Sun Java System Access Manager 7.1 配備計画ガイド

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–1242
2007 年 3 月

070829@18135

目次

表目次

図目次

はじめに

『Sun Java System Access Manager 7.1 配備計画ガイド』では、ソリューションライフサイクルに基づいて、Sun Java™ System Access Manager のための計画と配備のソリューションについて説明します。

Access Manager は Sun Java Enterprise System (Java ES) のコンポーネントの 1 つで、ネットワーク環境またはインターネット環境に分散したエンタープライズアプリケーションをサポートするために必要なサービスを提供するソフトウェアコンポーネントで構成されています。

対象読者

本書は、Access Manager 配備の計画、分析、および設計を担当する配備設計者およびビジネスプランナ向けに書かれています。また、本書は Access Manager 配備の特定の分野で設計や実装に携わるシステムインテグレータにとっても役立ちます。

お読みになる前に

次のコンポーネントおよび概念に精通することをお勧めします。

- 『Sun Java System Access Manager 7.1 Technical Overview』に記載されている Access Manager の技術的な概念
- 配備先のプラットフォーム: Solaris™、Linux、HP-UX、または Windows オペレーティングシステム
- Access Manager を実行する Web コンテナ: Sun Java System Application Server、Sun Java System Web Server、BEA WebLogic、または IBM WebSphere Application Server
- 技術的な概念: LDAP (Lightweight Directory Access Protocol)、 Java™ テクノロジ、JSP (JavaServer Pages™) テクノロジ、HTTP (HyperText Transfer Protocol)、HTML (HyperText Markup Language)、および XML (eXtensible Markup Language)

内容の紹介

このガイドは、配備計画のさまざまなフェーズについて説明するソリューションライフサイクルに基づいています。第 1 章では、ソリューションライフサイクルの説明を提供します。

関連マニュアル

次の関連マニュアルを利用できます。

- 12 ページの「Access Manager 7.1 マニュアルセット」
- 13 ページの「Sun Java Enterprise System 5 マニュアル」

Access Manager 7.1 マニュアルセット

次の表では、下記 Web サイトから入手できる Access Manager のマニュアルセットについて説明しています。

http://docs.sun.com/coll/1292.2

表 P–1　Access Manager 7.1 マニュアルセット

タイトル	説明
『Sun Java System Access Manager 7.1 Documentation Center』	Access Manager 7.1 マニュアルコレクションでよく参照される情報へのリンクを提供します。
『Sun Java System Access Manager 7.1 リリースノート』	新しい機能、修正された問題、インストールに関する注意事項、既知の問題および制限事項について説明します。リリースノートは、パッチ、新機能、問題などを説明する目的で、初回のリリース後定期的に更新されます。
『Sun Java System Access Manager 7.1 Technical Overview』	Access Manager の基本的な概念や用語について説明します。また、アクセス制御機能を統合したり、企業の資産や Web ベースアプリケーションを保護したりするために、Access Manager のコンポーネントがどのように連携するかの概要を提供します。
『Sun Java System Access Manager 7.1 配備計画ガイド』 (このガイド)	ソリューションライフサイクルに基づく、Access Manager の計画および配備のソリューションを提供します。

表 P–1　Access Manager 7.1 マニュアルセット　(続き)

タイトル	説明
『Sun Java System Access Manager 7.1 Postinstallation Guide』	インストール後の Access Manager の設定についての情報を提供します。通常、インストール後のタスクを実行するのは数回だけです。たとえば、Access Manager の追加インスタンスを配備したり、セッションフェイルオーバー用に Access Manager を設定することが必要な場合があります。
『Sun Java System Access Manager 7.1 管理ガイド』	レルム管理、ポリシー管理、認証、ディレクトリ管理などのさまざまな管理タスクについて説明します。
『Sun Java System Access Manager 7.1 Administration Reference』	Access Manager のコマンド行インタフェース (CLI)、設定属性、`AMConfig.properties` の属性、`serverconfig.xml` ファイルの属性、ログファイル、およびエラーコードのリファレンス情報を提供します。
『Sun Java System Access Manager 7.1 Federation and SAML Administration Guide』	Liberty Alliance Project 仕様に基づく Federation Manager についての情報を提供します。その中には、この仕様に基づいた統合サービスに関する情報、Liberty ベースの環境を有効にするための手順、およびフレームワークを拡張するためのアプリケーションプログラミングインタフェース (API) の要約が含まれます。
『Sun Java System Access Manager 7.1 Developer's Guide』	Access Manager のカスタマイズと、その機能を組織の現在の技術インフラストラクチャーに統合する方法についての情報を提供します。製品のプログラミング上の側面および API についても詳しく説明します。
『Sun Java System Access Manager 7.1 C API Reference』	Access Manager 公開 C API を構成するデータ型、構造、および関数の要約を提供します。
『Sun Java System Access Manager 7.1 Java API Reference』	Access Manager での Java パッケージの実装についての情報を提供します。
『Sun Java System Access Manager 7.1 Performance Tuning Guide』	最適なパフォーマンスのために Access Manager とその関連コンポーネントを調整する方法についての情報を提供します。
『Sun Java System Access Manager Policy Agent 2.2 User's Guide』	現在利用可能な Web エージェントおよび J2EE エージェントを含む、Policy Agent ソフトウェアの概要を提供します。Access Manager Policy Agent 2.2 のマニュアルコレクションを表示するには、次のサイトを参照してください。 `http://docs.sun.com/coll/1322.1`

Sun Java Enterprise System 5 マニュアル

次の表に、関連する Java ES 製品のマニュアルコレクションへのリンクを示します。

表 P–2　関連する Sun Java Enterprise System 5 マニュアル

製品	リンク
Sun Java Enterprise System 5	http://docs.sun.com/coll/1657.1
Sun Java System Directory Server Enterprise Edition 6.0	http://docs.sun.com/coll/1660.1
Sun Java System Web Server 7.0	http://docs.sun.com/coll/1664.1
Sun Java System Application Server Enterprise Edition 8.2	http://docs.sun.com/coll/1659.1
Sun Java System Message Queue 3.7 UR1	http://docs.sun.com/coll/1661.1
Sun Java System Web Proxy Server 4.0.4	http://docs.sun.com/coll/1665.1
Sun Java System Identity Manager 7	http://docs.sun.com/coll/1704.1

Sun 製品マニュアルの検索

Sun 製品マニュアルは docs.sun.com℠ Web サイトで検索できるだけでなく、検索エンジンの検索フィールドに次の構文を入力することによっても検索できます。

search-term `site:docs.sun.com`

たとえば、「ブローカー」を検索するには次のように入力します。

ブローカー `site:docs.sun.com`

その他の Sun Web サイト (たとえば、java.sun.com、www.sun.com、developers.sun.com) を検索に含めるには、検索フィールドで `docs.sun.com` の代わりに `sun.com` を使用します。

サードパーティーの関連 Web サイト

このリリースノートに掲載されているサードパーティーの URL を参照すると、追加および関連情報を入手できます。

注 – Sun は、本書に記載されたサードパーティーの Web サイトの有効性について責任を負いません。Sun は、これらのサイトまたはリソースを通じて入手可能なコンテンツ、広告、製品、その他の内容についていかなる保証もせず、かつ責任や義務を負いません。Sun は、これらのサイトやリソースを通じて入手したコンテンツ、製品、またはサービスを使用または信頼することに起因または関連する、またはそう主張された、現実のまたは主張されたいかなる損害や損失についても責任や義務を負わないものとします。

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	内容
マニュアル	http://jp.sun.com/documentation/	PDF 文書および HTML 文書をダウンロードできます。
サポートおよびトレーニング	http://jp.sun.com/supportraining/	技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P–3　表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 machine_name% you have mail.
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	machine_name% **su** Password:
aabbcc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *filename* と入力します。

表 P–3 表記上の規則 *(続き)*

字体または記号	意味	例
『』	参照する書名を示します。	『コードマネージャー・ユーザーズガイド』を参照してください。
「」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	`sun% grep '^#define \` `XV_VERSION_STRING'`

コード例は次のように表示されます。

- C シェル

  machine_name% **command y|n** [*filename*]

- C シェルのスーパーユーザー

  machine_name# **command y|n** [*filename*]

- Bourne シェルおよび Korn シェル

  $ **command y|n** [*filename*]

- Bourne シェルおよび Korn シェルのスーパーユーザー

  # **command y|n** [*filename*]

[ ] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

ご意見、ご要望をお寄せください

Sun はマニュアルをより良いものにするために、ご意見やご提案をお待ちしております。

ご意見をお寄せいただく場合は、http://docs.sun.com にアクセスし、「コメントの送信」をクリックしてください。オンラインフォームの場合は、マニュアルのタイトルとパーツ番号を指定してください。パーツ番号は、マニュアルのタイトルページまたはマニュアルの上部にある7 桁または9 桁の番号です。

たとえば、本書のタイトルは、『Sun Java System Access Manager 7.1 配備計画ガイド』で、パーツ番号は 820-1242 です。

◆ ◆ ◆ 第 1 章

1 Access Managerの配備計画について

Sun Java™ System Access Manager (Access Manager) は Sun アイデンティティー管理インフラストラクチャーの一部を成すもので、エンタープライズ内および企業間 (B2B) のバリューチェーンにおいて、組織が Web アプリケーションやほかのリソースへのセキュリティー保護されたアクセスを管理できるようにします。この章では、 Access Manager の配備を計画する際の基本的な方針について説明します。次の節で構成されています。

