化キーの再生成	4-27
特殊文字の使用	4-29

5 カスタム認証モジュールの使用

概要	5-1
使用事例の例と制限	5-3
前提条件	5-3
設計およびコーディングに関する考慮事項	5-3
カスタム認証モジュールのデプロイ	5-8

6 EPM System の保護のガイドライン

SSL の実装	6-1
管理パスワードの変更	6-1
暗号化キーの再生成	6-2
データベース・パスワードの変更	6-2
Cookie の保護	6-3
SSO トークンのタイムアウトの低減	6-4
セキュリティ・レポートの確認	6-4
認証システムの強力な認証としてのカスタマイズ	6-4
EPM Workspace のデバッグ・ユーティリティを使用不可にする	6-4
デフォルトの Web サーバー・エラー・ページの変更	6-5
サードパーティ製ソフトウェアのサポート	6-5

A カスタム認証サンプル・コード

サンプル・コード 1	A-1
サンプル・コード 2	A-2
サンプル・コード 2 のデータ・ファイル	A-4

B カスタム・ログイン・クラスの実装

カスタム・ログイン・クラス・サンプル・コード	B-1
カスタム・ログイン・クラスのデプロイ	B-4

C ユーザー・ディレクトリ全体のユーザーとグループの移行

概要	C-1
前提条件	C-1
移行手順	C-2
個々の製品の更新	C-5

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility Program の Web サイト(<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>)を参照してください。

Oracle サポートへのアクセス

サポートをご契約のお客様には、My Oracle Support を通して電子支援サービスを提供しています。詳細情報は <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> か、聴覚に障害のあるお客様は <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> を参照してください。

ドキュメントのフィードバック

このドキュメントに対するフィードバックを送るには、Oracle Help Center トピックのページの下部にあるフィードバック・ボタンをクリックします。
epmdoc_ww@oracle.com に電子メールを送信することもできます。

1

EPM System セキュリティについて

次も参照:

- [EPM System について](#)
- [必要な知識](#)
- [セキュリティ・インフラストラクチャのコンポーネント](#)
- [ユーザー認証](#)
- [プロビジョニング\(役割ベースの認証\)](#)
- [Shared Services Console の起動](#)

EPM System について

Oracle Enterprise Performance Management System 製品は、財務管理アプリケーションおよびプランニング・アプリケーションのモジュール式スイートと、レポートおよび分析の最も総合的なビジネス・インテリジェンス機能を統合する、総合的なエンタープライズ・システムを形成できます。EPM System 製品の主なコンポーネントは次のとおりです:

- Oracle Hyperion Foundation Services
- Oracle Essbase
- Oracle Hyperion Financial Management
- Oracle Hyperion Planning

これらの各製品ファミリの製品およびコンポーネントの詳細は、*Oracle Enterprise Performance Management System* インストール概要を参照してください。

必要な知識

このガイドは、Oracle Enterprise Performance Management System コンポーネントの構成、保護および管理を担当するシステム管理者を対象としています。前提条件となる知識は次のとおりです:

- 次のような、所属組織のセキュリティ・インフラストラクチャに関する十分な理解:
 - Oracle Internet Directory、Sun Java System Directory Server および Microsoft Active Directory などのディレクトリ・サーバー
 - 通信チャンネルを保護する Secure Socket Layer (SSL)の使用
 - Oracle Access Manager および SiteMinder などのアクセス管理システム
 - Kerberos などのシングル・サインオン(SSO)インフラストラクチャ
- 所属組織に関連する EPM System セキュリティの概念に関する知識

セキュリティ・インフラストラクチャのコンポーネント

Oracle Enterprise Performance Management System では、いくつかのセキュリティ・コンポーネントを統合することによって、堅牢なアプリケーション・セキュリティが確保されます。EPM System は、セキュアなインフラストラクチャと統合することにより、データおよびアクセスのセキュリティを保証する高度なセキュア・アプリケーション・スイートを提供します。EPM System の保護に使用できるインフラストラクチャ・コンポーネントは次のとおりです：

- オプションのアクセス管理システム(EPM System コンポーネントへの SSO アクセスを提供する Oracle Access Manager など)
- 統合 SSO インフラストラクチャ(Kerberos など)の使用。

Kerberos 認証をアクセス管理システム(SiteMinder)とともに使用すると、Windows ユーザーは、SiteMinder および EPM System コンポーネントに透過的にログインできます。

- EPM System コンポーネントおよびクライアント間の通信チャンネルを保護する Secure Socket Layer (SSL)の使用

ユーザー認証

ユーザー認証により、各ユーザーのログイン情報を検証して認証済ユーザーを判別することで、Oracle Enterprise Performance Management System コンポーネント全体でシングル・サインオン(SSO)機能が使用可能になります。コンポーネント固有の認可とともにユーザー認証は、EPM System コンポーネントへユーザー・アクセスを認めます。権限を付与するプロセスは、プロビジョニングと呼ばれます。

認証コンポーネント

次の項では、SSO をサポートするコンポーネントについて説明します。

- [ネイティブ・ディレクトリ](#)
- [外部ユーザー・ディレクトリ](#)

ネイティブ・ディレクトリ

ネイティブ・ディレクトリとは、Oracle Hyperion Shared Services がプロビジョニングのサポート、およびデフォルト・ユーザー・アカウントなどのシード・データの保管に使用するリレーショナル・データベースを指します。

ネイティブ・ディレクトリ機能:

- デフォルトの EPM System ユーザー・アカウントの維持と管理
- 全 EPM System プロビジョニング情報(ユーザー、グループおよび役割間の関係)の保管

ネイティブ・ディレクトリは、Oracle Hyperion Shared Services Console を使用してアクセスおよび管理します。Oracle Enterprise Performance Management System [ユ](#)

ユーザー・セキュリティ管理ガイドのネイティブ・ディレクトリの管理を参照してください。

外部ユーザー・ディレクトリ

ユーザー・ディレクトリとは、EPM System コンポーネントと互換性のある、企業ユーザーおよびアイデンティティの管理システムを指します。

EPM System コンポーネントは、Oracle Internet Directory、Sun Java System Directory Server (旧 SunONE Directory Server)、Microsoft Active Directory などの LDAP ベースのユーザー・ディレクトリを含むいくつかのユーザー・ディレクトリでサポートされています。リレーショナル・データベースもユーザー・ディレクトリとしてサポートされています。このドキュメントでは、ネイティブ・ディレクトリ以外のユーザー・ディレクトリを外部ユーザー・ディレクトリと呼びます。

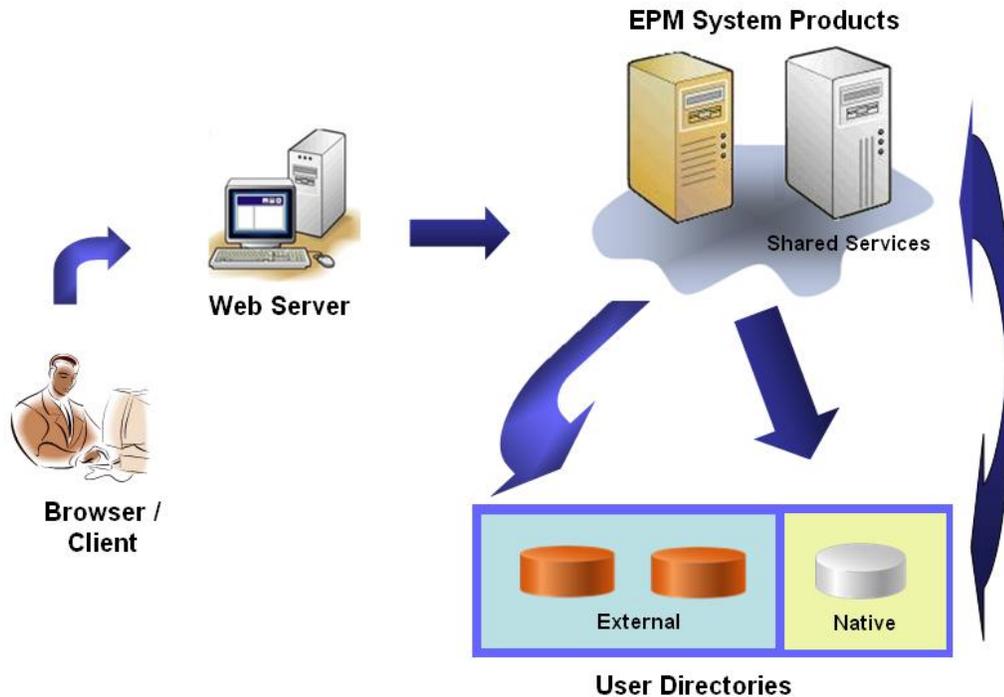
サポートされているユーザー・ディレクトリのリストは、Oracle Technology Network (OTN) の [Oracle Fusion Middleware Supported System Configurations](#) ページに掲載されている *Oracle Enterprise Performance Management System* の動作保証マトリックスを参照してください。

Shared Services Console から、多くの外部ユーザー・ディレクトリを EPM System ユーザーおよびグループのソースとして構成できます。各 EPM System ユーザーは、構成済ユーザー・ディレクトリ内で一意のアカウントを持っている必要があります。通常、EPM System ユーザーは、プロビジョニングを促進するためにグループに割り当てられます。

デフォルトの EPM System シングル・サインオン

EPM System では、EPM System アプリケーション全体での SSO がサポートされており、あるアプリケーションの認証済ユーザーは、資格証明を再入力することなく別のアプリケーションにシームレスに移動できます。SSO は、ユーザー認証およびプロビジョニング(役割ベースの認証)を処理する共通のセキュリティ環境を EPM System コンポーネント全体で統合することによって実装されます。

デフォルトの SSO プロセスを次の図に示します。



1. ユーザーは、ブラウザ経由で EPM System コンポーネントのログイン画面にアクセスし、ユーザー名とパスワードを入力します。
EPM System コンポーネントにより、構成済ユーザー・ディレクトリ(ネイティブ・ディレクトリなど)への問合せが行われ、ユーザー資格証明が確認されます。ユーザー・ディレクトリで一致するユーザー・アカウントが見つかったら、検索は終了し、ユーザー情報が EPM System コンポーネントに戻されます。
ユーザー・アカウントがどの構成済ユーザー・ディレクトリにもない場合、アクセスは拒否されます。
2. 取得したユーザー情報を使用して、EPM System コンポーネントにより、ネイティブ・ディレクトリへの問合せが行われ、ユーザーのプロビジョニングの詳細が入手されます。
3. EPM System コンポーネントにより、コンポーネントのアクセス制御リスト(ACL)がチェックされ、ユーザーがアクセスできるアプリケーション・アーティファクトが決定されます。

ネイティブ・ディレクトリからプロビジョニング情報を受け取ると、EPM System コンポーネントはユーザーに対して使用可能になります。この時点で、SSO は、ユーザーがプロビジョニングされているすべての EPM System コンポーネントで使用可能です。

アクセス管理システムからのシングル・サインオン

EPM System コンポーネントのセキュリティをさらに強化するには、Oracle Access Manager または SiteMinder など、サポートされているアクセス管理システムを実装できます。これらの製品では、認証済ユーザー資格証明を EPM System コンポーネントに提供し、事前定義済のアクセス権に基づいてアクセスを制御できます。

セキュリティ・エージェントからの SSO は EPM System Web アプリケーションでのみ使用可能です。このシナリオでは、EPM System コンポーネントは、セキュリティ・

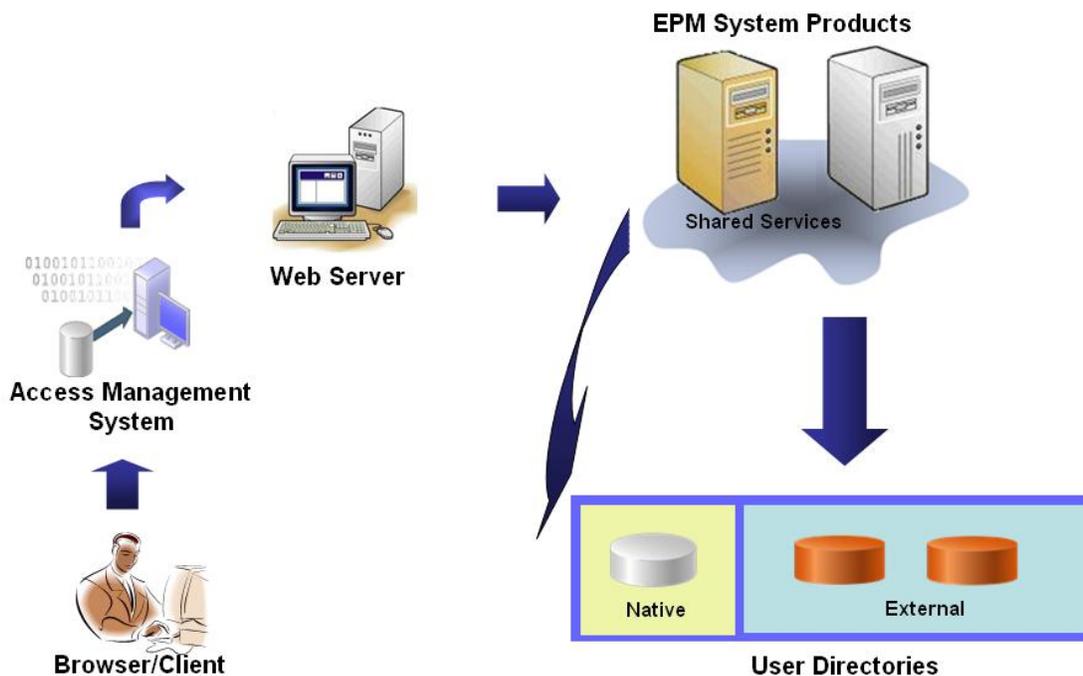
エージェントから提供されるユーザー情報を使用して、ユーザーのアクセス権限を判別します。セキュリティを強化するには、すべての要求が SSO ポータルを経由するように、サーバーへの直接のアクセスをファイアウォールでブロックすることをお勧めします。

アクセス管理システムからの SSO は、条件を満たした SSO メカニズム経由で認証済のユーザー資格証明を受け入れることによりサポートされます。[サポートされている SSO の方法](#)を参照してください。アクセス管理システムにより、ユーザーが認証され、ユーザーのログイン名が EPM System に渡されます。EPM System により、構成済のユーザー・ディレクトリに対してログイン名が確認されます。

次のトピックを参照してください。

- [Oracle Access Manager からのシングル・サインオン](#)
- [OracleAS シングル・サインオン](#)
- [SiteMinder SSO](#)
- [Kerberos シングル・サインオン](#)

概念を示した図:



1. ブラウザを使用して、ユーザーはアクセス管理システム(Oracle Access Manager、SiteMinder など)で保護されているリソースへのアクセスを要求します。

ノート:

EPM System コンポーネントは、アクセス管理システムで保護されているリソースとして定義されます。

アクセス管理システムは、要求をインターセプトし、ログイン画面を表示します。ユーザーはユーザー名とパスワードを入力します。これらは、ユーザーの信頼性を確認するためにアクセス管理システムで構成済ユーザー・ディレクトリに対して検証されます。EPM System コンポーネントは、これらのユーザー・ディレクトリと連動するようにも構成されています。

認証済ユーザーに関する情報は、EPM System コンポーネントに渡され、そのコンポーネントで有効なものとして受け入れられます。

アクセス管理システムは、条件を満たした SSO メカニズムを使用して、ユーザーのログイン名(ログイン属性の値)を EPM System コンポーネントに渡します。サポートされている SSO の方法を参照してください。

2. ユーザー資格証明を確認するために、EPM System コンポーネントにより、ユーザー・ディレクトリでユーザーの検索が試みられます。一致するユーザー・アカウントが見つかったら、ユーザー情報が EPM System コンポーネントに戻されます。EPM System セキュリティにより、EPM System コンポーネント全体で SSO を使用可能にする SSO トークンが設定されます。

3. 取得したユーザー情報を使用して、EPM System コンポーネントにより、ネイティブ・ディレクトリへの問合せが行われ、ユーザーのプロビジョニングの詳細が入手されます。

ユーザー・プロビジョニング情報を受け取ると、EPM System コンポーネントはユーザーに対して使用可能になります。SSO は、ユーザーがプロビジョニングされているすべての EPM System コンポーネントで使用可能です。

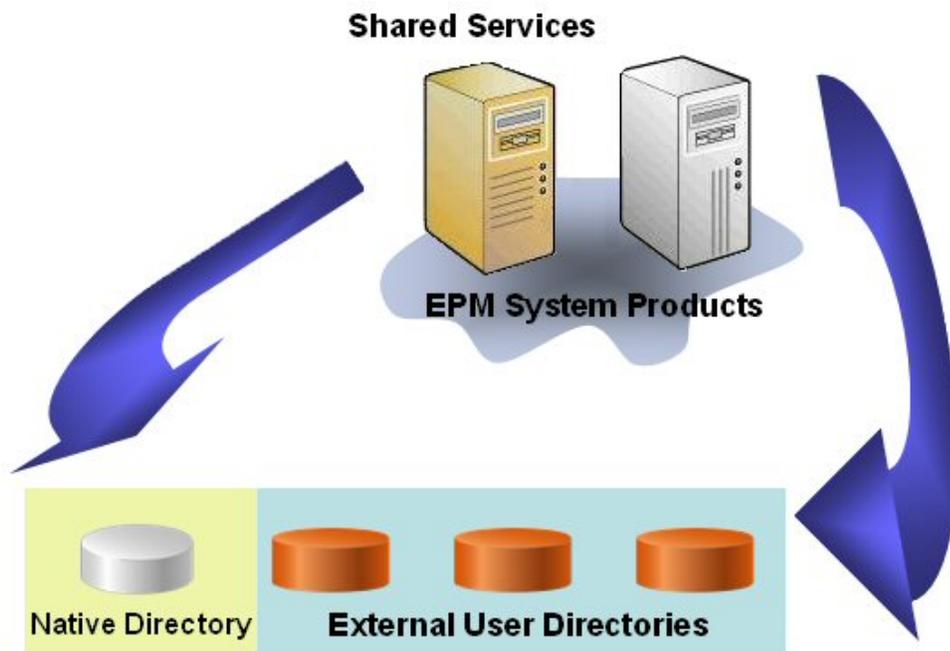
プロビジョニング(役割ベースの認証)

Oracle Enterprise Performance Management System のセキュリティでは、役割の概念を使用してアプリケーションに対するユーザーのアクセス権を決定します。役割とは、アプリケーション機能へのユーザー・アクセスを判別する権限です。一部の EPM System コンポーネントでは、レポートおよびメンバーなどのアーティファクトへのユーザー・アクセスをさらに詳細に制限するために、オブジェクトレベルの ACL が使用されます。

各 EPM System コンポーネントでは、様々な業務上の必要に対して調整された数個のデフォルトの役割が提供されます。EPM System コンポーネントに属する各アプリケーションはこの役割を継承します。Oracle Hyperion Shared Services に登録されているアプリケーションの事前定義済役割は、Oracle Hyperion Shared Services Console から使用できます。特定の要件に合うように、デフォルトの役割を集約する追加の役割も作成できます。この役割はプロビジョニングで使用されます。EPM System アプリケーションおよびそのリソースに属する固有の役割をユーザーおよびグループに付与するプロセスをプロビジョニングと呼びます。

ネイティブ・ディレクトリおよび構成済ユーザー・ディレクトリは、プロビジョニング・プロセス用のユーザーとグループ情報のソースです。Shared Services Console から、すべての構成済ユーザー・ディレクトリのユーザーとグループを参照およびプロビジョニングできます。また、ネイティブ・ディレクトリで作成されたアプリケーション固有の集約役割をプロビジョニング・プロセスで使用することもできます。

承認プロセスの概要図:



1. ユーザーが認証されたら、EPM System コンポーネントにより、ユーザー・ディレクトリへの問合せが行われ、ユーザーのグループが判別されます。
2. EPM System コンポーネントにより、グループ情報とユーザー情報を使用して、Shared Services からユーザーのプロビジョニング・データが取得されます。このデータを使用してユーザーがアクセスできるリソースが決定されます。

製品固有のアクセス制御を設定するなどの製品固有のプロビジョニング・タスクは、各製品向けに完成されます。このデータは、プロビジョニング・データと組み合わされて、ユーザーの製品アクセスを決定します。

EPM System 製品の役割ベースのプロビジョニングでは、これらのコンセプトが使用されます。

役割

役割は、EPM System リソースで機能を実行するユーザーとグループに許可されるアクセス権限を定義するコンストラクト(アクセス制御リストに類似)です。役割は、リソースまたはリソース・タイプ(レポートなどのユーザーがアクセスできるもの)と、ユーザーがリソースで実行できるアクション(表示や編集など)の組合せです。

EPM System アプリケーション・リソースへのアクセスは制限されています。アクセスを提供する役割がユーザー、またはユーザーが属するグループに割り当てられてからのみ、ユーザーはこれらのリソースにアクセスできます。役割に基づいたアクセス制限では、管理者は、アプリケーション・アクセスを制御および管理できます。

グローバル役割

グローバル役割、つまり複数の製品に及ぶ Shared Services の役割により、ユーザーは EPM System 製品間で特定のタスクを実行できます。たとえば、Shared Services 管理者は、すべての EPM System アプリケーションについてユーザーをプロビジョニングできます。

事前定義済役割

事前定義済役割は、EPM System 製品における組込みの役割です。これらを削除することはできません。EPM System 製品に属する各アプリケーション・インスタンスは、EPM System 製品の事前定義済役割を継承します。各アプリケーションのこれらの役割は、アプリケーションの作成時に Shared Services に登録されます。

集約役割

カスタム役割という名でも知られる集約役割では、アプリケーションに属する複数の事前定義済役割が集約されます。集約役割には、他の集約役割を含めることができます。たとえば、Shared Services 管理者またはプロビジョニング・マネージャは、Oracle Hyperion Planning アプリケーションのプランナと表示ユーザーの役割を組み合わせた集約役割を作成できます。役割を集約することにより、複数の細かい役割を持つアプリケーションの管理を簡略化できます。グローバル Shared Services の役割は、集約役割に含めることができます。複数のアプリケーションまたは製品に及ぶ集約役割は作成できません。

ユーザー

ユーザー・ディレクトリには、EPM System 製品にアクセスできるユーザーに関する情報が保管されています。認証および承認プロセスの双方でユーザー情報が使用されます。ネイティブ・ディレクトリ・ユーザーの作成と管理は、Shared Services Console でのみ行うことができます。

すべての構成済ユーザー・ディレクトリからのユーザーは、Shared Services Console から確認できます。これらのユーザーは、Shared Services に登録された EPM System アプリケーションでアクセス権を許可するように個別にプロビジョニングできます。個別ユーザーへのプロビジョニングはお勧めしません。

デフォルトの EPM System 管理者

管理者アカウント(デフォルト名 admin)は、デプロイメント・プロセス中にネイティブ・ディレクトリに作成されます。これは最も強力な EPM System アカウントで、EPM System セキュリティおよび環境の管理の責任を負う情報テクノロジーの専門家であるシステム管理者の設定にのみ使用される必要があります。

EPM System 管理者のユーザー名およびパスワードは Oracle Hyperion Foundation Services のデプロイメント中に設定されます。このアカウントは企業のアカウント・パスワード・ポリシーの対象にできないため、システム管理者アカウントを作成した後非アクティブにすることをお勧めします。

通常、デフォルトの EPM System 管理者アカウントは次のタスクを実行するために使用します:

- 企業ディレクトリを外部ユーザー・ディレクトリとして構成します。[ユーザー・ディレクトリの構成](#)を参照してください。
- 企業の情報テクノロジーの専門家に Shared Services 管理者の役割をプロビジョニングして、システム管理者アカウントを作成します。[Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド](#)のユーザーとグループのプロビジョニングを参照してください。

システム管理者

システム管理者は通常、企業の情報テクノロジーの専門家で、EPM System デプロイメントに含まれるすべてのサーバーに対する読取り、書込みおよび実行アクセス権を持っています。

一般に、システム管理者は次のタスクを実行します:

- デフォルトの EPM System 管理者アカウントを無効にします。
- 機能の管理者を少なくとも 1 つ作成します。
- Shared Services Console を使用して EPM System 用のセキュリティ構成を設定します。
- オプションで、ユーザー・ディレクトリを外部ユーザー・ディレクトリとして構成します。
- ログ分析ツールを定期的に行って EPM System をモニターします。

機能の管理者が実行するタスクは、このガイドに記載されています。

機能の管理者を作成する手順:

- 企業ディレクトリを外部ユーザー・ディレクトリとして構成します。[ユーザー・ディレクトリの構成](#)を参照してください。
- ユーザーまたはグループに機能の管理者の作成に必要な役割をプロビジョニングします。*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*のユーザーとグループのプロビジョニングを参照してください。

機能の管理者には、次の役割がプロビジョニングされている必要があります:

- Shared Services の LCM 管理者の役割
- デプロイされている各 EPM System コンポーネントの管理者およびプロビジョニング・マネージャの役割

機能の管理者

機能の管理者は、EPM System の専門家である企業ユーザーです。通常、このユーザーは外部ユーザー・ディレクトリとして Shared Services に構成されている企業ディレクトリで定義されます。

機能の管理者は、他の機能の管理者の作成、委任された管理の設定、アプリケーションやアーティファクトの作成およびプロビジョニング、EPM System 監査の設定などの EPM System 管理タスクを実行します。機能の管理者が実行するタスクは、*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*に記載されています。

グループ

グループは、ユーザーまたは他のグループのコンテナです。Shared Services Console からネイティブ・ディレクトリ・グループを作成して、管理できます。すべての構成済ユーザー・ディレクトリからのグループは、Shared Services Console に表示されます。これらのグループをプロビジョニングして、Shared Services に登録された EPM System 製品の権限を許可できます。

Shared Services Console の起動

Oracle Hyperion Enterprise Performance Management Workspace のメニュー・オプションを使用して、Oracle Hyperion Shared Services Console にアクセスします。

Shared Services Console を起動するには:

1. 次に移動します:

```
http://web_server_name:port_number/workspace
```

URL の中で、*web_server_name* は Oracle Hyperion Foundation Services が使用する Web サーバーが実行されているコンピュータの名前を示し、*port_number* は Web サーバー・ポートを示します。たとえば、`http://myWebserver:19000/workspace` のようになります。

 **ノート:**

セキュアな環境の EPM Workspace にアクセスする場合、プロトコルとして `https` (`http` ではなく) を使用し、セキュアな Web サーバー・ポート番号を使用します。たとえば、`https://myserver:19043/workspace` のような URL を使用します。

2. 「アプリケーションの起動」をクリックします。

 **ノート:**

ポップアップ・ブロックが原因で EPM Workspace が開かない場合があります。

3. 「ログオン」で、ユーザー名とパスワードを入力します。

最初は、Shared Services Console へアクセスできる唯一のユーザーは、ユーザー名とパスワードがデプロイメント・プロセス中に指定された Oracle Enterprise Performance Management System 管理者です。

4. 「ログオン」をクリックします。
5. 「ナビゲート」、「管理」、「Shared Services Console」の順に選択します。

2

EPM System コンポーネントの SSL 使用可能化

次も参照:

- [前提条件](#)
- [情報ソース](#)
- [場所の参照](#)
- [EPM System 製品の SSL 使用可能化について](#)
- [サポートされている SSL シナリオ](#)
- [必須の証明書](#)
- [SSL オフローダでの SSL 停止](#)
- [EPM System の完全な SSL デプロイメント](#)
- [Web サーバーでの SSL の停止](#)
- [Essbase 11.1.2.4 用 SSL](#)
- [Essbase 21c 用 SSL](#)

前提条件

- デプロイメント・トポロジを判別し、SSL を使用して保護する通信リンクを識別したものとします。
- よく知られている証明機関(CA)または独自の CA から必要な証明書を取得しているか、自己署名証明書を作成しているものとします。[必須の証明書](#)を参照してください。
- SSL の概念および証明書のインポートなどの手順に精通しているものとします。
参照ドキュメントのリストは、[情報ソース](#)を参照してください。

情報ソース

Oracle Enterprise Performance Management System を SSL 使用可能にするには、SSL を使用して通信するアプリケーション・サーバー、Web サーバー、データベース、ユーザー・ディレクトリなどのコンポーネントを準備する必要があります。このドキュメントでは、これらのコンポーネントを SSL 使用可能にするタスクに精通していることを前提としています。

- **Oracle WebLogic Server:** *WebLogic Server の保護ガイド*の [SSL の構成](#)を参照してください。
- **Oracle HTTP Server:** 『Oracle HTTP Server 管理者ガイド』の次のトピックを参照してください:

- セキュリティの管理
- Oracle HTTP Server の SSL の使用可能化に関する説明
- **ユーザー・ディレクトリ:** ユーザー・ディレクトリ・ベンダーのドキュメントを参照してください。次のリンクが役に立ちます:
 - **Oracle Internet Directory:** [Oracle Internet Directory 管理者ガイド](#)を参照してください
 - **Sun Java System Directory Server:** *Sun Java System Directory Server 管理ガイド*の [Directory Server のセキュリティ](#)を参照してください
 - **Active Directory:** Microsoft のドキュメントを参照してください。
- **データベース:** データベース・ベンダーのドキュメントを参照してください。

場所の参照

このドキュメントでは、次のインストールおよびデプロイメントの場所を参照します:

- **MIDDLEWARE_HOME** は、Oracle WebLogic Server などのミドルウェア・コンポーネントの場所、およびオプションで 1 つ以上の **EPM_ORACLE_HOME** を参照します。**MIDDLEWARE_HOME** は、Oracle Enterprise Performance Management System 製品のインストール時に定義されます。デフォルトの **MIDDLEWARE_HOME** ディレクトリは、Oracle/Middleware です。
- **EPM_ORACLE_HOME** は、EPM System 製品をサポートするのに必要なファイルを含むインストール・ディレクトリを参照します。**EPM_ORACLE_HOME** は **MIDDLEWARE_HOME** 内にあります。デフォルトの **EPM_ORACLE_HOME** は **MIDDLEWARE_HOME/EPMSysstem11R1** で、たとえば、Oracle/Middleware/EPMSysstem11R1 となります。

EPM System 製品は、**EPM_ORACLE_HOME/products** ディレクトリ(たとえば、Oracle/Middleware/EPMSysstem11R1/products)にインストールされます。

さらに、一部の EPM System 製品の構成時には、**MIDDLEWARE_HOME/user_projects/epmsystem1** (Oracle/Middleware/user_projects/epmsystem1 など)にコンポーネントがデプロイされます。
- **EPM_ORACLE_INSTANCE** は、一部の製品によってコンポーネントがデプロイされる構成プロセス時に定義される場所を表します。**EPM_ORACLE_INSTANCE** のデフォルトの場所は、**MIDDLEWARE_HOME/user_projects/epmsystem1** (Oracle/Middleware/user_projects/epmsystem1 など)です。

EPM System 製品の SSL 使用可能化について

Oracle Enterprise Performance Management System デプロイメント・プロセスでは、自動的に Oracle EPM System 製品が SSL モードおよび非 SSL モードの両方で機能するようにデプロイされます。

 ノート:

- EPM System では、HTTP および JDBC を介した SSL のみをサポートしています。安全な通信のために、Thrift や ODBC などの他の標準はサポートしていません。
- SSLv3 プロトコルを攻撃する Poodle (Padding Oracle On Downgraded Legacy Encryption)の脆弱性から保護するには、サーバーおよび EPM System コンポーネントにアクセスするために使用されるブラウザで SSLv3 サポートを無効にする必要があります。SSLv3 サポートを無効にする詳細は、サーバーおよびブラウザのドキュメントを参照してください。
- SSL を構成した後で非 SSL モードを使用不可にすると、EPM System サーバーが起動しなくなる可能性があります。ドメイン内のすべての EPM System サーバーに対してセキュア・レプリケーションを使用可能にすると、非 SSL モードが使用不可であってもサーバーが起動するようになります。

EPM System の共通設定を指定する場合、デプロイメント内のすべてのサーバー間通信で SSL を使用可能にするかどうかを指定します。

デプロイメント・プロセス中に SSL 設定を選択しても、各自の環境が自動的に SSL 用に構成されることはありません。この操作では、Oracle Hyperion Shared Services レジストリに対して、この Shared Services レジストリを使用するすべての EPM System コンポーネントでサーバー間通信にセキュア・プロトコル(HTTPS)を使用する必要があることを示すフラグが設定されるのみです。各自の環境を SSL 使用可能にするには、追加の手順を実行する必要があります。このドキュメントでは、このような手順について説明します。

 ノート:

アプリケーションを再デプロイすると、SSL を使用可能にするために指定したカスタム・アプリケーション・サーバー設定と Web サーバー設定が削除されます。

 ノート:

Enterprise Performance Management System リリース 11.2.x では、リポジトリ作成ユーティリティ (RCU) で MS SQL Server の Secure Sockets Layer (SSL) がサポートされていません。

サポートされている SSL シナリオ

次の SSL シナリオがサポートされています:

- SSL オフローダでの SSL 停止。[SSL オフローダでの SSL 停止](#)を参照してください。
- 完全な SSL デプロイメント。[EPM System の完全な SSL デプロイメント](#)を参照してください。

必須の証明書

SSL 通信では、コンポーネント間の信頼の確立に証明書が使用されます。よく知られたサードパーティ CA からの証明書を使用して、本番環境の Oracle Enterprise Performance Management System を SSL 使用可能化することをお勧めします。

ノート:

EPM System では、1 つの SSL 証明書で複数のサブドメインをセキュアにできるワイルドカード証明書の使用をサポートしています。ワイルドカード証明書を使用すると、管理の時間とコストを削減できます。

ワイルドカード証明書を使用して通信を暗号化している場合、Oracle WebLogic Server でホスト名検証を使用不可にする必要があります。

EPM System コンポーネントのホストの各サーバーに次の証明書が必要です:

- ルート CA 証明書

ノート:

ルート証明書が Java キーストアにすでにインストールされた、よく知られたサードパーティ CA からの証明書を使用している場合、Java キーストアにルート CA 証明書をインストールする必要はありません。

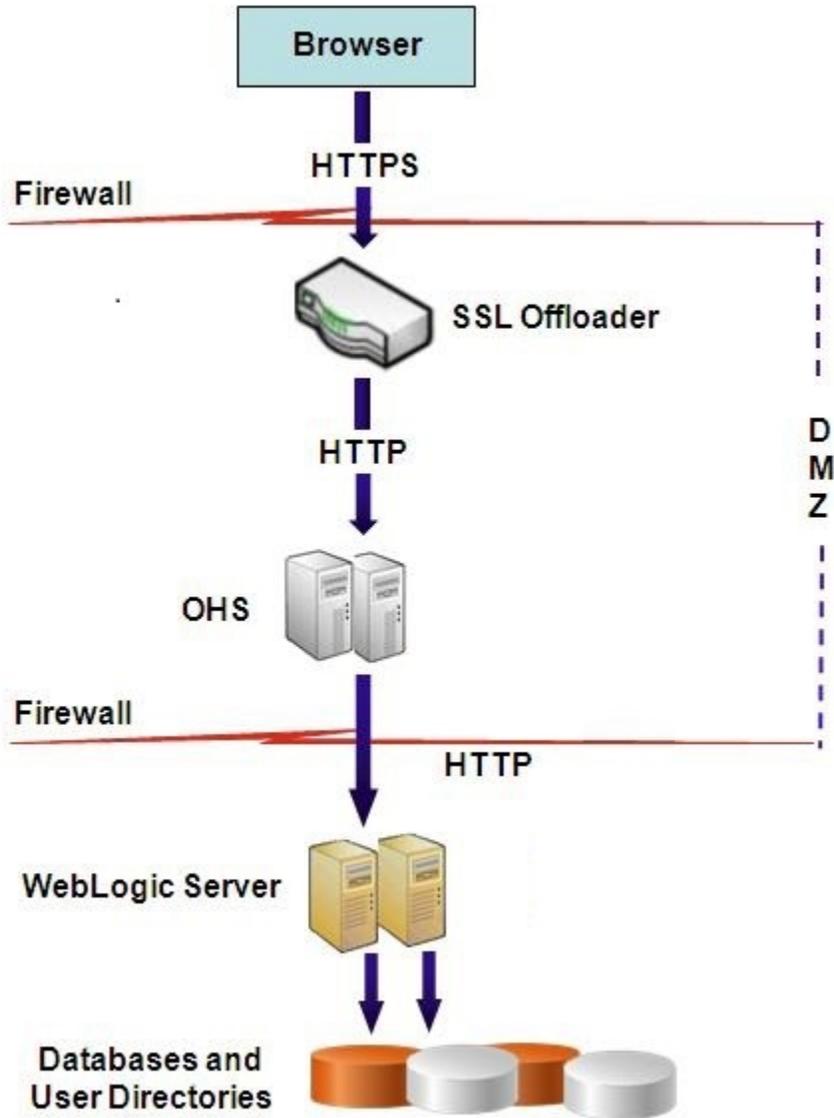
Firefox および Internet Explorer には、よく知られたサードパーティ CA の証明書があらかじめロードされています。CA として機能するには、これらのブラウザからアクセスされるクライアントで使用されるキーストアに CA ルート証明書をインポートする必要があります。CA としての役割を担う場合は、独自に発行した CA ルート証明書を対象のブラウザから取得できるようにしないと、そのブラウザからアクセスする Web クライアントでサーバーとの SSL ハンドシェイクを確立できません。

- デプロイメント内の各 Oracle HTTP Server の署名付き証明書
- WebLogic Server ホスト・マシンの署名付き証明書。このマシンの管理対象サーバーも、この証明書を使用できます
- SSL オフローダおよびロード・バランサの 2 つの証明書。これらの証明書の 1 つは外部通信用で、もう 1 つは内部通信用です

SSL オフローダでの SSL 停止

デプロイメント・アーキテクチャ

このシナリオでは、SSL を使用して、Oracle Enterprise Performance Management System クライアント(ブラウザなど)と SSL オフローダの間の通信リンクを保護します。概念を示した図:



前提条件

SSL オフローダおよびロード・バランサ

完全に構成された SSL オフローダがロード・バランサとともにデプロイメント環境に存在している必要があります。

ロード・バランサは、仮想ホストから受信したすべての要求を **Oracle HTTP Server** に転送するように構成されている必要があります。

仮想ホスト

SSL オフローダで停止する SSL の構成では、SSL オフローダ/ロード・バランサで 2 つのサーバー別名(たとえば、`epm.myCompany.com`、`empinternal.myCompany.com`)が使用され、1 つはオフローダとブラウザの間の外部通信用、もう 1 つは **EPM System** サーバー間の内部通信用です。サーバー別名がマシンの IP アドレスを示し、DNS を介して解決可能であることを確認してください。

オフローダとブラウザ間の外部通信(epm.myCompany.com を使用)をサポートする署名付き証明書が、オフローダ/ロード・バランサにインストールされている必要があります。

EPM System の構成

EPM System コンポーネントのデフォルト・デプロイメントは、SSL オフローダでの SSL 停止をサポートしています。追加のアクションは必要ありません。

EPM System を構成する際、Web アプリケーションの論理アドレスが、内部通信用に作成された別名(例: empinternal.myCompany.com)をポイントしていることを確認します。EPM System をインストールし、構成するには、次の情報ソースを参照してください:

- [Oracle Enterprise Performance Management System インストールおよび構成ガイド](#)
- [Oracle Enterprise Performance Management System インストール概要](#)
- [Oracle Enterprise Performance Management System インストレーションおよび構成トラブルシューティング・ガイド](#)

デプロイメントのテスト

配置プロセスが完了したら、セキュアな Oracle Hyperion Enterprise Performance Management Workspace の URL に接続して、すべてが機能していることを確認します:

```
https://virtual_host_external:SSL_PORT/workspace/index.jsp
```

たとえば、<https://epm.myCompany.com:443/workspace/index.jsp>(443 は SSL ポート)などです。

EPM System の完全な SSL デプロイメント

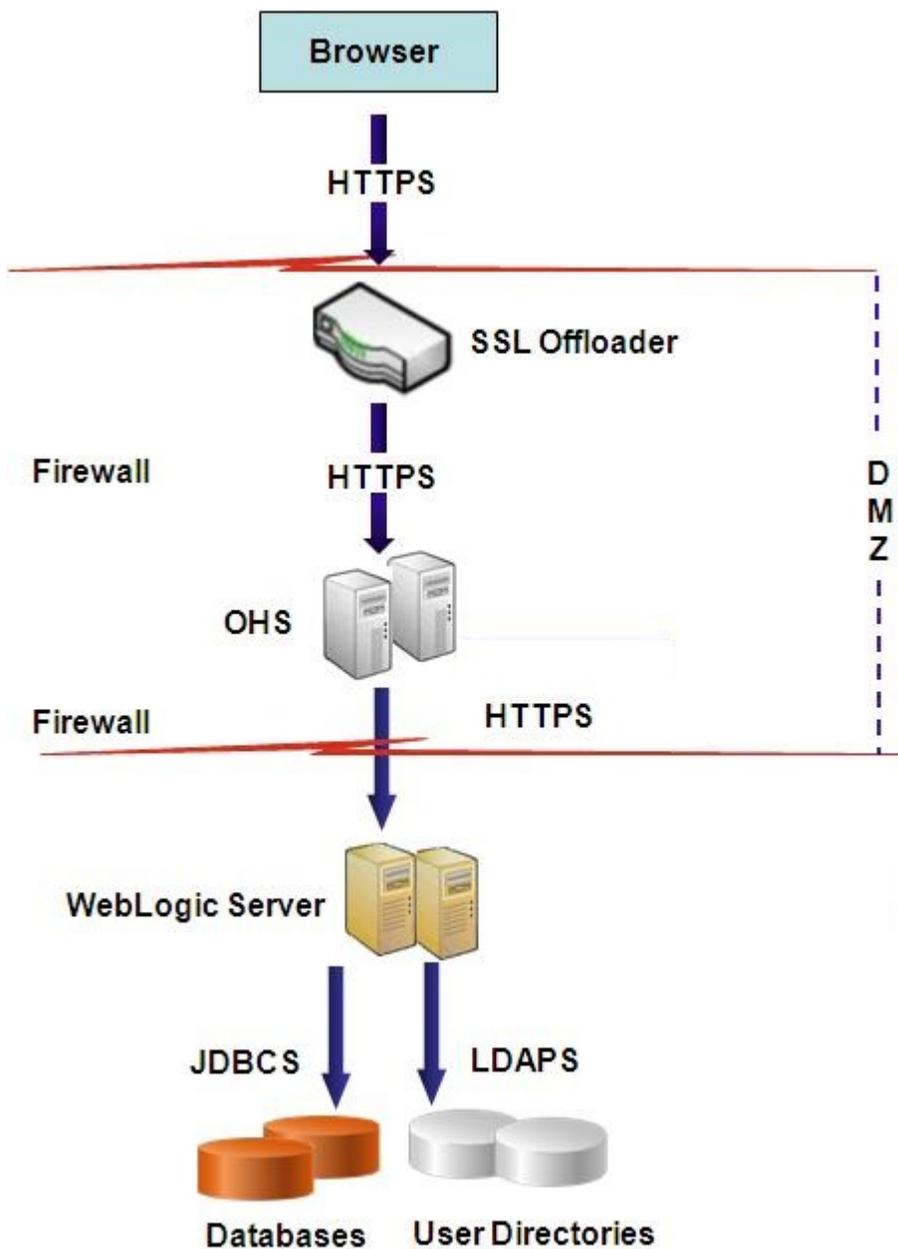
次も参照:

- [デプロイメント・アーキテクチャ](#)
- [前提条件](#)
- [完全 SSL 用 EPM System の構成](#)

デプロイメント・アーキテクチャ

完全 SSL モードでは、すべてのセキュリティ 保護可能なチャネル間の通信は、SSL を使用して保護されます。この Oracle Enterprise Performance Management System デプロイメント・シナリオが最もセキュアです。

概念を示した図:



前提条件

データベース

データベース・サーバーおよびクライアントが **SSL** 使用可能です。データベース・サーバーおよびクライアントの **SSL** 使用可能化の詳細は、データベースのドキュメントを参照してください。

EPM System

Oracle WebLogic Server や Oracle HTTP Server などの Oracle Enterprise Performance Management System コンポーネントがインストールされてデプロイされています。さらに、

EPM System 環境は、すべてが非 SSL モードで動作していることがテストによって確認されています。次の情報ソースを参照してください:

- [Oracle Enterprise Performance Management System インストールおよび構成ガイド](#)
- [Oracle Enterprise Performance Management System インストール概要](#)
- [Oracle Enterprise Performance Management System インストレーションおよび構成トラブルシューティング・ガイド](#)

データベースの接続を SSL 使用可能にする場合、構成プロセス中、各データベースの構成画面で「**詳細設定オプション**」リンクを選択し、次に示す必須の設定を指定する必要があります:

- 「**データベースに対して保護された接続を使用(SSL)**」を選択し、安全なデータベース URL を入力します。たとえば、
`jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS)(HOST=myDBhost)(PORT=1529)(CONNECT_DATA=(SERVICE_NAME=myDBhost.myCompany.com)))` です。
- **信頼できるキーストア**
- **信頼できるキーストア・パスワード**

詳細は、[Oracle Enterprise Performance Management System インストールおよび構成ガイド](#)を参照してください。

SSL オフローダおよびロード・バランサ

完全に構成された SSL オフローダがロード・バランサとともにデプロイメント環境に存在している必要があります。

完全 SSL 構成では、2 つのサーバー別名(`epm.myCompany.com` と `empinternal.myCompany.com` など)を SSL オフローダで使用します。1 つはオフローダとブラウザ間の外部通信用、もう 1 つは EPM System サーバー間の内部通信用です。サーバー別名がマシンの IP アドレスを示し、DNS を介して解決可能であることを確認してください。

ロード・バランサは、仮想ホストから受信したすべての要求を Oracle HTTP Server に転送するように構成されている必要があります。

2 つの署名付き証明書(1 つは、`epm.myCompany.com` を介したオフローダとブラウザ間の外部通信のサポート用、もう 1 つは、`empinternal.myCompany.com` を介したアプリケーション間の内部通信のサポート用)がオフローダ/ロード・バランサにインストールされている必要があります。サーバー名の公開を防ぎ、セキュリティを強化するために、これらの証明書はサーバー別名に結び付けることをお勧めします。

完全 SSL 用 EPM System の構成

次も参照:

- [EPM System の共通設定の再構成](#)
- [オプション: WebLogic Server に対するルート CA 証明書のインストール](#)
- [WebLogic Server に対する証明書のインストール](#)
- [WebLogic Server の構成](#)

- SSL 対応 Oracle Database との HFM サーバーの接続の有効化
- Oracle HTTP Server に関する手順
- WebLogic Server に配置された EPM System Web コンポーネントの構成
- ドメイン構成の更新
- サーバーおよび EPM System の再起動
- デプロイメントのテスト
- SSL 使用可能な外部ユーザー・ディレクトリの構成

EPM System の共通設定の再構成

このプロセスでは、Oracle Enterprise Performance Management System コンポーネントに強制的に SSL 通信を使用させるための設定を選択します。

ノート:

Oracle Hyperion Financial Management Web サーバーを SSL 使用可能にする場合、Financial Management の構成を開始する前に、weblogic.xml で HFM WebApp の session-descriptor を編集することによって、Cookie を保護する必要があります。

1. 7 Zip などのツールを使用して、Financial Management Web アーカイブを展開します。アーカイブ内の weblogic.xml の場所は、
`EPM_ORACLE_HOME\products\FinancialManagement\AppServer\InstallableApps\HFMWebApplication.ear\HFMWeb.war\WEB-INF\weblogic.xml` です。
2. weblogic.xml で HFM WebApp の session-descriptor に次のディレクティブを含めます:
`<cookie-secure>true</cookie-secure>`
3. weblogic.xml を保存します
4. 7 Zip によってアーカイブを更新するかどうか確認されたら、「Yes」をクリックします。

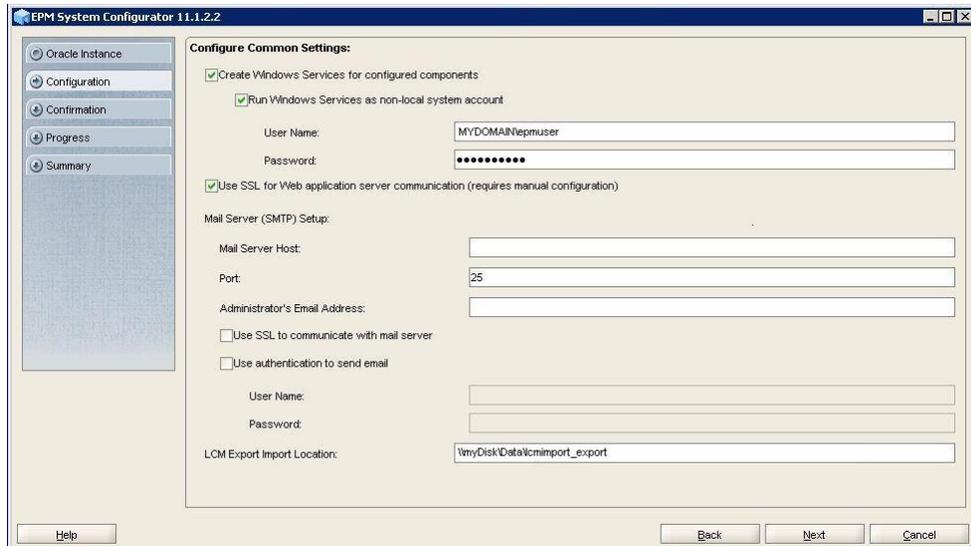
SSL 用に EPM System を再構成するには:

1. EPM System コンフィグレータを起動します。
2. 「構成を適用する EPM Oracle インスタンスを選択してください」で、次のステップを実行します:
 - a. 「EPM Oracle インスタンス名」で、EPM System コンポーネントの最初の構成時に使用したインスタンス名を入力します。
 - b. 「次」をクリックします。
3. 「構成」画面で、次のステップを実行します:
 - a. 「すべて選択解除」をクリアします。
 - b. Hyperion Foundation 構成タスクを展開し、「共通設定の構成」を選択します。
 - c. 「次」をクリックします。

4. 「共通設定の構成」で、次のステップを実行します:

▲ 注意:

電子メール・サーバーとの通信に SSL を使用するための設定を選択する前に、電子メール・サーバーが SSL 用に構成されていることを確認します。



- a. 「Java Web アプリケーション・サーバー通信に SSL を使用(手動構成が必要)」を選択し、EPM System が通信に SSL を使用する必要があることを指定します。
 - b. オプション: 「メール・サーバー・ホスト」および「ポート」に情報を入力します。SSL 通信をサポートするには、SMTP メール・サーバーで使用されるセキュアなポートを指定する必要があります。
 - c. オプション: SMTP メール・サーバーとの SSL 通信をサポートするには、「メール・サーバーとの通信に SSL を使用」を選択します。
 - d. 残りのフィールドに、設定を選択または入力します。
 - e. 「次」をクリックします。
5. 後続の「EPM System コンフィグレータ」画面で「次へ」をクリックします。
 6. デプロイメント・プロセスが完了すると、「要約」画面が表示されます。「終了」をクリックします。

オプション: WebLogic Server に対するルート CA 証明書のインストール

ほとんどのよく知られたサードパーティ CA のルート CA 証明書は、JVM キーストアにすでにインストールされています。よく知られたサードパーティ CA の証明書を使用

してインストールしていない場合、この項の手順を実行します(非推奨)。デフォルトの JVM キーストアの場所は、`MIDDLEWARE_HOME/jdk/jre/lib/security/cacerts` です。

 ノート:

各 Oracle Enterprise Performance Management System サーバーでこの手順を実行します。

ルート CA 証明書をインストールするには:

1. ルート CA 証明書を Oracle WebLogic Server がインストールされているマシンのローカル・ディレクトリにコピーします。
2. コンソールから、ディレクトリを `MIDDLEWARE_HOME/jdk/jre/bin` に変更します。
3. 次のような `keytool` コマンドを実行して、ルート CA 証明書を JVM キーストアにインストールします:

```
keytool -import -alias ALIAS -file CA_CERT_FILE -keystore KEYSTORE -storepass KEYSTORE_PASSWORD -trustcacerts
```

たとえば、次のコマンドを使用して現在のディレクトリに格納されている証明書 `CAcert.crt` を JVM キーストアに追加し、キーストア内の証明書別名として `Blister` を使用できます。ストア・パスワードは `example_pwd` としています。

```
keytool -import -alias Blister -file CAcert.crt -keystore ../lib/security/cacerts -storepass example_pwd -trustcacerts
```

 ノート:

前述のコマンドと例では、`keytool` を使用した証明書のインポートに構文の一部が使用されます。インポート構文の完全なリストは、`keytool` のドキュメントを参照してください。

WebLogic Server に対する証明書のインストール

デフォルトの Oracle WebLogic Server インストールでは、SSL をサポートするためにデモ用の証明書が使用されます。ご使用の環境を強化するために、よく知られたサードパーティが発行した証明書をインストールすることをお勧めします。

WebLogic Server をホストする各マシンで、`keytool` などのツールを使用して、WebLogic Server および Oracle Enterprise Performance Management System Web コンポーネントの署名付き証明書を保管するためのカスタム・キーストアを作成します。

カスタム・キーストアを作成して証明書をインポートするには:

1. コンソールから、ディレクトリを `MIDDLEWARE_HOME/jdk/jre/bin` に変更します。

2. 次のような **keytool** コマンドを実行して、既存のディレクトリで(コマンドの-keystore ディレクティブで識別される)カスタム・キーストアを作成します:

```
keytool -genkey -dname "cn=myserver, ou=EPM, o=myCompany, c=US" -
alias epm_ssl -keypass password -keystore
C:\oracle\Middleware\EPMSysstem11R1\ssl\keystore -storepass password
-validity 365 -keyalg RSA
```

ノート:

設定する共通名(cn)はサーバー名と一致する必要があります。cn に完全修飾ドメイン名(FQDN)を使用する場合、Web コンポーネントのデプロイの際に FQDN を使用する必要があります。

3. 証明書要求を生成します。

```
keytool -certreq -alias epm_ssl -file C:/certs/epmssl_csr -keypass
password -storetype jks -keystore
C:\oracle\Middleware\EPMSysstem11R1\ssl\keystore -storepass password
```

4. WebLogic Server マシンの署名付き証明書を取得します。
5. 署名付き証明書をキーストアにインポートします:

```
keytool -import -alias epm_ssl -file C:/certs/epmssl_cert -keypass
password -keystore C:\Oracle\Middleware\EPMSysstem11R1\ssl\keystore -
storepass password
```

WebLogic Server の構成

Oracle Enterprise Performance Management System Web コンポーネントをデプロイしたら、それらを SSL 通信用に構成する必要があります。

SSL 用に Web コンポーネントを構成するには:

1. `MIDDLEWARE_HOME/user_projects/domains/EPMSysstem/bin/startWebLogic.cmd` を実行して Oracle WebLogic Server を起動します:
2. 次の URL にアクセスして WebLogic Server 管理コンソールを起動します:

```
http://SERVER_NAME:Port/console
```

たとえば、myServer のデフォルト・ポートにデプロイされている WebLogic Server コンソールにアクセスするには、`http://myServer:7001/console` を使用する必要があります。

3. 「ようこそ」画面で、EPM System コンフィグレータで指定した WebLogic Server 管理者のユーザー名とパスワードを入力します。
4. 「チェンジ・センター」で、「ロックして編集」をクリックします。
5. コンソールの左側のペインで、「環境」を展開して、「サーバー」を選択します。

6. 「サーバーのサマリー」画面で、SSL 使用可能にするサーバーの名前をクリックします。
たとえば、Oracle Hyperion Foundation Services コンポーネントを SSL 使用可能にするには、EPMServer0 サーバーを使用します。
7. 「リスニング・ポートの有効化」をクリアして、HTTP リスニング・ポートを使用不可にします。
8. 「SSL リスニング・ポートの有効化」が選択されていることを確認します。
9. 「SSL リスニング・ポート」に、このサーバーが要求をリスニングする SSL リスニング・ポートを入力します。
10. 使用するアイデンティティと信頼キーストアを指定するには、「キーストア」を選択し、「キーストア」タブを開きます。
11. 「変更」をクリックします。
12. 次のいずれかのオプションを選択します:
 - よく知られたサードパーティ CA からのサーバー証明書を使用していない場合、「**カスタム・アイデンティティとカスタム信頼**」を選択します
 - よく知られたサードパーティ CA からのサーバー証明書を使用している場合、「**カスタム・アイデンティティと Java 標準信頼**」を選択します
13. 「保存」をクリックします。
14. 「カスタム・アイデンティティ・キーストア」で、署名付き WebLogic Server 証明書がインストールされているキーストアのパスを入力します。
15. 「カスタム・アイデンティティ・キーストアのタイプ」で、jks と入力します。
16. 「カスタム・アイデンティティ・キーストアのパスフレーズ」および「**カスタム・アイデンティティ・キーストアのパスフレーズを確認**」に、キーストアのパスワードを入力します。
17. 「キーストア」で「**カスタム ID とカスタム信頼**」を選択した場合、次を実行します:
 - 「**カスタム信頼キーストア**」で、サーバー証明書に署名した CA のルート証明書が使用できるカスタム・キーストアのパスを入力します。
 - 「**カスタム信頼キーストアのタイプ**」で、jks と入力します。
 - 「**カスタム信頼キーストアのパスフレーズ**」および「**カスタム信頼キーストアのパスフレーズを確認**」に、キーストアのパスワードを入力します。
18. 「保存」をクリックします。
19. SSL 設定を指定します:
 - 「**SSL**」を選択します。
 - 「**秘密キーの別名**」で、署名付き WebLogic Server 証明書のインポートの際に指定した別名を入力します。
 - 「**秘密鍵のパスフレーズ**」および「**秘密鍵のパスフレーズを確認**」に、秘密鍵の取得に使用するパスワードを入力します。
 - 「**保存**」をクリックします。

 **ノート:**

SHA-2 証明書を使用している場合、EPM System のサポートに使用される、それぞれの管理対象サーバーについて「**JSSE SSL の使用**」設定を選択する必要があります。この設定は、SSL ページの「詳細」タブにあります。この変更をアクティブにするには、WebLogic Server を再起動する必要があります。

20. サーバーに対してセキュア・レプリケーションを使用可能にします:
 - a. コンソールの左側のペインで、「**環境**」を展開し、「**クラスタ**」をクリックします。
 - b. 「クラスタのサマリー」で、セキュア・レプリケーションを使用可能にするサーバーの名前(Foundation Services など)をクリックします。
選択したサーバーの「設定」画面の「構成」タブが表示されます。
 - c. 「**レプリケーション**」をクリックして「レプリケーション」タブを開きます。
 - d. 「**セキュア・レプリケーションの有効化**」を選択します。このオプションを選択するには、事前に「**ロックして編集**」をクリックする必要がある場合があります。
 - e. 「**保存**」をクリックします。
21. このホストに属する各管理対象サーバーに対して、ステップ 6 からステップ 20 を実行します。
22. セキュアなレプリケーションを使用可能にして、クラスタのレプリケーション呼出し用のチャンネルを提供します。
詳細は、Oracle metalink のドキュメント 1319381.1 を参照してください。
 - 管理コンソールで、「**環境**」を展開し、「**クラスタ**」を選択します。
 - 「**レプリケーション**」を選択します。
 - 「**レプリケーション**」で、「**セキュア・レプリケーションの有効化**」を選択(チェック)します。
 - 「**保存**」をクリックします。
23. 「**チェンジ・センター**」で、「**変更のアクティブ化**」をクリックします。

SSL 対応 Oracle Database との HFM サーバーの接続の有効化

HFM データソースと Oracle データベースの間のネットワーク接続は、SSL を使用して暗号化できます。これを機能させるには、Oracle Wallet を、[Oracle ドキュメント](#)で説明されているとおりに構成する必要があります。また、SSL で暗号化された接続のための新しいポートをリスニングするように、TNS リスナーを構成する必要があります。さらに、HFM データソースをホストするサーバー上のキーストアおよびトラストストアに、適切な証明書をロードする必要があります。下の説明は、[Oracle Database ドキュメント](#)を参考にしています。

前提条件

下の手順に進む前に、次の前提条件を満たしていることを確認してください。

- データベース・サーバーが機能していること。
- ローカルまたはネットワーク・ファイアウォールによって、SSL 対応の TNS リスナーが実行されているポートでそのサーバーとの通信がブロックされていないことを確認してください。

下の例では、MS Windows Server 2016 上で Oracle 12c (12.1.0.2)バージョンが使用されています。これらの手順は、ウォレット・ファイル用に指定されているパスが Linux ファイルシステムのパスであり、環境変数の代替が、データベース・サーバーで使用するシェル向けに正しく変更されている場合、Linux インストールでも同様に機能します。これらと同じ手順を 19c の開発事例およびサポート事例で使用し成功しています。

この記事の例では自己署名付き証明書が使用されていますが、必要に応じて、適切な認証局証明書を使用することもできます。認証局によって発行された証明書のインストール時に従う必要がある正確な手順については、[Oracle Database ドキュメント](#)を参照してください。

Oracle Database の構成

Oracle Database を構成するには、次の手順を実行します。

1. データベース・サーバー上に新しい自動ログイン・ウォレットを作成します。

ノート:

これらの手順は、Oracle Wallet を以前に作成していない場合のみ必要です。次の手順は、データベース・サーバーで GUI の Oracle Wallet ツールを使用している場合は必要ありません。

```
C:\> cd %ORACLE_HOME%
C:\oracledb\12.1.0\home> mkdir wallet
C:\oracledb\12.1.0\home> orapki wallet create -wallet wallet -pwd
password1 -auto_login
```

orapki コマンドラインで `-auto_login_local` を使用するよう求めるメッセージが表示されても、無視してかまいません。SSL 認証失敗エラーが発生した場合は、[ドキュメント ID 2238096.1](#) を参照してこの問題をトラブルシューティングします。

また、ファイル `cwallet.sso` (ウォレット・ディレクトリの下)のセキュリティ権限を確認し、Oracle リスナー・サービス・ユーザーにこのファイルへの読取り権限があることを確認します。読取り権限がないと、後で SSL ハンドシェイクに失敗します。この状況は、ログインを許可されていない、Oracle ユーザー候補を使用して Oracle データベースをインストールしてある場合に発生します。Oracle データベースをその Oracle ユーザーでインストールしてある場合は、TNS リスナーを別のユーザーとして実行する必要があります。

2. 自己署名付き証明書を作成し、それをウォレットにロードします

```
C:\oracledb\12.1.0\home> orapki wallet add -wallet wallet -pwd password1 -
dn "CN={FQDN of db server}" -
keysize 1024 -self_signed -validity 3650
```

上の例でのパスワード `password1` が、手順1で指定したパスワードと一致する必要があります。

3. 新しく作成した自己署名付き証明書をエクスポートします

```
C:\oracledb\12.1.0\home> orapki wallet export -wallet wallet -pwd
password1 -dn "CN={FQDN of db server}"
-cert %COMPUTERNAME%-certificate.crt
```

4. エクスポートした Base64 証明書ファイルを HFM サーバーにコピーします。

5. SQL*NET および TNS リスナーを構成するには、次の手順を実行します。

a. データベース・サーバー上の使用されていないポートを特定します。下の例では、ポート 1522 で新しいリスナーが作成されます。SSL 接続に使用される一般的なポートは 2484 ですが、空いているどのポートでも使用できます。先に進む前に、データベース・サーバー上の使用したいポートが空いていることを確認し、必要に応じて調整してください。

b. SQLNET.ORA を更新します。WALLET_LOCATION 宣言の DIRECTORY 要素が、上の手順1で作成したウォレットを指している必要があります。

```
SQLNET.AUTHENTICATION_SERVICES= (TCPS, NTS, BEQ)
NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT)
WALLET_LOCATION=
(SOURCE =
(METHOD = FILE)
(METHOD_DATA =
(DIRECTORY = C:\oracledb\12.1.0\home\wallet)
)
)
SSL_CLIENT_AUTHENTICATION = FALSE
```

c. LISTENER.ORA を更新して新しいリスナーを定義します。上の手順5aで特定したポートを使用します。

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(SID_NAME = CLRExtProc)
(ORACLE_HOME = C:\oracledb\12.1.0\home)
(PROGRAM = extproc)
(ENVS =
"EXTPROC_DLLS=ONLY:C:\oracledb\12.1.0\home\bin\oraclr12.dll")
)
)
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION=
(SOURCE =
(METHOD = FILE)
(METHOD_DATA =
(DIRECTORY = C:\oracledb\12.1.0\home\wallet)
)
)
LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP) (HOST = myServer) (PORT = 1521))
```

```

)
(DESCRIPTION =
(AADDRESS = (PROTOCOL = TCPS)(HOST = myServer)(PORT = 1522))
)
(DESCRIPTION =
(AADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
)
)
)
ADR_BASE_LISTENER = C:\oracledb

```

- d. その新しいポート用に TNSNAMES.ORA 内に新しいエントリを作成します。

```

ORCL_SSL =
(DESCRIPTION =
(AADDRESS = (PROTOCOL = TCPS)(HOST = myServer)(PORT = 1522))
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = myServer_service)
)
)
)

```

上の手順 5a で特定し手順 5c で使用したのと同じポートを指定する必要があります。

- e. TNS リスナーを再起動します。

```

C:\oracledb\12.1.0\home>lsnrctl stop
C:\oracledb\12.1.0\home>lsnrctl start

```

- f. 新しい TNS リスナーが動作していることを確認します

```

C:\oracledb\12.1.0\home>tnsping orcl_ssl
TNS Ping Utility for 64-bit Windows: Version 12.1.0.2.0 - Production
on 10-SEP-2019 15:43:22
Copyright (c) 1997, 2014, Oracle. All rights reserved.
Used parameter files:
C:\oracledb\12.1.0\home\network\admin\sqlnet.ora
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS = (PROTOCOL = TCPS)
(HOST = myServer)
(PORT = 1522)) (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME =
myServer_service)))
OK (130 msec)

```

SSL データベース接続を使用するための HFM サーバーの構成

HFM サーバー上のトラストストアへのデータベースの証明書の追加

HFM データソースが実行されている各 EPM サーバーおよびすべての EPM サーバーで、次の手順を実行する必要があります。下で使用されている %MW_HOME% 環境変数は、Oracle Middleware インストールの場所です。この環境変数は、デフォルトでは EPM のインストール中に作成されず、ここでは EPM インストールの親ディレクトリを示すために使用されています。

EPM インストールの場所は、`EMP_ORACLE_HOME` 環境変数で指定します。下の例では、キーストアとトラストストアは、EPM インストールと同じ場所にあるディレクトリに配置されています。キーストア・ファイルとトラストストア・ファイルは、HFM サーバーのファイルシステム上の任意の場所に配置できます。

1. `%MW_HOME%`の下に、Java キーストアと PKCS12 トラストストアを保管するための新しいディレクトリを作成します。
 - a. `cd %MW_HOME%`
 - b. `mkdir certs`
2. JDK から Java キーストア・ファイル `cacerts` をコピーします。
 - a. `cd %MW_HOME%\certs`
 - b. `copy %MW_HOME%\jdk1.8.0_181\jre\lib\security\cacerts testing_cacerts`
 JDK のキーストアをコピーして、JDK のデフォルト・キーストアのかわりにそれを使用するのは、JDK がアップグレードされ、以前の JDK が削除された場合に、デフォルト・キーストアに挿入されたキーおよび証明書が失われるためです。
3. Base 64 証明書を `%MW_HOME%\certs` にコピーします。
4. その証明書を Java キーストア・ファイル `testing_cacerts` にインポートします。
 - a. たとえば、`keytool -importcert -file bur00cbb-certificate.crt -keystore testing_cacerts -alias "myserver"`のようになります。
 - i. キーストアのパスワードを指定する必要があります。
 - ii. "myserver"をデータベース・サーバーの完全修飾ドメインに置き換える必要があります。
 - b. その証明書が信頼できるものであるかを尋ねるプロンプトが表示されたら、**y** を指定します。
5. JDK の Java キーストア・ファイルから PKCS12 形式でトラストストアを作成します。次に例を示します。

```
keytool -importkeystore -srckeystore testing_cacerts -srcstoretype
JKS -deststoretype PKCS12 -destkeystore testing_cacerts.pfx
```

SSL を使用するための HFM の JDBC 接続の更新

1. SSL を使用するように、HFM データベースの JDBC 接続を再構成します。
 - a. EPM 構成ツールを起動します。
 - i. 「Financial Management」ノードの下で「データベースの構成」ノードと「アプリケーション・サーバーへのデプロイ」ノードを選択します。
 - ii. 「次」をクリックします。
 - iii. HFM の JDBC 接続について、次の各手順を実行します
 - i. 「ポート」、「サービス名」、「ユーザー名」および「パスワード」列で、その接続について、SSL ポート、サービス名、ユーザー名およびパスワードを入力します。
 - ii. (+)をクリックして詳細設定データベース・オプションを開きます。

- iii. 「セキュアな接続を使用」 チェックボックスを選択します。
 - iv. 手順2 で作成した Java キーストアの場所を入力します。
 - v. 「適用」 をクリックします。
 - vi. (+) をクリックして **詳細設定データベース・オプション** を開きます。
 - vii. 「変更された JDBC URL の編集および使用」 をクリックします。なお、表示される JDBC URL に変更を加える必要はありません。
 - viii. 「適用」 をクリックします。
 - ix. 「次」 をクリックします。
- b. EPM ドキュメントに記載されているとおりに、残りの手順を実行して HFM アプリケーションをデプロイします。
2. コマンド・ウィンドウまたはシェルを開いて、そのデータソースによって使用される ODBC 接続で SSL を有効にできるように、EPM レジストリを手動で更新します。次に示す各コマンドを実行します。

```
epmsys_registry.bat addproperty FINANCIAL_MANAGEMENT_PRODUCT/
DATABASE_CONN/@ODBC_TRUSTSTORE "C:
\Oracle\Middleware\certs\testing_cacerts.pfx"
epmsys_registry.bat addencryptedproperty FINANCIAL_MANAGEMENT_PRODUCT/
DATABASE_CONN
/@ODBC_TRUSTSTOREPASSWORD <truststorepassword>
epmsys_registry.bat addproperty FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN
/@ODBC_VALIDATESERVERCERTIFICATE false
```

上の例では、パス C:\Oracle\Middleware は、手順 1、2 および 3 での %MW_HOME% の値です。

プロパティ FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN/@ODBC_VALIDATESERVERCERTIFICATE は、自己署名付き証明書を使用している場合のみ **false** に設定する必要があります。FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN/@ODBC_TRUSTSTOREPASSWORD の値は、手順 2 でコピーした、元の Java キーストアのパスワードである必要があります。

HFM によって使用される TNS 名エントリの更新

TNSNAMES.ORA を編集して新しいエントリを作成し、古いエントリの名前を変更します。次の例では、HFM サーバー上の、必要な変更が適用された、更新後の TNSNAMES.ORA ファイルを示します。これらの変更を加えるのは、HFM によって HFMTNS という名前の TNS 名エントリが検索され使用されるためです。XFMDDataSource が正しく機能するように、このエントリのプロトコルおよびポートを変更しておく必要があります。

```
HFMTNS_UNENC =
(DESCRIPTION =
(AADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP) (HOST = myserver) (PORT = 1521))
)
(CONNECT_DATA =
(SERVICE_NAME = myserver_service)
```

```
(SERVER = DEDICATED)
)
)
HFMTNS =
(DESCRIPTION =
(AADDRESS_LIST =
(AADDRESS = (PROTOCOL = TCPS) (HOST = myserver) (PORT = 1522))
)
(CONNECT_DATA =
(SERVICE_NAME = myserver_service)
(SERVER = DEDICATED)
)
)
```

元の HFMTNS エントリの名前は、HFMTNS_UNENC に変更されています。HFMTNS_UNENC エントリをコピーし、その名前を HFMTNS に変更することで、新しい HFMTNS が作成されました。次に、プロトコルが TCPS に更新され、ポートが 1522 に変更されました。指定されたポートは、TNS LISTENER.ORA ファイル内で指定されているポートと同じである必要があります。

Oracle HTTP Server に関する手順

ウォレットの作成および Oracle HTTP Server の証明書のインストール

デフォルトのウォレットは、Oracle HTTP Server とともに自動的にインストールされます。デプロイメント内の各 Oracle HTTP Server に、実際のウォレットを構成する必要があります。

ノート: 11.2.x 以降では、Oracle Wallet Manager は Oracle HTTP Server とともにインストールされません。Oracle Wallet Manager は、Oracle Database Client をインストールした場合のみインストールされます。ウォレットの作成と証明書のインポートには、Database Client で入手可能なウォレット・マネージャを使用する必要があります。Oracle HTTP Server を SSL 用に構成している場合は、必ず、使用する EPM System 製品のインストールの一環として Oracle Database Client 64 ビットをインストールしてください。

Oracle HTTP Server の証明書を作成し、インストールするには:

1. Oracle HTTP Server のホストとなる各マシンで、Wallet Manager を起動します。
「スタート」、**「すべてのプログラム」**、**「Oracle-OHxxxxxx」**、**「統合管理ツール」**、**「Wallet Manager」** を選択します。
 xxxxxx は Oracle HTTP Server インスタンス番号です。
2. 新規で空のウォレットを作成します。
 - a. Oracle Wallet Manager で、**「ウォレット」**、**「新規」** を選択します。
 - b. **「はい」** をクリックしてデフォルトのウォレット・ディレクトリを作成するか、**「いいえ」** をクリックして選択した場所にウォレット・ファイルを作成します。
 - c. 「新規ウォレット」画面の **「ウォレット・パスワード」** および **「パスワードの確認」** に、使用するパスワードを入力します。
 - d. **「OK」** をクリックします。

- e. 確認のダイアログ・ボックスで、「いいえ」をクリックします。
3. **オプション:** Oracle HTTP Server にとって既知の CA を使用していない場合、ルート CA 証明書をウォレットにインポートします。
 - a. Oracle Wallet Manager で、「信頼できる証明書」を右クリックして「信頼できる証明書のインポート」を選択します。
 - b. ルート CA 証明書を参照して選択します。
 - c. 「オープン」を選択します。
4. 証明書要求を作成します。
 - a. Oracle Wallet Manager で、「証明書: [空]」を右クリックして「証明書リクエストの追加」を選択します。
 - b. 証明書要求の作成で、必要な情報を入力します。
共通名について、システムの hosts ファイルで使用可能な完全修飾サーバー別名(たとえば、epm.myCompany.com または epminternal.myCompany.com)を入力します。
 - c. 「OK」をクリックします。
 - d. 確認のダイアログ・ボックスで、「OK」をクリックします。
 - e. 作成した証明書要求を右クリックして、「証明書リクエストのエクスポート」を選択します。
 - f. 証明書要求ファイルの名前を指定します。
5. 証明書要求ファイルを使用して、署名付き証明書を CA から取得します。
6. 署名付き証明書をインポートします。
 - a. Oracle Wallet Manager で、署名付き証明書の取得に使用した証明書要求を右クリックし、「ユーザー証明書のインポート」を選択します。
 - b. 「証明書のインポート」で、「OK」をクリックして証明書をファイルからインポートします。
 - c. 「証明書のインポート」で、証明書ファイルを選択して「オープン」をクリックします。
7. ウォレットを便利な場所(`EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/keystores/epmsystem` など)に保存します。
8. 「ウォレット」、「自動ログイン」を選択して自動ログインをアクティブ化します。

ORAPKI を使用した Oracle Wallet の設定(Linux 上)

ORAPKI コマンドラインを使用して Oracle Wallet を設定するには、次のステップを実行します:

1. ウォレットのフォルダを作成します:

```
$ mkdir /MIDDLEWARE_HOME/oracle_common/wallet
```

2. orapki ユーティリティの場所をパスに追加します:

```
$ export PATH=$PATH:$MIDDLEWARE_HOME/oracle_common/bin
```

3. 証明書を保持するウォレットを作成します:

```
>$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet create -wallet
[wallet_location] -auto_login
```

コマンドラインにパスワードが指定されていない場合は、ウォレット・パスワードの入力および再入力が必要されます。-wallet に指定した場所にウォレットが作成されます。

4. 証明書署名要求(CSR)を生成し、ウォレットに追加します:

```
$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
[wallet_location] -dn 'CN=<CommonName>,OU=<OrganizationUnit>,
O=<Company>, L=<Location>, ST=<State>, C=<Country>' -keysize 512|
1024|2048|4096 -pwd [Wallet_Password]
```

5. ルートおよび中間証明書を信頼できるキーストアに追加します

```
$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
[wallet_location] -trusted_cert -cert [certificate_location] [-pwd]
```

6. CA (証明機関)を使用して、CSR (証明書署名要求)に署名します。Oracle Wallet から証明書要求をエクスポートするには:

```
$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet export -wallet
[wallet_location] -dn 'CN=<CommonName>,OU=<OrganizationUnit>,
O=<Company>, L=<Location>, ST=<State>, C=<Country>' -request
[certificate_request_filename] [-pwd]
```

7. 署名付き CSR をウォレットにインポートします:

```
$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
[wallet_location] -user_cert -cert [certificate_location] [-pwd]
```

8. ウォレットのコンテンツを表示するには:

```
$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet display -wallet
[wallet_location] [-pwd]
```

Oracle HTTP Server の SSL 使用可能化

Oracle HTTP Server をホストする各マシンで Web サーバーを再構成した後、作成したウォレットの場所で、デフォルト・ウォレットの場所を置き換え、Oracle HTTP Server 構成ファイルを更新します。

SSL 用に Oracle HTTP Server を構成するには:

1. デプロイメント内の各 Oracle HTTP Server ホスト・マシン上の Web サーバーを再構成します。
2. このインスタンスの EPM System コンフィグレータを開始します。
3. 構成タスクの選択画面で、次のステップを完了し、「次へ」をクリックします。

- a. **すべて選択解除**で、選択をクリアします。
- b. **Hyperion Foundation** タスク・グループを展開し、「**Web サーバーの構成**」を選択します。
4. 「**Web サーバーの構成**」で、「**次へ**」をクリックします。
5. 「**確認**」で、「**次へ**」をクリックします。
6. 「**要約**」で、「**終了**」をクリックします。
7. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig/ohs/config/fmwconfig/components/OHS/ohs_component/ssl.conf` を開きます。
8. 次に示すように、使用する SSL ポートが OHS Listen port の下にリストされていることを確認します。

SSL 通信ポートとして 19443 を使用する場合、エントリは次のようになります。

```
Listen 19443
```

9. SSLSessionCache パラメータ値を none に設定します。
10. デプロイメント内の各 Oracle HTTP Server の構成設定を更新します。
 - a. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig//ohs/config/fmwconfig/components/OHS/ohs_component/ssl.conf` を開きます。
 - b. SSLWallet ディレクティブを見つけ、その値を、証明書をインストールしたウォレットを示すように変更します。ウォレットを `EPM_ORACLE_INSTANCEhttpConfig/ohs/config/OHS/ohs_component/keystores/epmsystem` に作成した場合、SSLWallet ディレクティブは次のようになります:

```
SSLWallet "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/keystores/epmsystem"
```

- c. `ssl.conf` を保存して閉じます。
11. デプロイメント内の各 Oracle HTTP Server の `mod_wl_ohs.conf` を更新します。
 - a. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig//ohs/config/fmwconfig/components/OHS/ohs_component/mod_wl_ohs.conf` を開きます。
 - b. WLSSLWallet ディレクティブが、SSL 証明書が格納されている Oracle Wallet を示していることを確認します。

```
WLSSLWallet MIDDLEWARE_HOME/ohs/bin/wallets/myWallet
```

例: `C:/Oracle/Middleware/ohs/bin/wallets/myWallet`

- c. SecureProxy ディレクティブの値は ON に設定します。


```
SecureProxy ON
```
- d. デプロイされた Oracle Enterprise Performance Management System コンポーネントの LocationMatch 定義は、次の Oracle Hyperion Shared Services の例と同じよう

にします。この例では、SSL ポート 28443 を使用する myserver1 および myserver2 上の Oracle WebLogic Server クラスタを想定しています:

```
<LocationMatch /interop/>
  SetHandler weblogic-handler
  pathTrim /
  WeblogicCluster myServer1:28443,myServer2:28443
  WLProxySSL ON
</LocationMatch>
```

- e. mod_wl_ohs.conf を保存して閉じます。

WebLogic Server に配置された EPM System Web コンポーネントの構成

Oracle Enterprise Performance Management System Web コンポーネントをデプロイしたら、それらを SSL 通信用に構成する必要があります。

SSL 用に Web コンポーネントを構成するには:

1. `EPM_ORACLE_INSTANCE/domains/EPMSystem/bin/startWebLogic.cmd` に格納されているファイルを実行して Oracle WebLogic Server を起動します:
2. 次の URL にアクセスして WebLogic Server 管理コンソールを起動します:

`http://SERVER_NAME:Port/console`

たとえば、myServer のデフォルト・ポートにデプロイされている WebLogic Server コンソールにアクセスするには、`http://myServer:7001/console` を使用する必要があります。

3. 「ようこそ」画面で、ユーザー名とパスワードを入力して EPMSystem にアクセスします。ユーザー名とパスワードは構成プロセス中に EPM System コンフィグレータで指定されます。
4. 「**チェンジ・センター**」で、「**ロックして編集**」をクリックします。
5. コンソールの左側のペインで、「**環境**」を展開して、「**サーバー**」を選択します。
6. 「サーバーのサマリー」画面で、SSL 使用可能にするサーバーの名前をクリックします。

たとえば、すべての Oracle Hyperion Foundation Services コンポーネントをインストールした場合、これらのサーバーを SSL 使用可能にできます:

- CalcManager
- FoundationServices

7. 「**リスニング・ポートの有効化**」をクリアして、HTTP リスニング・ポートを使用不可にします。
8. 「**SSL リスニング・ポートの有効化**」が選択されていることを確認します。
9. 「**SSL リスニング・ポート**」に、WebLogic Server SSL リスニング・ポートを入力します。
10. 使用する ID と信頼キーストアを指定します。

- 「**キーストア**」を選択し、「キーストア」タブを開きます。
- 「**キーストア**」でオプションを選択します:
 - a. 「**キーストア**」を選択し、「キーストア」タブを開きます。
 - b. 「**キーストア**」でオプションを選択します:
 - よく知られたサードパーティ CA からのサーバー証明書を使用していない場合、「**カスタム・アイデンティティとカスタム信頼**」を選択します
 - よく知られたサードパーティ CA からのサーバー証明書を使用している場合、「**カスタム・アイデンティティと Java 標準信頼**」を選択します
 - c. 「**カスタム・アイデンティティ・キーストア**」で、署名付き WebLogic Server 証明書がインストールされているキーストアのパスを入力します。
 - d. 「**カスタム・アイデンティティ・キーストアのタイプ**」で、jks と入力します。
 - e. 「**カスタム・アイデンティティ・キーストアのパスフレーズ**」および「**カスタム・アイデンティティ・キーストアのパスフレーズを確認**」に、キーストアのパスワードを入力します。
 - f. 「**キーストア**」で「**カスタム ID とカスタム信頼**」を選択した場合、次を実行します:
 - 「**カスタム信頼キーストア**」で、サーバー証明書に署名した CA のルート証明書が使用できるカスタム・キーストアのパスを入力します。
 - 「**カスタム信頼キーストアのタイプ**」で、jks と入力します。
 - 「**カスタム信頼キーストアのパスフレーズ**」および「**カスタム信頼キーストアのパスフレーズを確認**」に、キーストアのパスワードを入力します。
 - g. 「**保存**」をクリックします。
- 11. SSL 設定を指定します。
 - 「**SSL**」を選択します。
 - 「**秘密キーの別名**」で、署名付き WebLogic Server 証明書のインポートの際に指定した別名を入力します。
 - 「**秘密鍵のパスフレーズ**」および「**秘密鍵のパスフレーズを確認**」に、秘密鍵の取得に使用するパスワードを入力します。
 - **Oracle Hyperion Provider Services Web アプリケーションのみ**: WebLogic Server と他の EPM System サーバー・コンポーネントの間の通信の暗号化にワイルドカード証明書を使用する場合は、Provider Services Web アプリケーションに対してホスト名検証を使用不可にします。
 - 「**詳細**」を選択します。
 - 「**ホスト名の検証**」で、「**なし**」を選択します。
 - 「**保存**」をクリックします。
- 12. 「**チェンジ・センター**」で、「**変更のアクティブ化**」をクリックします。

ドメイン構成の更新

このプロセスは、ドメイン構成を更新します。この手順を開始する前に、デプロイメントの完全バックアップを作成します。本番のデプロイメントを変更する前に、テスト・デプロイメントでこの手順をテストすることをお勧めします。

ドメイン構成を更新するには:

1. 次のように、MIDDLEWARE_HOME/oracle_common/bin directory ディレクトリに移動します。

```
cd MIDDLEWARE_HOME/oracle_common/bin
```

2. ORACLE_HOME、WL_HOME および JAVA_HOME を設定します。

```
set ORACLE_HOME= /Oracle/Middleware
set WL_HOME= /Oracle/Middleware/wlserver
set JAVA_HOME= /Oracle/Middleware/jdk
```

3. WebLogic コンソールで、管理サーバーの HTTP ポートを使用可能にします。

4. 次のようなコマンドを使用してキーストアを作成します:

```
libovdconfig.bat -host HOSTNAME -port 7001 -userName USERNAME -
domainPath %MWH%\user_projects\domains\EPMSysystem -createKeystore
```

このコマンドで、HOSTNAME と USERNAME をそれぞれ WebLogic サーバーのホスト名と管理者のユーザー名に置換します。出力で、OVD キーストアの正常な作成がレポートされていることを確認します。

5. AdminServer から SSL 証明書をエクスポートします。

Note:

この手順は、埋込み LDAP (デフォルトの認証プロバイダ)にのみ適用されます。その他の LDAP では、適切な LDAP 固有のコマンドを使用して証明書をエクスポートする必要があります。証明書ファイルの形式は、**Base 64 Encoded x.509** である必要があります

- a. Internet Explorer を使用して https://HOSTNAME:7002/console に接続することで、WebLogic 管理コンソールにアクセスします
- b. 「証明書の表示」、「詳細」の順にクリックし、**ファイルにコピー**を選択して SSL 証明書をエクスポートします。
- c. 証明書を **Base 64 Encoded x.509** 証明書ファイルとしてローカル・ディレクトリ(たとえば、C:\certificate\s1c17rby.cer など)に保存します。
- d. 証明書をサーバーに移動します。

6. keytool を使用して、ステップ 4 で作成したキーストアに証明書をインポートします。次のようなコマンドを使用します(JAVA_HOME (および keytool 実行ファイル)がパスにあると仮定しています):

```
export PATH=$JAVA_HOME/bin:$PATH
keytool -importcert -keystore
DOMAIN_HOME\config\fmwconfig\ovd\default\keystores/adapters.jks -
storepass PASSWORD -alias wcp_ssl -file CERTIFICATE_PATH -noprompt など、次に例を示します:
```

```
keytool -importcert -keystore %MWH%
\user_projects\domains\EPMSysystem\config\fmwconfig\ovd\default\keystore
s/adapters.jks -storepass examplePWD -alias wcp_ssl -file
C:\certificate\s1c17rby.cer -noprompt
```

 **Note:**

- このコマンドで使用するパスワードは、ステップ 4 でキーストアを生成する際に使用したパスワードと一致する必要があります。
- `CERTIFICATE_PATH` は、証明書の場所と名前です
- 別名は任意の別名にできます。

証明書が正常にインポートされると、Certificate was added to keystore という `keytool` のメッセージが表示されます。

7. WebLogic コンソールで、HTTP ポートに加えて管理サーバーの SSL ポートを使用可能にします。
8. Weblogic 管理サーバーおよび管理対象サーバーを再起動します。
9. 安全な接続を使用して Oracle Hyperion Enterprise Performance Management Workspace にログインし、すべてが機能していることを確認します。

サーバーおよび EPM System の再起動

デプロイメント内のすべてのサーバーを再起動してから、各サーバーで Oracle Enterprise Performance Management System を起動します。

デプロイメントのテスト

SSL デプロイメントが完了したら、すべてが機能していることを確認します。

デプロイメントをテストするには:

1. ブラウザを使用して、セキュアな Oracle Hyperion Enterprise Performance Management Workspace URL にアクセスします:

外部通信用のサーバー別名に `epm.myCompany.com` を、SSL ポートに 4443 を使用した場合、EPM Workspace の URL は次のようになります