Access Manager について

Access Manager は Sun Java Enterprise System (Java ES) のコンポーネントの 1 つで、ネットワーク環境またはインターネット環境に分散したエンタープライズアプリケーションをサポートするためのサービスを提供するソフトウェアコンポーネントで構成されています。Access Manager には、次の主な機能が備えられています。

- ロールとルールの両方に基づくアクセス制御を使用した集中化された認証および認証サービス
- 組織の Web ベースアプリケーションへのシングルサインオン (SSO) アクセス
- Liberty Alliance Project および SAML (Security Assertions Markup Language) と連携したアイデンティティーサポート
- 管理者およびユーザーのアクティビティーを含む重要な情報の Access Manager コンポーネントによるロギング。この記録をあとで分析、報告、監査に使用できます。ロギングは J2SE ロギング API (`java.util.logging`) をベースに実行されます。

Access Manager は、ディレクトリサービス、アクセス管理、プロビジョニング、連携などのアイデンティティー情報の利用、共有、および管理を行うために必要な機能を提供する Sun アイデンティティー管理製品群の一部でもあります。アイデンティティー管理製品群には次の製品があります。

- Sun Java System Access Manager
- Sun Java System Directory Server Enterprise Edition
- Sun Java System Federation Manager
- Sun Java System Identity Manager

各コンポーネントの詳細については、次の Sun ソフトウェア Web サイトを参照してください。http://www.sun.com/software/

次の図に、Access Manager、Identity Manager、および Directory Server のアイデンティティー管理コンポーネントを示します。

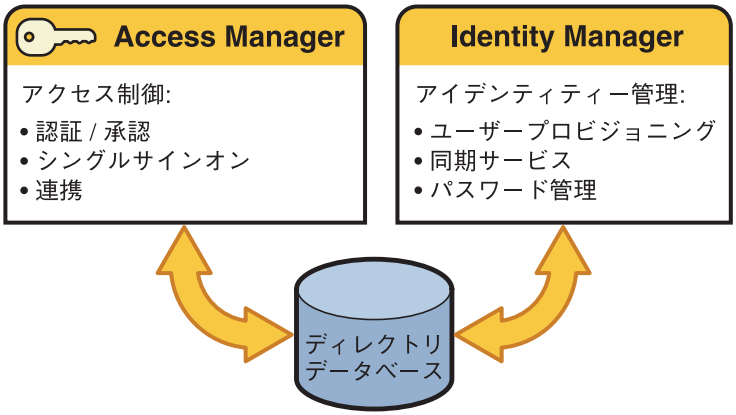


図 1–1　Sun アイデンティティー管理コンポーネント

Sun Java System Identity Manager には、ユーザープロビジョニング、パスワード管理、同期サービス、包括的な監査とレポート作成、および管理の委任といった機能があります。Identity Manager は Sun Java Enterprise System のコンポーネントではありません。Identity Manager をご使用の配備環境で使用する場合、または詳細情報の入手をご希望の場合は、Sun Microsystems の技術担当者、または次のサイトにある Sun 事業所にご連絡ください。http://www.sun.com/sales-n-service/WWSales.jsp

Access Manager に関する詳細な説明については、『Sun Java System Access Manager 7.1 Technical Overview』を参照してください。

Access Manager の配備計画

アイデンティティー管理ソリューションの実装を成功させる上で、配備計画の段階は非常に重要です。目標、要件、そして優先事項は、企業によって異なります。配備計画がうまくいくかどうかは、入念な準備、分析、および設計にかかっています。計画作業のどこかにミスや手違いがあると、いろいろな面でシステムに不具合が生じることになりかねません。システムの計画がずさんなために、重大な問題が起きる可能性もあります。たとえば、システムのパフォーマンスが上がらない、維

持管理が面倒である、操作コストがかかりすぎる、リソースに無駄がある、需要の増大に見合った拡張ができないといった問題が生じることがあります。

このマニュアルで説明する Access Manager 配備計画は、ソリューションのライフサイクルに沿ったものです。ソリューションのライフサイクルには、Java Enterprise System をベースにした Access Manager エンタープライズソフトウェアソリューションを計画し、設計し、実装するという各プロセスが含まれます。

ソリューションのライフサイクル

次の図に示されているソリューションのライフサイクルは、配備プロジェクトの計画 と進捗管理の手だてとして役立ちます。このライフサイクル構造には、配備計画を成功させるために必要な準備、分析、および設計が一連の段階として順番に組み込まれています。それぞれの段階は関連するいくつかのタスクからなり、1 つの段階で行われたタスクのアウトプットが次の段階のタスクのインプットとなります。それぞれの段階で行われるタスクを繰り返し行い、分析と設計を尽くした結果としてその段階のアウトプットを生成することが必要となります。

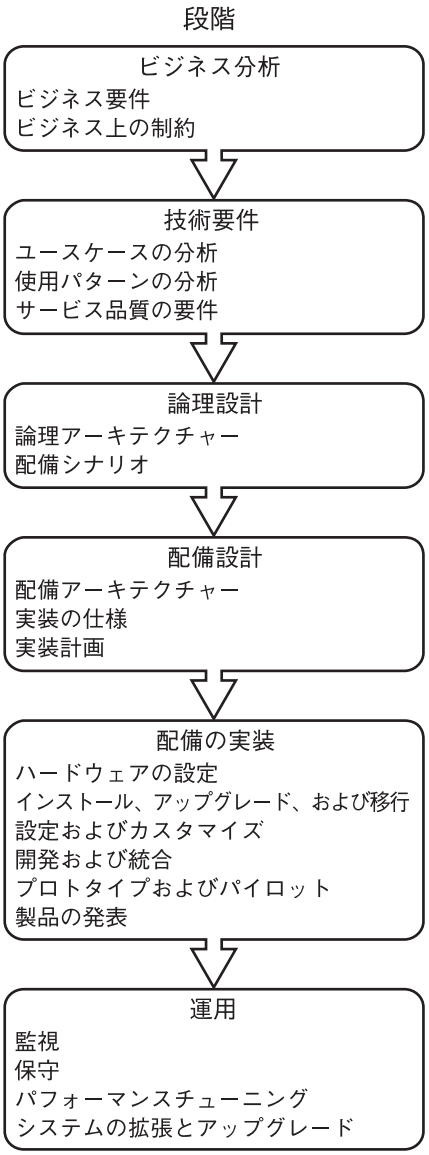


図1–2 ソリューションのライフサイクル

このマニュアルは、ソリューションのライフサイクルにおける各段階に基づいた構成になっています。この章の次の項では、ライフサイクルの各段階の要約を示します。これらの段階の詳細な説明については、『Sun Java Enterprise System 2006Q3 Deployment Planning Guide』を参照してください。

ビジネス分析段階

ビジネス分析を行う際には、配備プロジェクトの業務目標を定義し、その目標を達成するために満たす必要のあるビジネス要件を明示します。ビジネス要件を明示するにあたっては、業務目標の達成に影響を与える可能性のあるビジネス上の制約すべてを考慮します。適正なビジネス分析を行わないと、ソリューションが不完全になる危険性があります。

ビジネス分析段階の間に、ビジネス要件に関する資料を作成し、あとで技術要件段階のタスクのインプットとして使用できるようにしておきます。

第2章を参照してください。

技術要件段階

技術要件の段階では、ビジネス分析段階で定義したビジネス要件とビジネス上の制約の処理から始まり、それらの情報を次の配備アーキテクチャーの設計に使用できる技術仕様に書き換えます。技術要件では、パフォーマンス、可用性、セキュリティーなどのQoS (Quality of Service) 特性を指定します。

技術要件段階の間に、次の情報を含む資料を作成します。

- ユーザーのタスクと使用状況パターンの分析
- 計画しているシステムでのユーザー対話のユースケース
- ビジネス要件から導き出したQoS要件。ユーザーのタスクと使用状況パターンの分析を考慮に入れることも可能

作成した使用状況分析、ユースケース、およびQoS要件の資料は、ソリューションのライフサイクルにおける論理設計のインプットとなります。使用状況分析は、配備設計段階でも重要な役割を果たします。

第3章を参照してください。

論理設計段階

論理設計では、技術要件段階で作成したユースケースを入力として使用し、ソリューションを実装するために必要なAccess Managerコンポーネントを特定します。Java ESコンポーネントをサポートするコンポーネントや、ビジネス要件を満たすために必要な独自開発のコンポーネントも指定します。次に、それらのコンポーネントを、コンポーネント間の相互関係を示す論理アーキテクチャーにマップします。論理アーキテクチャー上には、ソリューションの実装に必要なハードウェアの指定は行いません。

論理設計段階のアウトプットが論理アーキテクチャーです。論理アーキテクチャーと技術要件段階で作成したQoS要件とによって配備シナリオが作成され、このシナリオが配備設計段階でのインプットとなります。

第4章を参照してください。

配備設計段階

配備設計では、論理アーキテクチャーに指定したコンポーネントを物理的な環境に対応させ、ハイレベルの配備アーキテクチャーを作成します。配備アーキテクチャーを構築する方法のローレベルの詳細を具体的に指定する、実装仕様も作成します。また、ソフトウェアソリューションの実装に関係するさまざまな面を詳述した一連の計画と仕様も作成します。

この配備設計段階でプロジェクトの承認が行われます。プロジェクトの承認にあたっては、配備のコストが評価されます。承認が下りたら、配備の実装に関わる契約を結び、プロジェクトを構築するためのリソースを確保します。通常、プロジェクトの承認は、実装仕様が細部まで明らかになったあとに行われます。しかし、配備アーキテクチャーが完成した時点で承認が下りる場合もあります。

配備設計段階のアウトプットには、次の内容が含まれます。

- 配備アーキテクチャー。コンポーネントとネットワーク上のハードウェアおよびソフトウェアとの対応付けを示すハイレベルの設計資料。
- 実装仕様。配備構築の設計図として使用する詳細な仕様。
- 実装計画。エンタープライズソフトウェアソリューションの実装におけるさまざまな面を網羅した一連の計画と仕様。実装計画には、移行計画、インストール計画、ユーザー管理計画、テスト計画などが含まれます。

第5章を参照してください。

実装段階

実装段階では、配備設計で作成した仕様と計画に基づいて、配備アーキテクチャーを構築し、ソリューションを実装します。

第6章を参照してください。

◆ ◆ ◆ 第 2 章

Access Manager のためのビジネス分析

Sun Java™ System Access Manager を使用すると、組織は、従業員、請負業者、顧客、およびサプライヤのためのアイデンティティー管理ソリューションを配備できます。ソリューションライフサイクルのビジネス分析段階では、ビジネス上の問題点を分析することでビジネス上の目標を定義し、その目標を達成するにあたってのビジネス要件とビジネス制約を明確にします。この章で説明する内容は、次のとおりです。

ビジネス分析について

ビジネス分析は、ビジネス目標を明らかにすることから始まります。次に、解決が必要なビジネス上の問題を分析し、ビジネス目標を達成するために満たす必要のあるビジネス要件を特定します。また、目標の達成能力を制限するビジネス制約がある場合は、それらの制約についても考慮します。ビジネス要件とビジネス制約を分析することにより、一連のビジネス要件ドキュメントを作成します。

ここで作成された一連のビジネス要件ドキュメントを、技術要件段階で技術要件を導き出すための基盤として使用します。ソリューションライフサイクル全体にわたり、このビジネス分析段階で実施された分析に従って、配備計画、そして最終的にはソリューションが成功するかどうかを判定します。

Access Manager ビジネス要件の定義

ここでは、Access Manager のために考慮する必要のある特定のビジネス要件、つまり、Access Manager ソリューションに必要なビジネス要件について説明します。

Sun Java System Access Manager は、複雑な分散型アイデンティティー管理システムであり、適切な配備によって、企業の組織にまたがる広範なデータおよびサービスへのアクセスがセキュリティー保護されます。企業リソースの適正な制御を確実にするため、配備プロセスの適切な計画が必要になります。この章では、配備の計画方法について説明します。次の節で構成されています。

リソースの定義

アイデンティティー管理ソリューションには、組織全体にわたる多様なシステムが関係するため、Access Manager を適切に配備するにはさまざまなリソースが必要になります。配備プロセスに関係する、または必要とされる企業のリソースを、以下に示します。

人事情報

組織内のさまざまな取引関係および政治的な関係を考慮する必要があります。直接的または多面的な報告体制を備えたチームを編成する必要があります。通常、Access Manager の配備は、1 人のプロジェクトマネージャーと数人の専任システム管理者で構成される小規模なチームで行われます。これらの人々は、チームリーダー、さらには多数の関連プロジェクトの責任を担うオーナーに報告を行います。また、経営権を持つスポンサーに直接報告することもよくあります。このグループは往々にして、Sun の技術リソースや、必要に応じて使用される LOB アプリケーション管理者で構成される仮想のチームメンバーによって増強されます。

この構成では実際のニーズと完全には一致しない可能性もありますが、ほぼ一般的な配備チームモデルを表しています。個々の役割を必ずしも明確に識別する必要はありませんが、さまざまなスキルセットを表す次の抽象技術ロールを使用することで、Access Manager の典型的な配備チームをさらに定義できます。

経営権を持つスポンサー

従来より、アイデンティティー管理の配備を成功させるには、組織的および政治的な境界を超えた企業の決定権を持つ人物の承認やサポートが必要です。このため、経営権を持つスポンサーが管理に関係することは重要です。プランニングのためのミーティングは、配備に関する既得権を保持する人物からの見識を得る上で重要なプロセスです。プロジェクト計画を策定する際、成果が全体としての企業目標に沿っていることを確認してください。たとえば、コスト削減が主な経営目標である場合、現在のアイデンティティー管理コストに関する統計を収集し、パスワードリセット関連のヘルプデスク利用コストなどのコストを判断します。具体的な統計が入手できれば、配備チームが経営陣のサポートを得るために必要な、明確な投資収益率 (ROI) の定義に役立ちます。関連するほかの問題には、以下が含まれます。

- だれがアイデンティティー管理の配備からメリットを得るか
- アイデンティティー管理ソリューションは、どのような組織上の問題を解決するか
- 配備を遅らせる可能性のある内部的な課題にどのように取り組んでいくか

多くの場合、アイデンティティー管理の概念や Access Manager 配備の価値を、ほかの経営陣にも納得させることが必要になります。経営面および技術面の「エバンジェリスト」は、新しいインフラストラクチャーを経営陣に売り込み、統合化への要求を喚起したり、インフラストラクチャーの変更を受け入れさせて、最終的な成功を収めるのに寄与します。

チームリーダー

プロジェクトの成功に責任を持つ当事者として、チームリーダーを 1 人選ぶ必要があります。このチームリーダーは、プロジェクトの目標を達成するという責任を担い、そのための権限を持ちます。このチームリーダーは、テクニカルリーダー、プロジェクトマネージャー、執行責任者などに論理的に分散されたロールである場合があります。このロールをどのように定義するとしても、その目標は配備プロセスが着実に前進し、成功していることを示して、経営陣の支援を維持することです。

プロジェクト管理

プロジェクトマネージャーは、スケジュールの調整を担当します。また、利用可能なサービス、コア IT グループの提供するサポート、およびさまざまな事業分野 (LOB) のアプリケーション統合に関連するスケジュールを管理します。この役割を担う担当者は、優れたコミュニケーション能力を持ち、企業の政治的側面を理解できる必要があります。プロジェクトマネージャーはまた、環境に追加される新規アプリケーションが円滑に機能するために、内部顧客のニーズとリソースの利用状況とのバランスを取る必要もあります。

LOB アプリケーションは、組織の運営に欠かすことのできないものです。通常、これらは、データベースおよびデータベース管理システムとの接続機能を持つ大規模なプログラムです。会計、サプライチェーン管理、およびリソース計画アプリケー

ションがこれに含まれます。現在、LOB アプリケーションは、ユーザーインタフェースを備えたネットワークアプリケーションや、電子メールや住所録などの個人向けアプリケーションとの接続機能を持つようになりつつあります。

システムアナリスト

システムアナリストは、Access Manager 配備に統合されるさまざまなデータやサービスの評価および分類を担当します。システムアナリストは、LOB アプリケーションのオーナーにインタビューを行い、プラットフォーム、アーキテクチャー、およびその配備スケジュールの技術要件に関する詳細情報を収集します。システムアナリストは、この情報を使用して顧客の要件を満たすようにアプリケーションを配備に統合する方法を計画します。システムアナリストは、さまざまなアプリケーションのアーキテクチャーおよびプラットフォームに関する広範な知識を持つ、IT ゼネラリストでなければなりません。また、Access Manager のアーキテクチャー、サービス、エージェント、および API に関する詳細な知識も必要になります。

事業分野別 (LOB) アプリケーション管理者

LOB アプリケーション管理者は、LOB アプリケーションに関する詳細な知識を持つと同時に、それらのアプリケーションを管理することのできる技術の専門家であり、Access Manager ポリシーエージェント (ポリシー適用ポイント) のアプリケーションへの統合を担当します。LOB アプリケーション管理者には、LOB アプリケーションのアーキテクチャー、統合ポイント、および適切なスケジュールに関する明確な意思伝達能力が必要です。通常、LOB アプリケーション管理者は、Access Manager ポリシーに示されるアクセス制御モデルの定義を担当します。この人物は、カスタムプログラミングを行って、Access Manager とそのアプリケーション (セッション調整など) 間の統合を拡張することもあります。さらに、通常は品質保証 (QA) および新たに配備された環境でのアプリケーションの回帰試験を担当します。

システム管理者

Access Manager を配備し、その可用性を維持する上で、適切なリソースが存在することは重要です。以下に示すレベルで、システム管理者が必要です。補助的な管理者には、Access Manager の配備先ソフトウェアコンテナの配備およびパフォーマンスを担当する Web コンテナ管理者も含まれます。

Access Manager 管理者

Access Manager 管理者は、Access Manager の配備と保守を担当します。この管理者は、共通サービスの可用性を保証し、一般的にインフラストラクチャーに必要な拡張を加え、特にポリシーとロールを設定します。また、ガイドラインを策定して統合作業のサポートに役立てるとともに、LOB アプリケーション管理者への技術サポートも行います。Java、XML、LDAP、HTTP、および Web アプリケーションアーキテクチャーの理解が必須です。