```
https://epm.myCompany.com:4443/workspace/index.jsp
```

2. 「ログオン」画面で、ユーザー名とパスワードを入力します。
3. 「ログオン」をクリックします。
4. デプロイされた Oracle Enterprise Performance Management System コンポーネントにセキュアにアクセスできることを確認します。

SSL 使用可能な外部ユーザー・ディレクトリの構成

前提条件

- Oracle Hyperion Shared Services Console で構成する予定の外部ユーザー・ディレクトリが SSL 使用可能であるものとします。

- ユーザー・ディレクトリを SSL 使用可能にするために、よく知られたサードパーティ CA からの証明書を使用しなかった場合、サーバー証明書に署名した CA のルート証明書のコピーがあるものとします。

ルート CA 証明書のインポート

ユーザー・ディレクトリを SSL 使用可能にするために、よく知られたサードパーティ CA からの証明書を使用しなかった場合、サーバー証明書に署名した CA のルート証明書を次のキーストアにインポートする必要があります:

ノート:

アプリケーションのデプロイメント過程で、WebLogic により、DemoTrust.jks を指す `-Djavax.net.ssl.trustStore` ディレクティブが `setDomainEnv.sh` または `setDomainEnv.cmd` に追加されます。デフォルトの WebLogic 証明書を使用しない場合は、`setDomainEnv.sh` または `setDomainEnv.cmd` から `-Djavax.net.ssl.trustStore` を削除します。

keytool などのツールを使用して、ルート CA 証明書をインポートします。

- すべての Oracle Enterprise Performance Management System サーバー:
JVM キーストア: `MIDDLEWARE_HOME/jdk/jre/lib/security/cacerts`
- 各 EPM System コンポーネント・ホスト・マシンの JVM に使用されるキーストア。デフォルトでは、EPM System コンポーネントは次のキーストアを使用します:
`MIDDLEWARE_HOME/jdk/jre/lib/security/cacerts`

外部ユーザー・ディレクトリの構成

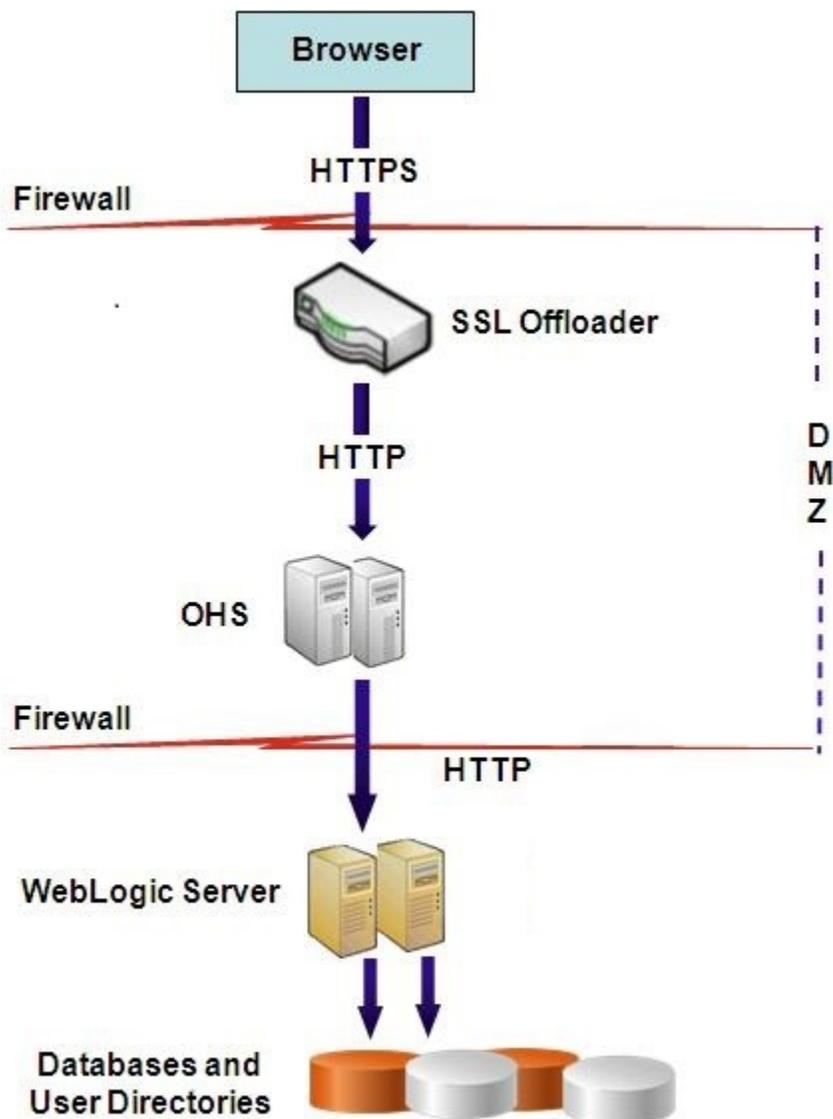
Shared Services Console を使用して、ユーザー・ディレクトリを構成します。ユーザー・ディレクトリの構成時、「SSL 使用可能」オプションを選択し、EPM System セキュリティでセキュア・プロトコルを使用してユーザー・ディレクトリと通信するよう指定する必要があります。EPM System セキュリティと LDAP 対応のユーザー・ディレクトリ(Oracle Internet Directory、Microsoft Active Directory など)の間の接続で SSL 使用可能にすることができます。

*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*のユーザー・ディレクトリの構成を参照してください。

Web サーバーでの SSL の停止

デプロイメント・アーキテクチャ

このシナリオでは、SSL を使用して、Oracle Enterprise Performance Management System クライアント(ブラウザなど)と Oracle HTTP Server の間の通信リンクを保護します。概念を示した図:



前提条件

この構成では、Web サーバー上で 2 つのサーバー別名(例: `epm.myCompany.com` と `empinternal.myCompany.com`)を使用します。1 つは Web サーバーとブラウザの間の外部通信用、もう 1 つは EPM System サーバー間の内部通信用です。サーバー別名がマシンの IP アドレスを示し、DNS を介して解決可能であることを確認してください。

ブラウザとの外部通信(例: `epm.myCompany.com` を使用)をサポートする署名付き証明書が、(セキュアな外部通信をサポートする仮想ホストが定義されている) Web サーバーにインストールされている必要があります。この仮想ホストは、SSL を終了し、Oracle HTTP Server に HTTP 要求を転送する必要があります。

EPM System の構成

EPM System コンポーネントのデフォルト・デプロイメントは、Web サーバーでの SSL 停止をサポートしています。追加のアクションは必要ありません。

EPM System を構成する際、論理 Web アプリケーションが、内部通信用に作成された仮想ホスト(例: empinternal.myCompany.com)をポイントしていることを確認します。EPM System をインストールし、構成するには、次の情報ソースを参照してください:

- *Oracle Enterprise Performance Management System* インストールおよび構成ガイド
- *Oracle Enterprise Performance Management System* インストール概要

デプロイメントのテスト

配置プロセスが完了したら、セキュアな Oracle Hyperion Enterprise Performance Management Workspace の URL に接続して、すべてが機能していることを確認します:

```
https://virtual_host_external:SSL_PORT/workspace/index.jsp
```

たとえば、`https://epm.myCompany.com:443/workspace/index.jsp`(443 は SSL ポート)などです。

Essbase 11.1.2.4 用 SSL

概要

この項では、Oracle Essbase インスタンスとコンポーネント (MaxL、Oracle Essbase Administration Services サーバー、Oracle Essbase Studio サーバー、Oracle Hyperion Provider Services、Oracle Hyperion Foundation Services、Oracle Hyperion Planning、Oracle Hyperion Financial Management および Oracle Hyperion Shared Services レジストリなど)の間の通信の保護に使用されるデフォルト証明書を置き換える手順を説明します。

デフォルトのデプロイメント

Essbase は、SSL モードおよび非 SSL モードで機能するようデプロイできます。Essbase エージェントは、セキュアでないポートでリスニングしますが、セキュアなポートでリスニングするよう構成することもできます。セキュアなポートにアクセスするすべての通信は SSL 接続として処理されます。クライアントが Essbase エージェントに非 SSL ポートで接続すると、接続は非 SSL 接続として処理されます。コンポーネントは、Essbase エージェントに対して非 SSL 接続と SSL 接続を同時に確立できます。

ログイン時にセキュアなプロトコルとポートを指定することで、セッションごとに SSL を制御できます。[セッションごとの SSL 接続の確立](#)を参照してください。

SSL が使用可能な場合、Essbase インスタンス内の通信はすべて暗号化され、データのセキュリティが確保されます。

セキュアなモードでの Essbase コンポーネントのデフォルトのデプロイメントでは、主にテストを目的とする場合は、自己署名の証明書を使用して SSL 通信を使用可能にします。本番環境で Essbase を SSL 使用可能にするには、よく知られたサードパーティ CA から発行された証明書を使用することをお勧めします。

コンポーネントの場合、ルート CA 証明書を Java キーストアに保管する必要があります。必要な証明書とその場所を次の表に示します。

 ノート:

ルート証明書が Oracle Wallet にすでにインストールされた、よく知られたサードパーティ CA からの証明書を使用している場合、ルート CA 証明書をインストールする必要はありません。

- Essbase サーバーと Essbase エージェント用の署名付き証明書。

表 2-1 必要な証明書とその場所

コンポーネント ¹	キーストア	証明書 ²
MaxL	Oracle Wallet	ルート CA 証明書
Administration Services サーバー	Oracle Wallet	ルート CA 証明書
Provider Services	Oracle Wallet	ルート CA 証明書
Oracle Enterprise Performance Management System データベース	Oracle Wallet	ルート CA 証明書
Essbase Studio サーバー	Java キーストア	ルート CA 証明書
Planning	<ul style="list-style-type: none"> • Oracle Wallet • Java キーストア 	ルート CA 証明書
Financial Management	Java キーストア	ルート CA 証明書
Essbase (サーバーおよびエージェント) ³	<ul style="list-style-type: none"> • Oracle Wallet • Java キーストア 	<ul style="list-style-type: none"> • ルート CA 証明書 • Essbase サーバーとエージェント用の署名付き証明書
Oracle Hyperion Shared Services リポジトリ		

¹ 同様のキーストアを使用する複数のコンポーネントのサポートには、1 つのキーストアのインスタンスのみ必要です。

² 複数のコンポーネントで、キーストアにインストールされているルート 証明書を 使用 でき ます。

³ 証明書を、デフォルトの Oracle Wallet および Java キーストアにインストールする必要があります。

Essbase コンポーネントのインストールとデプロイ

構成プロセスで、セキュアなエージェント・ポート(デフォルトは 6423)を選択できます。このポートは Oracle Essbase の構成時に変更できます。デフォルトでは、デプロイメント・プロセスで自己署名された必要な証明書がインストールされ、テスト用に機能上セキュアなデプロイメントが作成されます。

Oracle HTTP Server がインストールされている場合、EPM System インストーラで、Oracle Wallet と自己署名の証明書が Essbase インスタンスをホストするマシンの ARBOR_PATH 内にインストールされます。単一ホストのデプロイメントでは、この証明書がすべての Essbase コンポーネントで共有されます。

信頼できるサードパーティ CA 証明書の Essbase への使用

証明書要求の作成と証明書の取得

証明書要求を生成して、Oracle Essbase サーバーと Essbase エージェントをホストするサーバー用の証明書を取得します。証明書要求には、識別名(DN)に固有の暗号化された情報が含まれます。証明書要求を署名機関に送信して SSL 証明書を取得します。

keytool や Oracle Wallet Manager などのツールを使用して証明書要求を作成します。証明書要求の作成の詳細は、使用しているツールのドキュメントを参照してください。

keytool を使用する場合、次のようなコマンドを使用して証明書要求を作成します:

```
keytool -certreq -alias essbase_ssl -file C:/certs/essbase_server_csr -  
keypass password -storetype jks -keystore  
C:\oracle\Middleware\EPMSysstem11R1\Essbase_ssl\keystore -storepass password
```

ルート CA 証明書の取得とインストール

ルート CA 証明書により、SSL のサポートに使用される証明書の有効性が検証されます。これには、証明書を確認するために証明書の署名に使用された秘密キーを照合する対象の公開キーが含まれます。SSL 証明書に署名した証明機関からルート CA 証明書を取得できます。

Essbase サーバーまたはエージェントに接続するクライアントに、Essbase サーバー証明書に署名した CA のルート証明書をインストールします。ルート証明書は、必ずクライアントに適したキーストアにインストールします。[必要な証明書とその場所](#)を参照してください。

ノート:

複数のコンポーネントで、サーバー・マシンにインストールされているルート CA 証明書を使用できます。

Oracle Wallet

CA ルート証明書を Oracle Wallet で必要とするコンポーネントのリストは、[必要な証明書とその場所](#)を参照してください。ウォレットを作成するか、デフォルトの自己署名付き証明書がインストールされているデモ・ウォレットに証明書をインストールします。

ウォレットの作成とルート CA 証明書のインポートの詳細な手順については、Oracle Wallet Manager のドキュメントを参照してください。

Java キーストア

CA ルート証明書を Java キーストアで必要とするコンポーネントのリストは、[必要な証明書とその場所](#)を参照してください。デフォルトの自己署名付き証明書がインストールされているキーストアに証明書を追加するか、証明書を保管するためのキーストアを新たに作成します。

 ノート:

多くのよく知られたサードパーティ CA のルート CA 証明書は、Java キーストアにすでにインストールされています。

詳細な手順については、使用しているツールのドキュメントを参照してください。**keytool** を使用している場合、次のようなコマンドを使用してルート証明書をインポートします:

```
keytool -import -alias blister_CA -file c:/certs/CA.crt -keypass  
password -trustcacerts -keystore  
C:\Oracle\Middleware\EPMSysstem11R1\Essbase_ssl  
\keystore -storepass password
```

署名付き証明書のインストール

Essbase サーバーと Essbase エージェントをホストするサーバーに署名付き SSL 証明書をインストールします。Essbase RTC (C API) を使用して Essbase サーバーまたはエージェントとの接続を確立するコンポーネントの場合、証明書をルート CA 証明書とともに Oracle Wallet に保管する必要があります。JAPI を使用して Essbase サーバーまたはエージェントとの接続を確立するコンポーネントの場合、ルート CA 証明書と署名付き SSL 証明書を Java キーストアに保管する必要があります。詳細は、次の情報ソースを参照してください:

- Oracle Wallet Manager のドキュメント
- 証明書のインポートに使用するツール(keytool など)のドキュメントまたはオンライン・ヘルプ

keytool を使用する場合、次のようなコマンドを使用して証明書をインポートします:

```
keytool -import -alias essbase_ssl -file C:/certs/essbase_ssl.crt -  
keypass password -keystore  
C:\Oracle\Middleware\EPMSysstem11R1\Essbase_ssl\keystore -storepass  
password
```

Essbase サーバーのレジストリ値の更新

Windows

1. コマンド・プロンプトで、ディレクトリを `EPM_ORACLE_INSTANCE/epmsystem1/bin` に変更します。
2. 次のコマンドを実行して Windows レジストリを更新します。

```
epmsys_registry.bat updateproperty "#<Object ID>/@EnableSecureMode"  
true
```

```
epmsys_registry.bat updateproperty "#<Object ID>/@EnableClearMode"  
false
```

必ず<Object ID>を Essbase サーバーのコンポーネント ID に置き換えてください。これは、Essbase サーバー構成プロセスの完了後に生成されるレジストリ・レポート内にあります。

Linux

1. コンソールで、ディレクトリを `EPM_ORACLE_INSTANCE/epmsystem1/bin` に変更します。
2. 次のコマンドを実行してレジストリを更新します。

```
epmsys_registry.sh updateproperty "#<Object ID>/@EnableSecureMode" true
epmsys_registry.sh updateproperty "#<Object ID>/@EnableClearMode" false
```

必ず<Object ID>を **Essbase** サーバーのコンポーネント ID に置き換えてください。これは、**Essbase** サーバー構成プロセスの完了後に生成されるレジストリ・レポート内にあります。

Essbase の SSL 設定の更新

`essbase.cfg` 内で次の値を指定することで、**Essbase** サーバーおよびクライアントの SSL 設定をカスタマイズします。

- セキュア・モードを有効にする設定
- クリア・モードを有効にする設定
- クライアントとの通信で優先されるモード(クライアントでのみ使用)
- セキュアなポート
- 暗号スイート
- Oracle Wallet のパス

ノート:

`essbase.cfg` で、必須パラメータ(特に `EnableSecureMode`、`AgentSecurePort`)が欠落している場合は追加し、その値を設定します。

`essbase.cfg` を更新するには:

1. Oracle ウォレットを **Essbase** サーバーの証明書とともに `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/wallet` にコピーします。これは、**Essbase** サーバーに受け入れられる Oracle Wallet の唯一の場所です。
2. テキスト・エディタを使用して `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/essbase.cfg` を開きます。
3. 必要に応じて設定を入力します。デフォルトの **Essbase** 設定は暗黙的です。デフォルトの動作を変更する必要がある場合、`essbase.cfg` 内のカスタム動作の設定を追加します。たとえば、デフォルトで `EnableClearMode` が適用され、それにより、**Essbase** サーバーは、暗号化されていないチャンネルで通信することが有効化されます。**Essbase** サーバーの、暗号化されていないチャンネルで通信する機能をクリアするには、`essbase.cfg` で `EnableClearMode FALSE` を指定する必要があります。次の表を参照してください。

表 2-2 Essbase の SSL 設定

設定	説明 ¹
EnableClearMode ²	<p>Essbase アプリケーションと Essbase エージェントとの間で暗号化されていない通信を有効にします。このプロパティが FALSE に設定されている場合、Essbase は非 SSL 要求を処理できません。</p> <p>デフォルト: EnableClearMode TRUE</p> <p>例: EnableClearMode FALSE</p>
EnableSecureMode	<p>Essbase クライアントと Essbase エージェントとの間で SSL 暗号化通信を有効にします。SSL をサポートするには、このプロパティを TRUE に設定する必要があります。</p> <p>デフォルト: FALSE</p> <p>例: EnableSecureMode TRUE</p>
SSLCipherSuites	<p>SSL 通信で使用される暗号スイートの優先順のリスト。Essbase エージェントで、これらの暗号スイートの 1 つが SSL 通信に使用されます。エージェントが暗号スイートを選択する際、リスト内の最初の暗号スイートに最も高い優先度が適用されます。</p> <p>デフォルト: SSL_RSA_WITH_RC4_128_MD5</p> <p>例: SSLCipherSuites SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384</p>
APSPRESOLVER	<p>Oracle Hyperion Provider Services の URL。複数の Provider Services サーバーを使用している場合は、セミコロンを使用して各 URL を区切ります。</p> <p>例: APSPRESOLVER https://exampleAPShost1:PORT/aps;https://exampleAPShost2:PORT/aps</p>
AgentSecurePort	<p>エージェントがリスニングするセキュアなポート。</p> <p>デフォルト: 6423</p> <p>例: AgentSecurePort 16001</p>
WalletPath	<p>ルート CA 証明書と署名付き証明書を保管する Oracle Wallet の場所(1,024 文字未満)。</p> <p>デフォルト: ARBORPATH/bin/wallet</p> <p>例: WalletPath/usr/local/wallet</p>
ClientPreferredMode ³	<p>クライアント・セッションのモード(セキュアまたはクリア)。このプロパティが Secure に設定されている場合、SSL モードがすべてのセッションに使用されます。このプロパティが Clear に設定されている場合、クライアント・ログイン要求にセキュアなトランスポート・キーワードが含まれているかどうかに基づいてトランスポートが選択されます。セッションごとの SSL 接続の確立を参照してください。</p> <p>デフォルト: CLEAR</p> <p>例: ClientPreferredMode SECURE</p>

表 2-2 (続き) Essbase の SSL 設定

設定	説明 ¹
	¹ <code>essbase.cfg</code> 内にこれらのプロパティがない場合は、デフォルト値が強制適用されます。
	² <code>EnableClearMode</code> と <code>EnableSecureMode</code> を両方とも <code>FALSE</code> に設定すると、Essbase が動作しなくなります。
	³ クライアントはこの設定を使用して、Essbase でセキュアな通信を確立するかセキュアでない通信を確立するかを決定します。

4. `essbase.cfg` を保存して閉じます。

SSL を使用するための分散された Essbase ノードの更新

ノート:

この項の説明は、分散された Essbase デプロイメントにのみ適用されます。

ルート CA 証明書および署名付き証明書を含むウォレット・フォルダ(たとえば、`WalletPath/usr/local/wallet`)が、各分散ノード上の必要な場所にあることを確認してください。

1. ウォレット・フォルダを、各分散ノード内の次の場所にコピーします。
 - `EPM_ORACLE_HOME/common/EssbaseRTC/11.1.2.0/bin`
 - `EPM_ORACLE_HOME/common/EssbaseRTC-64/11.1.2.0/bin`
2. 各分散ノード内に次の場所が存在する場合は、これらの場所にウォレット・フォルダをコピーします。
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseServer/bin`
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseServer-32/bin`
 - `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin`
3. 各分散ノード上の次の場所に、`EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/essbase.cfg` をコピーします。
 - `EPM_ORACLE_HOME/common/EssbaseRTC/11.1.2.0/bin`
 - `EPM_ORACLE_HOME/common/EssbaseRTC-64/11.1.2.0/bin`
4. 各分散ノード上に次の場所が存在する場合は、これらの場所に `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/essbase.cfg` をコピーします。
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseServer/bin`
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseServer-32/bin`
 - `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin`
5. ウォレット・フォルダを、各分散ノード上の次の Essbase クライアント・インストール場所にコピーします。
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseClient/bin`

- `EPM_ORACLE_HOME/products/Essbase/EssbaseClient-32/bin`
6. 各分散ノード上の次の Essbase クライアント・インストール場所に、`EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/essbase.cfg` をコピーします。
- `EPM_ORACLE_HOME/products/Essbase/EssbaseClient/bin`
 - `EPM_ORACLE_HOME/products/Essbase/EssbaseClient-32/bin`

7. 次のプロパティを `essbase.properties` ファイルに追加します。

- `essbase.ssleverywhere=true`
- `olap.server.ssl.alwaysSecure=true`
- `APSRESOLVER=http[s]://host:httpsPort/aps`
必ずこの値を適切な URL に置き換えてください。

各分散ノード内に次の場所が存在する場合は、これらの場所にある `essbase.properties` ファイルを更新する必要があります。

- `EPM_ORACLE_HOME/common/EssbaseJavaAPI/11.2.0/bin/essbase.properties`
- `EPM_ORACLE_HOME/products/Essbase/aps/bin/essbase.properties`
- `EPM_ORACLE_INSTANCE/aps/bin/essbase.properties`

8. 各分散ノード上に `EPM_ORACLE_HOME/products/Essbase/eas` ディレクトリがある場合は、このディレクトリに `EPM_ORACLE_HOME/products/Essbase/aps/bin/essbase.properties` をコピーします。

9. **Oracle Hyperion Planning の場合のみ:** 次の 3 つのプロパティを `essbase.properties` ファイルに追加します。

- `essbase.ssleverywhere=true`
- `olap.server.ssl.alwaysSecure=true`
- `APSRESOLVER=APS_URL`
`APS_URL` を **Provider Services** の URL に置き換えます。複数の **Provider Services** サーバーを使用している場合は、セミコロンを使用して各 URL を区切ります。たとえば、`https://exampleAPShost1:PORT/aps;https://exampleAPShost2:PORT/aps` のようにします。

各分散ノードで、次の場所にある `essbase.properties` ファイルを更新する必要があります。

- `EPM_ORACLE_HOME/products/Planning/config/essbase.properties`
- `EPM_ORACLE_HOME/products/Planning/lib/essbase.properties`

10. **Oracle Hyperion Financial Reporting の場合のみ:** 次の 3 つのプロパティを `EPM_ORACLE_HOME/products/financialreporting/bin/EssbaseJAPI/bin/essbase.properties` ファイルに追加します。

- `essbase.ssleverywhere=true`
- `olap.server.ssl.alwaysSecure=true`
- `APSRESOLVER=APS_URL`
`APS_URL` を **Provider Services** の URL に置き換えます。複数の **Provider Services** サーバーを使用している場合は、セミコロンを使用して各 URL を区

切ります。たとえば、`https://exampleAPShost1:PORT/aps;https://exampleAPShost2:PORT/aps` のようにします。

ノート:

完全 SSL 環境では、**Financial Reporting** は接続を確立するために **Essbase** クラスタ名を必要とします。ホスト名が接続に使用されている場合、接続が失敗します。

11. a. 環境変数を設定します。

- **Windows:** `API_DISABLE_PEER_VERIFICATION` という名前の新しいシステム変数を作成し、その値を 1 に設定します。
- **Linux:** `setCustomParamsPlanning.sh` 内にディレクティブ `API_DISABLE_PEER_VERIFICATION=1` を追加します。

- b. `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/setEssbaseenv.bat` または `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/setEssbaseenv.sh` 内にディレクティブ `API_DISABLE_PEER_VERIFICATION=1` を追加します。

環境変数を設定します。

JAPI クライアント用の SSL プロパティのカスタマイズ

JAPI に依存する **Essbase** コンポーネントに対して、いくつかのデフォルト・プロパティが事前定義されます。 `essbase.properties` にプロパティを含めることによって、デフォルト・プロパティをオーバーライドできます。

ノート:

次の表で識別された SSL プロパティのうちいくつかのみが `essbase.properties` で外部化されます。外部化されないプロパティを追加する必要があります。

JAPI クライアントの SSL プロパティを更新するには:

1. テキスト・エディタを使用して、`EPM_ORACLE_HOME/common/EssbaseJavaAPI/11.1.2.0/bin/essbase.properties` を開きます。
2. 必要に応じてプロパティを更新します。カスタマイズ可能な JAPI クライアント・プロパティの説明は、次の表を参照してください。目的のプロパティが `essbase.properties` に含まれていない場合、そのプロパティを追加します。

表 2-3 JAPI クライアントのデフォルト SSL プロパティ

プロパティ	説明
<code>olap.server.ssl.alwaysSecure</code>	すべての Essbase インスタンスに対してクライアントで使用されるモードを設定します。このプロパティ値を true に設定すると、SSL モードが適用されます。 デフォルト: false
<code>olap.server.ssl.securityHandler</code>	プロトコルの処理用パッケージ名。この値を変更して別のハンドラを指定できます。 デフォルト: java.protocol.handler.pkgs
<code>olap.server.ssl.securityProvider</code>	Oracle では Sun SSL プロトコル実装が使用されます。この値を変更すると、別のプロバイダを指定できます。 デフォルト: com.sun.net.ssl.internal.www.protocol
<code>olap.server.ssl.supportedCiphers</code>	セキュアな通信用に有効化される追加の暗号のカンマ区切りリスト。Essbase でサポートされる暗号のみを指定する必要があります。 例: SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384
<code>olap.server.ssl.trustManagerClass</code>	署名の確認と証明書の有効期限のチェックによる SSL 証明書の検証に使用する TrustManager クラス。 デフォルトでは、すべての検証チェックを実施するにはこのプロパティは設定されません。 誤りチェックが実施されないようにするには、このパラメータの値を com.essbase.services.olap.security.EssDefaultTrustManager に設定します。これは、すべての検証チェックを成功とするデフォルトの TrustManager クラスです。 カスタム TrustManager を実装するには、javax.net.ssl.X509TrustManager インタフェースを実装する TrustManager クラスの完全修飾クラス名を指定します。 例: com.essbase.services.olap.security.EssDefaultTrustManager

3. `essbase.properties` を保存して閉じます。
4. すべての Essbase コンポーネントを再起動します。

セッションごとの SSL 接続の確立

MaxL などの Oracle Essbase コンポーネントでは、トランスポート・キーワードとして `secure` を使用して Essbase エージェントに接続すると、セッション・レベルで SSL

を制御できます。たとえば、次のいずれかのコマンドを **MaxL Console** から実行すると、**MaxL** と **Essbase** エージェントとの間にセキュアな接続を確立できます。

```
login admin example_password on hostA:PORT:secure
```

```
login admin example_password on hostA:secure
```

セッションごとの制御は、`essbase.cfg` に指定された構成設定より優先されます。トランスポート・キーワードが指定されていない場合、**Essbase** クライアントでは、`ClientPreferredMode` に設定された値を使用して、**Essbase** とセキュアな接続を開始するかどうかを決定します。`ClientPreferredMode` 設定が **secure** に設定されていない場合、通信は非セキュアなチャネルで行われます。

Essbase 21c 用 SSL

概要

この項では、**Oracle Essbase** インスタンスとコンポーネント (**MaxL**、**Oracle Essbase Administration Services** サーバー、**Oracle Hyperion Provider Services**、**Oracle Hyperion Foundation Services**、**Oracle Hyperion Planning**、**Oracle Hyperion Financial Management** および **Oracle Hyperion Shared Services** レジストリなど)の間の通信の保護に使用されるデフォルト証明書を置き換える手順を説明します。

ノート:

Essbase Administration Services (EAS) Lite は、**EPM** コンフィグレータを使用して構成された **HTTP** サーバー **SSL** ポート(たとえば **443**)を使用しません。`easconsole.jnlp` ファイルのセキュア **URL** は、非 **SSL** ポート(**80**)がデフォルトになります。

回避策: `easconsole.jnlp` で識別されるセキュア **URL** のデフォルトの非 **SSL** ポートを更新されたセキュア **URL** で置き換えます:

デフォルトのセキュア **URL**: `https://myserver:SECURE_PORT/easconsole/console.html`。たとえば、`https://myserver:80/easconsole/console.html` のようになります

更新されたセキュア **URL**: `https://myserver:SECURE_PORT/easconsole/console.html`。たとえば、`https://myserver:443/easconsole/console.html` のようになります

詳細は、**My Oracle Support (MOS)**の記事 - [ドキュメント ID 1926558.1 - SSL ポートが EAS Web コンソールの easconsole.jnlp に含まれない](#)を参照してください。

デフォルトのデプロイメント

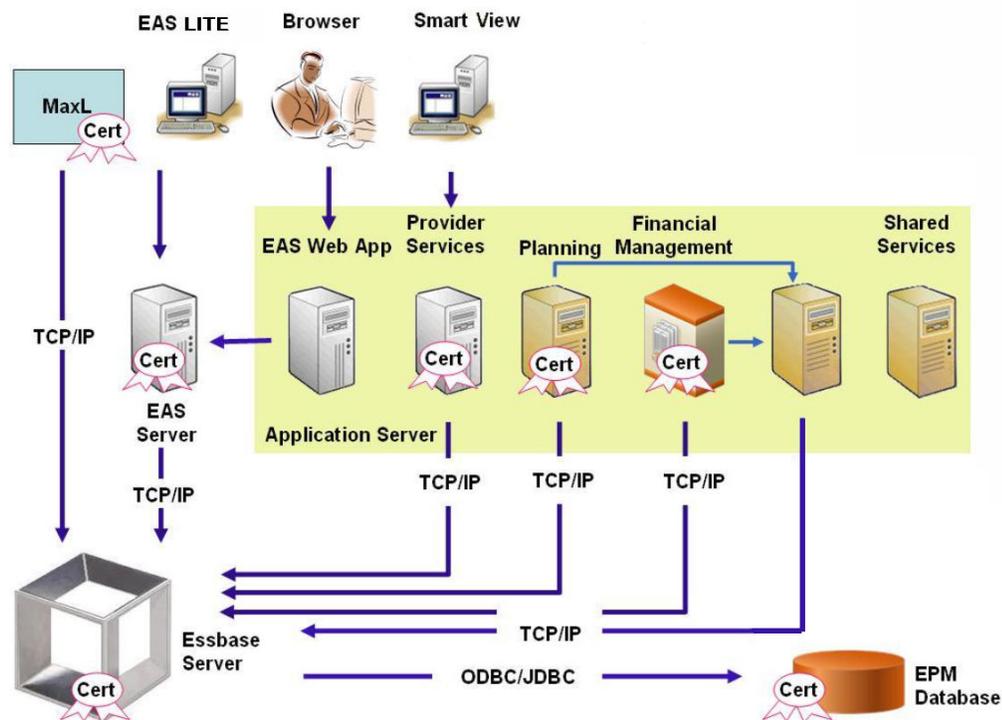
Essbase は、**SSL** モードおよび非 **SSL** モードで機能するようデプロイできます。**Essbase** エージェントは、セキュアでないポートでリスニングしますが、セキュアなポートでリスニングするよう構成することもできます。セキュアなポートにアクセスするすべての通信は **SSL**

接続として処理されます。クライアントが Essbase エージェントに非 SSL ポートで接続すると、接続は非 SSL 接続として処理されます。コンポーネントは、Essbase エージェントに対して非 SSL 接続と SSL 接続を同時に確立できます。

ログイン時にセキュアなプロトコルとポートを指定することで、セッションごとに SSL を制御できます。[セッションごとの SSL 接続の確立](#)を参照してください。

SSL が使用可能な場合、Essbase インスタンス内の通信はすべて暗号化され、データのセキュリティが確保されます。

セキュアなモードでの Essbase コンポーネントのデフォルトのデプロイメントでは、主にテストを目的とする場合は、自己署名の証明書を使用して SSL 通信を使用可能にします。本番環境で Essbase を SSL 使用可能にするには、よく知られたサードパーティ CA から発行された証明書を使用することをお勧めします。



通常、Oracle Wallet に、Essbase RTC を使用するクライアントとの SSL 通信を使用可能にする証明書が保管され、Java キーストアに、通信に JAPI を利用するコンポーネントとの SSL 通信を使用可能にする証明書が保管されます。SSL 通信を確立するために、Essbase クライアントとツールは、Essbase サーバーおよびエージェントの証明書に署名した CA のルート証明書を保管します。

必要な証明書とその場所

本番環境で Essbase を SSL 使用可能にするには、よく知られたサードパーティ CA から発行された証明書を使用することをお勧めします。デフォルトの自己署名付き証明書は、テスト目的で使用します。

 ノート:

Essbase では、1 つの SSL 証明書で複数のサブドメインをセキュアにできるワイルドカード証明書の使用をサポートしています。ワイルドカード証明書を使用すると、管理の時間とコストを削減できます。

ホスト名チェックが有効な場合、ワイルドカード証明書は使用できません。

次の証明書が必要です:

- ルート CA 証明書。
Essbase RTC を使用して Essbase との接続を確立するコンポーネントの場合、ルート CA 証明書を Oracle Wallet に保管する必要があります。JAPI を使用して接続を確立するコンポーネントの場合、ルート CA 証明書を Java キーストアに保管する必要があります。必要な証明書とその場所を次の表に示します。

 ノート:

ルート証明書が Oracle Wallet にすでにインストールされた、よく知られたサードパーティ CA からの証明書を使用している場合、ルート CA 証明書をインストールする必要はありません。

- Essbase サーバーと Essbase エージェント用の署名付き証明書。

表 2-4 必要な証明書とその場所

コンポーネント ¹	キーストア	証明書 ²
MaxL	Oracle Wallet	ルート CA 証明書
Administration Services サーバー	Oracle Wallet	ルート CA 証明書
Provider Services	Oracle Wallet	ルート CA 証明書
Oracle Enterprise Performance Management System データベース	Oracle Wallet	ルート CA 証明書
Planning	<ul style="list-style-type: none"> • Oracle Wallet • Java キーストア 	ルート CA 証明書
Financial Management	Java キーストア	ルート CA 証明書
Essbase (サーバーおよびエージェント) ³	<ul style="list-style-type: none"> • Oracle Wallet • Java キーストア 	<ul style="list-style-type: none"> • ルート CA 証明書 • Essbase サーバーとエージェント用の署名付き証明書

Oracle Hyperion Shared Services
リポジット

¹ 同様のキーストアを使用する複数のコンポーネントのサポートには、1 つのキーストアのインスタンスのみ必要です。

² 複数のコンポーネントで、キーストアにインストールされているルート証明書を使用できます。

³ 証明書を、デフォルトの Oracle Wallet および Java キーストアにインストールする必要があります。

Essbase コンポーネントのインストールとデプロイ

構成プロセスで、セキュアなエージェント・ポート(デフォルトは 6423)を選択できます。このポートは Oracle Essbase の構成時に変更できます。デフォルトでは、デプロイメント・プロセスで自己署名された必要な証明書がインストールされ、テスト用に機能上セキュアなデプロイメントが作成されます。

Oracle HTTP Server がインストールされている場合、EPM System インストーラで、Oracle Wallet と自己署名の証明書が Essbase インスタンスをホストするマシンの `ARBOR_PATH` 内にインストールされます。単一ホストのデプロイメントでは、この証明書がすべての Essbase コンポーネントで共有されます。

信頼できるサードパーティ CA 証明書の Essbase への使用

証明書要求の作成と証明書の取得

証明書要求を生成して、Oracle Essbase サーバーと Essbase エージェントをホストするサーバー用の証明書を取得します。証明書要求には、識別名(CN)に固有の暗号化された情報が含まれます。証明書要求を署名機関に送信して SSL 証明書を取得します。

keytool や Oracle Wallet Manager などのツールを使用して証明書要求を作成します。証明書要求の作成の詳細は、使用しているツールのドキュメントを参照してください。

keytool の使用例:

Java キーストア(JKS)を作成して、秘密キーを生成します:

```
keytool.exe -genkey -dname "cn=myserver, ou=EPM, o=Oracle, c=US"  
-alias essbase_ssl -keypass password -keystore  
C:\oracle\Middleware\EPMSysstem11R1\ssl\EPM.JKS -storepass password  
-validity 365 -keyalg RSA -keysize 2048 -sigalg SHA256withRSA -noprompt
```

証明書要求を生成します:

```
keytool -certreq -alias essbase_ssl -file  
C:\oracle\Middleware\EPMSysstem11R1\ssl\essabse_server.csr -keypass  
password  
-keystore C:\oracle\Middleware\EPMSysstem11R1\ssl\EPM.JKS -storepass  
password
```

秘密キーをエクスポートします(次のステップを実行するには、openssl ユーティリティが必要になります):

1. `openssl.exe pkcs12 -in C:\oracle\Middleware\EPMSysstem11R1\ssl\EPM.JKS -passin pass:password -legacy -nocerts -out c:\Apache24\ssl\Apache24.key -passout pass:password`
2. CA (認証局)を使用して新しく生成された証明書要求に署名し、それを次のファイルに貼り付けます:
`C:\oracle\Middleware\EPMSysstem11R1\ssl\essbase.cer。`

ルート CA 証明書の取得とインストール

ルート CA 証明書により、SSL のサポートに使用される証明書の有効性が検証されます。これには、証明書を確認するために証明書の署名に使用された秘密キーを照合する対象の公開キーが含まれます。SSL 証明書に署名した証明機関からルート CA 証明書を取得できます。

Essbase サーバーまたはエージェントに接続するクライアントに、Essbase サーバー証明書に署名した CA のルート証明書をインストールします。ルート証明書は、必ずクライアントに適したキーストアにインストールします。[必要な証明書とその場所](#)を参照してください。

ノート:

複数のコンポーネントで、サーバー・マシンにインストールされているルート CA 証明書を使用できます。

CA 署名付き証明書のインストール

CA 署名付き証明書のインストールについては、次のリンクを参照してください:

- [Essbase 用の Weblogic TLS 接続の設定](#)
- [TLS 証明書の更新](#)

次の下にある `tls.properties` ファイルを更新します

```
%EPM_HOME%\essbase\bin\tls_tools.properties:  
certCA=c:\\ssl\\ca.crt;c:\\ssl\\intermediate.crt;c:\\ssl\\essbase.key;c:\\  
ssl\\essbase.cer;
```

場所:

```
C:\\ssl\\ca.crt - root CA certificate.  
C:\\ssl\\intermediate.crt - intermediate CA certificate.  
C:\\ssl\\essbase.key - your private key generated in the previous step.  
C:\\ssl\\essbase.cer - your server's signed certificate issued by your CA.
```

次を実行して、新しい証明書で Essbase サーバーを更新します:

```
set ORACLE_HOME=c:\\OracleSSL  
set EPM_HOME=%ORACLE_HOME%  
set WL_HOME=%ORACLE_HOME%\\wlserver  
set JAVA_HOME=%ORACLE_HOME%\\jdk  
set DOMAIN_HOME=%ORACLE_HOME%\\user_projects\\domains\\essbase_domain  
%EPM_HOME%\essbase\bin\tls_tools.properties:  
%ORACLE_HOME%\\jdk\\bin\\java.exe -Xmx256m -jar %ORACLE_HOME%  
\\essbase\\lib\\tlsTools.jar %EPM_HOME%\essbase\bin\tls_tools.properties
```

Essbase の SSL 設定の更新

`essbase.cfg` 内で次の値を指定することで、Essbase サーバーおよびクライアントの SSL 設定をカスタマイズします。

- セキュア・モードを有効にする設定
- クリア・モードを有効にする設定
- クライアントとの通信で優先されるモード(クライアントでのみ使用)
- セキュアなポート
- 暗号スイート
- Oracle Wallet のパス

 **ノート:**

essbase.cfg で、必須パラメータ(特に EnableSecureMode、AgentSecurePort)が欠落している場合は追加し、その値を設定します。

次の下にある essbase.cfg を更新するには:

```
ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase
```

1. 必要に応じて設定を入力します。デフォルトの **Essbase** 設定は暗黙的です。デフォルトの動作を変更する必要がある場合、essbase.cfg 内のカスタム動作の設定を追加します。たとえば、デフォルトで EnableClearMode が適用され、それにより、**Essbase** サーバーは、暗号化されていないチャンネルで通信することが有効化されます。**Essbase** サーバーの、暗号化されていないチャンネルで通信する機能をクリアするには、essbase.cfg で EnableClearMode FALSE を指定する必要があります。次の表を参照してください:

表 2-5 Essbase の SSL 設定

設定	説明 ¹
EnableClearMode ²	Essbase アプリケーションと Essbase エージェントとの間で暗号化されていない通信を有効にします。このプロパティが FALSE に設定されている場合、Essbase は非 SSL 要求を処理できません。 デフォルト: EnableClearMode TRUE 例: EnableClearMode FALSE
EnableSecureMode	Essbase クライアントと Essbase エージェントとの間で SSL 暗号化通信を有効にします。SSL をサポートするには、このプロパティを TRUE に設定する必要があります。 デフォルト: FALSE 例: EnableSecureMode TRUE

表 2-5 (続き) Essbase の SSL 設定

設定	説明 ¹
SSLCipherSuites	SSL 通信で使用される暗号スイートの優先順のリスト。 Essbase エージェントで、これらの暗号スイートの 1 つが SSL 通信に使用されます。エージェントが暗号スイートを選択する際、リスト内の最初の暗号スイートに最も高い優先度が適用されます。 デフォルト: SSL_RSA_WITH_RC4_128_MD5 例: SSLCipherSuites SSL_RSA_WITH_AES_256_CBC_SHA256, SSL_RSA_WITH_AES_256_GCM_SHA384
APSPRESOLVER	Oracle Hyperion Provider Services の URL。複数の Provider Services サーバーを使用している場合は、セミコロンを使用して各 URL を区切ります。 例: https://exampleAPShost1:PORT/essbase;https://exampleAPShost2:PORT/essbase
AgentSecurePort	エージェントがリスニングするセキュアなポート。 デフォルト: 6423 例: AgentSecurePort 16001
WalletPath	ルート CA 証明書と署名付き証明書を保管する Oracle Wallet の場所(1,024 文字未満)。 デフォルト: ARBORPATH/bin/wallet 例: WalletPath/usr/local/wallet
ClientPreferredMode ³	クライアント・セッションのモード(セキュアまたはクリア)。このプロパティが Secure に設定されている場合、SSL モードがすべてのセッションに使用されます。このプロパティが Clear に設定されている場合、クライアント・ログイン要求にセキュアなトランスポート・キーワードが含まれているかどうかに基づいてトランスポートが選択されます。 セッションごとの SSL 接続の確立 を参照してください。 デフォルト: CLEAR 例: ClientPreferredMode SECURE

- ¹ essbase.cfg 内にこれらのプロパティがない場合は、デフォルト値が強制適用されます。
- ² EnableClearMode と EnableSecureMode を両方とも FALSE に設定すると、Essbase が動作しなくなります。
- ³ クライアントはこの設定を使用して、Essbase でセキュアな通信を確立するかセキュアでない通信を確立するかを決定します。

2. essbase.cfg を保存して閉じます。

SSL を使用するための分散された Essbase ノードの更新

ノート:

この項の説明は、分散された Essbase デプロイメントにのみ適用されます。

ルート CA 証明書および署名付き証明書を含むウォレット・フォルダ(たとえば、WalletPath/usr/local/wallet)が、各分散ノード上の必要な場所にあることを確認してください。

1. TLS ツールを使用して、新しい CA 証明書をすべてインポートします。

詳細は、次のリンクを参照してください:

- [Essbase 用の Weblogic TLS 接続の設定](#)
- [TLS 証明書の更新](#)

2. ソースの場所: ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase に移動して、essbase.properties ファイルの次のプロパティを変更します:

- `essbase.ssleverywhere=true`
- `olap.server.ssl.alwaysSecure=true`
- `APSPRESOLVER=APS_URL`
APS_URL を Provider Services URL で置き換えます。複数の Provider Services サーバーを使用している場合は、セミコロンを使用して各 URL を区切ります。