Directory Server 管理者

Access Manager の配備を考慮する前から、認証および承認に使用される企業のディレクトリサービスが組織内のグループにより管理されていることがよくあります。Directory Server 管理者は、ディレクトリサービスの可用性の管理だけでなく、アイデンティティー管理インフラストラクチャーのサポートに必要な変更も含め、現在定義されているLDAP スキーマやアイデンティティーデータへの追加や変更の受け入れおよび統合も担当します。

ハードウェア、データセンター、ネットワーク管理者

大規模な組織は、通常、ハードウェア、オペレーティングシステム、データセンター、およびネットワーク管理をミドルウェア管理から切り離すことでスケールメリットを追求します。この種の企業の場合、これらの管理者間で明確に意思を疎通させることが重要になります。配備を成功させる上で、特定のマシンにアクセスできることや、特定のネットワーク構成を確立することが重要な場合があります。これらの管理者に常にプロジェクトの主要管理点や要件を意識させることで、スムーズなロールアウトが可能になります。

独立系ソフトウェアベンダー

Sun Microsystems およびほかの独立系ソフトウェアベンダー (ISV) は、Access Manager の配備を成功させる上で重要なパートナーです。パッケージソフトウェアを購入することで、企業は複数の組織にまたがるソフトウェア開発を行うコストとリスクを軽減および分散させることができます。

ISV は、1 つ以上のタイプのコンピュータハードウェアまたはオペレーティングシステムプラットフォームで実行可能な製品を製造および販売します。プラットフォームを製造する企業 (Sun、IBM、Hewlett-Packard、Apple、Microsoft など) は、ISV を支援し、サポートを提供します。

ISV に関係する当事者すべてにとって最大の関心事は、協力関係を強化して配備を成功させることです。Sun テクニカルサービスやほかの ISV に働きかけ、プロジェクトの立ち上げの支援や、以前の Access Manager 配備で得た知識を提供してもらってください。対策チーム (Access Manager のエンジニアと配備チームとの仲介役として機能できる) と率直な討議を行うとともに、テクニカルサービスを利用すると、投資を有効に活用したり、配備を成功させるのに役立ちます。

提携先のサードパーティー

Access Manager の連携管理機能を活用することを計画している場合、外部パートナーや提携しているサードパーティーと共同して作業を行います。連携管理機能の初期配備を、独自の内部配備とともに考慮してください。この場合、重要なのは、提供

するビジネス機能を保持するLOBアプリケーションを含めること、および当事者すべての技術リソースとの通信を管理することです。弁護士も、関係する当事者間の良好な関係を築く助けになります。

資金の調達

多くの場合、配備プロジェクトのコスト面の責任はコアITグループが担います。実際、内部資金をLOBアプリケーションからコアグループに移して、アイデンティティー管理プロジェクトの資金の一部にあてるのが一般的な方法です。ただし、単一のLOBアプリケーショングループが内部資金を提供する場合でも、より大きな組織のニーズと資金調達グループのニーズとのバランスを取る必要があります。

目標の設定

組織は、目標を設定することにより、Access Manager配備が完了したあとのあるべき状態を定義します。配備の戦略とは、これらの目標に達するためのロードマップを計画し、目標に向かって進むことです。目標は、関係する当事者すべてが期待することを定義し、プロセスの初期にその承認を得て作成します。

一般に、アイデンティティー管理ソリューションでは、セキュリティーを拡張し、インフラストラクチャーの管理機能を向上させるとともに、コストを削減します。より具体的にいえば、Access Managerが組織に設定を許可する一般的な目標(およびそのメリット)には、以下が含まれます。

- 予想されるデジタルアイデンティティー(従業員、パートナー、顧客)の増加に対応する、スケーラブルなインフラストラクチャーを実装する
- アイデンティティープロファイルの作成および管理を、独自データを制御する各グループと統合する
- ベンダーの統合、ユーザーの自己管理、および関連する管理コストによりコストを削減する
- アイデンティティープロファイルの迅速な終了により、セキュリティーを改善する
- セキュリティーモデルおよびアクセス権の透過性を改善する
- クリティカルシステムへのアクセスに必要な時間を短縮する
- 組織内部のロールまたは連携が変更されたら、クリティカルシステムへのユーザー権限を削除する

最終的に、これらの目標を、関係するグループすべてのモチベーションの理解および実地調査から得られた情報と結び付けて、配備用のインフラストラクチャーの設計に使用できます。また、これらを配備プロセス全体で使用して、当事者の関係を維持し、プロジェクトの支持を得られるようにします。

情報の収集

実地調査を行い、配備に統合するアプリケーションやデータストアに関する情報を収集できます。さらに、これらの部門間の情報収集は、特定の機能および目標を定義し、関係するグループのモチベーションの理解を深めるのに役立ちます。収集が済んだら、情報を設計の青写真として、および経営権を持つスポンサーから確実な承認を得るために活用できます。実地調査の際、以下のグループの支援を得られます。

- ユーザーは、日常業務で使用しているアプリケーションに関するフィードバックを提供する
- 人事担当者は、雇用および雇用終了処理に関する情報を提供する
- サポート担当者は、組織の境界をまたがる問題に関して貴重な情報を提供する
- アプリケーション管理者および開発者は、配備に統合する事業分野別 (LOB) アプリケーションに関する技術情報を提供できる
- ネットワーク管理者は、組織のパフォーマンスや標準に関する技術的基盤の知識を保持している

初期調査には、次の項目に関する情報の収集が含まれます。

- 31 ページの「ビジネスプロセス」
- 32 ページの「IT インフラストラクチャー」
- 33 ページの「仮想データ」

ビジネスプロセス

ビジネスプロセスとは、組織内のさまざまなグループがそれぞれの業務の実行を定義する手順です。プロセスには、次の手順を含めることができます。

- 給与の支給
- 購買および買掛金勘定
- 従業員の出張の承認
- 部門の予算管理
- 従業員の雇用終了

通常、これらのプロセスは、業務単位ごとに使用されるアプリケーションによりサポートされるため、これらのプロセスの評価は必須です。考慮すべき内容は、次のとおりです。

- 現在のプロセスで遅延が発生するかどうか
- 同じ機能を実行する異なるプロセスが多数存在するかどうか
- 業務単位の境界をまたがってプロセスを標準化できるかどうか
- プロセスはどの程度複雑か。プロセスを集約したり簡略化したりできるか
- 現在のプロセスで組織上の変更を処理できるか

プロセスに加える変更はすべて、配備を開始する前に行う必要があります。

IT インフラストラクチャー

IT インフラストラクチャーには、Access Manager の配備に統合されるすべてのハードウェアサーバー、オペレーティングシステム、および統合化アプリケーションが含まれます。次の点について考慮します。

- Access Manager を利用するアプリケーション

  アプリケーションには、人事および会計用のアプリケーションなどの重要な内部アプリケーション、またはあまり重要性の高くない従業員のポータルを含めることができます。また、Access Manager の機能を利用するアプリケーションとして、機密性の高い財務情報と機密性の高くない販売物資の両方を処理する外部 B2B アプリケーション、またはクレジットカードデータや購入履歴に関係する B2C のショッピングカートがあります。

- Access Manager を利用するシステム

  アプリケーションの配備先のハードウェアとそのオペレーティングシステムについて考慮してください。Access Manager の配備には、アプリケーションを実行するための Web コンテナ、Sun Java System Directory Server (または既存のデータストア)、および Access Manager が最低限含まれます。また、企業リソースを使用して独自の Web コンテナを実行する、追加のハードウェアサーバーが含まれることもあります。さらに、セキュリティーを向上させるために、そこに Access Manager ポリシーエージェントをインストールすることもできます。

- 各部門が利用する Access Manager のサービス

  Access Manager 内に統合するデフォルトのサービスやカスタムサービスについて考慮します。ロールおよびポリシーの戦略は、部門ごとに割り当て、定義する必要があります。認証モジュールを評価する必要があります。カスタムサービスが存在する場合はそれを整備する必要があります。

さらに考慮する必要のある技術的な内容は、次のとおりです。

- インフラストラクチャー内に非互換性が存在するかどうか
- 現在のシステムで性能低下やダウンタイムが発生しているかどうか
- アプリケーションは十分にセキュリティー保護されているか
- ウイルスを制御する手段が存在するか
- アプリケーションは、ユーザーの資格に基づいてカスタマイズ可能か

詳細は、34 ページの「アプリケーションの評価」を参照してください。

仮想データ

仮想データとは、Access Manager にアクセスするプロファイル、Access Manager からアクセス可能な設定、およびAccess Manager によってセキュリティー保護されるデータなど、あらゆる状況で利用されるデータを指します。仮想データには、ユーザープロファイル (従業員、顧客など)、データおよびサービスアクセスルール、およびほかのタイプの企業データが含まれます。ただし、含まれるデータはこれだけに限定されるものではありません。

- Access Manager で保護する対象

  Access Manager は、すべての種類のデータおよびサービスへのアクセスをセキュリティー保護します。管理者は、Access Manager データの表示や設定が可能なユーザーを制限したり、アプリケーション、ポータル、およびサービスへのアクセスを制御できます。

- Access Manager を利用するユーザー

  ユーザーには、従業員、ビジネスパートナー、サプライヤ、現在の顧客、および潜在顧客が含まれます。各ユーザーが保持するプロファイルには、最低限、ユーザー ID とパスワードが含まれます。従業員は、外部の販売情報にアクセスする顧客よりも、明らかにより広範で機密性の高いプロファイルを保持します。

- アクセス可能なデータ

  データには、公開情報、内部情報、機密情報、秘密データなどが含まれます。さらに、外部 Web サイトの販売情報、従業員の機密プロファイル、企業のリソースを保護するアクセスルール、サーバー設定情報、連携顧客プロファイルなどが含まれる場合もあります。

- 信頼すべきデータの入手元

  多くの場合、異なる種類のデータを定義する複数のスキーマが使用されます。これらの定義を、配備内で調和させる必要があります。データの所有権の問題を銘記しておき、必要な場合には、さまざまな LOB アプリケーションがデータの制御を維持できるようにしてください。より大規模な組織にはすべてのサービスが重要であるため、企業全体を代表するサービスを提供するために、サテライトグループの需要のバランスを取ることが重要です。

さらに考慮する必要のある技術的な内容は、次のとおりです。

- 複数の属性内で同じ情報が定義されているか
- ユーザーは組織の境界をまたがる複数のプロファイルを保持しているか
- データストアは、ファイアウォールの内側に存在するか
- データは異なるデータストア間で一貫しているか
- 新規データが追加されたり、既存のデータが変更されたりする頻度はどれくらいか

詳細は、36 ページの「データの分類」を参照してください。

アプリケーションの評価

アイデンティティー管理サービスは、一般に、拡張されたシステムを構成する企業および業務単位向けアプリケーションを備えた、集中管理された IT 機能として提供されます。このシステム階層の維持には、サーバーインフラストラクチャーを管理および保守するコア IT グループ、および LOB アプリケーションを保守する従業員のサテライトグループが関係します。

大規模な組織には、たいてい、数百 (または数千) の内部アプリケーションがあります。それらのすべてを評価するには時間と費用がかかります。アプリケーションの調査を実行する場合は、次の条件を満たすアプリケーションを集中的に調査してください。

- 組織にとって特に大きな価値を持つアプリケーション
- シングルサインオンインフラストラクチャーへの統合で、メリットが期待されるアプリケーション
- 組織内部の標準的なプログラミングおよび配備プラットフォームを示すアプリケーション
- 通常、アイデンティティー管理インフラストラクチャーに受け入れられるアプリケーション
- 現在、配備の初期段階にあり、論理的に Access Manager の配備とスケジュールが一致する可能性のあるアプリケーション

スプレッドシートを作成すると、最も将来性の高いアプリケーションから得られた情報の整理に活用できます。全体的な測定基準を策定して、アプリケーション間の統合化の複雑性を比較できます。この測定基準により、アプリケーションがどの程度配備に適しているかを判断できます。適合性の高いアプリケーションの例は、セキュリティー目的で Access Manager ポリシーエージェントがインストールされたアプリケーションサーバーに認証を委任する Web アプリケーションです。すべてのユーザー情報は、LDAP ディレクトリに格納されます。

適合性の低いアプリケーションの例には、メインフレームコンピュータ上で動作する、テキストベースのインタフェースを備えたアプリケーションがあります。この場合、メインフレームアプリケーションの新しいバージョンを待つ間に、ほかのアプリケーションを統合する方が好都合です。

次の節では、組織のアプリケーションを評価する際に収集可能な情報の種類について説明します。この手順は、保護されるリソースを判別するのにも役立ちます。

プラットフォームの情報

既存のテクノロジおよびハードウェアに基づく一般的なプラットフォーム情報を使用して、アプリケーションが統合化に適切かどうかを評価できます。収集されるプラットフォーム情報には、次のものが含まれます。

- アプリケーションが動作するオペレーティングシステム (バージョンを含む)
- アプリケーションが動作する Web コンテナ (バージョンを含む)
- アプリケーションの開発に使用したプログラミングモデル(Java、ASP/.NET、C など)
- アプリケーションをアップグレードする計画があるかどうか。あるとすれば、そのスケジュールはいつか

LOB アプリケーションも、サードパーティー製のアプリケーション (ポータル、コンテンツ管理データベース、人事管理システムなど) を実行している場合があります。これらのアプリケーションが、Access Manager ポリシーエージェントでサポートされるプラットフォーム上に常に配備されているとは限りません。ポリシーエージェントが必要な場合は、これらのアプリケーションの配備基準を確認し、ポリシーエージェントの可用性に基づいてその配備のスケジュール設定を行ってください。

セキュリティーモデル

LOB アプリケーション内で使用する既存のセキュリティーモデルをドキュメント化しておくことは重要です。通常、外部の認証や承認を使用するアプリケーションは、外部のディレクトリサービスに依存するアプリケーションとともに、配備の候補になります。セキュリティー情報には、次のものが含まれます。

- 現在どの認証メカニズムを使用しているか
- 特殊な認証要件 (2 ファクター認証など) は存在するか
- 外部認証メカニズム用のプラグイン可能なインタフェースが存在するか
- 現在どの承認メカニズムを使用しているか
- 承認を外部で行うことは可能か (または、そうすべきか)
- どのユーザーデータリポジトリを使用しているか。それを外部で行うことは可能か
- 誰がアプリケーションにアクセス可能か。既存のロールまたはグループが所定の位置に存在しているか。どのような特殊条件が存在する場合、彼らにアクセスが許可されるのか

セッションのライフサイクル

アイデンティティーのセッションライフサイクルは、認証アプリケーションを評価する場合に考慮する重要な項目です。ユーザーセッションが作成、管理、および破棄される方法について明確に把握しておいてください。アプリケーションの統合時に参照できるよう、このプロセスを明確にドキュメント化してください。

カスタマイズおよびブランド設定

アプリケーションにブランド設定やルックアンドフィールに関する特定の要件がある場合は、それについて検討します。多くの場合、アプリケーション単体のルックアンドフィールを重視するか、ユーザーにとって一貫した使い勝手を重視するかは重要な問題です。カスタマイズやブランド設定を行う場合は、そのための時間もスケジュールに組み込む必要があるため、アプリケーションの評価にその要件が含まれているかどうかを確認してください。

データの分類

アプリケーションの分析と、その適切なレベルへの分類を完了したら、次に、これらのアプリケーションから提供されるデータおよびサービスの分類を開始する必要があります。この情報は、セキュリティーモデルの構築に使用されます。分類自体は、データおよびサービスを分類するプロセスです。それに続き、既存の認証および承認システムのカタログ作成が行われます。

この前者のプロセスに、アプリケーションの評価で収集された情報が使用されます。収集した情報をさまざまなセキュリティー層に編成するのは、良い方法です。これらの層は、データの紛失、アプリケーションの妥協的使用、誤用、その他の不正なアクセスタイプに関係したリスクの量を示すものとなります。正しく定義されたカテゴリを使用すると、リソースをセキュリティーモデルにマッピングして、認証および承認要件を組み込む作業が簡単になります。

データまたはサービスは、4 つのセキュリティーレベルに分類されます。X 軸はデータまたはサービス、Y 軸は関連付けられるセキュリティーレベルを表します。層 1 は、セキュリティーが最小であることを示します。これは、公開された Web サイトに適用可能なデータです。一方、層 4 は、セキュリティーが最大であることを示します。これは財務や人事 (HR) データなどに適用されます。実際の組織の分類に使用される層はこれより多い場合も少ない場合もありますが、この図は、大量のデータには関連するリスクも低く、そのためセキュリティー要件も少なくなるという典型的な例を示しています。関連するリスクが大きくなるにつれ、セキュリティー要件も多くなります。通常、高度なセキュリティー要件が必要なデータはごくわずかであり、多くのデータはセキュリティー要件をほとんど必要としません。

次の図は、典型的な組織内のデータおよびサービスのセキュリティー要件を示しています。

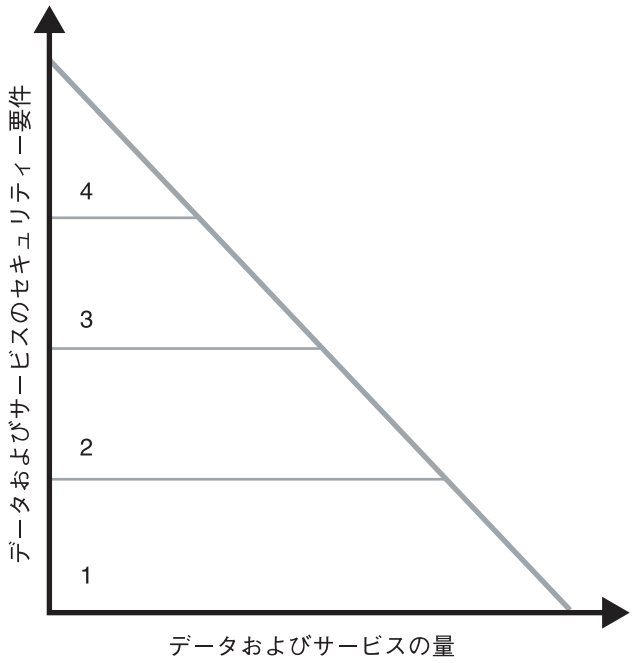


図 2–1 データおよびサービスのセキュリティー要件

認証および承認機能をマッピングできるように、データおよびサービスタイプを機能的にグループ分けしようとしていることを念頭に置いてください。層の数が多すぎるとプロセスが過度に複雑になり、層の数が少なすぎると柔軟性に欠けたものになります。さらに、ネットワーク上に配置すること自体が非常に危険なデータもあることも考慮しておくことは重要です。可能な場合は、内部で利用可能なデータと外部で利用可能なデータを明確に区別するようにしてください。これらの層を構築する際、認証および承認要件とともに、データのアクセス時刻およびネットワーク上の位置などの修飾条件も念頭に置くようにしてください。