```
https://exampleAPShost1:PORT/essbase;https://exampleAPShost2:PORT/essbase.
```

3. Wallet フォルダ、Walletssl フォルダ、essbase.cfg ファイルおよび essbase.properties ファイルを次の宛先パスにコピーします。

表 2-6 宛先パス

宛先パス	Walle t	Walle tssl	essb ase.c fg	essbas e. properti es
EPM_ORACLE_HOME\common\EssbaseRTC-21C\11.1.2.0\bin	はい	はい	はい	はい
EPM_ORACLE_HOME\common\EssbaseJavaAPI-21C\11.1.2.0\bin	はい	はい	はい	はい
ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\aps	はい	はい	はい	はい
ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase	はい	はい	はい	はい
MIDDLEWARE_HOME\essbase\products\Essbase\template_files\essbase	はい	はい	はい	はい
MIDDLEWARE_HOME\essbase\products\Essbase\EssbaseServer\bin	はい	はい	はい	はい
MIDDLEWARE_HOME\essbase\products\Essbase\aps\bin	はい	はい	はい	はい
MIDDLEWARE_HOME\essbase\products\Essbase\ears	はい	はい	はい	はい

表 2-6 (続き) 宛先パス

宛先パス	Walle t	Walle tssl	essb ase.c fg	essbas e. properti es
MIDDLEWARE_HOME\essbase\common\EssbaseJavaAPI\bin	はい	はい	はい	はい
Oracle Hyperion Financial Reporting の場合のみ EPM_ORACLE_HOME/products/ financialreporting/bin/EssbaseJAPI/bin/ ノート: 完全 SSL 環境では、Financial Reporting は接続 を確立するために Essbase クラスタ名を必要とします。 ホスト名が接続に使用されている場合、接続が失敗しま す。	はい	はい	はい	はい
Oracle Hyperion Planning Only の場合のみ EPM_ORACLE_HOME/products/Planning/config/ EPM_ORACLE_HOME/products/Planning/lib/	はい	はい	はい	はい

4. 環境変数を設定します。

- **Windows:** API_DISABLE_PEER_VERIFICATION という名前の新しいシステム変数を作成し、その値を 1 に設定します。
- **Linux:** setCustomParamsPlanning.sh 内にディレクティブ
API_DISABLE_PEER_VERIFICATION=1 を追加します。

JAPI クライアント用の SSL プロパティのカスタマイズ

JAPI に依存する Essbase コンポーネントに対して、いくつかのデフォルト・プロパティが事前定義されます。essbase.properties にプロパティを含めることによって、デフォルト・プロパティをオーバーライドできます。

 **ノート:**

次の表で識別された SSL プロパティのうちいくつかのみが essbase.properties で外部化されます。外部化されないプロパティを追加する必要があります。

JAPI クライアントの SSL プロパティを更新するには:

1. テキスト・エディタを使用して、EPM_ORACLE_HOME/common/EssbaseJavaAPI-21C/11.2.0/bin/essbase.properties を開きます。
2. 必要に応じてプロパティを更新します。カスタマイズ可能な JAPI クライアント・プロパティの説明は、次の表を参照してください。
目的のプロパティが essbase.properties に含まれていない場合、そのプロパティを追加します。

表 2-7 JAPI クライアントのデフォルト SSL プロパティ

プロパティ	説明
<code>olap.server.ssl.alwaysSecure</code>	すべての Essbase インスタンスに対してクライアントで使用されるモードを設定します。このプロパティ値を <code>true</code> に設定すると、SSL モードが適用されます。 デフォルト: <code>false</code>
<code>olap.server.ssl.securityHandler</code>	プロトコルの処理用パッケージ名。この値を変更して別のハンドラを指定できます。 デフォルト: <code>java.protocol.handler.pkgs</code>
<code>olap.server.ssl.securityProvider</code>	Oracle では Sun SSL プロトコル実装が使用されます。この値を変更すると、別のプロバイダを指定できます。 デフォルト: <code>com.sun.net.ssl.internal.www.protocol</code>
<code>olap.server.ssl.supportedCiphers</code>	セキュアな通信用に有効化される追加の暗号のカンマ区切りリスト。Essbase でサポートされる暗号のみを指定する必要があります。 例: <code>SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384</code>
<code>olap.server.ssl.trustManagerClass</code>	署名の確認と証明書の有効期限のチェックによる SSL 証明書の検証に使用する TrustManager クラス。デフォルトでは、すべての検証チェックを実施するにはこのプロパティは設定されません。誤りチェックが実施されないようにするには、このパラメータの値を <code>com.essbase.services.olap.security.EssDefaultTrustManager</code> に設定します。これは、すべての検証チェックを成功とするデフォルトの TrustManager クラスです。 カスタム TrustManager を実装するには、 <code>javax.net.ssl.X509TrustManager</code> インタフェースを実装する TrustManager クラスの完全修飾クラス名を指定します。 例: <code>com.essbase.services.olap.security.EssDefaultTrustManager</code>

3. `essbase.properties` を保存して閉じます。
4. すべての Essbase コンポーネントを再起動します。

セッションごとの SSL 接続の確立

MaxL などの Oracle Essbase コンポーネントでは、トランスポート・キーワードとして `secure` を使用して Essbase エージェントに接続すると、セッション・レベルで SSL を制御できます。たとえば、次のいずれかのコマンドを MaxL Console から実行すると、MaxL と Essbase エージェントとの間にセキュアな接続を確立できます。

```
login admin example_password on hostA:PORT:secure
```

```
login admin example_password on hostA:secure
```

セッションごとの制御は、`essbase.cfg` に指定された構成設定より優先されます。トランスポート・キーワードが指定されていない場合、Essbase クライアントでは、`ClientPreferredMode` に設定された値を使用して、Essbase とセキュアな接続を開始するかどうかを決定します。`ClientPreferredMode` 設定が `secure` に設定されていない場合、通信は非セキュアなチャネルで行われます。

3

セキュリティ・エージェントでの SSO の使用可能

次も参照:

- [サポートされている SSO の方法](#)
- [Oracle Access Manager からのシングル・サインオン](#)
- [OracleAS シングル・サインオン](#)
- [SSO 用の EPM System 製品の保護](#)
- [アイデンティティ 管理製品を使用したヘッダーベース SSO](#)
- [Oracle Identity Cloud Services を使用したヘッダーベース SSO のための EPM System の構成](#)
- [SiteMinder SSO](#)
- [Kerberos シングル・サインオン](#)
- [SSO 用の EPM System の構成](#)
- [Smart View に対するシングル・サインオンのオプション](#)

サポートされている SSO の方法

SSO では、Web アイデンティティ 管理ソリューションを使用して、認証済ユーザーのログイン名を Oracle Enterprise Performance Management System 製品に渡す必要があります。次の標準的な EPM System の方法を使用して、EPM System と市販あるいはカスタムの Web ベースの SSO ソリューションを統合できます。

- [HTTP ヘッダー](#)
- [カスタム・ログイン・クラス](#)
- [HTTP 認証ヘッダー](#)
- [HTTP 要求からリモート・ユーザーを取得](#)
- [アイデンティティ 管理製品を使用したヘッダーベース認証](#)

▲ 注意:

ヘッダーをユーザー・アイデンティティの伝播手段とする方法を使用する場合は、セキュリティ対策として、Web サーバーとアプリケーション・サーバーの間にクライアント証明書認証(双方向 SSL)を実装することをお勧めします。

HTTP ヘッダー

Oracle シングル・サインオン(OSSO)、SiteMinder または Oracle Access Manager を Web アイデンティティ 管理ソリューションとして使用する場合、EPM System セキュリティでは自動的にカスタム HTTP ヘッダーを選択して、認証済ユーザーのログイン名を EPM System コンポーネントに渡します。

EPM System 製品ユーザーのログイン名は、Oracle Hyperion Shared Services でユーザー・ディレクトリを構成する際に指定されるログイン属性によって判別されます。ログイン属性の簡単な説明は、*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*の OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。

HTTP ヘッダーには、ログイン属性として設定される属性の値が含まれている必要があります。たとえば、uid がログイン属性値である場合、HTTP ヘッダーは、uid 属性の値を持っている必要があります。

カスタム HTTP ヘッダーの定義および発行の詳細は、Web アイデンティティ 管理ソリューションのドキュメントを参照してください。

EPM System セキュリティにより、HTTP ヘッダーが分析され、Shared Services で構成されたユーザー・ディレクトリに対して持っているログイン名が検証されます。

カスタム・ログイン・クラス

ユーザーがログインすると、Web アイデンティティ 管理ソリューションでは、ユーザーがディレクトリ・サーバーに対して認証され、SSO メカニズムで認証済ユーザーの資格証明がカプセル化されて、下流のシステムで SSO が使用可能になります。Web アイデンティティ 管理ソリューションで EPM System 製品によってサポートされないメカニズムが使用されるか、ログイン属性の値が SSO メカニズムで使用できない場合、カスタム・ログイン・クラスを使用してログイン属性の値を導出し、EPM System 製品に渡すことができます。

カスタム・ログイン・クラスを使用することによって、X509 証明書ベースの認証を使用するセキュリティ・エージェントと EPM System を統合できます。この認証メカニズムを使用するには、EPM System コンポーネント間の SSO インタフェースを定義するための標準 Shared Services API の実装と、Web アイデンティティ 管理ソリューションが必要です。カスタム・ログイン・クラスでは、ログイン属性の値が EPM System 製品に渡される必要があります。ログイン属性の簡単な説明は、*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*の OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。サンプル・コードと実装ステップは、[カスタム・ログイン・クラスの実装](#)を参照してください。

カスタム・ログイン・クラス(デフォルト名

`com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl`)を使用するには、`com.hyperion.css.CSSSecurityAgentIF` インタフェースの実装をこのクラスパスで使用できる必要があります。CSSSecurityAgentIF では、ユーザー名とパスワードを取得するゲッター・メソッドが定義されます(オプション)。インタフェースで `null` のパスワードが戻される場合、セキュリティ 認証ではプロバイダが信頼済として扱われ、構成済プロバイダにおけるユーザーの存在が確認されます。インタフェースでパスワードの `null` 以外の値が戻される場合、EPM System では、この実装により戻されるユーザー名とパスワードを使用して要求の認証が試みられます。

CSSSecurityAgentIF は、getUserName と getPassword の 2 つのメソッドから構成されています。

getUserName メソッド

このメソッドでは、認証用のユーザー名が戻されます。

```
java.lang.String getUserName (
    javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse res)
    throws java.lang.Exception
```

req パラメータでは、ユーザー名を判別するために使用される情報を持つ HTTP 要求が識別されます。res パラメータは使用されません(下位互換性にプリセット)。

getPassword メソッド

このメソッドでは、認証用のクリアテキストのパスワードが戻されます。パスワードの取得はオプションです。

```
java.lang.String getPassword (
    javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse res)
    throws java.lang.Exception
```

req パラメータでは、パスワードを判別するために使用される情報を持つ HTTP 要求が識別されます。res パラメータは使用されません(下位互換性にプリセット)。

HTTP 認証ヘッダー

EPM System セキュリティでは、HTTP 認証ヘッダーを使用して、ログイン属性の値を Web アイデンティティ管理ソリューションから EPM System 製品に渡すことができます。EPM System 製品は、認証ヘッダーを分析して、ユーザーのログイン名を取得します。

HTTP 要求からリモート・ユーザーを取得

EPM System セキュリティでは、HTTP 要求を使用して、ログイン属性の値を Web アイデンティティ管理ソリューションから EPM System 製品に渡すことができます。Web アイデンティティ管理ソリューションがログイン属性(setRemoteUser 関数を使用して設定される)の値を含む HTTP 要求を渡す場合、この SSO メソッドを使用します。

アイデンティティ管理製品を使用したヘッダーベース認証

EPM System は、Oracle Identity Cloud Services、Microsoft Azure AD、Okta など、ヘッダーベース認証をサポートするアイデンティティ管理製品をサポートします。概念的なワークフローは次のとおりです:

- リバース・プロキシとして機能するゲートウェイ・アプリケーションが、認証されていないネットワーク・アクセスを制限することで、EPM System コンポーネントを保護します。

- ゲートウェイ・アプリケーションが EPM System コンポーネントへの HTTP(S)要求をインターセプトし、要求を EPM System コンポーネントに転送する前に、アイデンティティ管理製品がユーザーを認証するようにします。
- 要求を EPM System コンポーネントに転送している間、ゲートウェイ・アプリケーションは、認証されたユーザーのアイデンティティを HTTP ヘッダー要求を介して EPM System コンポーネントに伝播します。

この認証シナリオをサポートするには、HTTP(S)ヘッダー要求を介して伝播される認証済ユーザーのアイデンティティで機能するように EPM System を構成する必要があります。

Oracle Access Manager からのシングル・サインオン

Oracle Enterprise Performance Management System は、ログイン属性値を含むカスタム HTTP ヘッダー(デフォルト名は HYPLOGIN)を受け入れることで Oracle Access Manager と統合されます。ログイン属性は、Oracle Hyperion Shared Services で外部ユーザー・ディレクトリを構成する際に設定されます。ログイン属性の簡単な説明は、*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*の OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。

EPM System にログイン属性を提供する任意のヘッダー名を使用できます。ヘッダー名は、Oracle Access Manager からの SSO 用に Shared Services を構成する際に使用します。

EPM System は、ログイン属性の値を使用して、構成済ユーザー・ディレクトリ(この場合は、Oracle Access Manager がユーザーの認証に使用するユーザー・ディレクトリ)に対してユーザーを認証し、EPM System 全体で SSO を使用可能にする EPM SSO トークンを生成します。ユーザーのプロビジョニング情報がネイティブ・ディレクトリで確認され、ユーザーに EPM System リソースに対する権限が与えられます。

ノート:

シック・クライアントである Oracle Essbase Administration Services コンソールは、Oracle Access Manager からの SSO をサポートしていません。

Oracle Access Manager の構成および HTTP ヘッダーやポリシー・ドメインの設定などのタスクの実行に関する情報は、Oracle Access Manager のドキュメントに記載されています。このガイドは、次のタスクが完了し、機能している Oracle Access Manager のデプロイメントを想定しています:

- EPM System コンポーネントに必要なポリシー・ドメインの設定
- ログイン属性値を EPM System に渡す HTTP ヘッダーの構成
- **保護するリソース**にリストされた EPM System リソースの保護。保護されたリソースへのアクセス要求は Oracle Access Manager によって処理されます。
- **保護しないリソース**にリストされた EPM System リソースの保護解除。保護されていないリソースへのアクセス要求は Oracle Access Manager によって処理されません。

Oracle Access Manager からの SSO 用に EPM System を構成するには:

1. Oracle Access Manager がユーザーの認証に使用するユーザー・ディレクトリを外部ユーザー・ディレクトリとして EPM System に追加します。Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイドの OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。

 **ノート:**

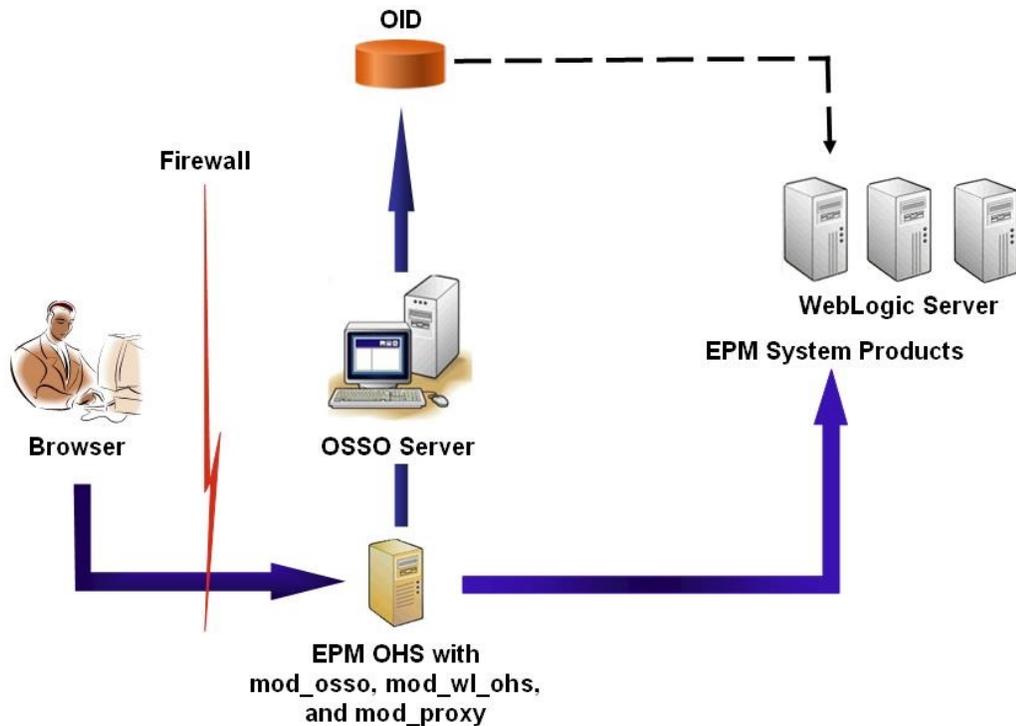
「接続情報」画面で、ユーザー・ディレクトリが信頼できる SSO ソースであることを示す「**信頼済**」チェック・ボックスが選択されていることを確認します。

2. EPM System に SSO を構成します。SSO 用の EPM System の構成を参照してください。
「SSO プロバイダ/エージェント」 リストから、「Oracle Access Manager」を選択します。Oracle Access Manager からの HTTP ヘッダーで HYPLOGIN 以外の名前を使用する場合、**「SSO メカニズム」** リストの隣にあるテキスト・ボックスにカスタム・ヘッダーの名前を入力します。
3. Oracle Data Relationship Management のみ:
 - a. Shared Services の認証に Data Relationship Management を構成します。
 - b. Data Relationship Management コンソールで SSO を使用可能にします。
 詳細は、Data Relationship Management のドキュメントを参照してください。

OracleAS シングル・サインオン

OracleAS Single Sign-on (OSSO) ソリューションでは、Oracle Internet Directory (OID) をユーザー・ディレクトリとして使用して、Web アプリケーションへの SSO アクセスを提供します。ユーザーは、OID で定義されたユーザー名とパスワードを使用して、Oracle Enterprise Performance Management System 製品にログインします。

プロセス・フロー



OSSO プロセス:

1. EPM System URL (`http://OSSO_OHS_Server_NAME:OSSO_OHS_Server_PORT/interop/index.jsp` など)を使用して、OSSO で保護されたアプリケーションとして定義される EPM System コンポーネントにアクセスします。
2. URL が OSSO で保護されているため、Oracle HTTP Server にデプロイされた `mod_osso` は要求をインターセプトします。`mod_osso` を使用して、Oracle HTTP Server は有効な cookie を確認します。有効な Cookie が要求で使用不可能な場合、Oracle HTTP Server は OSSO サーバーにユーザーをリダイレクトし、ユーザーはそこで資格証明を要求されて OID に対して認証されます。
3. OSSO サーバーは `obSSOCookie` を作成し、ブラウザに `obSSOCookie` を設定する Oracle HTTP Server 上の `mod_osso` モジュールに制御を返します。また、`mod_wl_ohs` (Oracle WebLogic Server) を介して、EPM System リソースに要求をリダイレクトします。EPM System リソースに要求を転送する前に、Oracle HTTP Server は、EPM System セキュリティで SSO を使用可能にするのに使用する `Proxy-Remote-User` ヘッダーを設定します。
4. EPM System コンポーネントは、`Proxy-Remote-User` から取得されたアイデンティティを持つユーザーが OID に存在することを確認します。このプロセスを機能させるには、OSSO サーバーを使用して構成される OID を Oracle Hyperion Shared Services の外部ユーザー・ディレクトリとして構成する必要があります。

前提条件

1. 完全な機能の Oracle Application Server インフラストラクチャ。

Oracle Application Server インフラストラクチャを確立するには、Oracle Identity Management Infrastructure 10.1.4 をインストールおよび構成します。OSSO が使用可能であることを確認します。Oracle Identity Management Infrastructure

10.1.4 のインストールには、OSSO をサポートするための次のコンポーネントが含まれています。

- Oracle 10g OSSO サーバー。
- OSSO サーバーで資格証明を検証するのに使用する OID。次のガイドを参照してください:
 - *Oracle Fusion Middleware Oracle Identity Management インストレーション・ガイド*
 - *Oracle Fusion Middleware Oracle Internet Directory 管理者ガイド*
- OSSO サーバーへのフロントエンドとしての Oracle HTTP Server。このインストールには、OSSO のパートナ・アプリケーションを定義できる `mod_osso` が含まれます。

ノート:

この Oracle HTTP Server インスタンスは、OSSO インフラストラクチャの一部です。EPM System コンポーネントの OSSO の構成に直接は使用されません。

インストール・プロセス中に、`mod_osso` がパートナ・アプリケーションとして OSSO サーバーに登録されていることを確認します。

2. 完全な機能の EPM System デプロイメント。
EPM System コンポーネントに Web サーバーを構成する場合、EPM System コンフィグレータは、Oracle HTTP Server で `mod_wl_ohs.conf` を構成して、WebLogic Server に要求をプロキシします。

デプロイメントのテスト

SSL デプロイメントが完了したら、すべてが機能していることを確認します。

デプロイメントをテストするには:

1. ブラウザを使用して、セキュアな Oracle Hyperion Enterprise Performance Management Workspace URL にアクセスします:

外部通信用のサーバー別名に `epm.myCompany.com` を、SSL ポートに 4443 を使用した場合、EPM Workspace の URL は次のようになります

```
https://epm.myCompany.com:4443/workspace/index.jsp
```

2. 「ログオン」画面で、ユーザー名とパスワードを入力します。
3. 「ログオン」をクリックします。
4. デプロイされた Oracle Enterprise Performance Management System コンポーネントにセキュアにアクセスできることを確認します。

EPM System 向けの OSSO の使用可能化

この項では、完全に構成された OSSO インフラストラクチャがあることを前提としています。『Oracle Application Server 管理者ガイド』を参照してください。

パートナ・アプリケーションとしての EPM System Web サーバーの登録

Oracle Identity Manager の SSO 登録ツール(ssoreg.sh または ssoreg.bat)を使用して、OSSO サーバーをフロントエンドする Oracle HTTP Server のパートナ・アプリケーションとして、Oracle Enterprise Performance Management System Web サーバーを登録します。

OSSO サーバーをフロントエンドする Oracle HTTP Server のホストとなるサーバー上で、この手順を実行します。このプロセスでは、選択した場所に難読化された osso.conf を生成および格納します。

パートナ・アプリケーションとして EPM System Web サーバーを登録するには:

1. OSSO サーバーをフロントエンドする Oracle HTTP Server のホストとなるサーバー上のコンソールを開き、Oracle HTTP Server の `ORACLE_HOME/sso/bin` ディレクトリ(Windows の場合、`C:\OraHome_1\sso/bin` など)に移動します。
2. `-remote_midtier` オプションで次のようなコマンドを実行します:

```
ssoreg.bat -site_name epm.myCompany.com
-mod_osso_url http://epm.myCompany.com:19400
-config_mod_osso TRUE
-update_mode CREATE
-remote_midtier
-config_file C:\OraHome_1\myFiles\osso.conf
```

次に、このコマンドで使用されるパラメータについて説明します。この説明では、パラメータ・アプリケーションは、EPM System Web サーバーとして使用される Oracle HTTP Server を参照します。

- `-site_name` は、パートナ・アプリケーションの Web サイト (epm.myCompany.com など)を識別します。
- `-mod_osso_url` は、パートナ・アプリケーションの URL を `PROTOCOL://HOST_NAME:PORT` 形式で示します。これは、EPM System Web サーバーが受信クライアント要求を受け入れる URL(http://epm.myCompany.com:19000 など)です。
- `-config_mod_osso` は、パートナ・アプリケーションで `mod_osso` を使用することを示します。config_mod_osso パラメータを含めて osso.conf を生成する必要があります。
- `-update_mode` は、更新モードを示します。デフォルトの CREATE を使用して、新規レコードを生成します。
- `-remote_midtier` は、mod_osso パートナ・アプリケーションが離れた中間層にあることを示します。パートナ・アプリケーションが OSSO サーバーとは異なる `ORACLE_HOME` にある場合に、このオプションを使用します。
- `-virtualhost` は、パートナ・アプリケーションの URL が仮想ホストであることを示します。仮想ホストを使用していない場合は、このパラメータを使用しないでください。
仮想ホストに結び付けられたパートナ・アプリケーションの URL を登録している場合、httpd.conf に仮想ホストを定義する必要があります。[オプション: 仮想ホストの定義](#)を参照してください。

- `-config_file` は、`osso.conf` ファイルを生成するパスを示します。

オプション: 仮想ホストの定義

パートナー・アプリケーションの登録時に仮想ホスト URL を使用する場合、EPM System Web サーバーとして使用される Oracle HTTP Server の `httpd.conf` を更新することによって、仮想ホストを定義する必要があります。

仮想ホストを定義するには:

1. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/httpd.conf` を開きます。
2. 次の記述に類似した定義を追加します。この定義は、Web サーバーが、仮想サーバー `epm.myCompany.com`、ポート `epm.myCompany.com:19400` で実行されていることを前提としています。各自の要件に合うように設定を変更してください。

```
NameVirtualHost epm.myCompany.com:19400
Listen 19400
    <VirtualHost epm.myCompany.com:19400>
DocumentRoot "C:/Oracle/Middleware/user_projects/epmsystem1/httpConfig/ohs
    /config/OHS/ohs_component/private-docs"
    include "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}
    /${COMPONENT_NAME}/mod_osso.conf"
    </VirtualHost>
```

mod_osso.conf の作成

EPM System Web サーバーをフロントエンドする Oracle HTTP Server 上に、`mod_osso.conf` を作成します。

`mod_osso.conf` を作成するには:

1. テキスト・エディタを使用して、ファイルを作成します。
2. 次のコンテンツをファイルにコピーして、使用する環境に合わせてファイルを変更します。

```
LoadModule osso_module C:/Oracle/Middleware/ohs/ohs/modules/mod_osso.so
<IfModule mod_osso.c>
    OsoIpCheck off
    OsoIdleTimeout off
    OsoSecureCookies off
    OsoConfigFile C:/Oracle/Middleware/user_projects/epmsystem1/
httpConfig/
    ohs/config/OHS/ohs_component/osso/osso.conf
```

3. `<IfModule mod_osso.c` 定義内に、次の記述に類似した場所定義を含めて、OSSO を使用して保護する予定の各リソースを識別します。

```
    <Location /interop/>
        require valid user
        AuthType Oso
    </Location>
</IfModule>
```

4. `mod_osso.conf` という名前でファイルを保存します。

osso.conf の再配置

EPM System Web サーバーをパートナ・アプリケーションとして登録するプロセス([パートナ・アプリケーションとしての EPM System Web サーバーの登録を参照](#))では、`-config_file` ディレクティブで識別された場所に、難読化された `osso.conf` ファイルを作成します。

`osso.conf` を再配置するには:

1. EPM System Web サーバーをパートナ・アプリケーションとして登録([パートナ・アプリケーションとしての EPM System Web サーバーの登録を参照](#))したときに作成された `osso.conf` を検索します。
2. `mod_osso.conf` ([mod_osso.conf の作成を参照](#))に定義された `OssosConfigFile` プロパティで識別されたディレクトリ(OSSO サーバーをフロントエンドする Oracle HTTP Server 上)に、`osso.conf` をコピーします。

OSSO 用 EPM System の構成

OSSO ソリューションと統合される OID を EPM System の外部ユーザー・ディレクトリとして構成し、SSO を使用可能にします。

OSSO 用 EPM System を構成するには:

1. OSSO ソリューションで使用する OID を外部ユーザー・ディレクトリとして構成します。*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*の OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。
2. EPM System で SSO を使用可能にします。[SSO 用の EPM System の構成](#)を参照してください。

ノート:

OSSO をアイデンティティ管理ソリューションとして構成するには、「**SSO プロバイダ/エージェント**」で「その他」を選択し、「**SSO メカニズム**」で「カスタム HTTP ヘッダー」を選択します。「Proxy-Remote-User」をカスタム HTTP ヘッダーの名前として入力します。

3. 少なくとも 1 つの OID ユーザーを Oracle Hyperion Shared Services 管理者としてプロビジョニングします。
4. EPM System 製品、および Shared Services セキュリティ API を使用するカスタム・アプリケーションを再起動します。

ノート:

Shared Services で構成済の OID が EPM System 製品を開始する前に必ず実行しているようにします。

オプション: OSSO サーバー上のデバッグ・メッセージの使用可能化

OSSO サーバー上のデバッグ・メッセージを記録するには、`policy.properties` を変更します。デバッグ・メッセージは `ORACLE_HOME/sso/log/ssoServer.log` に書き込まれます。

デバッグ・メッセージを記録するには:

1. テキスト・エディタを使用して、OSSO サーバー上の `ORACLE_HOME/sso/conf/policy.properties` (`C:\OraHome_1\sso\conf\policy.properties` など)を開きます。
2. `debugLevel` プロパティの値を `DEBUG` に設定します。

```
debugLevel = DEBUG
```

3. `policy.properties` を保存して閉じます。

オプション: 保護されたリソースのデバッグ・メッセージの使用可能化

`mod_osso.conf` を使用して保護されたリソースの OSSO デバッグ・メッセージを記録するには、EPM System Web サーバー上の `httpd.conf` を変更します。デバッグ・メッセージは、`EPM_ORACLE_INSTANCE/httpConfig/ohs/diagnostics/logs/OHS/ohs_component/ohs_component.log` に書き込まれます。

保護されたリソースのデバッグ・メッセージを記録するには:

1. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/httpd.conf` を開きます。
2. `OraLogSeverity` プロパティの値を `TRACE` に設定します。

```
OraLogSeverity TRACE:32
```

3. `httpd.conf` を保存して閉じます。

SSO 用の EPM System 製品の保護

ユーザーからの SSO 要求がセキュリティ・エージェント(OAM、OSSO または SiteMinder)にリダイレクトされるように、Oracle Enterprise Performance Management System リソースを保護する必要があります。

Oracle HTTP Server では、`mod_osso` を使用して OSSO サーバーにユーザーがリダイレクトされます。ユーザーは、要求する URL が保護される `mod_osso` で構成される場合にのみ、リダイレクトされます。*Oracle HTTP Server 管理者ガイド*の[セキュリティの管理](#)を参照してください。

SiteMinder SSO のリソース保護は、SiteMinder のドキュメントを参照してください。

OAM のみ: レスポンスに対するデフォルト・ヘッダーの追加の防止

デフォルトでは、OAM により 2 つのヘッダー(`Pragma: no-cache` および `Cache-Control: no-cache`)が保護 URL に追加されます。これらのヘッダーは EPM System および Web アプリケーションにより追加された類似のキャッシング・ディレクティブと競合するため、ブラウザではパフォーマンス低下の原因となる保護 URL のコンテンツのキャッシュを実行しない可能性があります。

これらの OAM ヘッダーがレスポンスに追加されるのを防止する手順は、*Fusion Middleware Oracle Access Manager with Oracle Security Token Service 管理者ガイド* の [Oracle Access Management のパフォーマンス・チューニング](#) の項の *OAM エージェントのチューニング* を参照してください。

保護するリソース

次の表に、保護する必要のあるコンテキストをリストします。OSSO 用にリソースを保護する構文(例として `interop` を使用)は、次のとおりです:

```
<Location /interop>
Require valid-user
AuthType Basic
order deny,allow
deny from all
allow from myServer.myCompany.com
satisfy any
</Location>
```

`allow from` パラメータでは、コンテキストの保護をバイパスできる開始サーバーを指定します。

Oracle Hyperion Enterprise Performance Management Workspace と Oracle Hyperion Financial Reporting については、次の例に示すパラメータのみを設定する必要があります:

```
<Location /workspace>
Require valid-user
AuthType Basic
</Location>
```

表 3-1 保護する EPM System リソース

EPM System 製品	保護するコンテキスト
Oracle Hyperion Shared Services	<ul style="list-style-type: none"> /interop /interop/.../*
EPM Workspace	<ul style="list-style-type: none"> /workspace /workspace/.../*
Financial Reporting	<ul style="list-style-type: none"> /hr /hr/.../*
Oracle Hyperion Planning	<ul style="list-style-type: none"> /HyperionPlanning /HyperionPlanning/.../*
Oracle Integrated Operational Planning	<ul style="list-style-type: none"> /interlace /interlace/.../*
Oracle Hyperion Financial Management	<ul style="list-style-type: none"> /hfmadf /hfmadfe/.../* /hfmoofficeprovider /hfmoofficeprovider/.../* /hfmsmartviewprovider /hfmsmartviewprovider/.../*

表 3-1 (続き) 保護する EPM System リソース

EPM System 製品	保護するコンテキスト
Oracle Hyperion Financial Reporting Web Studio	/frdesigner/**
Oracle Data Relationship Management	<ul style="list-style-type: none"> • /drm-web-client • /drm-web-client/.../*
Oracle Essbase Administration Services	<ul style="list-style-type: none"> • /hbrilauncher • /hbrilauncher/.../*
Oracle Hyperion Financial Data Quality Management	<ul style="list-style-type: none"> • /HyperionFDM • /HyperionFDM/.../*
Oracle Hyperion Calculation Manager	<ul style="list-style-type: none"> • /calcmgr • /calcmgr/.../*
Oracle Hyperion Provider Services	<ul style="list-style-type: none"> • /aps • /aps/.../*
Oracle Hyperion Profitability and Cost Management	<ul style="list-style-type: none"> • /profitability • /profitability/.../*
Account Reconciliation Manager	<ul style="list-style-type: none"> • /arm • /arm/.../*
Oracle Hyperion Financial Close Management	<ul style="list-style-type: none"> • /fcc • /fcc/.../*
Oracle Hyperion Financial Data Quality Management, Enterprise Edition	<ul style="list-style-type: none"> • /aif • /aif/.../*
Oracle Hyperion Tax Governance Tax Operations	/tss /taxop
Oracle Hyperion Tax Provision Supplemental Data Manager	/taxprov <ul style="list-style-type: none"> • /sdm* • /sdm/** • /sdm/./** • /SDM-Datamodel-context-root/**

保護しないリソース

次の表に、保護しないコンテキストをリストします。OSSO用にリソースを保護しない構文(例として/interop/framework(.*)を使用)は、次のとおりです。

```
<LocationMatch /interop/framework(.*)>
  Require valid-user
  AuthType Basic
  allow from all
  satisfy any
</LocationMatch>
```

表 3-2 保護しない EPM System リソース

EPM System 製品	保護しないコンテキスト
Shared Services	<ul style="list-style-type: none"> • /interop/framework • /interop/framework* • /interop/framework.* • /interop/framework/.../* • /interop/Audit • /interop/Audit* • /interop/Audit.* • /interop/Audit/.../* • /interop/taskflow • /interop/taskflow* • /interop/taskflow/.../* • /interop/WorkflowEngine • /interop/WorkflowEngine/* • /interop/WorkflowEngine/.../* • /interop/TaskReceiver • /framework/lcm/HSSMigration
EPM Workspace	<ul style="list-style-type: none"> • /epmstatic/.../* • /workspace/bpmstatic/.../* • /workspace/static/.../* • /workspace/cache/.../*
Planning	<ul style="list-style-type: none"> • /HyperionPlanning/Smartview • /HyperionPlanning/faces/PlanningCentral • /HyperionPlanning/servlet/HspDataTransfer • /HyperionPlanning/servlet/HspLCMServlet • /HyperionPlanning/servlet/HspADMServlet/.../* • /HyperionPlanning/servlet/HspADMServlet/** • /HyperionPlanning/servlet/HspADMServlet* • /HyperionPlanning/servlet/HspAppManagerServlet/.../* • /HyperionPlanning/servlet/HspAppManagerServlet/** • /HyperionPlanning/servlet/HspAppManagerServlet*

表 3-2 (続き) 保護しない EPM System リソース

EPM System 製品	保護しないコンテキスト
Financial Reporting	<ul style="list-style-type: none"> • /hr/common/HRLogon.jsp • /hr/services • /hr/services/* • /hr/services/.../* • /hr/modules/com/hyperion/reporting/web/reportViewer/HRStaticReport.jsp • /hr/modules/com/hyperion/reporting/web/repository/HRObjectListXML.jsp • /hr/modules/com/hyperion/reporting/web/reportViewer/HRHtmlReport.jsp • /hr/modules/com/hyperion/reporting/web/bookViewer/HRBookTOCFns.jsp • /hr/modules/com/hyperion/reporting/web/bookViewer/HRBookPdf.jsp
Data Relationship Management Calculation Manager	<ul style="list-style-type: none"> /drm-migration-client • /calcmgr/importexport.postExport.do • /calcmgr/common.performAction.do • /calcmgr/lcm.performAction.do* • /calcmgr/lcm.performAction.do/*
Administration Services	<ul style="list-style-type: none"> • /eas • /easconsole • /easdocs
Financial Management	<ul style="list-style-type: none"> • /hfm/EIE/EIListener.asp • /hfmapplicationsservice • /oracle-epm-fm-webservices • /hfmlcmsservice
Financial Close Management	<ul style="list-style-type: none"> • /FCC-DataModel-context-root • /oracle-epm-erpi-webservices/* • /ARM-DataModel-context-root • /oracle-epm-erpi-webservices/** • /arm/batch/armbatchexecutionservlet • /ARM-DataModel-context-root

表 3-2 (続き) 保護しない EPM System リソース

EPM System 製品	保護しないコンテキスト
Integrated Operational Planning	<ul style="list-style-type: none"> • /interlace/services/ • /interlace/services/* • /interlace/services/* • /interlace/services/.../* • /interlace/anteros • /interlace/anteros/* • /interlace/anteros/* • /interlace/anteros/.../* • /interlace/interlace • /interlace/interlace/* • /interlace/interlace/* • /interlace/interlace/.../* • /interlace/WebHelp • /interlace/WebHelp/* • /interlace/WebHelp/* • /interlace/WebHelp/.../* • /interlace/html • /interlace/html/* • /interlace/html/* • /interlace/html/.../* • /interlace/email-book • /interlace/email-book/* • /interlace/email-book/* • /interlace/email-book/.../*
Profitability and Cost Management	<ul style="list-style-type: none"> • /profitability/cesagent • /profitability/lcm • /profitability/control • /profitability/ApplicationListener • /profitability/HPMApplicationListener
FDME	<ul style="list-style-type: none"> • /aif/services/FDMRuleService • /aif/services/RuleService • /aif/LCMServlet

アイデンティティ 管理製品を使用したヘッダーベース SSO

前提条件

- 完全に構成された Oracle Enterprise Performance Management System。アイデンティティ 管理製品のディレクトリ・サーバーは、ユーザーを認可するためのユーザー・ディレクトリとして EPM System に構成する必要があります。
- ヘッダーベース認証をサポートする完全に構成されたアイデンティティ 管理製品 (Microsoft Azure AD、Okta など)。

次の一般的なプロセスには、互換性のあるアイデンティティ 管理製品を使用したヘッダーベース SSO のための EPM System の構成が含まれます。含まれる具体的な手順

は使用している製品によって異なるため、詳細な手順はアイデンティティ管理製品のマニュアルを参照してください。

Oracle Identity Cloud Services を使用してヘッダーベース認証を構成する詳細な手順は、[Oracle Identity Cloud Services を使用したヘッダーベース SSO のための EPM System の構成](#)を参照してください。

1. **EPM System** をアイデンティティ管理製品のエンタープライズ・アプリケーションとして登録します。このステップにより、アイデンティティ管理の管理者は、多要素認証などのサポートされている機能を含め、エンタープライズ・アプリケーションで認証を構成できます。
EPM System のエンタープライズ・アプリケーション URL として、workspace/index.jsp が付加されたゲートウェイの完全修飾ドメイン名(FQDN)を使用します(たとえば、https://gateway.server.example.com:443/workspace/index.jsp)。
HTTP ヘッダーを伝播するために、**EPM System** エンタープライズ・アプリケーションを構成します。
予約されていないヘッダー名を **HTTP** ヘッダーの名前として選択できます。ヘッダーの値は、**EPM System** ユーザーを一意に識別するプロパティである必要があります。
2. アプリケーション・ゲートウェイをインストール、構成および登録し、エンタープライズ・アプリケーションが、認証された要求のみを **EPM System** に転送するようにします。次の構成設定を使用します:
 - アクセス・ポイントとしてのゲートウェイ・サーバーの FQDN (たとえば、gateway.server.example.com:443)。
 - 認証された **HTTP(S)**要求の転送先となるリソースとしての **EPM System** の FQDN (たとえば、epm.server.example.com:443)。
3. アプリケーション・ゲートウェイによって伝播された **HTTP(S)**ヘッダーを順守するように、**EPM System** で **SSO** を使用可能にします。詳細は、[セキュリティ・オプションの設定](#)を参照してください。
SSO を使用可能にするには:
 - a. システム管理者として **Oracle Hyperion Shared Services Console** にアクセスします。[Shared Services Console の起動](#)を参照してください。
 - b. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
 - c. 「セキュリティ・オプション」をクリックします。
 - d. 「シングル・サインオン構成」セクションで次のようにします:
 - i. 「**SSO の使用可能**」チェック・ボックスを選択します。
 - ii. **SSO プロバイダ/セキュリティ・エージェント**・ドロップダウン・リストから、「その他」を選択します。
 - iii. 「**SSO メカニズム**」ドロップダウン・リストから、「**カスタム HTTP ヘッダー**」を選択し、セキュリティ・エージェントが **EPM System** に渡すヘッダーの名前を指定します。
 - e. 「OK」をクリックします。
4. **Oracle Hyperion Enterprise Performance Management Workspace** ポスト・ログオフ URL 設定を、**EPM System** からログアウトしたときにユーザーに表示する **Web** ページの設定に更新します。
EPM Workspace のポスト・ログオフ URL 設定を更新するには:

- a. システム管理者として EPM Workspace にアクセスします。 [EPM Workspace へのアクセス](#) を参照してください。
 - b. 「ナビゲート」、**Workspace 設定**、「**サーバー設定**」の順に選択します。
 - c. 「**Workspace サーバー設定**」で、「**ポスト・ログオフ URL**」を、EPM System からログアウトしたときにユーザーに表示する Web ページの URL に変更します。
 - d. 「**OK**」をクリックします。
5. Oracle Hyperion Foundation Services およびすべての EPM System 管理対象サーバーを再起動します。

Oracle Identity Cloud Services を使用したヘッダーベース SSO のための EPM System の構成

このシナリオでは、Oracle Identity Cloud Services が Oracle Enterprise Performance Management System ユーザーを認証し、SSO を使用可能にするために必要な HTTP ヘッダーを伝播します。

この項では、Oracle Identity Cloud Services を使用して SSO をサポートするための EPM System の設定および構成に関する手順を説明します。これらの手順は、アイデンティティ管理システム(Azure AD など)や、ヘッダーベース認証をサポートする IaaS (Infrastructure as a Service) プロバイダを使用した、EPM System のヘッダーベース認証をサポートするための手順と考えることができます。

概念的なワークフローは次のとおりです:

- リバース・プロキシとして機能するゲートウェイ・アプリケーションが、認証されていないネットワーク・アクセスを制限することで、EPM System コンポーネントを保護します。
- ゲートウェイ・アプリケーションが EPM System コンポーネントへの HTTP(S) 要求をインターセプトし、要求を EPM System コンポーネントに転送する前に、アイデンティティ管理製品がユーザーを認証するようにします。
- 要求を EPM System コンポーネントに転送している間、ゲートウェイ・アプリケーションは、認証されたユーザーのアイデンティティを HTTP ヘッダー要求を介して EPM System コンポーネントに伝播します。

前提条件およびサンプル URL

Oracle Identity Cloud Services でヘッダーベース SSO を確立するには、次の条件が整っている必要があります:

- 完全に構成された Oracle Enterprise Performance Management System。
- Oracle App Gateway が完全に構成されているホストまたはコンテナ。リバース・プロキシとして機能し、認証されていないアクセスを制限することで EPM System を保護します。

EPM System コンポーネントへの HTTP 要求をインターセプトし、EPM System に要求を転送する前に、ユーザーが Oracle Identity Cloud Services によって認識されるように Oracle App Gateway を構成する必要があります。要求を EPM System コンポーネントに転送している間、Oracle App Gateway は、認証されたユ

ーザーのアイデンティティを HTTP ヘッダー要求を介して伝播する必要があります。

- Oracle Identity Cloud Services へのドメイン管理者アクセス。

ここでは、次のサンプル URL が使用されます:

- Oracle Identity Cloud Services サーバー(アイデンティティ・プロバイダ)の完全修飾ドメイン名(FQDN)ベースの URL:
`https://identity.server.example.com:443/`
- Oracle App Gateway サーバー(ゲートウェイ・アプリケーションをホストする)の FQDN:
`https://gateway.server.example.com:443/`
- EPM System のエンタープライズ・アプリケーション URL。これは workspace/index.jsp が付加された Oracle App Gateway サーバーの FQDN です:
`https://gateway.server.example.com:443/workspace/index.jsp`

Note:

Oracle Identity Cloud Services および Oracle App Gateway は、HTTPS をサポートするように構成されます。EPM System の HTTPS サポートはオプションです。ここでは、EPM System が HTTPS サポートで構成されていることを想定しています。

EPM System のヘッダーベース認証の使用可能化

Oracle Enterprise Performance Management System のヘッダーベース認証を使用可能にするには、次の手順が含まれます:

- Oracle Identity Cloud Services への EPM System アプリケーションおよびゲートウェイの追加
- アプリケーション・ゲートウェイの構成
- 認証のためのユーザー・ディレクトリの構成
- EPM System における SSO の使用可能
- EPM Workspace 設定の更新

Oracle Identity Cloud Services への EPM System アプリケーションおよびゲートウェイの追加

ヘッダーベース認証を設定するには、Oracle Enterprise Performance Management System をエンタープライズ・アプリケーションとして作成する必要があります。

Oracle Cloud Identity Console での EPM System のエンタープライズ・アプリケーションとしての追加

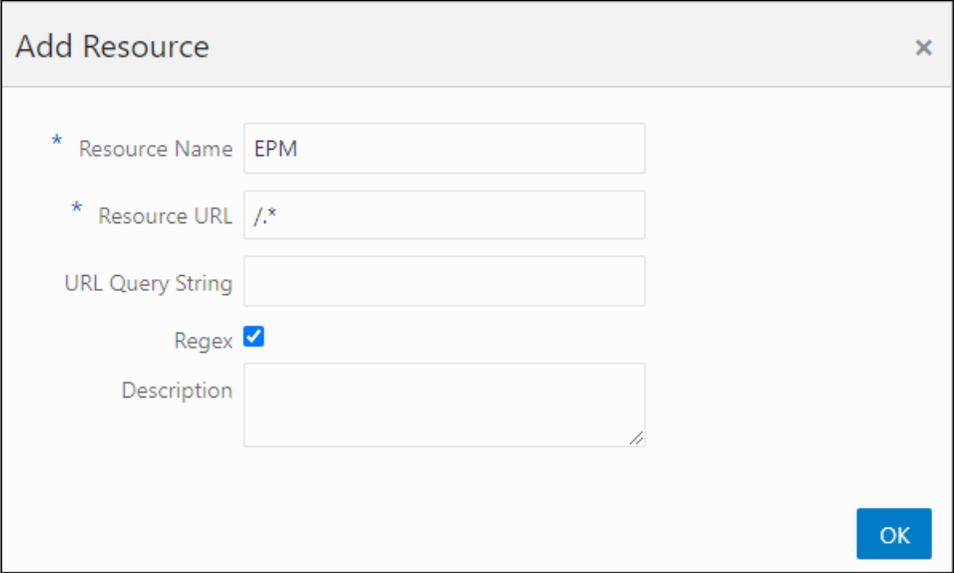
EPM System をエンタープライズ・アプリケーションとして追加するには:

1. ドメイン管理者として Oracle Cloud Identity Console にアクセスします。
 - a. ブラウザを使用して `https://www.oracle.com/cloud/sign-in.html` に移動します。

- b. Oracle Fusion Cloud EPM アカウント名を入力します。
 - c. Oracle Fusion Cloud EPM アカウント・サインイン・ページで、ユーザー名とパスワードを入力し、「サイン・イン」をクリックします。
 - d. ナビゲーション・ドロワーで、「ユーザー」、アイデンティティ (プライマリ) の順にクリックします。
 - e. アイデンティティ・コンソールをクリックします。
2. EPM System をエンタープライズ・アプリケーションとして追加します。
 - a. ナビゲーション・ドロワーで、「アプリケーション」をクリックします。
 - b. 「追加」、「エンタープライズ・アプリケーション」の順にクリックします。

3. アプリケーション詳細を追加します:
 - a. 「名前」に、EPM System エンタープライズ・アプリケーションを識別するための一意の名前を入力します。
 - b. オプションの説明を入力します。
 - c. 必要に応じて、EPM System のアプリケーション・アイコンをアップロードします。「アップロード」をクリックして、アイコンを選択してアップロードします。
 - d. 「アプリケーション URL」に、ゲートウェイがユーザーをリダイレクトする起動 URL を入力します。この URL は、EPM System アプリケーション・コンテキストである workspace/index.jsp が付加された Oracle App Gateway の FQDN です。

- e. 「設定」で、「自分のアプリケーション」に表示」を選択して、Oracle Cloud Identity Console の「個人用承認アイテム」ページの「シングル・サインオン構成」タブに EPM System エンタープライズ・アプリケーションを表示します。
 - f. 「次」をクリックします。
4. SSO 構成の詳細を指定します。
 - a. 「シングル・サインオン構成」をクリックします。
 - b. エンタープライズ・アプリケーションのリソースを追加します。「シングル・サインオン構成」で、「リソース」を展開します。
 - i. 「追加」をクリックします。



The screenshot shows a dialog box titled "Add Resource" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- Resource Name:** A text input field containing "EPM".
- Resource URL:** A text input field containing "/.*".
- URL Query String:** An empty text input field.
- Regex:** A checked checkbox.
- Description:** An empty text area.
- OK:** A blue button in the bottom right corner.

- ii. 一意のリソース名を指定します。
 - iii. 「リソース URL」に、/.*と入力します。
 - iv. 「正規表現」チェック・ボックスを選択します。
 - v. 「OK」をクリックします。
 - vi. 「シングル・サインオン構成」で、「リソース」を展開します。
- c. 認証ポリシーを追加します。「シングル・サインオン構成」で、「認証ポリシー」を展開します。
 - i. 「CORS を許可」および「セキュアな Cookie が必要」チェック・ボックスを選択します。
 - ii. 「管理対象リソース」で「追加」をクリックし、SSO リソースの認証方法として「フォームまたはアクセス・トークン」を定義します。

The screenshot shows a dialog box titled "Add Resource" with a close button (X) in the top right corner. Inside the dialog, there are several fields and sections:

- * Resource:** A text input field containing "EPM" with a search icon to its right.
- * Authentication Method:** A dropdown menu currently showing "Form or Access Token".
- Authentication Method Overrides:** A section with a plus sign (+) to its right.
- Headers:** A section with a plus sign (+) to its right.
- Header Table:** A table with two columns: "Name" and "Value". The first row contains "HYPLOGIN" in the "Name" column and "Work Email" in the "Value" column. There is a close button (X) to the right of the table.
- Add Button:** A blue button labeled "Add" located at the bottom right of the dialog.

- iii. 「リソース」で、前のステップで追加した SSO リソースを選択します。
 - iv. 「ヘッダー」を展開します。
 - v. EPM System に伝播される HTTP ヘッダーの名前を入力します。デフォルトの認証ヘッダー名は HYPLOGIN です。任意の名前を選択できます。
 - vi. 「値」で、EPM System ユーザーを一意に識別するプロパティを選択します。このフィールドの値は、EPM System のユーザーのアイデンティティと一致する必要があります。たとえば、EPM System のユーザーのアイデンティティが電子メール ID である場合は、値として勤務先電子メールを選択します。
 - vii. 「保存」をクリックします。
5. 「終了」をクリックして、エンタープライズ・アプリケーションを作成します。
 6. 「アクティブ化」をクリックして、アプリケーションを使用可能にします。
 7. アプリケーション・ゲートウェイを登録し、EPM System のホストおよびアプリケーションを設定します。
 - a. ナビゲーション・ドロワーで、「セキュリティ」、「アプリケーション・ゲートウェイ」の順にクリックします。
 - b. 「追加」をクリックします。
 - c. 「詳細」に、ゲートウェイの一意の名前を入力し、必要に応じて説明を入力します。
 - d. 「次」をクリックして、ホスト画面を開きます。
 - e. EPM System のアプリケーション・ゲートウェイのホストを追加します。
 - i. ホスト画面で、「追加」をクリックします。

The screenshot shows a dialog box titled "Add Host" with the following fields and values:

- Host Identifier:** EPMAppGateway
- Host:** gateway.server.example.com
- Port:** 443
- SSL Enabled:**
- Additional Properties:**