認証へのマッピング

データをセキュリティーレベルに応じて分類できたら、次の段階は、認証および承認メカニズムの一覧を作成することです。利用可能な認証メカニズムに関する手持ちのリストを使用して、これらのメカニズムを定義済みのセキュリティー層に関連付けます。たとえば、次の関連付けは、前の図で分類されたデータに対応しています。

- 層 1 のデータは、アクセス制御なしの匿名認証が適切と考えられます。
- 層 2 のデータは、パスワード保護のみを必要とします。
- 層 3 のデータは、ハードトークンまたは証明書認証が必要です。

- 層 4 のデータは、マルチファクター認証を必要とします。または、ネットワーク上には絶対に配置しないようにします。

認証要件とデータやサービスの分類とのマッピングが、単純で明快なものであるようにしてください。そうでない場合、一致しない項目間で共通の基準を見つけるようにしてください。論理的な差異が存在するなら、ためらわずに複数の図を作成してください。

たとえば、イントラネットとエクストラネット用のアプリケーションでは、それぞれ別個の図を作成できます。また、人事 (HR) や財務など、職務上のセキュリティードメインに基づいて、データを分類することもできます。このようなデータ分類手法は万能とは言えませんが、セキュリティー要件を理解し、論理的に管理可能なグループにマッピングする場合に役立ちます。

承認へのマッピング

アプリケーション評価により入手したデータを使用して各アプリケーションを検証し、スケーラブルな承認モデルを決定します。通常、最善の方法は、複数のアプリケーションにわたって使用する共通のグループやロールを見つけることです。これらのグループやロールが、組織内で機能的なロールにマッピングされていると理想的です。また、これらのグループやロールのソース (メンバーシップデータの存在位置およびそのモデリング方法) を判別する必要もあります。たとえば、データは Sun Java System Directory Server 内に存在する可能性があります。

存在しない場合は、カスタムプラグインが必要になる場合があります。堅牢なグループモデルが存在する場合、最初に、各アプリケーションを既存のグループまたはロールに関連付けてください。存在しない場合は、最初にロールまたはグループメカニズムを計画し、機能的観点からユーザータイプ間の共通の関係を見つけてから、特定のアプリケーションにとりかかります。作業の完了時点で、以下のものが入手できます。

- 既存のグループおよびロールの明確なマッピング
- データの存在する位置、その品質と管理に関する権限を持つ人物に関する明確な理解
- 配備を促進したり、配備のコストや複雑性を低減したりするために作成する必要のある新しいグループまたはロールについての明確な理解
- 既存および将来のグループメカニズムの、分類されたアプリケーションへのマッピング
- 特定のグループやロールへのアクセスを可能にするため、アプリケーションが必要とする追加条件に関する注意事項

この基本的なセキュリティーモデル (認証および承認メカニズムとの関連を利用したデータの分類) を使用して、配備を実行するスケジュールをまとめることができます。

スケジュールの作成

収集した情報から、予備段階のスケジュールを作成する必要があります。次の節では、一般的な配備スケジュールを作成するための手順について説明します。

配備の設計

スケジュールのこの段階では、これまでに入手した概念、ビジネスニーズ、およびユーザー要件を適切なコンテキストに配置します。ここで、配備の全体像が明確になります。コンポーネントが記述され、技術要件が定義され、完成したアーキテクチャーが立案作成されます。この設計段階を開始する 2 つの方法は、ログイン時の画面案を作成すること、およびデータフローチャートを作成することです。

コンセプト証明

コンセプト証明を使用すると、設計をビジネス環境でテストできます。組織には、たいていの場合、期待される結果と一体になった設定済みの一連のテストケースである、テストケースデータベースが存在します。このテストデータベースには、コンセプト証明を適用できます。そして、すべてが順調に進めば、ドキュメント化された結果は新しい結果と等しくなります。コンセプト証明の目的は、「配備の設計」で提起されたすべての疑問に答えることにより、すべてのニーズを効果的に、最小限のリスクで満たせることを証明することです。

これは一般にすばやく実行されるため、限られたデータセットに基づいて設計を改良するための十分な時間を取ることができます。通常は、コンセプト証明とそれに続く設計改良を、数回繰り返す必要があります。最後に実行するコンセプト証明は、いくつかの内部アプリケーションが統合されたものになるはずです。企業の共有サービス統合は、一般に、初期採用者によるサインオン標準モデル、次に一般の参加者によるサインオン標準モデル、最後に残された者たちによるサインオン標準モデルに準拠したものになります。初期の採用者で成功すると、そのアプリケーションを一般の採用での参照用として使用しやすくなります。

初期採用

ミッションクリティカルなアプリケーションや収益を創出するためのアプリケーションは、最初のアプリケーションとして選択すべきではありません。よりリスクの少ない戦略は、重要なアプリケーションの中でもロールアウト時に問題が発生しても企業運営に大きな混乱をきたさないものを選択することです。たとえば、部門ポータルは、会計年度末の会計システムよりも、シングルサインオン (SSO) ロールアウト用の拠点として適しています。

また、プロセスの弱点を排除し、結果が立証され、成功が迅速に認識されるように、初期段階でアプリケーションロールアウトの数を制限してください。最良の

ロールアウトの戦略は、可視性を最大限に高めながら、組織上のリスクを最小限に抑えることです。このため、製品に関する適切な経験を持つ配備チームがクリティカルアプリケーションを担当するようにします。

一般の参加

配備プロジェクトは単一のアプリケーションで始まりますが、多目的システムを構築できるように、他の内部顧客の要件も同時に評価する必要があります。中心的なIT グループは、より大規模な組織を代表するサービスを提供するため、サテライトグループのさまざまな基準およびスケジュールを受け入れることができなければなりません。スケジュールは十分に大きなウィンドウに表示し、サテライトグループが変更およびアップグレードを自分たちのアプリケーションの配備および品質保証 (QA) サイクルに組み入れる時間を取れるようにする必要があります。

製品環境

コンセプト証明が終了したら、改良された設計を製品環境内にレプリケートできます。製品環境の目的は、通常の環境で設計の機能を実証し、正しく動作することを確認することです。この環境は、コンセプト証明で観察された動作、および配備設計で定義された動作と比較されます。このテストは、安定性を確認する目的でも行われます。

評価が行われ、レポートが生成されます。初期採用アプリケーションは、準備段階が完了しているため、製品環境でも稼働します。新規アプリケーションを、テスト段階から製品段階へ、徐々に移行させてゆきます。その他のアプリケーションは、初期採用と同じようにコンセプト証明サイクルで稼働させたあとで、徐々に製品環境に追加されていきます。

スケジュールはプロジェクトの複雑さに応じて変動するため、サンプルスケジュールは利用できません。ただし、このプロセスは通常、2 〜 3 か月の期間をかけて行われます。

配備ロードマップ

Access Manager 統合を確実に成功させるには、詳細な計画が必要不可欠です。このプロセスには、ハードウェア、現在配備されているアプリケーション、アイデンティティーデータ、およびアクセス階層に関する情報の収集が含まれます。Access Manager の配備は、次の各段階に細分化できます。

1. 次に示すようなビジネスの目的を特定する
   - 業務効率を改善する
   - データのセキュリティーを確保する
   - 組織内の範囲と関係を理解し、ビジネス目的のサポートに必要な行動の変化を分析することにより、確実に生産性を進展させる

2. ビジネス目的の達成に必要なテクノロジサービスおよびツールを列挙することによって、高度なテクノロジ分析を開発し、それをビジネスの目的に割り当てる
3. 次に示すようなテクノロジサービスの具体策を定義する
   - パーソナライズにより蓄積された従業員の履歴およびデータを保管する
   - アイデンティティー管理を使用して、パスワード同期およびアイデンティティー管理を実行する
   - ロールのストラテジを綿密に練って、企業のセキュリティー保護を実現する
4. 統計の精度、予測可能性、範囲、費用、影響、複雑性、行動、インフラストラクチャー、利点、サポート、依存関係などの項目に基づいて、各具体策に優先順位をつける

◆ ◆ ◆ 第 3 章

3 技術要件

ソリューションのライフサイクルにおける技術要件段階では、使用状況を分析し、ユースケースを特定して、提案された配備ソリューションの QoS 要件を決定します。この章では、Sun Java™ System Access Manager 7.1 での、この処理に関する要件の大まかな技術概要を示します。次の節で構成されています。

配備オプション

Access Manager の配備を計画する際に、組織が考慮する必要のある重要な要素がいくつかあります。これらは、通常、リスク評価および成長戦略と関連があります。次に例を示します。

- 配備環境でどれくらいのユーザーをサポートすることが見込まれているか。予測される成長率はどれくらいか

  ユーザーの成長およびシステムの使用状況を監視し、この実データを予測されるデータと比較して、現在の能力で予測される成長を処理可能であることを確認することは重要です。