```
ssl_certificate /usr/local/gateway.server.example.com.crt;
ssl_certificate_key /usr/local/gateway.server.example.com.key;
ssl_password_file /usr/local/gateway.server.example.com.password.txt;
```

A green "Save" button is located at the bottom right of the dialog.

- ii. 「**ホスト識別子**」に、EPMAppGateway と入力します。
- iii. 「**ホスト**」に、アプリケーション・ゲートウェイ・サーバーをホストするコンピュータの完全修飾ドメイン名を入力します(たとえば、gateway.server.example.com)。
- iv. 「**ポート**」に、アプリケーション・ゲートウェイ・サーバーが **HTTPS** 要求に応答するポートを入力します。
- v. 「**SSL 有効**」チェック・ボックスを選択します。
- vi. 「**追加プロパティ**」で、次の情報を入力します:
 - SSL 証明書の場所
 - SSL 証明書キー
 - SSL パスワード・ファイル(必要な場合)

詳細は、*Oracle Identity Cloud Service の管理*のアプリケーション・ゲートウェイの設定の[アプリケーション・ゲートウェイの登録](#)を参照してください。
- vii. 「**保存**」をクリックします。
- viii. 「**次**」をクリックして、アプリケーション画面を開きます。
- f. EPM System エンタープライズ・アプリケーションをアプリケーション・ゲートウェイに追加します。
 - i. 「**アプリケーション**」で、「**追加**」をクリックします。
 - ii. 「**アプリケーション**」で、前に Oracle Cloud Identity Console に追加した EPM System エンタープライズ・アプリケーションを選択します。

Assign an App to gate ✕

* Application

* Select a Host

Policy default

* Resource Prefix

* Origin Server

Additional Properties

Save

- iii. 「ホストの選択」で、EPMAAppGateway (アプリケーション・ゲートウェイに追加した EPM System ホスト)を選択します。
 - iv. 「リソース接頭辞」に/と入力し、すべての要求を EPM System ホストに転送するようにします。
 - v. 「オリジン・サーバー」に、Oracle Hyperion Enterprise Performance Management Workspace をホストするコンピュータの完全修飾ドメイン名、および EPM Workspace が使用するポート番号を入力します。
 - vi. 「保存」をクリックします。
8. アプリケーション・ゲートウェイのクライアント ID およびクライアント・シークレットを記録します。これらの値は、アプリケーション・ゲートウェイを設定するために必要です。
- a. ナビゲーション・ドロワーで、「セキュリティ」、「アプリケーション・ゲートウェイ」の順にクリックします。
 - b. EPM System エンタープライズ・アプリケーション用に追加したゲートウェイの名前をクリックします。
 - c. クライアント ID (英数字文字列)をテキスト・エディタにコピーします。
 - d. 「シークレットの表示」をクリックして、クライアントのシークレット・コードを表示します。
 - e. クライアント・シークレット(英数字文字列)をテキスト・エディタにコピーします。
 - f. テキスト・ファイルを保存します。

Note:

Oracle Identity Cloud Services に対して構成の更新が行われるたびに、アプリケーション・ゲートウェイ・サーバーを再起動する必要があります。アプリケーション・ゲートウェイ・サーバーを起動および停止するには、[アプリケーション・ゲートウェイの起動および停止](#)を参照してください。

アプリケーション・ゲートウェイの構成

詳細は、*Oracle Identity Cloud Service の管理*の[アプリケーション・ゲートウェイの設定](#)を参照してください。

アプリケーション・ゲートウェイ・サーバーを構成するには、前の項で記録したクライアント ID とクライアント・シークレットが必要です。

認証のためのユーザー・ディレクトリの構成

Oracle Identity Cloud Services や Microsoft Azure などの一部のアイデンティティ管理製品は、Oracle Enterprise Performance Management System でユーザー・ディレクトリとして直接構成できません。このような製品は、Oracle Unified Directory または Oracle Virtual Directory で構成してから、それを EPM System でユーザー・ディレクトリとして構成します。ユーザー・ディレクトリの構成の詳細な手順は、[ユーザー・ディレクトリの構成](#)を参照してください。

EPM System における SSO の使用可能

Oracle Enterprise Performance Management System でセキュリティ・オプションを構成して、SSO を使用可能にします。手順の詳細は、[セキュリティ・オプションの設定](#)を参照してください。

SSO を使用可能にするには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。
[Shared Services Console の起動](#)を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 「セキュリティ・オプション」をクリックします。
4. 「シングル・サインオン構成」セクションで次のようにします:
 - a. 「SSO の使用可能」チェック・ボックスを選択します。
 - b. **SSO プロバイダ**/セキュリティ・エージェント・ドロップダウン・リストから、「その他」を選択します。
 - c. 「SSO メカニズム」ドロップダウン・リストから、「**カスタム HTTP ヘッダー**」を選択し、セキュリティ・エージェントが EPM System に渡すヘッダーの名前(HYPLOGIN、または Oracle Cloud Identity Console でエンタープライズ・アプリケーションのリソースを追加する際に指定したカスタム名)を指定します。
5. 「OK」をクリックします。

Note:

SSO 構成の変更の後に、すべての EPM System サービスを再起動するようにしてください。

EPM Workspace 設定の更新

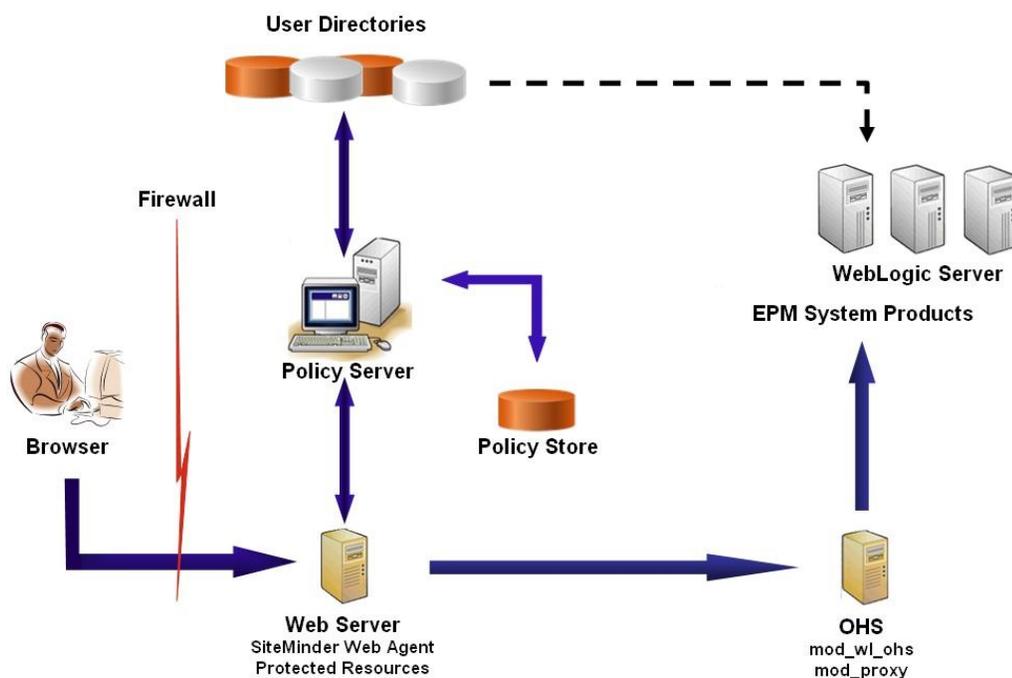
1. システム管理者として Oracle Hyperion Enterprise Performance Management Workspace にアクセスします。 [EPM Workspace へのアクセス](#) を参照してください。
2. 「ナビゲート」、Workspace 設定、「サーバー設定」の順に選択します。
3. 「Workspace サーバー設定」で、「ポスト・ログオフ URL」を、Oracle Enterprise Performance Management System からログアウトしたときにユーザーに表示する Web ページの URL に変更します。
4. 「OK」をクリックします。
5. Oracle Hyperion Foundation Services およびすべての EPM System コンポーネントを再起動します。

SiteMinder SSO

SiteMinder は Web 専用のソリューションです。デスクトップ・アプリケーションおよびそのアドイン(たとえば、Microsoft Excel や Report Designer)は、SiteMinder からの認証を使用できません。ただし、Oracle Smart View for Office では、SiteMinder 認証を使用できます。

プロセス・フロー

SiteMinder 使用可能な SSO の概要を図で示します:



SiteMinder の SSO プロセス:

1. ユーザーは、SiteMinder によって保護された Oracle Enterprise Performance Management System リソースにアクセスしようとします。SiteMinder ポリシー・サーバーをフロントエンドする Web サーバーに接続する URL(`http://WebAgent_Web_Server_Name:WebAgent_Web_ServerPort/interop/index.jsp` など)を使用します。
2. Web サーバーは、資格証明を要求するポリシー・サーバーにユーザーをリダイレクトします。構成済ユーザー・ディレクトリに対する資格証明の検証後、ポリシー・サーバーは、SiteMinder Web エージェントのホストとなる Web サーバーに資格証明を渡します。
3. SiteMinder Web エージェントのホストとなる Web サーバーは、EPM System をフロントエンドする Oracle HTTP Server に要求をリダイレクトします。Oracle HTTP Server は、Oracle WebLogic Server 上にデプロイされている、要求されたアプリケーションにユーザーをリダイレクトします。
4. EPM System コンポーネントは、プロビジョニング情報を確認し、コンテンツを提供します。このプロセスが機能するには、SiteMinder でユーザーの認証に使用するユーザー・ディレクトリを、EPM System の外部ユーザー・ディレクトリとして構成する必要があります。これらのディレクトリは信頼済として構成する必要があります。

注意事項

SiteMinder は Web 専用のソリューションです。デスクトップ・アプリケーションおよびそのアドイン(たとえば、Microsoft Excel や Report Designer)は、SiteMinder からの認証を使用できません。ただし、Smart View では、SiteMinder 認証を使用できます。

前提条件

1. 完全な機能の SiteMinder インストールは、次のコンポーネントで構成されています:
 - ポリシーおよびエージェント・オブジェクトが定義された SiteMinder ポリシー・サーバー
 - SiteMinder ポリシー・サーバーをフロントエンドする Web サーバーにインストールされた SiteMinder Web エージェント
2. 完全な機能の EPM System デプロイメント。
EPM System コンポーネントに Web サーバーを構成する場合、EPM System コンフィグレータは、WebLogic Server に要求をプロキシするように `mod_wl_ohs.conf` を構成します。

SiteMinder Web エージェントの使用可能化

Web エージェントは、EPM System リソースに対する要求をインターセプトする Web サーバー上にインストールされます。認証されていないユーザーが保護された EPM System リソースにアクセスしようとすると、Web エージェントではユーザーに対して SSO 資格証明を要求します。ユーザーが認証されると、ポリシー・サーバーは認証されたユーザーのログイン名を追加し、そのログイン名はヘッダーで渡されます。その後、HTTP 要求が EPM System Web サーバーに渡され、要求をリダイレクトします。EPM System コンポーネントはヘッダーから認証済ユーザー資格証明を抽出します。

SiteMinder は、異種の Web サーバー・プラットフォーム上で実行されている EPM System 製品全体の SSO をサポートしています。EPM System 製品が異なる Web サーバーを使用する場合、SiteMinder Cookie を同じドメイン内の Web サーバーに確実に渡せるようにする必要があります。各 Web サーバーの `WebAgent.conf` ファイルの `Cookiedomain` プロパティの値として適切な EPM System アプリケーション・ドメインを指定して、これを行います。

Netegrity SiteMinder エージェント・ガイドの Web エージェントの構成に関する項を参照してください。

ノート:

Oracle Hyperion Shared Services はそのコンテンツを保護するために基本認証を使用するため、Shared Services への要求をインターセプトする Web サーバーは、SiteMinder を使用した SSO をサポートできるように基本認証を使用可能にする必要があります。

SiteMinder Web エージェント構成ウィザードを実行して、Web エージェントを構成します(これを行うには、`WEBAGENT_HOME/install_config_info/nete-wa-config` を実行します。たとえば、Windows の場合、`C:\netegrity\webagent\install_config_info\nete-wa-config.exe` になります)。構成プロセスでは、SiteMinder Web サーバーの `WebAgent.conf` が作成されます。

SiteMinder Web エージェントを使用可能にするには:

1. テキスト・エディタを使用して、`WebAgent.conf` を開きます。このファイルの場所は、使用している Web サーバーによって異なります。
2. `enableWebAgent` プロパティの値をはいに設定します。

```
enableWebAgent="YES"
```
3. Web エージェント構成ファイルを保存して閉じます。

例 3-1 SiteMinder ポリシー・サーバーの構成

SiteMinder 管理者は、EPM System 製品に SSO が使用可能になるようにポリシー・サーバーを構成する必要があります。

構成プロセスは次のとおりです:

- SiteMinder Web エージェントの作成、および SiteMinder Web サーバーに適した構成オブジェクトの追加
- 保護する必要がある各 EPM System リソースのレルムの作成、および Web エージェントのレルムへの追加。保護するリソースを参照してください
- 保護する EPM System リソース用に作成されたレルム内で、保護しないリソース用のレルムを作成します。保護しないリソースを参照してください
- HTTP ヘッダー参照の作成。ヘッダーは、EPM System アプリケーションにログイン属性の値を提供する必要があります。ログイン属性の簡単な説明は、Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイドの OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。
- Web エージェント・アクションとして、取得、ポストおよび配置を使用したレルム内のルール作成
- 値が `hyplogin=<%userattr="SM_USERLOGINNAME"%>` のレスポンス属性の作成
- ポリシーの作成、ユーザー・ディレクトリ・アクセスの割当て、および EPM System 用に作成したルールの現在のメンバー・リストへの追加

- EPM System コンポーネント用に作成したルールに対する応答の設定

例 3-2 EPM System Web サーバーに要求を転送するための SiteMinder Web サーバーの構成

SiteMinder Web エージェントのホストとなる Web サーバーを構成して、認証済ユーザー(ユーザーを識別するヘッダーを含む)から EPM System Web サーバーに要求を転送します。

Apache ベースの Web サーバー用に、次の記述に類似したディレクティブを使用して、認証済要求を転送します:

```
ProxyPass / http://EPM_WEB_SERVER:EPM_WEB_SERVER_PORT/
ProxyPassReverse / http://EPM_WEB_SERVER:EPM_WEB_SERVER_PORT/
ProxyPreserveHost On
#If SiteMinder Web Server is using HTTPS but EPM Web Server is using HTTP
RequestHeader set WL-Proxy-SSL true
```

このディレクティブで、`EPM_WEB_SERVER` および `EPM_WEB_SERVER_PORT` を、各自の環境の実際の値に置き換えます。

例 3-3 EPM System で SiteMinder を使用可能にする

SiteMinder との統合により、EPM System 製品の SiteMinder 認証を使用可能にする必要があります。[SSO 用の EPM System の構成](#)を参照してください。

Kerberos シングル・サインオン

概要

Oracle Enterprise Performance Management System 製品は、EPM System 製品をホストするアプリケーション・サーバーが Kerberos 認証用に設定されている場合は、Kerberos SSO をサポートします。

Kerberos は信頼できる認証サービスで、各 Kerberos クライアントは他の Kerberos クライアント(ユーザー、ネットワーク・サービスなど)のアイデンティティを有効なものとして信頼します。

EPM System 製品にユーザーがアクセスする場合に行われる処理は、次のとおりです:

1. Windows コンピュータで、ユーザーは、Kerberos レルムでもある Windows ドメインにログインします。
2. 統合 Windows 認証を使用するように構成されているブラウザを使用して、ユーザーはアプリケーション・サーバー上で実行されている EPM System 製品にログインします。
3. アプリケーション・サーバー(ネゴシエート・アイデンティティ・アサーション・プロバイダ)は要求をインターセプトし、ブラウザの認証ヘッダーから Kerberos チケットとともに Simple and Protected Generic Security Services API (GSSAPI) Negotiation Mechanism (SPNEGO) トークンを取得します。
4. アサーション・プロバイダは、EPM System 製品にユーザーに関する情報を渡すために、そのアイデンティティ・ストアに対してトークンに含まれるユーザーのアイデンティティの妥当性を確認します。EPM System 製品は Active Directory に対してユーザー名を検証します。EPM System 製品は、すべての EPM System 製品間で SSO をサポートする SSO トークンを発行します。

サポート制約事項

Kerberos SSO は、すべての EPM System 製品に対してサポートされていますが、次の例外があります:

- Kerberos SSO は、Oracle Smart View for Office 以外のシック・クライアントに対してはサポートされていません。
- Smart View では、Oracle Essbase、Oracle Hyperion Planning および Oracle Hyperion Financial Management プロバイダに対してのみ Kerberos 統合をサポートしています

前提条件

このドキュメントにはアプリケーション・レベルの Kerberos 構成ステップが記載されていますが、システム・レベルでの Kerberos 構成に関する知識があることを前提としています。これらの手順を開始する前に、次のタスクの前提条件が満たされていることを確認してください。

このドキュメントでは、Windows クライアント・マシンが Kerberos 認証用に構成されているフル機能の Kerberos 対応ネットワーク環境で作業していることを前提としています。

- 企業の Active Directory が Kerberos 認証用に構成されているものとします。
[Microsoft Windows Server のドキュメント](#)を参照してください。
- EPM System 製品へのアクセスに使用されるブラウザは、Kerberos チケットを使用してネゴシエートするように構成されているものとします。
- KDC とクライアント・マシン間で、時間同期の誤差は 5 分以内です。
[Authentication Errors are Caused by Unsynchronized Clocks \(http://technet.microsoft.com/en-us/library/cc780011\(WS.10\).aspx\)](http://technet.microsoft.com/en-us/library/cc780011(WS.10).aspx)を参照してください。

WebLogic Server を使用した Kerberos SSO

Oracle WebLogic Server Kerberos SSO では、ネゴシエート・アイデンティティ・アサーション・プロバイダを使用して SPNEGO トークンのネゴシエーションとデコードを行うことによって、Microsoft クライアントでの SSO を実現します。WebLogic Server は、Kerberos チケットを取得するために SPNEGO トークンをデコードし、そのチケットを検証して WebLogic Server ユーザーにマップします。WebLogic Server の Active Directory 認証プロバイダをネゴシエート・アイデンティティ・アサーション・プロバイダとともに使用することで、Active Directory を WebLogic Server ユーザーのユーザー・ディレクトリとして構成できます。

ブラウザが EPM System 製品へのアクセスを要求すると、KDC はそのブラウザに Kerberos チケットを発行し、それによって、サポートされる GSS トークン・タイプを含む SPNEGO トークンが作成されます。ネゴシエート・アイデンティティ・アサーション・プロバイダは SPNEGO トークンをデコードし、GSSAPI を使用して、セキュリティ・コンテキストを受け入れます。要求を開始したユーザーのアイデンティティはユーザー名にマップされ、WebLogic Server に渡されます。また、WebLogic Server は、ユーザーが属するグループを決定します。この段階で、要求された EPM System 製品はユーザーに使用できるようになります。

 ノート:

ユーザーは SPNEGO をサポートするブラウザ(たとえば、Internet Explorer や Firefox など)を使用して、WebLogic Server で実行している EPM System 製品にアクセスする必要があります。

認証プロセスから取得されたユーザー ID を使用して、EPM System 製品認証プロセスはプロビジョニング・データをチェックします。EPM System 製品へのアクセスは、プロビジョニング・データに基づいて制限されます。

Kerberos 認証をサポートするための WebLogic Server での手順

Kerberos 認証をサポートするには、管理者は次のタスクを完了する必要があります:

- EPM System の WebLogic ドメインを作成します。[EPM System の WebLogic ドメインの作成](#)を参照してください。
- 認証プロバイダを作成します。[WebLogic Server での LDAP 認証プロバイダの作成](#)を参照してください。
- ネゴシエート・アイデンティティ・アサーション・プロバイダを作成します。[ネゴシエート・アイデンティティ・アサーション・プロバイダの作成](#)を参照してください。
- Kerberos 識別を作成します。[WebLogic Server 用の Kerberos 識別の作成](#)を参照してください。
- Kerberos 用の JVM オプションを更新します。[Kerberos 用の JVM オプションの更新](#)を参照してください。
- 認可ポリシーを構成します。[認可ポリシーの構成](#)を参照してください。
- SSODiag をデプロイして使用し、WebLogic Server が EPM System に対して Kerberos SSO をサポートできる状態にあるかどうかを確認します。[SSODiag を使用した Kerberos 環境のテスト](#)を参照してください。

EPM System の WebLogic ドメインの作成

通常、EPM System コンポーネントは、EPMSysWebLogic ドメイン(デフォルトの場所は `MIDDLEWARE_HOME/user_projects/domains/EPMSysWebLogic`)にデプロイされます。

Kerberos 認証用に EPM System WebLogic ドメインを構成するには:

1. EPM System コンポーネントをインストールします。
2. Oracle Hyperion Foundation Services のみをデプロイします。
Foundation Services のデプロイメントにより、デフォルトの EPM System WebLogic ドメインが作成されます。
3. Oracle Hyperion Shared Services Console にログインして、Foundation Services のデプロイメントが成功したことを確認します。[Shared Services Console の起動](#)を参照してください。

WebLogic Server での LDAP 認証プロバイダの作成

WebLogic Server 管理者は、LDAP 認証プロバイダを作成して、ユーザー情報およびグループ情報を外部 LDAP サーバーに格納します。LDAP v2-または v3-に準拠した LDAP サーバーは、WebLogic Server と連携して機能します。次のリファレンスを参照してください:

- [Oracle Fusion Middleware Oracle WebLogic Server の保護ガイドの LDAP 認証プロバイダの構成。](#)
- [Oracle Fusion Middleware Oracle WebLogic Server Administration Console オンライン・ヘルプの認証およびアイデンティティ・アサーション・プロバイダの構成。](#)

ネゴシエート・アイデンティティ・アサーション・プロバイダの作成

ネゴシエート・アイデンティティ・アサーション・プロバイダは、Microsoft クライアントによる SSO の使用を可能にします。SPNEGO トークンをデコードして Kerberos トークンを取得し、その Kerberos トークンを検証して WebLogic ユーザーにマップします。ネゴシエート・アイデンティティ・アサーション・プロバイダは、WebLogic Security フレームワークで定義されているように Security Service Provider Interface (SSPI)の実装で、クライアントの SPNEGO トークンに基づいたクライアントの認証に必要なロジックを提供します。

- [Oracle Fusion Middleware Oracle WebLogic Server の保護ガイドのネゴシエート・アイデンティティ・アサーション・プロバイダの構成。](#)
- [Oracle Fusion Middleware Oracle WebLogic Server Administration Console オンライン・ヘルプの認証およびアイデンティティ・アサーション・プロバイダの構成。](#)

ネゴシエート・アイデンティティ・アサーション・プロバイダの作成時、すべての認証プロバイダに対して JAAS 制御フラグ・オプションを SUFFICIENT に設定します。[Oracle Fusion Middleware Oracle WebLogic Server Administration Console オンライン・ヘルプ](#)の JAAS 制御フラグの設定を参照してください。

WebLogic Server 用の Kerberos 識別の作成

Active Directory ドメイン・コントローラ・マシンで、WebLogic Server および EPM System Web サーバーを表すユーザー・オブジェクトを作成し、Kerberos レルムの WebLogic Server および Web サーバーを表すサービス・プリンシパル名(SPN)にマップします。クライアントでは、SPN がないサービスを検索できません。SPN は、ログイン・プロセスで使用する WebLogic Server ドメインにコピーする keytab ファイルに格納します。

手順の詳細は、[Oracle Fusion Middleware Oracle WebLogic Server の保護ガイドの WebLogic Server 用の識別の作成](#)を参照してください。

WebLogic Server 用の Kerberos 識別を作成するには:

1. たとえば、WebLogic Server ドメインをホストするコンピュータに epmHost を作成するなど、Active Directory ドメイン・コントローラ・マシンでユーザー・アカウントを作成します。

 ノート:

マシンではなく、ユーザー・オブジェクトとして識別を作成します。コンピュータの簡易名を使用します。たとえば、ホスト名が `epmHost.example.com` の場合、`epmHost` を使用します。

ユーザー・オブジェクトの作成時に使用したパスワードを書き留めます。これは、**SPN** の作成に必要です。

パスワード・オプション、特に「ユーザーは次回ログオン時にパスワードの変更が必要」オプションを選択しないでください。

2. Kerberos プロトコルに準拠するようにユーザー・オブジェクトを変更します。アカウントは、**Kerberos 事前認証**を必要とします。
 - 「**Account**」タブで、使用する暗号化を選択します。
 - 他のアカウント・オプション(特に「**Kerberos 事前認証を必要としない**」)が選択されていないことを確認します。
 - 暗号化タイプを設定すると、オブジェクトのパスワードが破損する可能性があるため、パスワードをオブジェクトの作成時に設定したパスワードにリセットします。
3. **Active Directory** ドメイン・コントローラをホストするコンピュータで、コマンド・プロンプト・ウィンドウを開き、**Active Directory サポート・ツール**がインストールされているディレクトリに移動します。
4. 必要な **SPN** を作成して構成します。
 - a. 次のようなコマンドを使用して、この手順のステップ 1 で作成したユーザー・オブジェクト(`epmHost`)に **SPN** が関連付けられていることを確認します。

```
setspn -L epmHost
```

- b. 次のようなコマンドを使用して、**Active Directory** ドメイン・サービス(**AD DS**)で **WebLogic Server** の **SPN** を構成し、共有秘密キーを含む **keytab** ファイルを生成します。

```
ktpass -princ HTTP/epmHost.example.com@EXAMPLE.COM -pass password -mapuser epmHost -out c:\epmHost.keytab
```

5. **WebLogic Server** をホストするコンピュータで **keytab** ファイルを作成します。
 - a. コマンド・プロンプトを開きます。
 - b. `MIDDLEWARE_HOME/jdk/bin` に移動します。
 - c. 次のようなコマンドを実行します:

```
ktab -k keytab_filename -a epmHost@example.com
```
 - d. パスワードの入力を求められたら、この手順のステップ 1 でユーザーの作成時に設定したパスワードを入力します。
6. **WebLogic** ドメイン内の起動ディレクトリ (`C:\Oracle\Middleware\user_projects\domains\EPMSysystem など`)に **keytab** ファイルをコピーします。

7. Kerberos 認証が正しく機能していることを確認します。

```
kinit -k -t keytab-file account-name
```

このコマンドで、account-name は Kerberos プリンシパルを示します(例: HTTP/epmHost.example.com@EXAMPLE.COM)。このコマンドからの出力は次のようになります:

```
New ticket is stored in cache file C:\Documents and Settings\Username\krb5cc_MachineB
```

Kerberos 用の JVM オプションの更新

Oracle Fusion Middleware Oracle WebLogic Server の保護 11g リリース 1 (10.3.1)の WebLogic Server による Kerberos 認証での起動引数の使用および JAAS ログイン・ファイルの作成を参照してください。

EPM System 管理対象サーバーが Windows のサービスとして稼働している場合、Windows レジストリを更新して、JVM 起動オプションを設定します。

Windows レジストリで JVM 起動オプションを更新するには:

1. Windows レジストリ・エディタを開きます。
2. 「マイ コンピュータ」、「HKEY_LOCAL_MACHINE」、「Software」、「Hyperion Solutions」、「Foundationservices0」、「HyS9EPMServer_epmsystem1」の順に選択します。
3. 次の文字列値を作成します:

 **ノート:**

次の表にリストされた名前は例です。

表 3-3 Kerberos 認証用の JVM 起動オプション

名前	タイプ	データ
JVMOption44	REG_SZ	-Djava.security.krb5.realm=Active Directory Realm Name
JVMOption45	REG_SZ	-Djava.security.krb5.kdc=Active Directory host name or IP address
JVMOption46	REG_SZ	- Djava.security.auth.login.config=location of Kerberos login configuration file
JVMOption47	REG_SZ	- Djavax.security.auth.useSubjectCredsOnly=false

4. 追加した JVMOption を反映するように JVMOptionCount DWord の値を更新します(現在の 10 進数値に 4 を加えます)。

認可ポリシーの構成

EPM System にアクセスする Active Directory ユーザー用の認可ポリシーの構成の詳細は、*Oracle Fusion Middleware Oracle WebLogic Server* **ロールおよびポリシーによるリソースの保護ガイド**の [Web アプリケーションおよび EJB リソースの保護のオプション](#)を参照してください。

サンプル・ポリシーの構成ステップは、[SSODiag 用のポリシーの作成](#)を参照してください。

SSODiag を使用した Kerberos 環境のテスト

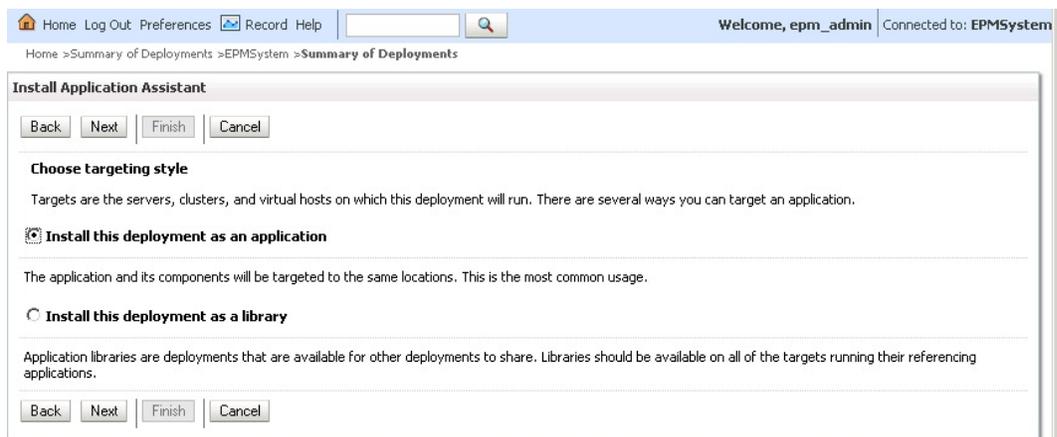
SSODiag は、Kerberos 環境で WebLogic Server が EPM System をサポートできる状態にあるかどうかをテストする診断 Web アプリケーションです。

SSODiag のデプロイ

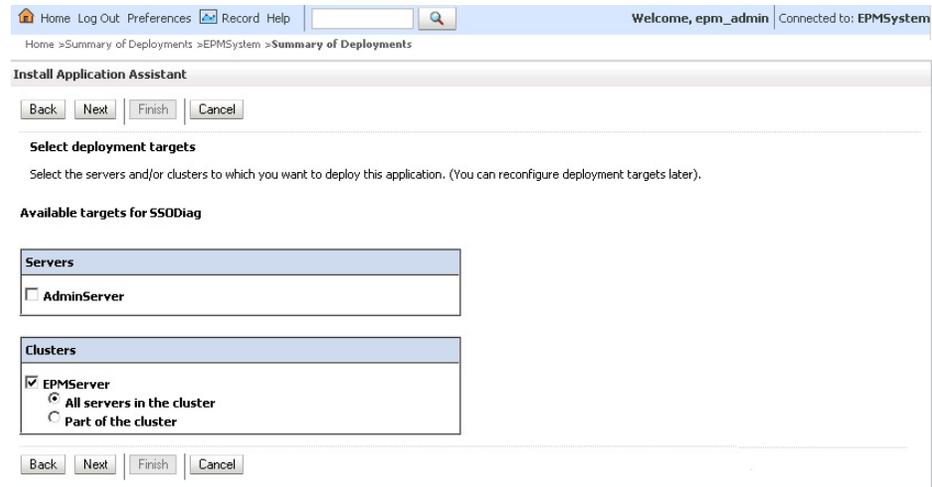
Foundation Services のデプロイ時に指定した WebLogic Server 管理者の資格証明(デフォルトのユーザー名は `epm_admin`)を使用して、SSODiag をデプロイします。

SSODiag をデプロイして構成するには:

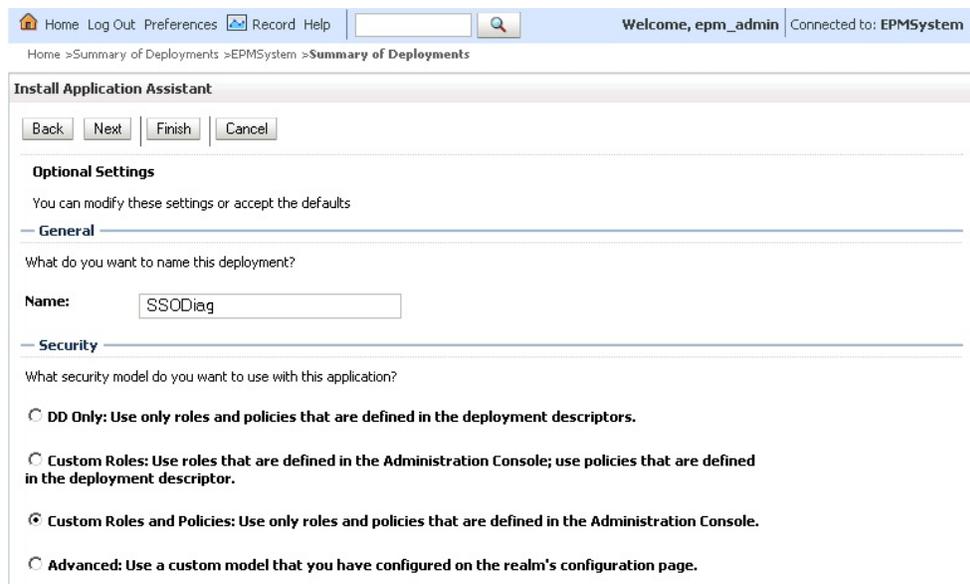
1. EPM System ドメインに対する WebLogic Server 管理コンソールにログオンします。
2. チェンジ・センターで、「**ロックして編集**」をクリックします。
3. 「**ドメイン構造**」の「**EPMSystem**」から、「**デプロイメント**」をクリックします。
4. **デプロイメントの要約**で、「**インストール**」をクリックします。
5. 「**パス**」で、`EPM_ORACLE_HOME/products/Foundation/AppServer/InstallableApps/common/SSODiag.war` を選択します。
6. 「**次**」をクリックします。
7. **ターゲット指定スタイルの選択**で、**このデプロイメントをアプリケーションとしてインストールする**が選択されていることを確認し、「**次へ**」をクリックします。



8. 「**デプロイ・ターゲットの選択**」で、次を選択し、「**次へ**」をクリックします。
 - EPMServer
 - クラスタのすべてのサーバー



9. 「オプション設定」で、「カスタム・ロールおよびポリシー: 管理コンソール内に定義されたロールとポリシーのみを使用します。」をセキュリティ・モデルとして選択します。



10. 「次」をクリックします。
11. 確認画面で、「いいえ、後で構成を確認します。」を選択します。
12. 「終了」をクリックします。
13. チェンジ・センターで、「変更のアクティブ化」を選択します。

SSODiag 用の Oracle HTTP Server の構成

mod_wl_ohs.conf を更新して、SSODiag URL 要求を WebLogic Server に転送するよう Oracle HTTP Server を構成します。

Oracle HTTP Server で URL 転送を構成するには:

1. テキスト・エディタを使用して、`EPM_ORACLE_INSTANCE/httpConfig/ohs/config/fmwconfig/components/OHS/ohs_component/mod_wl_ohs.conf` を開きます。
2. SSODiag の LocationMatch 定義を追加します:

```
<LocationMatch /SSODiag/>
    SetHandler weblogic-handler
    WeblogicCluster myServer:28080
</LocationMatch>
```

前述の例で、myServer は Foundation Services ホスト・マシンを表し、28080 は Oracle Hyperion Shared Services が要求をリスニングするポートを表します。

3. mod_wl_ohs.conf を保存して閉じます。
4. Oracle HTTP Server を再起動します。

SSODiag 用のポリシーの作成

WebLogic Server 管理コンソールでポリシーを作成し、次の SSODiag URL を保護します。

`http://OHS_HOST_NAME:PORT/SSODiag/krbssodiag`

この例では、OHS_HOST_NAME は Oracle HTTP Server をホストするサーバーの名前を表し、PORT は Oracle HTTP Server が要求をリスニングするポートを表します。

SSODiag を保護するポリシーを作成するには:

1. WebLogic Server 管理コンソールのチェンジ・センターで、EPM System に対して「**ロックして編集**」を選択します。
2. 「**デプロイメント**」、SSODiag、「**セキュリティ**」、「**URL パターン**」、「**ポリシー**」の順に選択します。
3. 次の URL パターンを作成します:
 - /
 - /index.jsp
4. 作成した各 URL パターンを変更します:
 - a. 「**スタンドアロン Web アプリケーションの URL パターン**」の URL パターンのリストから、作成したパターン(/)をクリックして開きます。
 - b. 「**条件の追加**」を選択します。
 - c. 「**述部リスト**」から「**ユーザー**」を選択します。
 - d. 「**次**」を選択します。
 - e. 「**ユーザー引数名**」で、Kerberos 認証用に構成されているクライアント・デスクトップへのアクセスに使用するアカウントを持つ Active Directory ユーザー(krbuser1 など)を入力し、「**追加**」を選択します。krbuser1 は Active Directory または Windows デスクトップ・ユーザーです。
 - f. 「**終了**」を選択します。
5. 「**保存**」を選択します。

SSODiag を使用した Kerberos 認証用の WebLogic Server 構成のテスト

Kerberos 認証用の WebLogic Server 構成が正しく機能する場合、*Oracle Hyperion Kerberos SSO 診断ユーティリティ V 1.0* ページに次のメッセージが表示されます:

```
Retrieving Kerberos User principal name... Success.
Kerberos principal name retrieved... SOME_USER_NAME
```

▲ 注意:

SSODiag が Kerberos プリンシパル名を取得できない場合、Kerberos 認証用に EPM System コンポーネントを構成しないでください。

Kerberos 認証用の WebLogic Server 構成をテストするには:

1. Foundation Services と Oracle HTTP Server を起動します。
2. WebLogic Server 管理コンソールを使用して、すべての要求を処理する SSODiag Web アプリケーションを起動します。
3. 有効な Active Directory 資格証明を使用して、Kerberos 認証用に構成されているクライアント・マシンにログオンします。
4. ブラウザを使用して、次の SSODiag URL に接続します:

```
http://OHS_HOST_NAME:PORT/SSODiag/krbssodiag
```

この例では、*OHS_HOST_NAME* は Oracle HTTP Server をホストするサーバーの名前を表し、*PORT* は Oracle HTTP Server が要求をリスニングするポートを表します。

Kerberos 認証が適切に機能する場合、SSODiag は次の情報を表示します:

```
Retrieving Kerberos User principal name... Success.
Kerberos principal name retrieved... SOME_USER_NAME
```

Kerberos 認証が適切に機能しない場合、SSODiag は次の情報を表示します:

```
Retrieving Kerberos User principal name... failed.
```

セキュリティ・モデルの変更

セキュリティ・レルムによって保護されている Web アプリケーションのデフォルト・セキュリティ・モデルは DOnly です。セキュリティ・モデルを CustomRolesAndPolicies に変更する必要があります。

セキュリティ・モデルを変更するには:

1. テキスト・エディタを使用して、*MIDDLEWARE_HOME/user_projects/domains/EPMSystem/config/config.xml* を開きます。

2. 各 Foundation Services コンポーネントのアプリケーション配置記述子で次の要素を特定します:

```
<security-dd-model>DDOnly</security-dd-model>
```

3. 各コンポーネントのセキュリティ・モデルを次のように変更します:

```
<security-dd-model>CustomRolesAndPolicies</security-dd-model>
```

4. config.xml を保存して閉じます。

EPM System セキュリティ構成の更新

EPM System セキュリティ構成を変更し、Kerberos SSO を使用可能にします。

Kerberos 認証用に EPM System を構成するには:

1. Shared Services Console に管理者としてログオンします。
2. Shared Services で、Kerberos 認証用に外部ユーザー・ディレクトリとして構成されている Active Directory ドメインを追加します。*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*の OID、およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。
3. SSO を使用可能にします。OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。**「セキュリティ・オプション」**で、次の表の設定を選択し、Kerberos SSO を使用可能にします。

表 3-4 Kerberos SSO を使用可能にする設定

フィールド	必要な設定
SSO の使用可能	選択
SSO プロバイダ/エージェント	その他
SSO メカニズム	HTTP 要求からリモート・ユーザーを取得

4. Foundation Services を再起動します。

Kerberos SSO のテスト

Foundation Services にログインし、Kerberos SSO が適切に機能することを確認します。

Kerberos SSO をテストするには:

1. Foundation Services と Oracle HTTP Server が稼働していることを確認します。
2. 有効な Active Directory 資格証明を使用して、Kerberos 認証用に構成されているクライアント・マシンにログオンします。
3. ブラウザを使用して、Foundation Services URL に接続します。

EPM System コンポーネントの構成

EPM System コンフィグレータを使用して、Foundation Services がデプロイされている WebLogic ドメインに他の EPM System コンポーネントを構成およびデプロイします。

Kerberos 認証用の EPM System 管理対象サーバーの構成

Microsoft Windows 環境では、EPM System 管理対象サーバーは Windows サービスとして実行されます。WebLogic 管理対象サーバーごとに JVM 起動オプションを変更する必要があります。非コンパクト・デプロイメント・モードの管理対象サーバーの包括的なリストは次のとおりです:

- AnalyticProviderServices0
- CalcMgr0
- ErpIntegrator0
- EssbaseAdminServer0
- FinancialReporting0
- HFMWeb0
- FoundationServices0
- HpsAlerter0
- HpsWebReports0
- Planning0
- Profitability0

EPM System の Web アプリケーションがコンパクト・デプロイメント・モードでデプロイメントされている場合は、EPMSystem0 管理対象サーバーの JVM 起動オプションのみを更新する必要があります。複数のコンパクト管理対象サーバーがある場合、すべての管理対象サーバーについて JVM 起動オプションを更新する必要があります。

Oracle Fusion Middleware Oracle WebLogic Server の保護ガイドの [WebLogic Server による Kerberos 認証での起動引数の使用](#) を参照してください。

ノート:

次の手順は、FoundationServices 管理対象サーバーの JVM 起動オプションを設定する方法を示しています。このタスクは、デプロイメント内の WebLogic 管理対象サーバーごとに実行する必要があります。

WebLogic Server 起動スクリプトで JVM オプションを構成する手順の詳細は、[Kerberos 用の JVM オプションの更新](#) を参照してください。

WebLogic Server 起動スクリプトで JVM オプションを構成する手順

認可ポリシーの構成

Foundation Services 以外の EPM System コンポーネントにアクセスする Active Directory ユーザー用の認可ポリシーを構成します。WebLogic 管理コンソールからのセキュリティ・ポリシーの構成の詳細は、[認可ポリシーの構成](#) を参照してください。

EPM System コンポーネントのデフォルトのセキュリティ・モデルの変更

EPM System 構成ファイルを編集して、デフォルトのセキュリティ・モデルを変更します。非コンパクト EPM System デプロイメントの場合、config.xml に記録されてい

各 EPM System Web アプリケーションのデフォルトのセキュリティ・モデルを変更する必要があります。EPM System Web アプリケーションのリストは次のとおりです:

- AIF
- APS
- CALC
- EAS
- FINANCIALREPORTING
- PLANNING
- PROFITABILITY
- SHARED SERVICES
- WORKSPACE

セキュリティ・モデルを変更するには:

1. テキスト・エディタを使用して、`MIDDLEWARE_HOME/user_projects/domains/EPMSystem/config/config.xml` を開きます
2. 各 EPM System コンポーネントの `app-deployment` 定義で、次の例に示すように、`<security-dd-model>` の値を `CustomRolesAndPolicies` に設定します:

```
<app-deployment>
  <name>SHARED SERVICES#11.1.2.0</name>
  <target>EPMServer</target>
  <module-type>ear</module-type>
  <source-path>C:\Oracle\Middleware\EPMSystem11R1/products/Foundation/
AppServer/InstallableApps/common/interop.ear</source-path>
  <security-dd-model>CustomRolesAndPolicies</security-dd-model>
  <staging-mode>nostage</staging-mode>
</app-deployment>
```

3. `config.xml` を保存して閉じます。
4. WebLogic Server を再起動します。

EPM System コンポーネントの URL 保護ポリシーの作成

各 EPM System コンポーネントの URL を保護するには、WebLogic Server 管理コンソールで URL 保護ポリシーを作成します。詳細は、*Oracle Fusion Middleware Oracle WebLogic Server* [ロールおよびポリシーによるリソースの保護ガイド](#)の [Web アプリケーションおよび EJB リソースの保護のオプション](#)を参照してください。

URL 保護ポリシーを作成するには:

1. EPM System ドメインに対する WebLogic Server 管理コンソールのチェンジ・センターで、「**ロックして編集**」をクリックします。
2. 「**デプロイメント**」をクリックします。
3. デプロイメント内の EPM System エンタープライズ・アプリケーション(PLANNING など)を展開し、その Web アプリケーション(HyperionPlanning など)をクリックします。EPM System コンポーネントのリストは、[EPM System コンポーネントのデフォルトのセキュリティ・モデルの変更](#)を参照してください。

 **ノート:**

一部のエンタープライズ・アプリケーション(Oracle Essbase Administration Services など)は、URL パターンの定義が必要な複数の Web アプリケーションから構成されます。

4. Web アプリケーションの URL パターン・スコープのポリシーを作成します。
 - AIF
 - APS
 - CALC
 - EAS
 - FINANCIALREPORTING
 - PLANNING
 - PROFITABILITY
 - SHARED SERVICES
 - WORKSPACE
 - a. 「セキュリティ」、「ポリシー」、「新規」の順にクリックします。
 - b. 「URL パターン」で、EPM System 製品の保護および保護解除された URL を入力します。詳細は、[EPM System リソースの保護および保護解除](#)を参照してください。
 - c. 「OK」をクリックします。
 - d. 作成した URL パターンをクリックします。
 - e. 「条件の追加」をクリックします。
 - f. 述部リストで、ポリシー条件を選択して「次へ」をクリックします。指定したグループのすべてのメンバーにこのセキュリティ・ポリシーを付与する「グループ」条件を使用することをお勧めします。
 - g. 選択した述部に関連する引数を指定します。たとえば、前のステップで「グループ」を選択した場合、次のステップを実行する必要があります:
 - h. 「グループ引数名」に、Web アプリケーションへのアクセスを許可するユーザーを含むグループの名前を入力します。入力する名前は、Active Directory グループ名と完全一致する必要があります。
 - 「追加」をクリックします。
 - さらにグループを追加するには、前述のステップを繰り返します。
 - i. 「終了」をクリックします。Active Directory でグループが見つからない場合は、WebLogic Server にエラー・メッセージが表示されます。続行する前に、このエラーを解決する必要があります。
 - j. 「保存」を選択します。
5. デプロイメントの他の EPM System について、ステップ 3 とステップ 4 を繰り返します。
6. チェンジ・センターで、「構成の解放」をクリックします。

7. WebLogic Server を再起動します。

Web アプリケーションでのクライアント証明書ベースの認証の使用可能化

`EPM_ORACLE_HOME/products/` 内にある次のアプリケーション・アーカイブの構成ファイルに `login-config` 定義を挿入します。

- `Essbase/eas/server/AppServer/InstallableApps/Common/eas.ear`
- `FinancialDataQuality/AppServer/InstallableApps/aif.ear`
- `financialreporting/InstallableApps/HReports.ear`
- `Profitability/AppServer/InstallableApps/common/profitability.ear`

クライアント証明書ベースの認証を使用可能にするには:

1. EPM System コンポーネントおよびプロセスを停止します。
2. 7 Zip を使用して、エンタープライズ・アーカイブに含まれる Web アーカイブ(たとえば、`EPM_ORACLE_HOME/products/Essbase/eas/server/AppServer/InstallableApps/Common/eas.ear/eas.war`)を展開します。
3. WEB-INF に移動します。
4. `web.xml` を変更します。具体的には、`</webapp>`要素の直前に次の `login_config` 定義を追加します:

```
<login-config>
  <auth-method>CLIENT-CERT</auth-method>
</login-config>
```

5. `web.xml` を保存します。
6. 7 Zip によってアーカイブを更新するかどうか確認されたら、「Yes」をクリックします。

EPM System セキュリティ構成の更新

SSO を順守するように EPM System セキュリティを構成します。[SSO 用の EPM System の構成](#)を参照してください。

SSO 用の EPM System の構成

Oracle Enterprise Performance Management System 製品は、SSO 用にセキュリティ・エージェントをサポートするように構成する必要があります。Oracle Hyperion Shared Services で指定する構成では、すべての EPM System 製品に対して次のことを決定します。

- セキュリティ・エージェントから SSO を受け入れるかどうか
- SSO を受け入れる認証メカニズム

SSO を使用可能な環境において、ユーザーが最初にアクセスする EPM System 製品では、SSO メカニズムが分析され、ここに含まれている認証済ユーザー ID が取得されます。EPM System 製品では、Shared Services で構成されたユーザー・ディレクトリに対してユーザー ID がチェックされ、ユーザーが有効な EPM System ユーザーであるかどうか判別されます。また、EPM System 製品全体で SSO を使用可能にするトークンも発行されます。

Shared Services で指定される構成により、SSO が使用可能になり、すべての EPM System 製品に対して SSO を受け入れる認証メカニズムが決定されます。

Web アイデンティティ 管理ソリューションから SSO を使用可能にするには:

1. Oracle Hyperion Shared Services Console を Shared Services 管理者として起動します。Shared Services Console の起動を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. Web アイデンティティ 管理ソリューションにより使用されるユーザー・ディレクトリが Shared Services で外部のユーザー・ディレクトリとして構成されることを確認します。

たとえば、Kerberos SSO を使用可能にする場合、Kerberos 認証用に構成されている Active Directory を外部ユーザー・ディレクトリとして構成する必要があります。

手順は、ユーザー・ディレクトリの構成を参照してください。

4. 「セキュリティ・オプション」を選択します。
5. 「詳細オプションの表示」を選択します。
6. 「定義済ユーザー・ディレクトリ」画面の「シングル・サインオン構成」で、次のステップを実行します:

- a. 「SSO の使用可能」を選択します。
- b. 「SSO プロバイダ/エージェント」から、Web アイデンティティ 管理ソリューションを選択します。Kerberos による SSO を構成している場合、「その他」を選択します。

推奨される SSO メカニズムが自動的に選択されます。次の表を参照してください。サポートされている SSO の方法も参照してください。

 ノート:

推奨される SSO メカニズムを使用していない場合、「SSO プロバイダ/エージェント」で「その他」を選択する必要があります。たとえば、SiteMinder の HTTP ヘッダー以外のメカニズムを使用するには、「SSO プロバイダ/エージェント」の「その他」を選択してから、「SSO メカニズム」で使用する SSO メカニズムを選択します。

表 3-5 Web アイデンティティ 管理ソリューションに適した SSO メカニズム

Web アイデンティティ 管理ソリューション	推奨 SSO メカニズム
Oracle Access Manager	カスタム HTTP ヘッダー ¹
OSSO	カスタム HTTP ヘッダー
SiteMinder	カスタム HTTP ヘッダー
Kerberos	HTTP 要求からリモート・ユーザーを取得

¹ デフォルトの HTTP ヘッダー名は、HYPLLOGIN です。カスタム HTTP ヘッダーを使用中の場合、名前を置き換えます。

7. 「OK」をクリックします。

Smart View に対するシングル・サインオンのオプション

Oracle Smart View for Office はシック・クライアントであり、ブラウザではありませんが、HTTP を使用してサーバー・コンポーネントに接続し、システム的にはブラウザのように動作します。Smart View では、ブラウザ・インターフェイスでサポートされるすべての標準的な Web ベースの統合方法がサポートされます。ただし、一部の制限があります:

- Oracle Enterprise Performance Management System コンポーネントに接続されている既存のブラウザ・セッションから Smart View が起動される場合、既存のセッションからの Cookie が共有されないため、ユーザーは Smart View に再度サイン・インする必要があります。
- デフォルトの Oracle Access Manager ログイン・フォームではなくカスタム HTML ベースのログイン・フォームを使用している場合、カスタム・フォームのソースに文字列 loginform が含まれていることを確認してください。これは、Smart View が Oracle Access Manager と統合して動作できるようにするために必要です。

4

ユーザー・ディレクトリの構成

次も参照:

- [ユーザー・ディレクトリおよび EPM System セキュリティ](#)
- [ユーザー・ディレクトリ構成に関連する操作](#)
- [Oracle Identity Manager と EPM System](#)
- [Active Directory の情報](#)
- [OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成](#)
- [リレーショナル・データベースをユーザー・ディレクトリとして構成する](#)
- [ユーザー・ディレクトリの接続のテスト](#)
- [ユーザー・ディレクトリ設定の編集](#)
- [ユーザー・ディレクトリ構成の削除](#)
- [ユーザー・ディレクトリの検索順の管理](#)
- [セキュリティ・オプションの設定](#)
- [暗号化キーの再生成](#)
- [特殊文字の使用](#)

ユーザー・ディレクトリおよび EPM System セキュリティ

Oracle Enterprise Performance Management System 製品は、ユーザー・ディレクトリと総称される多くのユーザーおよびアイデンティティ管理システムでサポートされています。その中には、Sun Java System Directory Server (旧 SunONE Directory Server)、Active Directory など、Lightweight Directory Access Protocol (LDAP)対応のユーザー・ディレクトリが含まれています。また、EPM System は、外部ユーザー・ディレクトリとしてリレーショナル・データベースもサポートします。

通常、EPM System 製品では、プロビジョニングにネイティブ・ディレクトリおよび外部ユーザー・ディレクトリが使用されます。サポートされているユーザー・ディレクトリのリストは、[Oracle Enterprise Performance Management System の動作保証マトリックス](#)を参照してください。

EPM System 製品では、製品にアクセスする各ユーザーにユーザー・ディレクトリ・アカウントが必要です。これらのユーザーは、プロビジョニングを円滑にするようグループに割り当てることができます。ユーザーおよびグループには、EPM System の役割とオブジェクト ACL をプロビジョニングすることができます。管理のオーバーヘッドのため、個別ユーザーのプロビジョニングはお薦めしません。すべての構成済ユーザー・ディレクトリからのユーザーおよびグループは、Oracle Hyperion Shared Services Console に表示されます。

デフォルトで、EPM System コンフィグレータにより、EPM System 製品をサポートする Shared Services リポジトリがネイティブ・ディレクトリとして構成されます。ディレクトリ・マネージャは、Shared Services Console からネイティブ・ディレクトリにアクセスして管理します。

ユーザー・ディレクトリ構成に関連する操作

SSO と承認をサポートするには、システム管理者が外部ユーザー・ディレクトリを構成する必要があります。Oracle Hyperion Shared Services Console から、システム管理者はユーザー・ディレクトリの構成と管理に関連する複数のタスクを実行できます。これらのトピックは、次の手順に示されています。

- ユーザー・ディレクトリの構成:
 - [OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成](#)
 - [リレーショナル・データベースをユーザー・ディレクトリとして構成する](#)
- [ユーザー・ディレクトリの接続のテスト](#)
- [ユーザー・ディレクトリ設定の編集](#)
- [ユーザー・ディレクトリ構成の削除](#)
- [ユーザー・ディレクトリの検索順の管理](#)
- [セキュリティ・オプションの設定](#)

Oracle Identity Manager と EPM System

Oracle Identity Manager は、エンタープライズ・リソース全体でユーザー・アカウントと属性レベルの権限の両方を追加、更新および削除するプロセスを自動化する、役割およびユーザーの管理ソリューションです。Oracle Identity Manager は、スタンドアロン製品として、あるいは Oracle Identity and Access Management Suite Plus の一部として使用できます。

Oracle Enterprise Performance Management System は、LDAP グループであるエンタープライズ・ロールの使用によって Oracle Identity Manager と統合されます。EPM System コンポーネントの役割は、エンタープライズ・ロールに割り当てることができます。Oracle Identity Manager エンタープライズ・ロールに追加されたユーザーまたはグループは、割り当てられている EPM System の役割を自動的に継承します。

たとえば、*Budget Planning* という名前の Oracle Hyperion Planning アプリケーションがあるとします。このアプリケーションをサポートするには、Budget Planning インタラクティブ・ユーザー、Budget Planning エンド・ユーザー、Budget Planning 管理者の 3 つの役割を Oracle Identity Manager で作成します。EPM System の役割をプロビジョニングする際には、プロビジョニング・マネージャに指示して、Oracle Identity Manager のエンタープライズ・ロールに *Budget Planning* およびその他の EPM System コンポーネント(Shared Services など)の必須役割を必ずプロビジョニングさせます。Oracle Identity Manager のエンタープライズ・ロールに割り当てられているユーザーとグループはすべて、EPM System の役割を継承します。Oracle Identity Manager のデプロイと管理の詳細は、Oracle Identity Manager のドキュメントを参照してください。

Oracle Identity Manager と EPM System を統合するには、管理者は次のステップを実行する必要があります。

- EPM System プロビジョニングに使用する予定の Oracle Identity Manager エンタープライズ・ロールのメンバー(ユーザーとグループ)が LDAP 対応のユーザー・ディレクトリ (OID、Active Directory など)で定義されていることを確認します。
- EPM System で、エンタープライズ・ロールのメンバーが定義されている LDAP 対応のユーザー・ディレクトリを外部ユーザー・ディレクトリとして構成します。OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成を参照してください。

Active Directory の情報

この項では、このドキュメントで使用される Microsoft Active Directory の概念について説明します。

DNS 検索とホスト名検索

システム管理者は、Oracle Hyperion Shared Services が静的ホスト名検索または DNS 検索を行って Active Directory を識別できるように Active Directory を構成できます。静的ホスト名検索は Active Directory フェイルオーバーをサポートしません。

高可用性を確保するために Active Directory を複数のドメイン・コントローラに構成するシナリオでは、DNS 検索を使用すると、Active Directory の高可用性が実現されます。DNS 検索を実行するように構成されている場合、Shared Services は登録されているドメイン・コントローラを識別する問合せを DNS サーバーに対して行い、最大の重みのドメイン・コントローラに接続します。Shared Services が接続されているドメイン・コントローラで障害が発生した場合、Shared Services は、次に使用可能な最大の重みのドメイン・コントローラに動的に切り替えます。

ノート:

DNS 検索は、フェイルオーバーをサポートする冗長 Active Directory 設定が使用可能な場合のみ構成できます。詳細は、Microsoft のドキュメントを参照してください。

グローバル・カタログ

グローバル・カタログは、フォレスト内のすべての Active Directory オブジェクトのコピーを保管するドメイン・コントローラです。そのホスト・ドメインのディレクトリ内のその他すべてのドメインのすべてのオブジェクトの完全なコピーおよびフォレスト内のその他すべてのドメインのすべてのオブジェクトの部分コピーを保管し、これらは通常ユーザー検索操作で使用されます。グローバル・カタログの設定については、Microsoft のドキュメントを参照してください。

組織でグローバル・カタログを使用する場合、次の方法のいずれかを使用して、Active Directory を構成します。

- 外部ユーザー・ディレクトリとしてグローバル・カタログ・サーバーを構成する(推奨)。
- 個別の外部ユーザー・ディレクトリとして各 Active Directory ドメインを構成する。

個々の Active Directory ドメインではなく、グローバル・カタログを構成することにより、Oracle Enterprise Performance Management System 製品がフォレスト内のローカルおよびユニバーサル・グループにアクセスできるようになります。

OID、Active Directory およびその他の LDAP ベースのユーザー・ディレクトリの構成

この項で示す手順を使用して、システム管理者は、OID、Sun Java System Directory Server、Oracle Virtual Directory、Active Directory、IBM Tivoli Directory Server などの LDAP ベースの企業ユーザー・ディレクトリを構成するか、あるいは構成画面に示されない LDAP ベースのユーザー・ディレクトリを構成します。

OID、Active Directory および他の LDAP ベースのユーザー・ディレクトリを構成するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。[Shared Services Console の起動](#)を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
「プロバイダ構成」タブが開きます。この画面には、ネイティブ・ディレクトリを含め、すでに構成済のすべてのユーザー・ディレクトリがリストされます。
3. 「新規」をクリックします。
4. 「ディレクトリ・タイプ」で、次のいずれかのオプションを選択します:
 - **Lightweight Directory Access Protocol (LDAP):** Active Directory ではなく、LDAP ベースのユーザー・ディレクトリを構成します。Oracle Virtual Directory を構成するには、このオプションを選択します。
 - **Microsoft Active Directory (MSAD):** Active Directory を構成します。
Active Directory および Active Directory Application Mode (ADAM)のみ:
カスタム ID 属性(ObjectGUID 以外の属性、たとえば sAMAccountName)を Active Directory または ADAM で使用する場合、「**Lightweight Directory Access Protocol (LDAP)**」を選択し、ディレクトリ・タイプ「その他」として構成します。
5. 「次」をクリックします。

The screenshot shows the Oracle Enterprise Performance Management System (EPM) configuration interface. The main window is titled "Configure User Directories" and is divided into three sections: "1. MSAD Connection Information", "2. MSAD User Configuration", and "3. MSAD Group Configuration". The "1. MSAD Connection Information" section is currently active and contains the following fields:

- Directory Server: Microsoft
- Name:
- Host Name: (Radio buttons for DNS Lookup and Host Name are present, with Host Name selected)
- Port: 389
- SSL Enabled:
- Base DN: (Fetch DN button is next to it)
- ID Attribute: objectguid
- Maximum Size: 0
- Trusted:
- Anonymous Bind:
- User DN: (Append Base DN checkbox is next to it)
- Password:

Below the "Server Information" section, there is a "Show Advanced Options" checkbox which is checked. This section contains three sub-sections:

- LDAP Options**
 - Referrals: ignore
 - Dereference Aliases: Always
 - Connection Read Timeout: 60 sec
- Connection Pooling**
 - Max Connections: 100
 - Timeout: 300000 ms
 - Evict Interval: 120 mins
 - Allowed Idle Connection Time: 120 mins
 - Grow Connections:
- Custom Module**
 - Enable Custom Authentication Module:

At the bottom of the configuration window, there are buttons for "Help", "Back", "Next", "Finish", and "Cancel".

6. 必要なパラメータを入力します。

表 4-1 接続情報画面

ラベル	説明
ディレクトリ・サーバー	<p>ユーザー・ディレクトリを選択します。ID 属性値が、選択した製品の推奨される一定の一意のアイデンティティ属性に変わります。</p> <p>ステップ 4 で Active Directory を選択した場合、このプロパティは自動的に選択されます。</p> <p>次のシナリオで、「その他」を選択します。</p> <ul style="list-style-type: none"> リストされていないユーザー・ディレクトリ・タイプ(Oracle Virtual Directory など)を構成しています。 リストされている LDAP 対応ユーザー・ディレクトリ(たとえば OID)を構成していますが、カスタム ID 属性は使用しません。 カスタム ID 属性を使用するように Active Directory または ADAM を構成しています。
名前	<p>例: Oracle Internet Directory</p> <p>ユーザー・ディレクトリのわかりやすい名前。複数のユーザー・ディレクトリが構成されている場合は、特定のユーザー・ディレクトリを識別するために使用します。「名前」には、空白とアンダースコア以外の特殊文字を含めることはできません。</p> <p>例: Corporate_OID</p>

 **ノート:**

Oracle Virtual Directory では、LDAP ディレクトリと RDMBS データ・リポジトリの抽象化が仮想化されて 1 つのディレクトリ・ビューで提供されるため、Oracle Virtual Directory でサポートされるユーザー・ディレクトリの数やタイプに関係なく、Oracle Enterprise Performance Management System では 1 つの外部ユーザー・ディレクトリとみなされます。

表 4-1 (続き) 接続情報画面

ラベル	説明
DNS 検索	<p>Active Directory のみ: このオプションを選択して DNS 検索を使用可能にします。DNS 検索とホスト名検索を参照してください。DNS 検索は、接続が失敗しないように、本番環境での Active Directory への接続方法として構成することをお勧めします。</p> <div style="border: 1px solid #0070C0; padding: 10px; margin: 10px 0;"> <p> ノート:</p> <p>グローバル・カタログを構成している場合は、このオプションを選択しないでください。</p> </div> <p>このオプションを選択すると、次のフィールドが表示されます:</p> <ul style="list-style-type: none"> • ドメイン: Active Directory フォレストのドメイン名です。 例: example.com または us.example.com • AD サイト: Active Directory サイト名で、通常は Active Directory 構成コンテナに保管されているサイト・オブジェクトの相対的な識別名です。一般的に AD サイトにより、市、都道府県、地域や国などの地理的な場所が識別されます。 例: Santa Clara または US_West_region • DNS サーバー: ドメイン・コントローラの DNS サーバー検索をサポートするサーバーの DNS 名。
ホスト名	<p>Active Directory のみ: このオプションを選択して静的なホスト名検索を使用可能にします。DNS 検索とホスト名検索を参照してください。</p> <div style="border: 1px solid #0070C0; padding: 10px; margin: 10px 0;"> <p> ノート:</p> <p>Active Directory グローバル・カタログを構成している場合は、このオプションを選択します。</p> </div>
ホスト名	<p>ユーザー・ディレクトリ・サーバーの DNS 名。SiteMinder から SSO をサポートするためにユーザー・ディレクトリを使用する場合は、完全修飾のドメイン名を使用します。ホスト名は、テスト目的で Active Directory 接続を確立する場合にのみ使用することをお勧めします。</p> <div style="border: 1px solid #0070C0; padding: 10px; margin: 10px 0;"> <p> ノート:</p> <p>Active Directory グローバル・カタログを構成している場合は、グローバル・カタログ・サーバーのホスト名を指定します。グローバル・カタログを参照してください。</p> </div> <p>例: MyServer</p>

表 4-1 (続き) 接続情報画面

ラベル	説明
ポート	ユーザー・ディレクトリが実行するポート番号。
	<p> ノート:</p> <p>Active Directory グローバル・カタログを構成している場合は、グローバル・カタログ・サーバーが使用するポート(デフォルトは 3268)を指定します。グローバル・カタログを参照してください。</p>
	例: 389
SSL 使用可能	このユーザー・ディレクトリとのセキュア通信を使用可能にするチェック・ボックス。ユーザー・ディレクトリは、セキュア通信として構成する必要があります。
ベース DN	ユーザーおよびグループの検索を開始するノードの識別名(DN)。また、「 DN のフェッチ 」ボタンを使用して、使用可能なベース DN のリストを表示し、そのリストから適切なベース DN を選択できます。
	<p> ノート:</p> <p>グローバル・カタログを構成している場合は、フォレストのベース DN を指定します。</p>
	<p>特殊文字の使用上の制限については、特殊文字の使用を参照してください。</p> <p>EPM System 製品のすべてのユーザーとグループを含む最下位の DN を選択することをお勧めします。</p> <p>例: dc=example,dc=com</p>
ID 属性	<p>この属性値は、「ディレクトリ・タイプ」で「その他」が選択されている場合のみ変更できます。この属性はディレクトリ・サーバー上のユーザーおよびグループ・オブジェクトに存在する共通の属性である必要があります。</p> <p>この属性の推奨値は、OID (orclguid)、SunONE (nsuniqueid)、IBM Directory Server (Ibm-entryUuid)、Novell eDirectory (GUID)および Active Directory (ObjectGUID)に自動的に設定されます。</p> <p>例: orclguid</p> <p>「ディレクトリ・サーバー」で「その他」を選択後、ID 属性値を手動で設定する場合(Oracle Virtual Directory を構成する場合など)、ID 属性値は次のようになります:</p> <ul style="list-style-type: none"> 一意の属性を指します 場所に固有ではありません 時間の経過とともに変わりません

表 4-1 (続き) 接続情報画面

ラベル	説明
最大サイズ	<p>検索が戻す結果の最大数。ユーザー・ディレクトリ設定でサポートする値よりもこの値が大きい場合は、ユーザー・ディレクトリ値がこの値をオーバーライドします。</p> <p>Active Directory 以外のユーザー・ディレクトリの場合、このフィールドを空白にすると、検索条件を満たすすべてのユーザーとグループが取得されます。</p> <p>Active Directory の場合、この値を 0 に設定すると、検索条件を満たすすべてのユーザーとグループが取得されます。</p> <p>委任された管理モードで Oracle Hyperion Shared Services を構成している場合は、この値を 0 に設定します。</p>
信頼済	<p>このプロバイダが信頼できる SSO ソースであることを示すチェック・ボックス。信頼できるソースからの SSO トークンにはユーザーのパスワードは含まれません。</p>
匿名のバインド	<p>Shared Services で匿名をユーザー・ディレクトリにバインドしてユーザーおよびグループを検索できることを示すチェック・ボックス。ユーザー・ディレクトリが匿名のバインドを許可する場合のみ使用できます。このオプションを選択しない場合は、ユーザー情報が保管されたディレクトリを検索するのに十分なアクセス権限を持つアカウントをユーザー DN に指定する必要があります。匿名のバインドを使用しないことをお勧めします。</p>
<div style="border: 1px solid #0070C0; padding: 10px; background-color: #E6F2FF;"> <p> ノート:</p> <p>匿名のバインドは OID ではサポートされません。</p> </div>	
ユーザー DN	<p>「匿名のバインド」 が選択されている場合、このオプションは使用不可です。</p> <p>Shared Services がユーザー・ディレクトリとのバインドに使用するユーザーの識別名。このユーザーには DN 内の RDN 属性に対する検索権限が必要です。たとえば、dn: cn=John Doe, ou=people, dc=myCompany, dc=com では、バインド・ユーザーには cn 属性への検索アクセス権が必要です。</p> <p>ユーザー DN の値に特殊文字を指定する場合はエスケープ文字を使用する必要があります。制限については、特殊文字の使用を参照してください。</p> <p>例: cn=admin,dc=myCompany,dc=com</p>
ベース DN の追加	<p>ベース DN をユーザー DN に追加するためのチェック・ボックス。ディレクトリ・マネージャ・アカウントをユーザー DN として使用している場合は、ベース DN を追加しないでください。</p> <p>「匿名のバインド」オプションが選択されている場合、このチェック・ボックスは使用不可です。</p>
パスワード	<p>ユーザー DN パスワードです</p> <p>「匿名のバインド」オプションが選択されている場合、このボックスは使用不可です。</p> <p>例: UserDNpassword</p>
詳細オプションの表示	<p>詳細オプションを表示するチェック・ボックス。</p>

表 4-1 (続き) 接続情報画面

ラベル	説明
参照	Active Directory のみ: 参照に従うように Active Directory が構成されている場合は、「従う」を選択すると、LDAP 参照に自動的に従います。「無視」を選択すると、参照は使用されません。
別名の逆参照	Shared Services の検索で使用するメソッドを選択すると、ユーザー・ディレクトリの別名が逆参照されます。これにより、別名の DN が指すオブジェクトが検索で取得されます。次を選択します: <ul style="list-style-type: none"> • 常時: 常に別名を逆参照します。 • なし: 別名を逆参照しません。 • 検索中: 名前解決の間にのみ別名を逆参照します。 • 検索中: 名前解決の後にのみ別名を逆参照します。
接続読取りタイムアウト	この間隔(秒数)が経過した後も応答がない場合、LDAP プロバイダは LDAP 読取り試行を中止します。 デフォルト: 60 秒
最大接続数	接続プール内の最大接続数。LDAP ベースのディレクトリ (Active Directory を含む)の場合、デフォルトは 100 です。 デフォルト: 100
タイムアウト	プールから接続を取得するまでのタイムアウト。この期間が過ぎると例外が発生します。 デフォルト: 300000 ミリ秒(5 分)
削除間隔	オプション: 削除プロセスを実行してプールを消去するための間隔。削除プロセスによって、「アイドル状態の接続許容時間」を超えたアイドル状態の接続が除去されます。 デフォルト: 120 分
アイドル状態の接続許容時間	オプション: 削除プロセスがプール内のアイドル状態の接続を除去するまでの許容時間。 デフォルト: 120 分
接続の拡大	このオプションは、接続プールが最大接続数を超える接続を保持できるかどうかを示します。デフォルトで選択されています。接続プールが接続を保持できず、接続がタイムアウトに設定された時間内に使用できない場合、システムはエラーを返します。
カスタム認証モジュールを使用可能にする	カスタム認証モジュールの使用を可能にして、このユーザー・ディレクトリで定義されたユーザーを認証するためのチェック・ボックス。認証モジュールの完全修飾 Java クラス名も、「セキュリティ・オプション」画面で入力する必要があります。 セキュリティ・オプションの設定 を参照してください。 カスタム認証モジュールの認証は、シン・クライアントおよびシック・クライアントに対して透過的で、クライアントのデプロイメント変更は必要ありません。 <i>Oracle Enterprise Performance Management System セキュリティ構成ガイド</i> のカスタム認証モジュールの使用を参照してください。

7. 「次」をクリックします。

Shared Services は、「ユーザー構成」画面に設定されたプロパティを使用して、ユーザーの検索を開始するノードの特定に利用されるユーザー URL を作成します。この URL を使用すると、検索効率が向上します。

▲ 注意:

ユーザー URL は別名をポイントできません。EPM System のセキュリティでは、ユーザー URL が実際のユーザーをポイントすることを求められます。

画面の「自動構成」領域を使用して、必要な情報を取得することをお勧めします。

✎ ノート:

ユーザー構成で使用できる特殊文字のリストについては、[特殊文字の使用](#)を参照してください。

8. 「自動構成」に、フォーマット `attribute=identifier` を使用して、一意のユーザー識別子を入力します。例: `uid=jdoe`。

ユーザーの属性は、「ユーザー構成」領域に表示されます。

OID を構成している場合は、OID のルート DSE がネーミング・コンテキスト属性内にエントリを含まないため、ユーザー・フィルタを自動的に構成できません。[Oracle Fusion Middleware Oracle Internet Directory 管理者ガイドのネーミング・コンテキストの管理](#)を参照してください。

 ノート:

「ユーザー構成」領域のテキスト・ボックスに、必要なユーザー属性を手動で入力できます。

表 4-2 ユーザー構成画面

ラベル	説明 ¹
ユーザー RDN	ユーザーの相対的な識別名。DN の各コンポーネントは RDN と呼ばれ、ディレクトリ・ツリー内の分岐を表します。ユーザーの RDN は一般に、uid または cn と同じです。 制限については、 特殊文字の使用 を参照してください。 例: ou=People
ログイン属性	ユーザーのログオン名を保管する一意の属性(カスタム属性も可能)。ユーザーは、EPM System 製品にログインするとき、この属性の値をユーザー名として使用します。 ユーザー ID (「ログイン属性」の値)は、すべてのユーザー・ディレクトリにわたって一意である必要があります。たとえば、SunONE 構成と Active Directory 構成の「ログイン属性」として、それぞれ uid と sAMAccountName を使用できます。これらの属性の値は、ネイティブ・ディレクトリを含むすべてのユーザー・ディレクトリにわたって一意である必要があります。
	<div data-bbox="742 1079 894 1119" data-label="Section-Header"> ノート:</div> <p>ユーザー ID では、大文字と小文字が区別されません。</p>

表 4-2 (続き) ユーザー構成画面

ラベル	説明 ¹
オブジェクト・クラス	ユーザーのオブジェクト・クラス(ユーザーに関連付けられる必須とオプションの属性)。Shared Services は、この画面に表示されたオブジェクト・クラスを検索フィルタで使用します。これらのオブジェクト・クラスを使用して、Shared Services は、プロビジョニングされたすべてのユーザーを検索する必要があります。

 **ノート:**

ユーザー・ディレクトリ・タイプ「その他」として Active Directory または ADAM を、カスタム ID 属性を使用するように構成している場合、この値を user に設定する必要があります。

オブジェクト・クラスは、必要に応じて手動で追加できます。オブジェクト・クラスを追加するには、「**オブジェクト・クラス**」ボックスにオブジェクト・クラス名を入力し、「**追加**」をクリックします。

オブジェクト・クラスを削除するには、オブジェクト・クラスを選択し、「**削除**」をクリックします。

デフォルト

- **Active Directory:** user
- **Active Directory 以外の LDAP ディレクトリ:** person, organizationalPerson, inetorgperson

ユーザーを制限するフィルタ

EPM System 製品の役割がプロビジョニングされるユーザーのみを取得する LDAP 問合せ。たとえば、LDAP 問合せ (uid=Hyp*) は、名前が Hyp で始まるユーザーのみを取得します。

ユーザー構成画面はユーザー RDN を検証します。必要な場合は、ユーザー・フィルタの使用をお勧めします。

ユーザー・フィルタは、問合せで戻されるユーザー数を制限します。ユーザー RDN によって識別されるノードが、プロビジョニングされる必要のない多くのユーザーを含む場合に特に重要です。ユーザー・フィルタは、プロビジョニングされる必要のないユーザーを除外するために使用できます。これにより、パフォーマンスが向上します。

表 4-2 (続き) ユーザー構成画面

ラベル	説明 ¹
複数属性の RDN 用のユーザー検索属性	<p>Active Directory 以外の LDAP 対応ユーザー・ディレクトリのみ: ディレクトリ・サーバーが複数属性の RDN を使用するよう構成される場合にのみ、この値を設定します。設定した値はいずれかの RDN 属性である必要があります。指定した属性の値は一意で、属性は検索可能である必要があります。たとえば、SunONE ディレクトリ・サーバーが、cn (cn=John Doe) および uid (uid=jDoe12345) 属性を組み合わせるように構成され、次のような複数属性の RDN を作成するとします:</p> <pre>cn=John Doe+uid=jDoe12345, ou=people, dc=myCompany, dc=com</pre> <p>この場合、これらの属性が次の条件を満たしている場合には、cn または uid のいずれかを使用できます:</p> <ul style="list-style-type: none"> この属性は「接続情報」タブにファイルされたユーザー DN で識別されたユーザーにより検索可能です この属性はユーザー・ディレクトリ全体で一意の値に設定する必要があります
カスタム・プライマリ・グループの解決	<p>Active Directory のみ: 効果的な役割を決定するためにユーザーのプライマリ・グループを識別するかどうかを示すチェック・ボックスです。このチェック・ボックスはデフォルトで選択されています。この設定は変更しないことをお勧めします。</p>
ユーザー・パスワードの期限が次の日数以内に切れ る場合に警告を表示	<p>Active Directory のみ: Active Directory ユーザーのパスワードが指定した日数以内に期限切れになる場合に警告メッセージを表示するかどうかを示すチェック・ボックス。</p>

¹ EPM System セキュリティでは、構成値がオプションの一部のフィールドにデフォルト値が使用されます。そのようなフィールドに値を入力しない場合、デフォルト値が実行時に使用されます。

9. 「次」をクリックします。

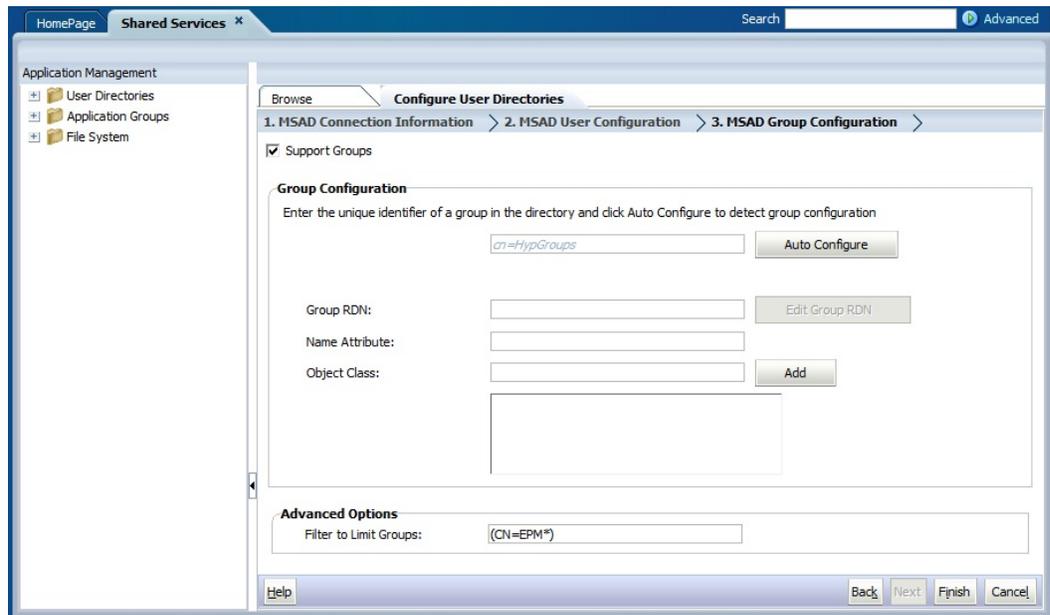
「グループ構成」画面が開きます。Shared Services は、この画面に設定されたプロパティを使用して、グループの検索を開始するノードを特定するグループ URL を作成します。この URL を使用すると、検索効率が向上します。

▲ 注意:

グループ URL は別名をポイントできません。EPM System のセキュリティでは、グループ URL が実際のグループをポイントすることを求められます。グループの別名を使用する Novell eDirectory を構成している場合、グループ URL 内でグループの別名とグループ・アカウントを使用できる必要があります。

 **ノート:**

「グループ構成」画面のデータ入力はオプションです。グループ URL の設定を入力しない場合、Shared Services は、ベース DN 内を検索してグループを見つけます。これは特に、ユーザー・ディレクトリに多くのユーザーが含まれている場合に悪影響をパフォーマンスに及ぼします。



10. 組織で、グループのプロビジョニングを予定していない場合、またはユーザーがユーザー・ディレクトリでグループに分類されない場合は、「**グループのサポート**」をクリアします。このオプションをクリアすると、この画面のフィールドは使用不可になります。

グループをサポートしている場合は、自動構成機能を使用して、必要な情報を取得することをお勧めします。

OID をユーザー・ディレクトリとして構成している場合は、自動構成機能を使用できません。理由は、OID のルート DSE がネーミング・コンテキスト属性内にエントリを含まないからです。Oracle Fusion Middleware Oracle Internet Directory 管理者ガイドの [ネーミング・コンテキストの管理](#) を参照してください。

11. 「**自動構成**」テキスト・ボックスに、一意のグループ識別子を入力し、「**検索**」をクリックします。

グループ識別子は、フォーマット `attribute=identifier` で指定する必要があります。
例: `cn=western_region`

グループの属性は、「グループ構成」領域に表示されます。

 **ノート:**

必要なグループ属性は、「グループ構成」テキスト・ボックスに入力できます。

▲ 注意:

ノード名に / (スラッシュ) または \ (円記号(バックスラッシュ)) を含むユーザー・ディレクトリにグループ URL が設定されていない場合、ユーザーおよびグループの検索は失敗します。たとえば、ユーザーおよびグループが存在するノード内(OU=child\ou, OU=parent/ou または OU=child/ou, OU=parent \ ou など)のユーザー・ディレクトリにグループ URL が指定されていない場合、ユーザーまたはグループを表示する操作は失敗します。

表 4-3 グループ構成画面

ラベル	説明 ¹
グループ RDN	<p>グループの相対 DN。この値は、ベース DN の相対パスで、グループ URL として使用されます。</p> <p>グループ RDN を指定します。これにより、プロビジョニングする予定のすべてのグループが使用可能な最下位のユーザー・ディレクトリ・ノードが識別されます。</p> <p>プロビジョニングに Active Directory プライマリ・グループを使用する場合、プライマリ・グループがグループ RDN 下にあることを確認します。Shared Services では、グループ URL のスコープ外のプライマリ・グループは取得されません。</p> <p>グループ RDN はログインと検索のパフォーマンスに重大な影響を及ぼします。グループ RDN はすべてのグループ検索の開始点であるため、EPM System 製品のすべてのグループが使用可能な最下位ノードを識別する必要があります。最適なパフォーマンスを保証するには、グループ RDN 内に存在するグループのメンバーが 10,000 を超えないようにする必要があります。これより多くのグループが存在する場合は、グループ・フィルタを使用して、プロビジョニングするグループのみを取得します。</p>
名前属性	<p>制限については、特殊文字の使用を参照してください。</p> <p>例: ou=Groups</p> <p>グループの名前を保管する属性</p> <p>デフォルト</p> <ul style="list-style-type: none"> • Active Directory を含む LDAP ディレクトリ: cn • ネイティブ・ディレクトリ: cssDisplayNameDefault

✎ ノート:

グループ URL 内の使用可能なグループ数が **10,000** を超えると、**Shared Services** は警告を表示します。

表 4-3 (続き) グループ構成画面

ラベル	説明 ¹
オブジェクト・クラス	<p>グループのオブジェクト・クラス。Shared Services は、この画面に表示されたオブジェクト・クラスを検索フィルタで使用します。これらのオブジェクト・クラスを使用して、Shared Services は、ユーザーに関連付けられたすべてのグループを検索する必要があります。</p> <div style="border: 1px solid #0070C0; padding: 10px; margin: 10px 0;"> <p> ノート:</p> <p>ユーザー・ディレクトリ・タイプ「その他」として Active Directory または ADAM を、カスタム ID 属性を使用するように構成している場合、この値を <code>group?member</code> に設定する必要があります。</p> </div> <p>オブジェクト・クラスは、必要に応じて手動で追加できます。オブジェクト・クラスを追加するには、「オブジェクト・クラス」テキスト・ボックスにオブジェクト・クラス名を入力し、「追加」をクリックします。</p> <p>オブジェクト・クラスを削除するには、オブジェクト・クラスを選択し、「削除」をクリックします。</p> <p>デフォルト</p> <ul style="list-style-type: none"> • Active Directory: <code>group?member</code> • Active Directory 以外の LDAP ディレクトリ: <code>groupofuniquenames?uniquemember,</code> <code>groupOfNames?member</code> • ネイティブ・ディレクトリ:: <code>groupofuniquenames?uniquemember, cssGroupExtend?cssIsActive</code>
グループを制限するフィルタ	<p>EPM System 製品の役割がプロビジョニングされるグループのみを取得する LDAP 問合せ。たとえば、LDAP 問合せ (<code> (cn=Hyp*) (cn=Admin*)</code>) は、名前が Hyp または Admin で始まるグループのみを取得します。</p> <p>グループ・フィルタは、問合せで戻されるグループ数を制限するために使用します。グループ RDN によって識別されるノードが、プロビジョニングされる必要のない多くのグループを含む場合に特に重要です。フィルタは、プロビジョニングされる必要のないグループを除外するために使用できます。これにより、パフォーマンスが向上します。</p> <p>プロビジョニングに Active Directory プライマリ・グループを使用する場合、設定したグループ・フィルタがグループ URL のスコープ内に含まれるプライマリ・グループを取得できることを確認します。たとえば、フィルタ (<code> (cn=Hyp*) (cn=Domain Users)</code>) は、名前が Hyp で始まるグループと Domain Users という名前のプライマリ・グループを取得します。</p>

¹ EPM System セキュリティでは、構成値がオプションの一部のフィールドにデフォルト値が使用されます。そのようなフィールドに値を入力しない場合、デフォルト値が実行時に使用されます。

12. 「終了」をクリックします。

Shared Services は構成を保存して、「定義済ユーザー・ディレクトリ」画面に戻ります。この画面には、今構成したユーザー・ディレクトリが表示されます。

13. 構成をテストします。[ユーザー・ディレクトリの接続のテスト](#)を参照してください。
14. 必要に応じて、検索順の割当てを変更します。詳細は、[ユーザー・ディレクトリの検索順の管理](#)を参照してください。
15. 必要に応じて、セキュリティ・オプションを指定します。詳細は、[セキュリティ・オプションの設定](#)を参照してください。
16. Oracle Hyperion Foundation Services とその他の EPM System コンポーネントを再起動します。

リレーショナル・データベースをユーザー・ディレクトリとして構成する

Oracle、SQL Server、および IBM DB2 リレーショナル・データベースのシステム表からのユーザーおよびグループ情報を使用して、プロビジョニングをサポートできます。グループ情報がデータベースのシステム・スキーマから取得できない場合、Oracle Hyperion Shared Services はそのデータベース・プロバイダからのグループのプロビジョニングはサポートしません。たとえば、Shared Services は、データベースがオペレーティング・システム上で定義されているグループを使用するため、古いバージョンの IBM DB2 からグループ情報を抽出できません。ただし、プロビジョニング・マネージャはネイティブ・ディレクトリのグループにこれらのユーザーを追加して、このグループをプロビジョニングできます。サポートされているプラットフォーム情報は、Oracle Technology Network (OTN)の [Oracle Fusion Middleware Supported System Configurations](#) ページに掲載されている *Oracle Enterprise Performance Management System* の動作保証マトリックスを参照してください。

ノート:

DB2 データベースを使用する場合、ユーザー名は 8 文字以上にする必要があります。Oracle および SQL Server データベースの場合は 256 文字、DB2 の場合は 1000 文字を超えないようにしてください。

ユーザーおよびグループのリストを取得するには、データベース管理者、たとえば、Oracle SYSTEM ユーザーとしてデータベースに接続できるように Shared Services を構成します。

ノート:

Shared Services は、プロビジョニングに対してアクティブなデータベース・ユーザーのみ取得します。非アクティブでロックされているデータベース・ユーザー・アカウントは無視されます。

データベース・プロバイダを構成するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。
Shared Services Console の起動を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 「新規」をクリックします。
4. 「ディレクトリ・タイプ」画面で、「リレーショナル・データベース(Oracle、DB2、SQL Server)」を選択します。
5. 「次」をクリックします。

The screenshot shows a configuration window titled 'Configure User Directories' with two tabs: '1. Database Configuration' and '2. Advanced Database Configuration'. The 'Advanced Database Configuration' tab is active. It contains the following fields:

- Database Type: Oracle (dropdown menu)
- Name: (text input field)
- Server: (text input field)
- Port: 1521 (text input field)
- Service/SID: (text input field)
- User Name: (text input field)
- Password: (text input field)
- Trusted:

At the bottom of the window, there are buttons for 'Help', 'Back', 'Next', 'Finish', and 'Cancel'.

6. 「データベースの構成」タブで、構成パラメータを入力します。

表 4-4 「データベース構成」タブ

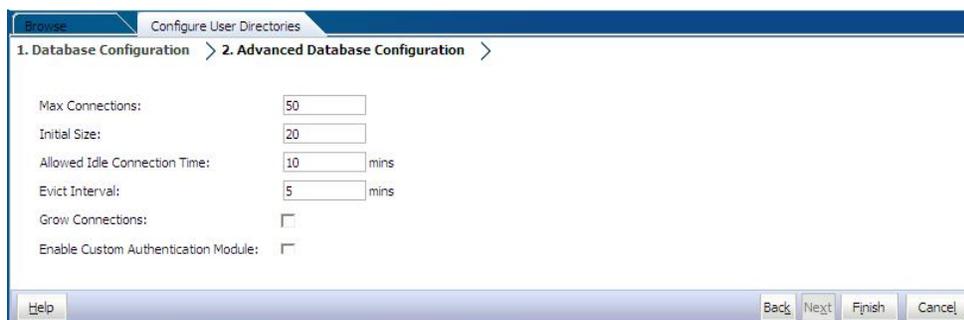
ラベル	説明
データベース・タイプ	リレーショナル・データベース・プロバイダ。Shared Services は、データベース・プロバイダとして Oracle および SQL Server データベースのみサポートしています。 例: Oracle
名前	データベース・プロバイダの一意の構成名。 例: Oracle_DB_FINANCE
サーバー	データベース・サーバーが稼働しているコンピュータの DNS 名。 例: myserver
ポート	データベース・サーバーのポート番号 例: 1521
サービス/SID (Oracle のみ)	システム識別子(デフォルトは orcl) 例: orcl
データベース(SQL Server および DB2 のみ)	Shared Services が接続する必要があるデータベース 例: master
ユーザー名	Shared Services がデータベースへのアクセスに使用するユーザー名。このデータベース・ユーザーには、データベース・システム表へのアクセス権が必要です。Oracle データベースには system アカウント、SQL Server データベースにはデータベース管理者のユーザー名を使用することをお勧めします。 例: SYSTEM

表 4-4 (続き) 「データベース構成」 タブ

ラベル	説明
パスワード	「 ユーザー名 」でユーザーを識別するパスワード。 例: system_password
信頼済	このプロバイダが信頼できる SSO ソースであることを指定するチェック・ボックス。信頼できるソースからの SSO トークンにはユーザーのパスワードは含まれません。

7. **オプション:** 接続プールを構成するには、「次へ」をクリックします。

「詳細なデータベース構成」タブが開きます。



8. 「詳細なデータベース構成」タブで、接続プールのパラメータを入力します。

表 4-5 「詳細なデータベース構成」 タブ

ラベル	説明
最大接続数	プールの最大接続数。デフォルトは 50 です。
初期サイズ	プールを初期化する場合に使用可能な接続数。デフォルトは 20 です。
アイドル状態の接続許容時間	オプション: 削除プロセスがプール内のアイドル状態の接続を除去するまでの許容時間。デフォルトは 10 分です。
削除間隔	オプション: プールを消去するために削除プロセスを実行する間隔。削除はアイドル状態の接続許容時間を超えたアイドル接続を除去します。デフォルトは 5 分です。
接続の拡大	接続プールが最大接続数を超える接続を保持できるかどうかを示します。デフォルトでは、このオプションはクリアされており、接続は保持できないことを示します。接続プールが接続を保持できず、接続がタイムアウトに設定された時間内に使用できない場合、システムはエラーを返します。
カスタム認証モジュールを使用可能にする	カスタム認証モジュールの使用を使用可能にして、このユーザー・ディレクトリで定義されたユーザーを認証するためのチェック・ボックス。認証モジュールの完全修飾 Java クラス名も、「セキュリティ・オプション」画面で入力する必要があります。 セキュリティ・オプションの設定 を参照してください。 カスタム認証モジュールの認証は、シン・クライアントおよびシック・クライアントに対して透過的に行われます。 Oracle Enterprise Performance Management System セキュリティ構成ガイド のカスタム認証モジュールの使用を参照してください。

9. 「終了」をクリックします。
10. 「定義済ユーザー・ディレクトリ」画面に戻るには、「OK」をクリックします。
11. データベース・プロバイダ構成をテストします。[ユーザー・ディレクトリの接続のテスト](#)を参照してください。
12. 必要に応じて、検索順の割当てを変更します。詳細は、[ユーザー・ディレクトリの検索順の管理](#)を参照してください。
13. 必要に応じて、セキュリティ設定を指定します。[セキュリティ・オプションの設定](#)を参照してください。
14. Oracle Hyperion Foundation Services およびその他の Oracle Enterprise Performance Management System コンポーネントを再起動します。

ユーザー・ディレクトリの接続のテスト

ユーザー・ディレクトリの構成後、Oracle Hyperion Shared Services が現在の設定を使用してユーザー・ディレクトリに接続できることを確認するため、接続をテストします。

ユーザー・ディレクトリ接続をテストするには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。[Shared Services Console の起動](#)を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. ユーザー・ディレクトリのリストから、テストする外部ユーザー・ディレクトリ構成を選択します。
4. 「テスト」、「OK」の順にクリックします。

ユーザー・ディレクトリ設定の編集

管理者は名前以外のユーザー・ディレクトリ構成のパラメータを変更できます。プロビジョニング用に使用されていたユーザー・ディレクトリの構成データは編集しないことをお勧めします。

▲ 注意:

たとえば、ユーザー・ディレクトリ構成の ID 属性などのいくつかの設定を編集すると、プロビジョニング・データが使用不可になります。プロビジョニングされたユーザー・ディレクトリの設定を変更する場合は、十分注意してください。

ユーザー・ディレクトリ構成を編集するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。[Shared Services Console の起動](#)を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 編集するユーザー・ディレクトリを選択します。
4. 「編集」をクリックします。

5. 構成設定を変更します。

 **ノート:**

構成名は変更できません。LDAP ユーザー・ディレクトリ構成を変更する場合、「ディレクトリ・サーバー」リストから別のディレクトリ・サーバーや「その他」(カスタム LDAP ディレクトリの場合)を選択できます。ネイティブ・ディレクトリ・パラメータは編集できません。

編集可能なパラメータの説明については、次の表を参照してください:

- **Active Directory** およびその他の LDAP ベースのユーザー・ディレクトリ、**OID、Active Directory** およびその他の LDAP ベースのユーザー・ディレクトリの構成の表を参照してください。
 - データベース: **リレーショナル・データベースをユーザー・ディレクトリとして構成する**の表を参照してください
6. 「OK」をクリックして、変更を保存します。

ユーザー・ディレクトリ構成の削除

システム管理者はユーザー・ディレクトリ構成をいつでも削除できます。構成を削除すると、ユーザー・ディレクトリから取得されたユーザーおよびグループのプロビジョニング情報がすべて使用不可になり、検索順からディレクトリが除去されます。

 **ヒント:**

プロビジョニング用に使用された構成済のユーザー・ディレクトリを使用しない場合、ユーザーおよびグループの検索に使用されないように検索順から除去します。このアクションにより、プロビジョニング情報の整合性を維持し、後でユーザー・ディレクトリを使用できます。

ユーザー・ディレクトリ構成を削除するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。**Shared Services Console の起動**を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. ディレクトリを選択します。
4. 「削除」をクリックします。
5. 「OK」をクリックします。
6. 再度「OK」をクリックします。
7. Oracle Hyperion Foundation Services およびその他の Oracle Enterprise Performance Management System コンポーネントを再起動します。

ユーザー・ディレクトリの検索順の管理

システム管理者が外部ユーザー・ディレクトリを構成すると、Oracle Hyperion Shared Services により自動的にユーザー・ディレクトリが検索順に追加され、ネイティブ・ディレクトリの検索順より上位の次の使用可能な検索順が割り当てられます。検索順は、Oracle Enterprise Performance Management System でユーザーとグループについて検索する際、構成されたユーザー・ディレクトリ間を循環するために使用されます。

システム管理者はユーザー・ディレクトリを検索順から除去できます。この場合、Shared Services により残りのディレクトリの検索順が自動的に再割当てされます。検索順に含まれないユーザー・ディレクトリは、認証およびプロビジョニングのサポートに使用されません。

ノート:

Shared Services は、指定されたアカウントを検出するとユーザーまたはグループの検索を停止します。EPM System ユーザーの大部分が存在する企業ディレクトリを検索順の一番上に配置することをお勧めします。

デフォルトでは、ネイティブ・ディレクトリは検索順の最後のディレクトリとして設定されます。管理者は検索順を管理するために、次のタスクを実行できます。

- [ユーザー・ディレクトリの検索順への追加](#)
- [検索順の変更](#)
- [検索順の割当ての除去](#)

ユーザー・ディレクトリの検索順への追加

新規に構成されたユーザー・ディレクトリは、検索順に自動的に追加されます。検索順からディレクトリを除去した場合、検索順の最後にそれを追加できます。

検索順にユーザー・ディレクトリを追加するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。
[Shared Services Console の起動](#)を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 検索順に追加する非アクティブなユーザー・ディレクトリを選択します。
4. 「含む」をクリックします。
このボタンは、検索順にないユーザー・ディレクトリを選択している場合のみ使用可能です。
5. 「定義済ユーザー・ディレクトリ」画面に戻るには、「OK」をクリックします。
6. Oracle Hyperion Foundation Services とその他の EPM System コンポーネントを再起動します。

検索順の割当ての除去

検索順からユーザー・ディレクトリを除去してもディレクトリ構成が無効にならず、ユーザー認証のために検索されるディレクトリのリストからユーザー・ディレクトリが除去されま

す。検索順に含まれないディレクトリは、「非アクティブ」ステータスに設定されます。管理者が検索順からユーザー・ディレクトリを除去すると、他のユーザー・ディレクトリに割り当てられている検索順は自動的に更新されます。

ノート:

ネイティブ・ディレクトリは検索順から削除できません。

検索順からユーザー・ディレクトリを除去するには:

1. システム管理者として **Shared Services Console** にアクセスします。 **Shared Services Console** の起動を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 検索順から除去するディレクトリを選択します。
4. 「除外」をクリックします。
5. 「OK」をクリックします。
6. 「ディレクトリの構成結果」画面で「OK」をクリックします。
7. **Foundation Services** とその他の **EPM System** コンポーネントを再起動します。

検索順の変更

各ユーザー・ディレクトリに割り当てられているデフォルトの検索順は、ディレクトリが構成されたシーケンスに基づきます。デフォルトでは、ネイティブ・ディレクトリは検索順の最後のディレクトリとして設定されます。

検索順を変更するには:

1. システム管理者として **Shared Services Console** にアクセスします。 **Shared Services Console** の起動を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 検索順を変更するユーザー・ディレクトリを選択します。
4. 「上へ移動」または「下へ移動」をクリックします。
5. 「OK」をクリックします。
6. **Foundation Services** とその他の **EPM System** コンポーネント、および **Shared Services** セキュリティ API を使用するカスタム・アプリケーションを再起動します。

セキュリティ・オプションの設定

セキュリティ・オプションは、検索順に含まれるすべてのユーザー・ディレクトリに適用可能なグローバル・パラメータから構成されています。

セキュリティ・オプションを設定するには:

1. システム管理者として **Oracle Hyperion Shared Services Console** にアクセスします。 **Shared Services Console** の起動を参照してください。

2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 「セキュリティ・オプション」を選択します。
4. 「セキュリティ・オプション」では、グローバル・パラメータを設定します。

表 4-6 ユーザー・ディレクトリ用のセキュリティ・オプション

パラメータ	説明
トークンのタイムアウト	Oracle Enterprise Performance Management System 製品または Web アイデンティティ 管理ソリューションによって発行された SSO トークンが期限切れになるまでの時間(分単位)。ユーザーは、この期間が過ぎてからログインする必要があります。トークンのタイムアウトは、サーバーのシステム・クロックに基づいて設定されます。デフォルトは 480 分です。

 **ノート:**

トークンのタイムアウトは、セッションのタイムアウトとは異なります。

キャッシュのリフレッシュ
間隔

Oracle Hyperion Shared Services でグループとユーザーの関係データのキャッシュをリフレッシュする間隔(分単位)。デフォルトは 60 分です。

Shared Services では、次のキャッシュ・リフレッシュ後にのみ、新しい外部ユーザー・ディレクトリ・グループと、既存のグループに追加された新しいユーザーに関する情報がキャッシュされます。新規に作成された外部ユーザー・ディレクトリ・グループを通じてプロビジョニングされたユーザーの役割は、キャッシュがリフレッシュされるまでプロビジョニングされません。

表 4-6 (続き) ユーザー・ディレクトリ用のセキュリティ・オプション

パラメータ	説明
今すぐリフレッシュ	グループを含む Shared Services キャッシュのユーザー関係データへのリフレッシュを手動で開始するには、このボタンをクリックします。外部ユーザー・ディレクトリに新規グループを作成し、それらをプロビジョニングした後、または新規ユーザーを既存のグループに追加した後に、キャッシュ・リフレッシュを開始することが必要な場合があります。キャッシュは、 Shared Services によってキャッシュ内のデータを使用する呼出しが行われた後にのみリフレッシュされます。
SSO 互換性の使用可能化	デプロイメントが Oracle Business Intelligence Enterprise Edition リリース 11.1.1.5 以前と統合した場合は、このオプションを選択します。
委任されたユーザー管理モードを使用可能にする	EPM System 製品の委任されたユーザー管理を使用可能にし、配布されたプロビジョニング・アクティビティの管理をサポートするオプション。 Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド の委任されたユーザー管理を参照してください。
SSO の使用可能	Oracle Access Manager などのセキュリティ・エージェントからの SSO のサポートを使用可能にするオプション
SSO プロバイダ/エージェント	EPM System 製品が SSO を受け入れる必要のある Web アイデンティティ管理ソリューションを選択します。 Web アイデンティティ管理ソリューション(Kerberos など)がリストされていない場合、「 その他 」を選択します。 SSO プロバイダを選択すると、希望する SSO メカニズムと名前が自動的に選択されます。必要に応じて、SSO メカニズム(HTTP ヘッダーまたはカスタム・ログイン・クラス)の名前を変更できます。 SSO プロバイダまたはエージェントとして「その他」を選択した場合、 EPM System のサポートする SSO メカニズムをサポートすることを確認する必要があります。 Oracle Enterprise Performance Management System セキュリティ構成ガイド のサポートされている SSO の方法を参照してください。
SSO メカニズム	ユーザーのログイン名を EPM System 製品に提供するために選択した Web アイデンティティ管理ソリューションで使用されるメソッド。使用可能な SSO メソッドの説明は、 Oracle Enterprise Performance Management System セキュリティ構成ガイド のサポートされている SSO の方法を参照してください。 <ul style="list-style-type: none"> カスタム HTTP ヘッダー: セキュリティ・エージェントが EPM System に渡すヘッダーの名前を設定します。 カスタム・ログイン・クラス: 認証用の HTTP 要求を処理するカスタム Java クラスを指定します。Oracle Enterprise Performance Management System セキュリティ構成ガイドのカスタム・ログイン・クラスを参照してください。

 ノート:

カスタム・ログイン・クラスは、カスタム認証と同じではありません。

- HTTP 認証ヘッダー: 標準 **HTTP** メカニズム。
- HTTP 要求からリモート・ユーザーを取得: セキュリティ・エージェントによって **HTTP** 要求にリモート・ユーザーが挿入される場合、このオプションを選択します。

表 4-6 (続き) ユーザー・ディレクトリ用のセキュリティ・オプション

パラメータ	説明
カスタム認証モジュール	<p>認証モジュールで、カスタム認証モジュールが選択されているすべてのユーザー・ディレクトリでユーザーの認証に使用される必要があるカスタム認証モジュールの完全修飾 Java クラス名(たとえば、com.mycompany.epm.CustomAuthenticationImpl)。</p> <p>認証モジュールは、ディレクトリ構成で使用可能(デフォルト)である場合にのみ、ユーザー・ディレクトリに使用されます。</p> <p>Oracle Hyperion Foundation Services では、カスタム認証 JAR ファイルの名前が CustomAuth.jar である必要があります。</p> <p>CustomAuth.jar は、 MIDDLEWARE_HOME\user_projects\domains\WEBLOGIC_DOMAIN\lib (通常は C:\Oracle\Middleware\user_projects\domains\EPMSys\lib)に存在する必要があります。</p> <p>いずれのクライアント・インストールにおいても、CustomAuth.jar は EPM_ORACLE_HOME/common/jlib/11.1.2.0 (通常は C:\Oracle\Middleware\EPMSys11R1\common\jlib\11.1.2.0)に存在する必要があります。</p> <p>JAR ファイル内では任意のパッケージ構成およびクラス名を使用できます。</p> <p>詳細は、<i>Oracle Enterprise Performance Management System セキュリティ構成ガイド</i>のカスタム認証モジュールの使用を参照してください。</p>

5. 「OK」をクリックします。
6. Foundation Services とその他の EPM System コンポーネントを再起動します。

暗号化キーの再生成

Oracle Enterprise Performance Management System では、次のキーを使用してセキュリティが確保されます:

- シングル・サインオン・トークン暗号化キー。EPM System SSO トークンの暗号化と復号化に使用されます。このキーは、Oracle Hyperion Shared Services レジストリに保管されます。
- 信頼できるサービス・キー。EPM System コンポーネントで、SSO トークンを要求しているサービスの認証の確認に使用されます
- プロバイダ構成暗号化キー。EPM System セキュリティで、構成されている外部ユーザー・ディレクトリとのバインドに使用されるパスワード(LDAP 対応ユーザー・ディレクトリのユーザー DN パスワード)の暗号化に使用されます。このパスワードは、外部ユーザー・ディレクトリの構成時設定されます。

EPM System セキュリティを強化するために、これらのキーを定期的に変更します。Oracle Hyperion Shared Services および EPM System のセキュリティ・サブシステムでは、128 ビットのキーの強度の AES 暗号化が使用されます。

▲ 注意:

Oracle Hyperion Financial Management および Oracle Hyperion Profitability and Cost Management で使用されるタスクフローは、シングル・サインオン暗号化キーを再生成すると無効化されます。キーを再生成した後に、タスクフローを開いて保存して再度有効にします。

シングル・サインオン暗号化キー、プロバイダ構成キーまたは信頼できるサービス・キーを再生成するには:

1. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。Shared Services Console の起動を参照してください。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 「暗号化オプション」を選択します。
4. 「暗号化オプション」で、再生成するキーを選択します。

表 4-7 EPM System の暗号化オプション

オプション	説明
シングル・サインオン・トークン	<p>EPM System SSO トークンの暗号化と復号化に使用される暗号化キーを再生成する場合に選択します。</p> <p>「セキュリティ・オプション」で「SSO 互換性の使用可能化」が設定されている場合、次のいずれかのボタンを選択します。</p> <ul style="list-style-type: none"> • 新規 SSO トークン暗号化キーを作成する場合、「新しいキーの生成」 • デフォルトの SSO トークン暗号化キーをリストアする場合、「デフォルトにリセット」
信頼できるサービス・キー	<p>EPM System コンポーネントで、SSO トークンを要求しているサービスの認証の確認に使用される信頼できる認証キーを再生成する場合、このオプションを選択します。</p>
プロバイダ構成キー	<p>EPM System セキュリティで、構成されている外部ユーザー・ディレクトリとのバインドに使用されるパスワード(LDAP 対応ユーザー・ディレクトリのユーザー DN パスワード)の暗号化に使用されるキーを再生成する場合、このオプションを選択します。このパスワードは、外部ユーザー・ディレクトリの構成時設定されます。</p>

✎ ノート:

デフォルトの暗号化キーに戻す場合は、既存のキーストア・ファイル(EPM_ORACLE_HOME/common/CSS/ssHandlerTK)を、すべての EPM System ホスト・マシンから削除する必要があります。

5. 「OK」をクリックします。
6. SSO 暗号化キーを新たに生成する場合、このステップを完了させます。

- a. 「ダウンロード」をクリックします。
 - b. 「OK」をクリックして、Oracle Hyperion Foundation Services をホストするサーバーのフォルダに ssHandlerTK (新規 SSO 暗号化キーをサポートするキーストア・ファイル)を保存します。
 - c. ssHandlerTK をすべての EPM System ホスト・マシン上の `EPM_ORACLE_HOME/common/CSS` にコピーします。
7. Foundation Services とその他の EPM System コンポーネントを再起動します。

特殊文字の使用

Active Directory およびその他の LDAP ベースのユーザー・ディレクトリでは、DN、ユーザー名、役割およびグループ名などのエンティティで特殊文字が使用可能です。このような文字を理解させるには、Oracle Hyperion Shared Services に対して特別な処理が必要になる場合があります。

通常、ユーザー・ディレクトリ設定(ベース DN やユーザーおよびグループの URL など)で特殊文字を指定する場合は、エスケープ文字を使用する必要があります。次の表は、ユーザー名、グループ名、ユーザー URL、グループ URL、およびユーザー DN の OU の値で使用可能な特殊文字をリストしています。

表 4-8 サポートされている特殊文字

文字	名前または意味	文字	名前または意味
(左カッコ	\$	ドル
)	右カッコ	+	プラス
"	二重引用符	&	アンパサンド
'	一重引用符	\	円記号(バックスラッシュ)
,	カンマ	^	キャレット
=	次と等しい	;	セミコロン
<	次より小さい	#	ポンド
>	次より大きい	@	アットマーク

ノート:

ベース DN 内の組織単位名に/(スラッシュ)を使用しないでください

- 特殊文字はログイン・ユーザー属性の値には使用できません。
- アスタリスク(*)は、ユーザー名、グループ名、ユーザー URL、グループ URL、およびユーザー DN の OU 名には使用できません。
- 特殊文字の組合せを含んだ属性値は使用できません。
- アンパサンド(&)は、エスケープ文字なしで使用できます。Active Directory の設定では、&は&のように指定する必要があります。

- ユーザー名とグループ名には円記号(バックスラッシュ)(\)とスラッシュ(/)の両方を使用できません。たとえば、test/\user や new\test/user のような名前は使用できません。

表 4-9 エスケープする必要がない文字

文字	名前または意味	文字	名前または意味
(左カッコ	'	一重引用符
)	右カッコ	^	キャレット
\$	ドル	@	アットマーク
&	アンパサンド		

 **ノート:**

&は&のように記述されている必要があります。

これらの文字は、ユーザー・ディレクトリの設定(ユーザー名、グループ名、ユーザー URL、グループ URL およびユーザー DN)で使用する場合にエスケープされる必要があります。

表 4-10 ユーザー・ディレクトリ構成設定における特殊文字のエスケープ

特殊文字	エスケープ	設定例	エスケープの例
カンマ(,)	円記号/バックスラッシュ (\)	ou=test,ou	ou=test\,ou
プラス記号(+)	円記号/バックスラッシュ (\)	ou=test+ou	ou=test\+ou
等しい(=)	円記号/バックスラッシュ (\)	ou=test=ou	ou=test\=ou
シャープ(#)	円記号/バックスラッシュ (\)	ou=test#ou	ou=test\#ou
セミコロン(;)	円記号/バックスラッシュ (\)	ou=test;ou	ou=test\;ou
より小さい(<)	円記号/バックスラッシュ (\)	ou=test<ou	ou=test\<<ou
より大きい(>)	円記号/バックスラッシュ (\)	ou=test>ou	ou=test\>ou
二重引用符(")	二重の円記号/バックスラッシュ(\)	ou=test"ou	ou=test\\"ou
円記号/バックスラッシュ(\)	三重の円記号/バックスラッシュ(\)	ou=test\ou	ou=test\\\ou

 ノート:

- ユーザー DN では、二重引用符(")は、1つの円記号(バックスラッシュ)でエスケープされる必要があります。たとえば、ou=test"ou は ou=test\"ou と指定する必要があります。
- ユーザー DN では、円記号(バックスラッシュ)(\)は、1つの円記号(バックスラッシュ)でエスケープされる必要があります。たとえば、ou=test\ou は ou=test\\ou と指定する必要があります。

 注意:

ユーザー URL が指定されていない場合、RDN ルート内で作成されるユーザーに/ (スラッシュ)または\ (円記号(バックスラッシュ))が含まれてはいけません。同様に、グループ URL が指定されない場合、これらの文字は RDN ルート内に作成されたグループ名で使用してはいけません。たとえば、OU=child\ou,OU=parent/ou または OU=child/ou,OU=parent\ou などのグループ名は、サポートされません。この問題は、ユーザー・ディレクトリ構成の ID 属性に一意の属性を使用している場合は該当しません。

ネイティブ・ディレクトリでの特殊文字

ネイティブ・ディレクトリのユーザー名とグループ名には、特殊文字がサポートされています。

表 4-11 サポートされている特殊文字: ネイティブ・ディレクトリ

文字	名前または意味	文字	名前または意味
@	アットマーク	,	カンマ
#	ポンド	=	次と等しい
\$	ドル	+	プラス
^	キャレット	;	セミコロン
(左カッコ	!	感嘆符
)	右カッコ	%	パーセント
'	一重引用符		

5

カスタム認証モジュールの使用

次も参照:

- [概要](#)
- [使用事例の例と制限](#)
- [前提条件](#)
- [設計およびコーディングに関する考慮事項](#)
- [カスタム認証モジュールのデプロイ](#)

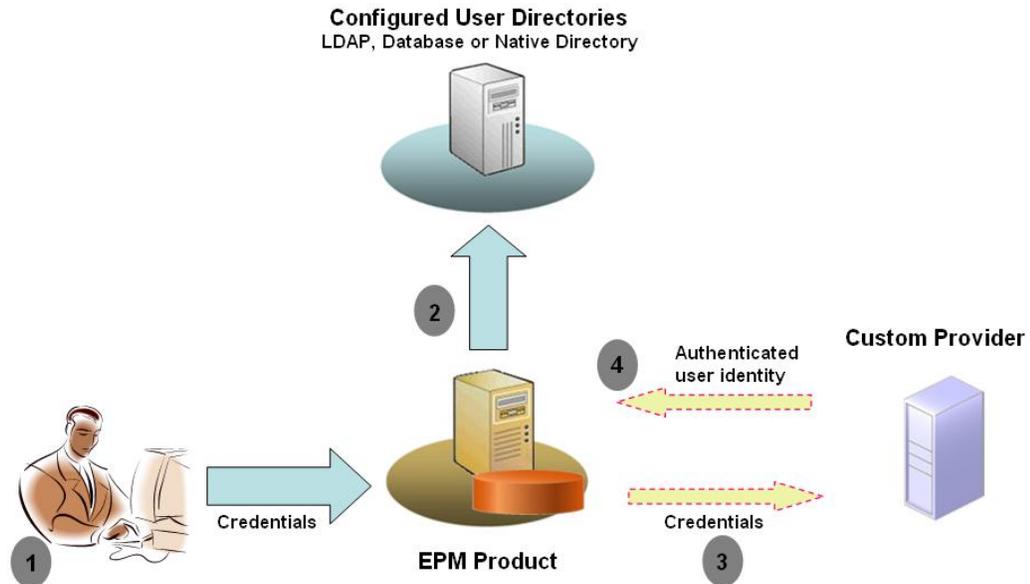
概要

カスタム認証モジュールは、Oracle Enterprise Performance Management System ユーザーを認証するためにユーザーが開発および実装する Java モジュールです。通常、EPM System 製品では、ユーザー名とパスワードの取得にログオン画面が使用されます。ユーザー名とパスワードはユーザーの認証に使用されます。EPM System 認証を使用するかわりに、カスタム認証モジュールを使用してユーザーを認証し、その後の処理のために認証済ユーザー資格証明を EPM System に渡すことができます。カスタム認証モジュールの実装には EPM System 製品の変更は含まれません。

カスタム認証モジュールは、シック・クライアント(Oracle Smart View for Office および Oracle Essbase Studio など)とシン・クライアント(Oracle Hyperion Enterprise Performance Management Workspace など)の両方で使用できます。

カスタム認証モジュールは、ユーザーが EPM System 製品にログインする際に入力する情報を使用します。ユーザー・ディレクトリに対して使用可能な場合、カスタム認証モジュールを使用してユーザーを認証します。ユーザーを正しく認証できた場合、カスタム認証モジュールは EPM System にユーザー名を戻します。

次の図は、カスタム認証のシナリオの例を示しています:



たとえば、RSA SecurID インフラストラクチャをカスタム・プロバイダとして使用し、EPM System への透過的で強力な認証を確保します。次に概要を示します：

1. ユーザーは資格証明(通常、ユーザー名とパスワード)を入力して EPM System 製品にアクセスします。これらの資格証明は、カスタム認証モジュールで使用されるプロバイダに対してユーザーを一意に識別する必要があります。たとえば、RSA SecurID インフラストラクチャを使用してユーザーを認証する場合、ユーザーは RSA ユーザー ID と PIN(EPM System ユーザー ID およびパスワードではなく)を入力します。
2. 検索順序(検索順序を参照)を使用すると、EPM System は構成済ユーザー・ディレクトリ内を循環し、ユーザーを検索します。
 - 現在のユーザー・ディレクトリがカスタム認証用に構成されていない場合、EPM System は、EPM System 認証を使用してユーザーを検索し、認証しようとします。
 - ユーザー・ディレクトリがカスタム認証用に構成されている場合、EPM System は認証プロセスをカスタム・モジュールに委任します。
3. EPM System が認証をカスタム・モジュールに委任した場合、カスタム認証モジュールは資格証明を受け入れ、その独自のロジックを使用してカスタム・プロバイダ(RSA SecurID インフラストラクチャなど)に対してユーザー認証を送ります。
4. カスタム認証モジュールでそのプロバイダに対してユーザーが認証される場合、ユーザー名が EPM System に戻されるか、Java 例外が戻されます。

カスタム認証モジュールで戻されるユーザー名は、カスタム認証で使用可能なユーザー・ディレクトリのユーザー名と同一である必要があります。

- カスタム認証モジュールでユーザー名が戻された場合、EPM System は、カスタム認証で使用可能なユーザー・ディレクトリでユーザーを検索します。この段階では、EPM System はカスタム認証用に構成されていないユーザー・ディレクトリは検索しません。
- カスタム認証モジュールで例外がスローされるか、null ユーザーが戻された場合、EPM System は、カスタム認証が使用可能になっていない、検索順序の残

りのユーザー・ディレクトリ内でユーザーの検索を続行します。資格証明が一致するユーザーが見つからない場合、EPM System にエラーが表示されます。

使用事例の例と制限

カスタム認証の実装シナリオには、次のようなものがあります：

- ワンタイム・パスワード・サポートの追加
- **Resource Access Control Facility (RACF)** に対する認証の実行
- 簡単な LDAP バインドのかわりに **Simple Authentication and Security Layer (SASL)** バインドを LDAP 対応ユーザー・ディレクトリに追加

チャレンジ/応答メカニズムの認証は、カスタム認証モジュールを実装している場合、うまく機能しない可能性があります。カスタム認証モジュールによってスローされたカスタム・メッセージは、クライアントに伝播されません。クライアント (**Oracle Hyperion Enterprise Performance Management Workspace** など) が通常のメッセージを表示するためにエラー・メッセージをオーバーライドするので、次のシナリオは有効ではありません：

- 2 つの連続する RSA SecurID PIN
- チャレンジのパスワード変形(パスワードの最初、最後および 3 番目の文字の入力など)

前提条件

- CustomAuth.jar という完全にテストされた Java アーカイブには、カスタム認証モジュール・ライブラリが含まれます。CustomAuth.jar では、標準 **Oracle Hyperion Shared Services API** の一部として com.hyperion.css パッケージに定義されたパブリック・インタフェース **CSSCustomAuthenticationIF** を実装する必要があります。http://download.oracle.com/docs/cd/E12825_01/epm.111/epm_security_api_11111/client/com/hyperion/css/CSSCustomAuthenticationIF.html を参照してください。
- **Shared Services** 管理者としての **Shared Services** へのアクセス権

設計およびコーディングに関する考慮事項

検索順序

Oracle Hyperion Shared Services では、ネイティブ・ディレクトリ以外に複数のユーザー・ディレクトリを構成できます。デフォルトの検索順序の位置は、すべての構成済ユーザー・ディレクトリに割り当てられます。検索順序を **Oracle Hyperion Shared Services Console** から変更できます。ネイティブ・ディレクトリを除き、構成済ユーザー・ディレクトリは検索順序から除去できます。**Oracle Enterprise Performance Management System** では、検索順序に含まれていないユーザー・ディレクトリは使用されません。**Oracle Enterprise Performance Management System** ユーザー・セキュリティ管理ガイドを参照してください。

検索順序により、ユーザーの認証のために **EPM System** がユーザー・ディレクトリ内を循環する順序が決定されます。ユーザーがユーザー・ディレクトリ内で認証されている場合、**EPM System** は検索を停止し、ユーザーを戻します。ユーザーが検索順序内のユーザー・ディレクトリに対して認証されていない場合、**EPM System** は認証を拒否してエラーを戻します。

検索順序でのカスタム認証の影響

カスタム認証は、EPM System セキュリティによる検索順序の解釈に影響を及ぼします。

カスタム認証モジュールでユーザー名が戻された場合、EPM System は、カスタム認証で使用可能なユーザー・ディレクトリのみでユーザーを検索します。この段階では、EPM System はカスタム認証用に構成されていないユーザー・ディレクトリを無視します。

カスタム認証のフローについて

次の使用事例シナリオを、カスタム認証のフローを調査するために使用します:

- [使用事例シナリオ 1](#)
- [使用事例シナリオ 2](#)
- [使用事例シナリオ 3](#)

使用事例シナリオ 1

次の表に、このシナリオで使用する EPM System ユーザー・ディレクトリ構成と検索順序を示します。このシナリオでは、カスタム認証モジュールが RSA インフラストラクチャを使用してユーザーを認証すると仮定します。

表 5-1 シナリオ 1 の設定

ユーザー・ディレクトリのタイプと名前	検索順序	カスタム認証	サンプル・ユーザー名	パスワード ¹
ネイティブ・ディレクトリ	1	使用不可	test_user_1 test_user_2 test_user_3	password
LDAP 対応 SunONE_West	2	使用不可	test_ldap1 test_ldap_2 test_user_3 test_ldap_4	ldappassword
LDAP 対応 SunONE_East	3	使用可能	test_ldap1 test_ldap_2 test_user_3	SunONE では ldappassword、 カスタム・モジュールでは RSA PIN

¹ 単純化するため、すべてのユーザーが同じユーザー・ディレクトリ・パスワードを使用すると仮定します。

認証プロセスを開始するには、ユーザーは EPM System 製品のログオン画面でユーザー名とパスワードを入力します。このシナリオでは、カスタム認証モジュールは次のアクションを実行します:

- ユーザー名と RSA PIN をユーザー資格証明として受け入れます

- ユーザー名を `username@providername` 形式(たとえば、`test_ldap_2@SunONE_East`)で EPM System セキュリティに戻します

表 5-2 ユーザーのやりとりと結果

ユーザー名およびパスワード	認証結果	ログイン・ユーザー・ディレクトリ
test_user_1/password	成功	ネイティブ・ディレクトリ
test_user_3/password	成功	ネイティブ・ディレクトリ
test_user_3/ ldappassword	成功	SunONE_West (検索順序 2) ¹
test_user_3/RSA PIN	成功	SunONE_East (検索順序 3) ²
test_ldap_2/ ldappassword	成功	SunONE_West (検索順序 2)
test_ldap_4/RSA PIN	失敗 EPM System に認証エラーが表示 されます。 ³	

- ¹ ユーザーは EPM System 資格証明を入力したので、カスタム認証ではこのユーザーは認証できません。EPM System はカスタム認証で使用可能でないユーザー・ディレクトリでのみこのユーザーを識別できます。ユーザーはネイティブ・ディレクトリ(検索順序番号 1)ではなく、SunONE West (検索順序番号 2)で識別されます。
- ² EPM System は、このユーザーをネイティブ・ディレクトリ(検索順序番号 1)または SunONE West (検索順序番号 2)で見つけられません。カスタム認証モジュールでは RSA サーバーに対してユーザーを検証し、`test_user_3@SunONE_EAST` を EPM System に戻します。EPM System はユーザーを SunONE East(検索順序番号 3)で検索します。これはカスタム認証が有効なディレクトリです。
- ³ カスタム・モジュールで認証されているユーザーはすべて、検索順序に含まれるカスタム認証が有効なユーザー・ディレクトリに含めることをお勧めします。カスタム認証モジュールで戻されるユーザー名が、検索順序に含まれるカスタム認証が有効なユーザー・ディレクトリにない場合、ログインは失敗します。

使用事例シナリオ 2

次の表に、このシナリオで使用する EPM System ユーザー・ディレクトリ構成と検索順序を示します。このシナリオでは、カスタム認証モジュールが RSA インフラストラクチャを使用してユーザーを認証すると仮定します。

このシナリオでは、カスタム認証モジュールは次のアクションを実行します:

- ユーザー名と RSA PIN をユーザー資格証明として受け入れます
- ユーザー名(たとえば、`test_ldap_2`)を EPM System セキュリティに戻します。

表 5-3 検索順序の例

ユーザー・ディレクトリ	検索順序	カスタム認証	サンプル・ユーザー名	パスワード ¹
ネイティブ・ディレクトリ	1	使用不可	test_user_1 test_user_2 test_user_3	password
LDAP 対応(たとえば、SunONE)	2	使用可能	test_ldap1 test_ldap2 test_user_3	SunONE では ldappassword、カスタム・モジュールでは RSA PIN

- 1 単純化するため、すべてのユーザーが同じユーザー・ディレクトリ・パスワードを使用すると仮定します。

認証プロセスを開始するには、ユーザーは EPM System 製品のログイン画面でユーザー名とパスワードを入力します。

表 5-4 ユーザーのやりとりと結果

ユーザー名およびパスワード	ログイン結果	ログイン・ユーザー・ディレクトリ
test_user_1/password	成功	ネイティブ・ディレクトリ
test_user_3/password	成功	ネイティブ・ディレクトリ
test_user_3/ldappassword	失敗	SunONE ¹
test_user_3/RSA PIN	成功	SunONE ²

- 1 ネイティブ・ディレクトリに対するユーザーの認証は、パスワードが一致していないために失敗します。カスタム認証モジュールを使用したユーザーの認証は、使用されたパスワードが有効な RSA PIN ではないため失敗します。EPM System は、カスタム認証設定がこのディレクトリの EPM System 認証をオーバーライドしたため、SunONE (検索順序 2) でこのユーザーの認証を試行しません。
- 2 ネイティブ・ディレクトリに対するユーザーの認証は、パスワードが一致していないために失敗します。カスタム認証モジュールによりユーザーが認証され、ユーザー名 test_user_3 が EPM System に戻されます。

使用事例シナリオ 3

次の表に、このシナリオで使用する EPM System ユーザー・ディレクトリ構成と検索順序を示します。このシナリオでは、カスタム認証モジュールが RSA インフラストラクチャを使用してユーザーを認証すると仮定します。

このようなシナリオの明確さのため、カスタム認証モジュールがユーザー名を username@providername 形式(たとえば、test_ldap_4@SunONE)で戻すことをお勧めします。

表 5-5 検索順序の例

ユーザー・ディレクトリ	検索順序	カスタム認証	サンプル・ユーザー名	パスワード ¹
ネイティブ・ディレクトリ	1	使用可能	test_user_1 test_user_2 test_user_3	RSA_PIN
LDAP 対応(たとえば、MSAD)	2	使用不可	test_ldap1 test_ldap4 test_user_3	ldappassword
LDAP 対応(たとえば、SunONE)	3	使用可能	test_ldap1 test_ldap4 test_user_3	SunONE では ldappassword、カスタム・モジュールでは RSA PIN

- 1 単純化するため、すべてのユーザーが同じユーザー・ディレクトリ・パスワードを使用すると仮定します。

認証プロセスを開始するには、ユーザーは EPM System 製品のログイン画面でユーザー名とパスワードを入力します。

表 5-6 ユーザーのやりとりと結果

ユーザー名およびパスワード	認証結果	ログイン・ユーザー・ディレクトリ
test_user_1/password	成功	ネイティブ・ディレクトリ
test_user_3/RSA_PIN	成功	ネイティブ・ディレクトリ
test_user_3/ldappassword	成功	MSAD (検索順序 2)
test_ldap_4/ldappassword	成功	MSAD (検索順序 2)
test_ldap_4/RSA PIN	成功	SunONE (検索順序 3)

ユーザー・ディレクトリおよびカスタム認証モジュール

カスタム認証モジュールを使用するには、EPM System ユーザーおよびグループ情報を含むユーザー・ディレクトリを、カスタム・モジュールに認証を委任するよう個別に構成できます。

カスタム・モジュールを使用して認証された EPM System ユーザーは、検索順序(検索順序を参照)に含まれたユーザー・ディレクトリの 1 つに存在する必要があります。また、ユーザー・ディレクトリは、認証をカスタム・モジュールに委任するよう構成されている必要があります。

カスタム・プロバイダのユーザーのアイデンティティ (たとえば、RSA SecurID インフラストラクチャの 1357642)は、Shared Services で構成されるユーザー・ディレクトリのユーザー名 (たとえば、Oracle Internet Directory の jDoe)と異なる場合があります。ユーザーの認証後、カスタム認証モジュールは、ユーザー名 jDoe を EPM System に戻す必要があります。

ノート:

ベスト・プラクティスとして、EPM System で構成されるユーザー・ディレクトリのユーザー名は、カスタム認証モジュールで使用されるユーザー・ディレクトリで使用可能なものと同一にすることをお勧めします。

CSSCustomAuthenticationIF Java インタフェース

カスタム認証モジュールは、EPM System セキュリティ・フレームワークとの統合に CSSCustomAuthenticationIF Java インタフェースを使用する必要があります。カスタム認証が成功した場合はユーザー名の文字列を、認証が失敗した場合はエラー・メッセージを戻す必要があります。認証プロセスが完了した場合、カスタム認証モジュールによって戻されたユーザー名は、Shared Services 検索順序に含まれるユーザー・ディレクトリの 1 つに存在する必要があります。EPM System セキュリティ・フレームワークでは、`username@providerName` 形式がサポートされています。

 ノート:

カスタム認証モジュールが戻すユーザー名に* (アスタリスク)を含めないでください。EPM System セキュリティ・フレームワークがユーザーの検索中にワイルドカード文字と解釈します。

CSSCustomAuthenticationIF インタフェース・シグネチャは、[サンプル・コード 1](#)を参照してください。

カスタム認証モジュール(クラス・ファイルを使用可)は、CustomAuth.jar に含まれている必要があります。パッケージ構造は重要ではありません。

CSSCustomAuthenticationIF インタフェースの詳細は、[セキュリティ API ドキュメント](#)。を参照してください

CSSCustomAuthenticationIF の authenticate メソッドではカスタム認証がサポートされます。authenticate メソッドは、EPM System にアクセスしようとする際にユーザーが入力した資格証明(ユーザー名とパスワード)を入力パラメータとして受け入れます。このメソッドは、カスタム認証が成功した場合に文字列(ユーザー名)を戻します。認証に失敗した場合は java.lang.Exception をスローします。メソッドにより戻されるユーザー名は、**Shared Services** 検索順序に含まれるユーザー・ディレクトリの1つでユーザーを一意に識別する必要があります。EPM System セキュリティ・フレームワークでは、username@providerName 形式がサポートされています。

 ノート:

リソース(たとえば、JDBC 接続プール)を初期化するには、クラス・コンストラクタを使用します。これにより、認証のたびにリソースをロードすることがなくなり、パフォーマンスが向上します。

カスタム認証モジュールのデプロイ

Oracle Enterprise Performance Management System デプロイメントでサポートされるカスタム・モジュールは1つのみです。検索順序の1つ以上のユーザー・ディレクトリに付いてカスタム認証を有効にできます。

カスタム認証モジュールは、com.hyperion.css パッケージで定義されるパブリック・インタフェース CSSCustomAuthenticationIF を実装する必要があります。このドキュメントでは、選択したユーザー・プロバイダに対してユーザーを認証するロジックを定義する完全な機能のカスタム・モジュールを持っていることを前提としています。カスタム認証モジュールを開発およびテストした後、EPM System 環境で実装する必要があります。

ステップの概要

カスタム認証コードではエラー・ロギングに log4j を使用しないでください。以前のリリースで使用したコードで log4j を使用している場合は、このリリースで使用する前に、コードから削除する必要があります。

カスタム認証モジュールを実装するには、次のステップを実行します:

- EPM System 製品(Oracle Hyperion Shared Services および Shared Services API を使用するあらゆるシステムを含む)を停止します。
- カスタム認証モジュールの Java アーカイブ CustomAuth.jar を次のデプロイメントにコピーします:

- **WebLogic:** CustomAuth.jar を `MIDDLEWARE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib` (通常は `C:/Oracle/Middleware/user_projects/domains/EPMSysstem/lib`) にコピーします。

カスタム認証モジュールの実装を含んだリリース 11.1.2.0 または 11.1.2.1 からアップグレードしている場合、CustomAuth.jar を `EPM_ORACLE_HOME/common/jlib/11.1.2.0` から `MIDDLEWARE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib` に移動します。

- **すべてのクライアント・デプロイメント:** CustomAuth.jar をすべての EPM System クライアント・デプロイメントの次の場所にコピーします:

`EPM_ORACLE_HOME/common/jlib/11.1.2.0` (通常は `Oracle/Middleware/common/jlib/11.1.2.0`) CustomAuth.jar ファイルが、常に `EPM_ORACLE_HOME/common/jlib/11.1.2.0` ディレクトリにあることを確認します。

カスタム認証を操作するすべてのサーバーおよびクライアントでは、CustomAuth.jar ファイルが次の 2 つの場所に存在している必要があります:

- * `MIDDLEWARE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib`
- * `EPM_ORACLE_HOME/common/jlib/11.1.2.0`

- Shared Services のユーザー・ディレクトリ設定を更新します。[Shared Services での設定の更新](#)を参照してください。
- Shared Services を開始してから、その他の EPM System 製品を開始します。
- 実装をテストします。[デプロイメントのテスト](#)を参照してください。

Shared Services での設定の更新

デフォルトでは、カスタム認証は、すべてのユーザー・ディレクトリで使用不可です。デフォルトの動作をオーバーライドして、特定の外部ユーザー・ディレクトリまたはネイティブ・ディレクトリに対して、カスタム認証を使用可能にできます。

ユーザー・ディレクトリ構成の更新

カスタム認証を使用可能にするユーザー・ディレクトリの構成を更新する必要があります。

ユーザー・ディレクトリ構成を更新するには:

1. Oracle Hyperion Foundation Services を起動します。
2. システム管理者として Oracle Hyperion Shared Services Console にアクセスします。
3. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
4. 「定義済ユーザー・ディレクトリ」画面で、カスタム認証設定を変更するユーザー・ディレクトリを選択します。

 ノート:

EPM System は、検索順序に含まれたユーザー・ディレクトリのみを使用します。

5. 「編集」をクリックします。
6. 「詳細オプションの表示」を選択します。
7. 「カスタム・モジュール」で、「認証モジュール」を選択し、現在のユーザー・ディレクトリに対してカスタム・モジュールを使用可能にします。
8. 「終了」をクリックします。
9. この手順を繰り返して、検索順序に含まれる他のユーザー・ディレクトリの構成を更新します。

セキュリティ・オプションの更新

CustomAuth.jar が `EPM_ORACLE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib` で使用可能であることを確認してから、次の手順を開始してください。

セキュリティ・オプションを更新するには:

1. システム管理者として Shared Services Console にアクセスします。
2. 「管理」、「ユーザー・ディレクトリの構成」の順に選択します。
3. 「セキュリティ・オプション」を選択します。
4. 「詳細オプションの表示」を選択します。
5. **認証モジュール**で、カスタム認証モジュールが選択されるすべてのユーザー・ディレクトリでユーザーの認証に使用されるカスタム認証モジュールの完全修飾クラス名を入力します。たとえば、`com.mycompany.epm.CustomAuthenticationImpl` です。
6. 「OK」をクリックします。

デプロイメントのテスト

ネイティブ・ディレクトリがカスタム認証に対して構成されていない場合、カスタム認証のテストにネイティブ・ディレクトリ・ユーザーを使用しないでください。

 ノート:

カスタム認証モジュールの問題を識別して修正する必要があります。カスタム・モジュールで使用するユーザー・ディレクトリからのユーザーを **EPM System** 検索順序で使用可能なカスタム認証が有効なユーザー・ディレクトリのユーザーにマップするために、カスタム・モジュールがフレームレスに機能することを想定しています。

デプロイメントをテストするには、カスタム・モジュールで使用されるユーザー・ディレクトリ(たとえば、RSA SecurID インフラストラクチャ)からのユーザー資格証明を

使用して **EPM System** にログインします。これらの資格証明は、**EPM System** の資格証明と異なる場合があります。

EPM System 製品でリソースへのアクセスを許可された場合、実装は成功したと考えられます。ユーザーが見つからなかったというエラーが常に実装の失敗を示しているわけではありません。このような場合、入力した資格証明がカスタム・ユーザー・ストアに存在するか、一致するユーザーが **EPM System** 検索順序でカスタム認証が有効なユーザー・ディレクトリの1つに存在するかを確認してください。

カスタム認証をテストするには:

1. **EPM System** 製品が実行されていることを確認します。
2. Oracle Hyperion Enterprise Performance Management Workspace などの **EPM System** コンポーネントにアクセスします。
3. カスタム認証が有効なユーザーディレクトリで定義されているユーザーとしてログインします。
 - a. 「**ユーザー名**」に、ユーザー ID(たとえば、RSA ユーザー ID)を入力します。
 - b. 「**パスワード**」に、パスワード(たとえば、RSA PIN)を入力します。
 - c. 「**ログイン**」をクリックします。
4. **EPM System** 製品のリソースにアクセスできたことを確認します。

6

EPM System の保護のガイドライン

次も参照:

- [SSL の実装](#)
- [管理パスワードの変更](#)
- [暗号化キーの再生成](#)
- [データベース・パスワードの変更](#)
- [Cookie の保護](#)
- [SSO トークンのタイムアウトの低減](#)
- [セキュリティ・レポートの確認](#)
- [認証システムの強力な認証としてのカスタマイズ](#)
- [EPM Workspace のデバッグ・ユーティリティを使用不可にする](#)
- [デフォルトの Web サーバー・エラー・ページの変更](#)
- [サードパーティ製ソフトウェアのサポート](#)

SSL の実装

SSL では、データを暗号化する暗号システムを使用します。SSL は、データを安全に送信できるクライアントとサーバー間の安全な接続を作成します。

Oracle Enterprise Performance Management System 環境をセキュリティ保護するには、Web アプリケーションおよびユーザー・ディレクトリ接続で使用されるすべての通信チャネルを SSL によって保護します。[EPM System コンポーネントの SSL 使用可能化](#)を参照してください。

さらに、ファイアウォールを使用して、すべてのエージェント・ポート(Oracle Hyperion Reporting and Analysis エージェント・ポートであるポート 6861 など)を保護します。エンド・ユーザーは、EPM System エージェント・ポートにアクセスする必要がありません。

管理パスワードの変更

ネイティブ・ディレクトリのデフォルトの **admin** ユーザー・アカウントでは、すべての Oracle Hyperion Shared Services 機能にアクセスできます。このパスワードは、Oracle Hyperion Foundation Services のデプロイ時に設定されます。このアカウントのパスワードを定期的に変更する必要があります。

パスワードを変更するには、**admin** ユーザー・アカウントを編集します。[Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド](#)のユーザー・アカウントの変更を参照してください。

暗号化キーの再生成

Oracle Hyperion Shared Services Console を使用して、次のものを定期的に再生成します:

- シングル・サインオン・トークン

▲ 注意:

Oracle Hyperion Financial Management および Oracle Hyperion Profitability and Cost Management で使用されるタスクフローは、新しいキーストアを生成すると無効化されます。キーストアを再生成した後に、タスクフローを開いて保存すると、タスクフローは再度有効化されます。

- 信頼できるサービス・キー
- プロバイダ構成キー

[暗号化キーの再生成](#)を参照してください。

✎ ノート:

Oracle Hyperion Shared Services および Oracle Enterprise Performance Management System のセキュリティ・サブシステムでは、128 ビットのキーの強度の AES 暗号化が使用されます。

データベース・パスワードの変更

すべての Oracle Enterprise Performance Management System 製品データベースのパスワードを定期的に変更します。Oracle Hyperion Shared Services レジストリでデータベースのパスワードを変更する手順の詳細は、この項で説明します。

EPM System 製品データベースのパスワード変更手順の詳細は、*Oracle Enterprise Performance Management System インストールおよび構成ガイド*を参照してください。

EPM System 製品のデータベースのパスワードを Shared Services レジストリで変更するには:

1. データベース管理コンソールを使用して、EPM System 製品のデータベースの構成に使用したアカウントを持つユーザーのパスワードを変更します。
2. EPM System 製品(Web アプリケーション、サービスおよびプロセス)を停止します。
3. EPM System コンフィグレータを使用して、次の手順のいずれかを実行してデータベースを再構成します。

Oracle Hyperion Shared Services のみ:

 ノート:

EPM System 製品が Shared Services と異なるマシンに存在する分散環境では、すべてのサーバーでこの手順を実行する必要があります。

- a. EPM System コンフィグレータの Foundation タスクから「**データベースの構成**」を選択します。
- b. 「Shared Services およびレジストリ・データベース構成」ページで、「**前に構成された Shared Services データベースに接続**」を選択します。
- c. Shared Services データベースを構成するのに使用したアカウントを持つユーザーの新パスワードを指定します。他の設定は変更しないでください。
- d. 構成を続行し、完了したら「**終了**」をクリックします。

Shared Services 以外の EPM System 製品: ノート:

現在のサーバーにデプロイされている EPM System 製品に対しのみ、次のステップを行います。

詳細な手順については、*Oracle Enterprise Performance Management System* インストールおよび構成ガイドを参照してください。

4. EPM System 製品およびサービスを開始します。

Cookie の保護

Oracle Enterprise Performance Management System Web アプリケーションでは、セッションを追跡するための Cookie が設定されます。特にセッションの cookie を設定しているとき、サーバーは保護フラグを設定できます。これにより、ブラウザは保護チャンネルを介して cookie を送信できます。この動作で、セッションが乗っ取られる危険性が低くなります。

 ノート:

EPM System 製品が SSL 使用可能の環境にデプロイされる場合のみ Cookie を保護します。

Oracle WebLogic Server セッションの記述子を変更して、WebLogic Server の Cookie を保護します。session-param 要素内の cookieSecure 属性の値を TRUE に設定します。[Oracle Fusion Middleware Oracle WebLogic Server セキュリティのプログラミング 11g](#) の Web アプリケーションの保護を参照してください

SSO トークンのタイムアウトの低減

SSO トークンのデフォルトのタイムアウトは 480 分です。SSO トークンのタイムアウトを、たとえば 60 分に縮小すると、表示されている場合はトークンの再利用を最小限にできます。*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*のセキュリティ・オプションの設定を参照してください。

セキュリティ・レポートの確認

セキュリティ・レポートには、監査を構成しているセキュリティ・タスクに関する監査情報が含まれています。特に *Oracle Enterprise Performance Management System* 製品で失敗したログイン試行とプロビジョニングの変更を識別するために、このレポートを *Oracle Hyperion Shared Services Console* で定期的に生成し確認します。レポート生成オプションとして「**詳細ビュー**」を選択し、変更された属性と新しい属性値に基づいてレポート・データをグループ化します。*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*のレポートの生成を参照してください。

認証システムの強力な認証としてのカスタマイズ

カスタム認証モジュールを使用して、EPM System に強力な認証を追加できます。たとえば、RSA SecurID two-factor 認証を nonchallenge 応答モードで使用できます。カスタム認証モジュールは、シン・クライアントおよびシック・クライアントに対して透過的であり、クライアント側のデプロイメント変更は必要ありません。[カスタム認証モジュールの使用](#)を参照してください。

EPM Workspace のデバッグ・ユーティリティを使用不可にする

- トラブルシューティングの目的で、*Oracle Hyperion Enterprise Performance Management Workspace* には未処理の JavaScript ファイルが付属しています。セキュリティの目的で、これらの未処理の JavaScript ファイルを本番環境から除去する必要があります：
 - `EPM_ORACLE_HOME/common/epmstatic/wspace/js/`ディレクトリのバックアップ・コピーを作成します。
 - ファイル `DIRECTORY_NAME.js` を除き、`EPM_ORACLE_HOME/common/epmstatic/wspace/js` の各サブディレクトリから .js ファイルを削除します。各サブディレクトリには、ディレクトリの名前を持つ .js ファイルが含まれています。たとえば、`EPM_ORACLE_HOME/common/epmstatic/wspace/js/com/hyperion/bpm/web/common` には `Common.js` が含まれています。ディレクトリの名前を持つファイル(この場合は `Common.js`)以外のすべての .js ファイルを除去します。
- *EPM Workspace* には、*EPM Workspace* がデバッグ・モードでデプロイされた場合にアクセス可能になるデバッグ・ユーティリティおよびテスト・アプリケーション

ンが用意されています。セキュリティの目的で、管理者は EPM Workspace のクライアント側のデバッグをオフにする必要があります。

デバッグ・モードをオフにするには:

1. EPM Workspace に管理者としてログインします。
2. 「ナビゲート」、「管理」、「Workspace サーバー設定」の順に選択します。
3. 「Workspace サーバー設定」の **ClientDebugEnabled** で、「いいえ」を選択します。
4. 「OK」をクリックします。

デフォルトの Web サーバー・エラー・ページの変更

アプリケーション・サーバーが要求を受け入れられないとき、バックエンド・アプリケーション・サーバーの Web サーバー・プラグイン(Oracle WebLogic Server の Oracle HTTP Server プラグインなど)はプラグインの構築情報が表示されたデフォルトのエラー・ページを戻します。これ以外にも、Web サーバーによってデフォルトのエラー・ページが表示される場合があります。攻撃者は、この情報から公共の Web サイトの既知の脆弱性を知ることができます。

Web アプリケーション・サーバー・プラグインおよび Web サーバーのエラー・ページをカスタマイズして、サーバーのバージョン、サーバー・タイプ、プラグインの作成日、プラグイン・タイプなどの本番環境用システム・コンポーネントに関する情報が含まれないようにできます。詳細は、ご使用のアプリケーション・サーバーおよび Web サーバーのベンダーのドキュメントを参照してください。

サードパーティ製ソフトウェアのサポート

オラクル社は、サードパーティ・ベンダーが明言している下位互換性を了承し、サポートします。したがって、ベンダーが下位互換を明言している場合、その後のメンテナンス・リリースやサービス・パックを使用できます。互換性がないことがわかると、オラクル社では、製品をデプロイすべきパッチ・リリースを指定(およびサポート・マトリックスから互換性のないバージョンを削除)するか、その Oracle 製品のメンテナンス・リリースまたはサービス・フィックスを提供します。

サーバー側の更新: サードパーティ製サーバー側コンポーネントのアップグレードに関するサポートは将来のメンテナンス・リリースに関する方針に従います。通常、Oracle では、サードパーティ製サーバー側コンポーネントについて、現在サポートしているリリースのサービス・パックの次回メンテナンス・リリースへのアップグレードをサポートします。次回の主要リリースへのアップグレードはサポートされません。

クライアント側の更新: Oracle ではクライアント・コンポーネントの自動更新をサポートしています。これには、サードパーティ製クライアント・コンポーネントの次回主要リリースへの更新が含まれます。たとえば、ブラウザの JRE バージョンを現在サポートされている JRE バージョンに更新できます。

A

カスタム認証サンプル・コード

サンプル・コード 1

ノート:

カスタム認証コードではエラー・ロギングに **log4j** を使用しないでください。以前のリリースで使用したカスタム認証コードで **log4j** を使用している場合は、このリリースで使用する前に、コードから削除する必要があります。

次のコード・スニペットは、カスタム・モジュールの空の実装です:

```
package com.hyperion.css.custom;

import java.util.Map;
import com.hyperion.css.CSSCustomAuthenticationIF;

public class CustomAuthenticationImpl implements CSSCustomAuthenticationIF {
    public String authenticate(Map context,String userName,
                               String password) throws Exception{
        try{
            //Custom code to find and authenticate the user goes here.
            //The code should do the following:
            //if authentication succeeds:
                //set authenticationSuccessFlag = true
                //return authenticatedUserName
            // if authentication fails:
                //log an authentication failure
                //throw authentication exception
        }
        catch (Exception e){
            //Custom code to handle authentication exception goes here
            //Create a new exception, set the root cause
            //Set any custom error message
            //Return the exception to the caller
        }
        return authenticatedUserName;
    }
}
```

入力パラメータ:

- コンテキスト: ロケール情報のキーと値のペアを含むマップ

- ユーザー名: カスタム・モジュールがユーザーを認証するユーザー・ディレクトリにユーザーを一意に識別する識別子。ユーザーは、**Oracle Enterprise Performance Management System** コンポーネントにログインする際にこのパラメータの値を入力します。
- パスワード: カスタム・モジュールがユーザーを認証するユーザー・ディレクトリのユーザーのパスワード・セット。ユーザーは、**EPM System** コンポーネントにログインする際にこのパラメータの値を入力します。

サンプル・コード 2

次のサンプル・コードは、フラット・ファイルに含まれるユーザー名とパスワードを使用したユーザーのカスタム認証を示します。カスタム認証を機能させるには、クラス・コンストラクタ内のユーザーとパスワードのリストを初期化する必要があります。

```
package com.hyperion.css.security;

import java.util.Map;
import java.util.HashMap;
import com.hyperion.css.CSSCustomAuthenticationIF;
import java.io.*;

public class CSSCustomAuthenticationImpl implements
CSSCustomAuthenticationIF{
    static final String DATA_FILE = "datafile.txt";

    /**
     * authenticate method includes the core implementation of the
     * Custom Authentication Mechanism. If custom authentication is
     * enabled for the provider, authentication operations
     * are delegated to this method. Upon successful authentication,
     * this method returns a valid user name, using which EPM System
     * retrieves the user from a custom authentication enabled provider.
     * User name can be returned in the format username@providerName,
     * where providerName indicates the name of the underlying provider
     * where the user is available. authenticate method can use other
     * private methods to access various core components of the
     * custom authentication module.

     * @param context
     * @param userName
     * @param password
     * @return
     * @throws Exception
     */
    Map users = null;

    public CSSCustomAuthenticationImpl(){
        users = new HashMap();
        InputStream is = null;
        BufferedReader br = null;
        String line;
        String[] userDetails = null;
    }
}
```

```
String userKey = null;
try{
    is = CSSCustomAuthenticationImpl.class.getResourceAsStream(DATA_FILE);
    br = new BufferedReader(new InputStreamReader(is));
    while(null != (line = br.readLine())){
        userDetails = line.split(":");
        if(userDetails != null && userDetails.length==3){
            userKey = userDetails[0]+ ":" + userDetails[1];
            users.put(userKey, userDetails[2]);
        }
    }
}
catch(Exception e){
    // log a message
}
finally{
    try{
        if(br != null) br.close();
        if(is != null) is.close();
    }
    catch(IOException ioe){
        ioe.printStackTrace();
    }
}
}

/* Use this authenticate method snippet to return username from a flat file
*/

public String authenticate(Map context, String userName, String password)
throws Exception{
    //userName : user input for the userName
    //password : user input for password
    //context : Map, can be used to additional information required by
    //          the custom authentication module.

    String authenticatedUserKey = userName + ":" + password;

    if(users.get(authenticatedUserKey)!=null)
        return (String)users.get(authenticatedUserKey);
    else throw new Exception("Invalid User Credentials");
}

/* Refer to this authenticate method snippet to return username in
   username@providername format */

public String authenticate(Map context, String userName, String password)
throws Exception{

    //userName : user input for userName
    //password : user input for password
    //context : Map can be used to additional information required by
    //          the custom authentication module.

    //Your code should uniquely identify the user in a custom provider and in
```

```

a configured
//user directory in Shared Services. EPM Security expects you to
append the provider
//name to the user name. Provider name must be identical to the name
of a custom
//authentication-enabled user directory specified in Shared Services.

//If invalid arguments, return null or throw exception with
appropriate message
//set authenticationSuccessFlag = false

String authenticatedUserKey = userName + ":" + password;
if(users.get(authenticatedUserKey)!=null)
    String userNameStr = (new StringBuffer())
        .append((String)users.get(authenticatedUserKey))
        .append("@").append(PROVIDER_NAME).toString();
    return userNameStr;
else throw new Exception("Invalid User Credentials");
}
}

```

サンプル・コード 2 のデータ・ファイル

データ・ファイルが `datafile.txt` という名前(サンプル・コードで使用される名前)で、作成する **Java** アーカイブに含まれていることを確認してください。

サンプル・コード 2 ([サンプル・コード 2](#) を参照)で実装されるカスタム認証モジュールをサポートするためにカスタム・ユーザー・ディレクトリとして使用されるフラット・ファイルのコンテンツとして次を使用します。

```

xyz:password:admin
test1:password:test1@LDAP1
test1:password:test1
test1@LDAP1:password:test1@LDAP1
test1@1:password:test1
user1:Password2:user1@SunONE1
user1_1:Password2:user1
user3:Password3:user3
DS_User1:Password123:DS_User1@MSAD1
DS_User1:Password123:DS_User1
DS_User1@1:Password123:DS_User1

```

ユーザー名を `username@providername` 形式で戻す予定の場合にカスタム・ユーザー・ディレクトリとして使用されるフラット・ファイルのコンテンツとして次を使用します:

```

xyz:password:admin
test1:password:test1
test1@1:password:test1

```

```
user1_1:Password2:user1
user3:Password3:user3
DS1_1G100U_User61_1:Password123:DS1_1G100U_User61
DS1_1G100U_User61_1@1:Password123:DS1_1G100U_User61
TUser:password:TUser
```

B

カスタム・ログイン・クラスの実装

Oracle Enterprise Performance Management System には、X509 証明書からユーザー・アイデンティティ (DN) を抽出するための `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl` が用意されています。

DN 以外の証明書にある属性からユーザーアイデンティティを取得する必要がある場合、この付録で説明しているように、`com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl` に類似したカスタム・ログイン・クラスを開発および実装する必要があります。

カスタム・ログイン・クラス・サンプル・コード

このサンプル・コードは、デフォルトの `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl` の実装を示しています。通常、この実装の `parseCertificate(String sCertificate)` メソッドをカスタマイズして、DN 以外の証明書属性からユーザー名を取得する必要があります。

```
package com.hyperion.css.sso.agent;

import java.io.ByteArrayInputStream;
import java.io.UnsupportedEncodingException;
import java.security.Principal;
import java.security.cert.CertificateException;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
import com.hyperion.css.CSSSecurityAgentIF;
import com.hyperion.css.common.configuration.*;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * X509CertificateAuthImpl implements the CSSSecurityAgentIF interface It
 * accepts
 * the X509 certificate of the authenticated user from the Web Server via a
 * header, parses the certificate, extracts the DN of the User and
 * authenticates the user.
 */
public class X509CertificateSecurityAgentImpl implements CSSSecurityAgentIF
{
    static final String IDENTITY_ATTR = "CN";
    String g_userDN = null;
    String g_userName = null;
    String hostAddress = null;
    /**
     * Returns the User name (login name) of the authenticated user,
     * for example demouser. See CSS API documentation for more information
     */
}
```

```
    */
    public String getUsername(HttpServletRequest req,
        HttpServletResponse res)
        throws Exception
    {
        hostAddress = req.getServerName();
        String certStr = getCertificate(req);

        String sCert = prepareCertificate(certStr);

        /* Authenticate with a CN */
        parseCertificate(sCert);

        /* Authenticate if the Login Attribute is a DN */
        if (g_username == null)
        {
            throw new Exception("User name not found");
        }
        return g_username;
    }

    /**
     * Passing null since this is a trusted Security agent
     authentication
     * See Security API documentation for more information on
     CSSSecurityAgentIF
     */
    public String getPassword(HttpServletRequest req,
        HttpServletResponse res)
        throws Exception
    {
        return null;
    }

    /**
     * Get the Certificate sent by the Web Server in the HYPLOGIN
     header.
     * If you pass a different header name from the Web server, change
     the
     * name in the method.
     */
    private String getCertificate(HttpServletRequest request)
    {
        String cStr = (String)request
            .getHeader(CSSConfigurationDefaults.HTTP_HEADER_HYPLGI
N);
        return cStr;
    }

    /**
     * The certificate sent by the Web server is a String.
     * Put a "\n" in place of whitespace so that the X509Certificate
     * java API can parse the certificate.
     */
    private String prepareCertificate(String gString)
```

```

    {
        String str1 = null;
        String str2 = null;

        str1 = gString.replace("-----BEGIN CERTIFICATE-----", "");
        str2 = str1.replace("-----END CERTIFICATE-----", "");
        String certStrWithNL = "-----BEGIN CERTIFICATE-----"
            + str2.replace(" ", "\n") + "-----END CERTIFICATE-----";
        return certStrWithNL;
    }

/**
 * Parse the certificate
 * 1. Create X509Certificate using the certificateFactory
 * 2. Get the Principal object from the certificate
 * 3. Set the g_userDN to a certificate attribute value (DN in this
sample)
 * 4. Parse the attribute (DN in this sample) to get a unique username
 */
private void parseCertificate(String sCertificate) throws Exception
{
    X509Certificate cert = null;
    String userID = null;
    try
    {
        X509Certificate clientCert = (X509Certificate)CertificateFactory
            .getInstance("X.509")
            .generateCertificate(
                new
                ByteArrayInputStream(sCertificate
                    .getBytes("UTF-8")));

        if (clientCert != null)
        {
            Principal princDN = clientCert.getSubjectDN();
            String dnStr = princDN.getName();
            g_userDN = dnStr;
            int idx = dnStr.indexOf(",");
            userID = dnStr.substring(3, idx);
            g_userName = userID;
        }
    }
    catch (CertificateException ce)
    {
        throw ce;
    }
    catch (UnsupportedEncodingException uee)
    {
        throw uee;
    }
} //end of getUserFromCert
} // end of class

```

カスタム・ログイン・クラスのデプロイ

カスタム・ログイン・クラスを実装するには、次のステップを実行します:

1. カスタム・ログイン・クラスを作成およびテストします。コードに log4j への参照がないことを確認します。[カスタム・ログイン・クラス・サンプル・コード](#)を参照してください。
カスタム・クラスには任意の名前を使用できます。
2. カスタム・ログイン・クラスを CustomAuth.jar にパッケージします
3. CustomAuth.jar を次のデプロイメントにコピーします:
 - **WebLogic:** CustomAuth.jar を `MIDDLEWARE_HOME/user_projects/domains/WebLogic_DOMAIN/lib` (通常は `Oracle/Middleware/user_projects/domains/EPMSysstem/lib`) にコピーします。

ノート:

カスタム・ログイン・クラスの実装を含んだリリース **11.1.2.0** または **11.1.2.1** からアップグレードしている場合、CustomAuth.jar を `EPM_ORACLE_HOME/common/jlib/11.1.2.0` から `MIDDLEWARE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib` に移動します。

- **クライアント・デプロイメント:** CustomAuth.jar をすべての Oracle Enterprise Performance Management System クライアント・デプロイメントの次の場所にコピーします:

`EPM_ORACLE_HOME/common/jlib/11.1.2.0` (通常は `Oracle/Middleware/common/jlib/11.1.2.0`)

カスタム・ログイン・クラスを使用する場合、クライアント証明書認証を有効にすることをお勧めします。

C

ユーザー・ディレクトリ全体のユーザーとグループの移行

概要

多くのシナリオにおいて、プロビジョニング済の Oracle Enterprise Performance Management System ユーザーのユーザー・アイデンティティおよびグループ・アイデンティティが陳腐化する可能性があります。EPM System コンポーネントは、コンポーネントで使用可能なプロビジョニング情報が陳腐化すると、アクセスできなくなります。陳腐化したプロビジョニング・データが作成される可能性のあるシナリオは次のとおりです:

- ユーザー・ディレクトリの処分: 組織でユーザーを別のユーザー・ディレクトリに移動した後、元のユーザー・ディレクトリを処分する場合があります。
- バージョンのアップグレード: ユーザー・ディレクトリのバージョンをアップグレードすると、ホスト・マシン名またはオペレーティング・システム環境の要件が変わる場合があります。
- ベンダーの変更: 組織で別のベンダーのユーザー・ディレクトリを使用することにしたため、元のユーザー・ディレクトリの使用を打ち切る場合があります。たとえば、組織で Oracle Internet Directory を SunONE Directory Server に切り替えたとします。

ノート:

- この付録では、廃止予定のユーザー・ディレクトリを ソース・ユーザー・ディレクトリと呼び、ユーザー・アカウントの移行先のユーザー・ディレクトリをターゲット・ユーザー・ディレクトリと呼びます。
- この移行手順ではソース・ユーザー・ディレクトリからターゲット・ユーザー・ディレクトリへのユーザー・アカウントの移行はサポートされておらず、EPM アプリケーションの関連付けのみがサポートされます。ユーザーはターゲット・ユーザー・ディレクトリに手動で作成する必要があります。このプロセスは、ネイティブ・ディレクトリを含む、あらゆるソース・ユーザー・ディレクトリのユーザーに適用されます。

Hyperion Shared Services が構成されているソース・ユーザー・ディレクトリにネイティブ・ディレクトリ以外のグループがある場合、それらのグループもターゲット・ユーザー・ディレクトリ内に作成する必要があります。

前提条件

- Oracle Enterprise Performance Management System ユーザーおよびグループのうち、ユーザー・ディレクトリ間でのプロビジョニング・データの移行の対象となるものがターゲット・ユーザー・ディレクトリに存在する必要があります。

ソース・ユーザー・ディレクトリ内にあるグループ関係は、ターゲット・ユーザー・ディレクトリで保持される必要があります。

- **EPM System** ユーザーのユーザー名は、ソース・ユーザー・ディレクトリとターゲット・ユーザー・ディレクトリで同一である必要があります。

移行手順

ネイティブ・ディレクトリ・データのエクスポート

ソース環境で次の手順に従います:

Oracle Hyperion Enterprise Performance Management System ライフサイクル管理を使用して、ネイティブ・ディレクトリから次の共有サービス・アーティファクトのみをエクスポートします:

- ネイティブ・ディレクトリ・グループ
- 割り当てられた役割
- 委任リスト

ライフサイクル管理では、通常、`EPM_ORACLE_INSTANCE/import_export/USER_NAME/EXPORT_DIR/resource/Native Directory` に複数のエクスポート・ファイルを作成します(`USER_NAME` は、`admin` などのエクスポート操作を行うユーザーのアイデンティティで、`EXPORT_DIR` は、エクスポート・ディレクトリの名前)。通常、次のファイルが作成されます:

- `Groups.csv`
- `Assigned Roles.csv`
- `Delegated Lists.csv`
- デプロイされているアプリケーションごとの `Assigned Roles/PROD_NAME.csv` (`PROD_NAME` は、`Shared Services` などの **Oracle Enterprise Performance Management System** コンポーネントの名前)。

ノート:

- ライフサイクル管理を使用したデータのエクスポート手順の詳細は、**Oracle Enterprise Performance Management System** ライフサイクル管理ガイドを参照してください。
- `Users.csv` ファイルがエクスポートされていないことを確認します。

アーティファクトをエクスポートしたら、移行ステータス・レポートで、最後のエクスポート操作のステータスが「完了」と表示されていることを確認します。

ネイティブ・ディレクトリ・データをエクスポートするには:

1. **Oracle Hyperion Shared Services Console** のビュー・ペインで、「**Foundation**」アプリケーション・グループ内の「**Shared Services**」アプリケーションを選択します。

2. 移行するには、次のリストから必要なアーティファクトのみを選択します：
 - ネイティブ・ディレクトリ・グループ
 - 割り当てられた役割
 - 委任リスト
3. 「**エクスポート**」をクリックします。
4. エクスポート・アーカイブの名前を入力します。デフォルトは admin *DATE* です(例: admin 13-03-18)。
5. 「**エクスポート**」をクリックします。

ネイティブ・ディレクトリ・データのインポート

ターゲット環境で次の手順に従います：

1. 手動で作成します：
 - a. ターゲット外部ユーザー・ディレクトリにユーザーを、ソース・ユーザー・ディレクトリと同じように。
 - b. ターゲット外部ユーザー・ディレクトリにグループを、ソース・ユーザー・ディレクトリと同じように(ネイティブ・ディレクトリ・グループを除く)。
2. ターゲット・ユーザー・ディレクトリを構成します。
ソース・ユーザー・ディレクトリから別のユーザー・ディレクトリにユーザー・アカウントを移行した場合、**EPM System** でターゲット・ユーザー・ディレクトリを外部ユーザー・ディレクトリとして追加します。たとえば、**Oracle Internet Directory** から **SunONE Directory Server** にユーザー・アカウントを移行した場合、**SunONE Directory Server** を外部ユーザー・ディレクトリとして追加します。**Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド**の第3章のユーザー・ディレクトリの構成を参照してください。

ノート:

データをソース・ユーザー・ディレクトリから移行する、すべての **EPM System** ユーザーのユーザー・アカウントとグループがターゲット・ユーザー・ディレクトリに含まれていることを確認します。

すでに外部ユーザー・ディレクトリとして定義されているユーザー・ディレクトリにユーザーを移行した場合、ユーザー・アカウントが **Oracle Hyperion Shared Services** からアクセス可能であることを確認します。これは、**Shared Services Console** からユーザーを検索することで実行できます。**Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド**のユーザー、グループ、役割および委任リストの検索を参照してください。

ターゲット・ユーザー・ディレクトリを外部ユーザー・ディレクトリとして構成する際、「ログイン属性」プロパティが、ソース・ユーザー・ディレクトリで元々そのユーザー名の属性値として使用されていた属性を指していることを確認します。[前提条件](#)を参照してください。

3. ターゲット・ユーザー・ディレクトリを検索順の最上部に移動します。

 ノート:

ターゲット・ユーザー・ディレクトリ名がソース・ディレクトリ名と同じ場合、ソース・ユーザー・ディレクトリを EPM System 構成から削除する必要があります。

Shared Services では、新たに追加されたユーザー・ディレクトリに、既存のディレクトリに割り当てられている検索順より低い順序が割り当てられます。ターゲット・ユーザー・ディレクトリの検索順がソース・ユーザー・ディレクトリよりも上になるよう検索順を変更します。この順序によって、Shared Services がソースを検索する前にターゲット・ユーザー・ディレクトリでユーザーを検出できるようになります。Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイドのユーザー・ディレクトリの検索順の管理を参照してください。

4. Oracle Hyperion Foundation Services とその他の EPM System コンポーネントを再起動し、変更を反映します。
5. (ソース環境からエクスポートされた)ネイティブ・ディレクトリ・データをインポートします:
作成/更新オプションを使用してライフサイクル管理を実行し、ネイティブ・ディレクトリからエクスポートしてあった(次にリストされている)データをインポートします。

- Groups.csv
- Assigned Roles.csv
- Delegated Lists.csv

 ノート:

- ライフサイクル管理を使用したデータのインポート手順の詳細は、Oracle Enterprise Performance Management System ライフサイクル管理ガイドを参照してください。
- Users.csv ファイルがインポートされていないことを確認します。

データをインポートしたら、移行ステータス・レポートで、最後のインポート操作のステータスが「完了」と表示されていることを確認します。

ネイティブ・ディレクトリ・データをインポートするには:

- a. Shared Services Console のビュー・ペインで、「**ファイル・システム**」を展開します。
- b. インポート・ファイルのファイル・システムの場所を選択します。
- c. プロビジョニング情報をインポートするアーティファクトのタイプを選択します。
- d. 「**インポート**」をクリックします。
- e. 「**OK**」をクリックします。

個々の製品の更新

▲ 注意:

個々の製品を更新する前に、Oracle Enterprise Performance Management System コンポーネントによって使用されているリポジトリのユーザーとグループのデータをバックアップすることをお勧めします。ローカル製品リポジトリの情報を更新した後は、バックアップからのみ、元のローカル製品リポジトリのユーザーとグループのデータに戻すことができます。

Planning

Oracle Hyperion Planning では、プロビジョニングされたユーザーとグループに関する情報は Planning リポジトリに保管されます。ユーザーとグループをユーザー・ディレクトリ間で移行した結果、ネイティブ・ディレクトリ内のユーザー・アイデンティティが変更された場合、「ユーザーとグループの移行」を選択して、Planning リポジトリの情報とネイティブ・ディレクトリを同期化する必要があります。このボタンは、Planning でデータ・フォーム、メンバーおよびタスク・リストへのアクセス権を割り当てる際に使用できます。

Financial Management

Oracle Hyperion Financial Management では、オブジェクトにアクセスできるようにプロビジョニングされたユーザーとグループに関する情報はローカル Financial Management リポジトリに記録されます。ユーザーとグループをユーザー・ディレクトリ間で移行した結果、ネイティブ・ディレクトリ内のユーザーとグループの情報が変更された場合、Financial Management リポジトリの情報とネイティブ・ディレクトリを同期化する必要があります。