- 環境にサービスを追加する計画はあるか。それは現在の設計に影響を及ぼす可能性があるか

  これで、開発中のアーキテクチャーは、現在のサービスに対して最適化されます。将来の計画も検討する必要があります。

さらに、アーキテクチャーは、以降の節で説明する目標を達成するための基礎を提供する必要があります。

セキュリティー

セキュリティーの確保された内部および外部ネットワーク環境を計画している場合には、以下の選択肢について考慮してください。

- サーバーベースのファイアウォールは、サーバーへのポートレベルのアクセスを制限することにより、追加セキュリティーレイヤーを提供します。標準のファイアウォールと同様、サーバーベースのファイアウォールは、着信および送信 TCP/IP トラフィックを制限します。
- 最小化とは、システムの脆弱性が悪用される可能性を最小限に抑えるために、不要なソフトウェアおよびサービスをすべてサーバーから削除することをいいます。
- 分割 DNS インフラストラクチャーには、1 つのドメイン内に 2 つのゾーンが作成されます。1 つのゾーンは組織の内部ネットワーククライアントにより使用され、もう1 つのゾーンは外部ネットワーククライアントにより使用されます。この手法は、より高度なセキュリティーレベルを実現するために推奨されています。DNS サーバーでロードバランサを使用することによって、パフォーマンスを向上させることもできます。

高可用性

IT の配備では、ユーザーに対して可用性を継続するとともに、SPOF (Single Point Of Failure) を発生させないことが重要です。可用性を高めるための手法は、クラスタリングやマルチマスターレプリケーションなど、製品ごとに異なります。望ましい高可用性とは、システムやコンポーネントが一定の期間、連続的に使用可能であるということです。システムは一般に複数のホストサーバーで構成されますが、ユーザーには 1 つの高可用性システムのように見えます。すべてのアプリケーションが 1 台のサーバーで動作する、最小構成の配備の場合、SPOF に含まれる要素として次のものが考えられます。

- Access Manager Web コンテナ
- Directory Server
- Java 仮想マシン (JVM)
- Directory Server ハードディスク
- Access Manager ハードディスク
- ポリシーエージェント

高可用性を実現する場合は、バックアップやフェイルオーバー処理、およびデータストレージやデータアクセスを中心に計画します。ストレージに関する 1 つの手法は、RAID (redundant array of independent disks) です。より高い可用性が求められるシ

ステムでは、システムの各部分が適切に設計され、本稼働に先立ち、十分にテストされていることが必要です。たとえば、テストが十分ではない新規のアプリケーションプログラムほど、本番での稼働中に、システム全体に影響するエラーを引き起こす可能性が高くなります。

クラスタリング

クラスタリングとは、単一の高可用性システムを構築するために複数のコンピュータを使用することを指します。Sun Java System Directory Server のデータストアでは、クラスタリングが非常に重要な手法となる場合が多くあります。たとえば、クラスタ化された 1 組の MMR サーバーでは、可用性が確保されることにより、各マスターインスタンスの可用性を向上させることができます。

スケーラビリティー

「水平スケーリング」は、複数のホストサーバーを接続して1 つの装置として動作させることで実現します。ロードバランスに対応したサービスは、サービスの速度と可用性が向上するため、水平スケーリングが行われていると見なされます。一方、「垂直スケーリング」は、1 台のホストサーバー内部にリソースを追加することにより、既存のハードウェアの容量を拡張します。スケーリング可能なリソースには、CPU、メモリー、および記憶装置が含まれます。水平スケーリングおよび垂直スケーリングは相互排他的なものではないため、配備ソリューションとして両者を併用することができます。通常、環境内のサーバーに容量いっぱいまでリソースがインストールされることはないため、垂直スケーリングを使用してパフォーマンスを改善します。サーバーの容量が限界に近づいた場合も、水平スケーリングを使用して、ほかのサーバーに負荷を分散することができます。

ハードウェア要件

Access Manager を配備する場合の最小構成は、Access Manager と Sun Java System Web Server などの Web コンテナを実行する 1 台のホストサーバーです。Directory Server が稼働するサーバーは、Access Manager と同じでも、違ってもかまいません。複数のサーバーが配備されている環境では、Access Manager インスタンスとそれに対応する Web コンテナは異なるホストサーバーにインストールされ、クライアントの要求は、ロードバランサによって各 Access Manager インスタンスに分配されます。通常、Directory Server と Access Manager は別々のサーバーにインストールされます。

パフォーマンスを最適化するために、Access Manager は 100M バイト以上の Ethernet ネットワーク上で実行してください。Access Manager と 1 つのWeb コンテナを実行する Access Manager 配備の最小構成には、1 個以上の CPU を搭載している必要がありますが、5 個以上の CPU を搭載してもプロセッサのパフォーマンスはそれほど向上しません。ホストサーバーごとに 2 ～ 4 個の CPU を強くお勧めします。ソフトウェアの基本的なテストを実行するために、512M バイト以上の RAM が必要です。

実際の配備では、スレッド、Access Manager SDK、HTTP サーバー、およびほかの内部処理用に 1G バイトの RAM、基本操作およびオブジェクト割り当て領域に 2G バイトの RAM、さらに 10,000 並行セッションごとに 100M バイトの RAM が推奨されています。各 Access Manager は、並行セッションが 100,000 でキャップアウトすることが推奨されており、それ以降は、水平ロードバランスを適用する必要があります (32 ビットアプリケーションの 4G バイトメモリー制限を前提とする)。

ソフトウェア要件

Access Manager には、次のような固有のソフトウェア要件があります。

サポートされるリリース、必要なパッチ、および既知の制限を含むソフトウェア要件の最新情報については、『Sun Java System Access Manager 7.1 リリースノート』を参照してください。

オペレーティングシステム要件

Access Manager 7.1 は、次のプラットフォームでサポートされています。

- Solaris™ 10 OS (SPARC®、x86、および x64 ベースのシステム)
- Solaris 9 OS (SPARC および x86 システム)
- Red Hat™ Linux OS
- Microsoft Windows OS
- HP-UX OS

サポートされる各オペレーティングシステムの特定のバージョンについては、『Sun Java System Access Manager 7.1 リリースノート』を参照してください。

OS パッチおよびパッチクラスタのダウンロード方法については、`http://sunsolve.sun.com/` にある SunSolve Online にアクセスしてください。

現在 Solaris システムにインストールされているパッチを表示するには、`showrev -p` コマンドを使用します。

Web コンテナ要件

Access Manager 7.1 は、完全インストール、または SDK のみのインストールのいずれかでの使用に関して、次の Web コンテナをサポートしています。

- Sun Java System Web Server
- Sun Java System Application Server
- BEA WebLogic
- IBM WebSphere Application Server

注 – BEA WebLogic および IBM WebSphere Application Server は、HP-UX システムではサポートされていません。

これらの Web コンテナのサポートされているバージョンについては、『Sun Java System Access Manager 7.1 リリースノート』を参照してください。

ポリシーエージェントを Access Manager Web コンテナにインストールする場合は、約 10M バイトのディスク容量が使用されます。Web コンテナを設定するときには、この追加容量を考慮に入れる必要があります。詳細は、『Sun Java System Access Manager Policy Agent 2.2 User's Guide 』を参照してください。

Directory Server 要件

Access Manager 7.1 には、LDAP ディレクトリサーバーに対する次の要件があります。

- Access Manager 情報ツリーが Sun Java System Directory Server に格納されていること。情報ツリーには次の情報が含まれます。
  - ユーザー認証の方法
  - ユーザーがアクセス可能なリソース
  - ユーザーがリソースへのアクセス権を取得したあとに、アプリケーションで使用可能になる情報
- Access Manager には、ユーザーやグループなどのユーザーデータを格納するためのアイデンティティーリポジトリが必要です。Access Manager 7.1 は、Sun Java System Directory Server または LDAP バージョン 3 (LDAP v3) 互換のディレクトリサーバーをアイデンティティーリポジトリとして使用できます。

Access Manager 情報ツリーおよびアイデンティティーリポジトリについては、『Sun Java System Access Manager 7.1 Technical Overview』を参照してください。

Java Development Kit (JDK) ソフトウェア要件

Access Manager 7.1 に必要な JDK ソフトウェアの特定のバージョンについては、『Sun Java System Access Manager 7.1 リリースノート』を参照してください。

Access Manager セッションフェイルオーバー要件

Access Manager セッションフェイルオーバーを実装することを計画している場合、以下のコンポーネントが必要です。

- Access Manager を実行する Web コンテナ: Sun Java System Web Server、 Sun Java System Application Server、 IBM WebSphere Application Server、 または BEA WebLogic。
- Sun Java System Directory Server: すべての Access Manager インスタンスが、同じ Directory Server にアクセスする必要があります。
- Sun Java System Message Queue。 Message Queue ブローカクラスタは、 Access Manager インスタンスとセッションストアデータベースの間のセッションメッセージを管理します。
- Berkeley DB (`http://www.oracle.com/database/berkeley-db.html`) はデフォルトのセッションストアデータベースです。 Sun Java Enterprise System 5 リリースによって配布されるバーションを使用します。

Access Manager のセッションフェイルオーバーは以下のプラットフォームでサポートされています。

- Solaris 10 OS (SPARC、 x86、 および x64 ベースのシステム)
- Solaris 9 OS (SPARC および x86 システム)
- Red Hat Linux OS
- Microsoft Windows OS
- HP-UX OS

これらのプラットフォームおよびコンポーネントがサポートされるバージョンに関する最新の情報については、 『Sun Java System Access Manager 7.1 リリースノート』 を参照してください。

詳細は、 『Sun Java System Access Manager 7.1 Postinstallation Guide』 の第 6 章 「Implementing Session Failover」 を参照してください。

Web ブラウザ要件

Access Manager 管理者およびエンドユーザーは、 Web ブラウザを使用して、管理タスクおよびユーザー管理タスクを実行します。このリリースでサポートされる Web ブラウザについては、 『Sun Java System Access Manager 7.1 リリースノート』 を参照してください。

Access Manager コンソールにアクセスするには、ブラウザで JavaScript を有効にする必要があります。

JSSE 暗号化要件

セキュリティー保護されたインターネット通信を実装するために、次の配備シナリオまたはアクティビティーでは、Access Manager 7.1 で JSS 暗号化の代わりに Java Secure Socket Extension (JSSE) 暗号化を使用する必要があります。

- SSL 対応 Web コンテナインスタンスへのクライアント SDK 配備
- SSL 対応 Web コンテナインスタンスへの分散認証配備
- SSL 対応 Web コンテナインスタンスへの単一の Access Manager 7.1 WAR ファイル配備
- SSL 対応 Access Manager サーバーでの `com.sun.identity.idm` API の使用
- SSL 対応サードパーティー Web コンテナインスタンス (BEA WebLogic または IBM WebSphere Application Server) への Access Manager 配備

JSSE に関する情報およびダウンロードについては、次の Sun Developer Network (SDN) Web サイトを参照してください。`http://java.sun.com/products/jsse/`

管理ユーザー

Access Manager 配備の技術要件を評価するときは、次の管理ユーザーおよび管理アカウントについて考慮します。

- 49 ページの「Access Manager の管理ロール」
- 52 ページの「Access Manager の管理アカウント」
- 54 ページの「ポリシーエージェントの管理ユーザー」

Access Manager の管理ロール

- 49 ページの「レルムモードの管理ロール」
- 50 ページの「旧バージョンモードの管理ロール」

レルムモードの管理ロール

Access Manager のレルムモードでは、委任プラグインがアイデンティティーリポジトリプラグインと連携して、ネットワーク管理者の権限の有効範囲を決定します。デフォルトの管理者ロールはアイデンティティーリポジトリプラグインで定義されます。委任プラグインは、個々のネットワーク管理者の権限の有効範囲を記述するルールを形成するとともに、そのルールが適用されるロールを指定します。次の表に、アイデンティティーリポジトリで定義されるロールと、委任プラグインが各ロールに適用するデフォルトルールの一覧を示します。

表 3–1 レルムモードでの Access Manager のロールと権限の有効範囲

アイデンティティーリポジトリのロール	委任ルール
レルム管理者	Access Manager 情報ツリーのすべてのレルム内にあるすべてのデータにアクセスできます。
サブレルム管理者	Access Manager 情報ツリーの特定のレルム内にあるすべてのデータにアクセスできます。
ポリシー管理者	Access Manager 情報ツリーのすべてのレルム内にあるすべてのポリシーにアクセスできます。
ポリシーレルム管理者	Access Manager 情報ツリーの特定レルム内部にあるポリシーのみにアクセスできます。

認証サービスとポリシーサービスは、集積されたデータを使用して認証および承認のプロセスを実行します。委任プラグインおよびアイデンティティーリポジトリプラグインのコードは、Access Manager で公開されていません。

旧バージョンモードの管理ロール

Access Manager の旧バージョンモードでは、LDAP エントリの委任管理 (Access Manager 内の各アイデンティティー関連オブジェクトにマップされる) は、定義済みのロールおよびアクセス制御命令 (ACI) を使用して実装されます。デフォルトの管理ロールおよびその定義済み ACI は、Access Manager のインストール時に作成され、Access Manager コンソールを使用して表示および管理できます。Access Manager 7.1 のレルムモードでは、ロールは ACI ではなくポリシーに依存します。

Access Manager のアイデンティティー関連オブジェクトが作成されると、適切な管理ロール (および対応する ACI) も作成され、そのオブジェクトの LDAP エントリに割り当てられます。そのあと、ロールは、そのオブジェクトのノード制御を管理する個々のユーザーに割り当てることができます。たとえば、Access Manager が組織を新規作成すると、いくつかのロールが自動的に作成され、Directory Server に格納されます。

- 組織管理者は設定済み組織のすべてのエントリに対する読み取りアクセス権と書き込みアクセス権を持っています。
- 組織のヘルプデスク管理者は、設定済み組織のすべてのエントリに対する読み取りアクセス権、およびこれらのエントリ内の `userPassword` 属性に対する書き込みアクセス権を持っています。
- 組織のポリシー管理者は、組織のすべてのポリシーに対する読み取りアクセス権と書き込みアクセス権を持っています。

これらのロールのいずれかをユーザーに割り当てると、そのロールに与えられたすべてのアクセス権がユーザーに与えられます。

次の表に、Access Manager 管理ロール、および各ロールに適用される権限の要約を示します。

表 3–2 旧バージョンモードでのデフォルトロールおよび動的ロールと各ロールのアクセス権

ロール	管理サフィックス	アクセス権
最上位組織の管理者 (amadmin)	ルートレベル	最上位組織内のすべてのエントリ (ロール、ポリシー、グループなど) に対する読み取りおよび書き込みアクセス権。
最上位組織のヘルプデスク管理者	ルートレベル	最上位組織内のすべてのパスワードに対する読み取りおよび書き込みアクセス権。
最上位組織のポリシー管理者	ルートレベル	すべてのレベルのポリシーに対する読み取りおよび書き込みアクセス権。参照ポリシー作成を委任するため、下位組織により使用されます。
組織管理者	組織のみ	作成された下位組織内のすべてのエントリ (ロール、ポリシー、グループなど) に対する読み取りおよび書き込みアクセス権のみ。
組織のヘルプデスク管理者	組織のみ	作成された下位組織内のすべてのパスワードに対する読み取りおよび書き込みアクセス権のみ。
組織ポリシー管理者 (Organization Policy Admin)	組織のみ	作成された下位組織内のすべてのポリシーに対する読み取りおよび書き込みアクセス権のみ。
コンテナ管理者 (Container Admin)	コンテナのみ	作成されたコンテナ内のすべてのエントリ (ロール、ポリシー、グループなど) に対する読み取りおよび書き込みアクセス権のみ。
コンテナヘルプデスク管理者 (Container Help Desk Admin)	コンテナのみ	作成されたコンテナ内のすべてのパスワードに対する読み取りおよび書き込みアクセス権のみ。
グループ管理者	グループのみ	作成されたグループ内のすべてのエントリ (ロール、ポリシー、グループなど) に対する読み取りおよび書き込みアクセス権のみ。
ピープルコンテナ管理者	ピープルコンテナのみ	作成されたピープルコンテナ内のすべてのエントリ (ロール、ポリシー、グループなど) に対する読み取りおよび書き込みアクセス権のみ。

表 3–2 旧バージョンモードでのデフォルトロールおよび動的ロールと各ロールのアクセス権 (続き)

ロール	管理サフィックス	アクセス権
ユーザー (自己管理者)	ユーザーのみ	ユーザーエントリ内のすべての属性に対する読み取りおよび書き込みアクセス権のみ (nsroledn や inetuserstatus などのユーザー属性を除く)。

グループベースの ACI の代わりにロールを使用すると、効率を高め、保守の手間を少なくすることができます。フィルタ処理されたロールは、ユーザーの nsRole 属性の確認のみを行うため、LDAP クライアントの処理が簡略化されます。ロールは、そのメンバーの親のピアでなければならない、という範囲制限の影響を受けます。

デフォルトACI については、Access Manager コンソールのオンラインヘルプを参照してください。

Access Manager の管理アカウント

Access Manager のインストール時には、次の管理アカウントが作成されます。

- 管理者ユーザー ID (amadmin) は、Access Manager の最上位管理者で、Access Manager によって管理されるすべてのエントリに無制限にアクセスできます。デフォルト名の amadmin を変更することはできません。

  インストール中に、amadmin のパスワードを入力する必要があります。インストール後に amadmin のパスワードを変更するには、Access Manager 管理コンソールを使用します。

- LDAP、メンバーシップ、およびポリシーサービスのバインド DN ユーザー (amldapuser) は、すべての Directory Server エントリに対する読み取りおよび検索アクセス権を持つ管理ユーザーです。デフォルト名の amldapuser を変更することはできません。

  インストール中に、amldapuser のパスワードを入力する必要があります。amadmin と同じパスワードは使用しないでください。インストール後に amldapuser のパスワードを変更するには、Directory Server コンソールか ldapmodify ユーティリティーを使用します。

  amldapuser のパスワードを変更した場合は、LDAP 認証サービスとポリシー設定サービスにも、この変更を反映させる必要があります。amldapuser は、これらのサービスで使用されているデフォルトユーザーだからです。この変更は、これらのサービスを登録している組織ごとに行う必要があります。

- プロキシユーザー (puser) は、任意のユーザー (組織管理者またはエンドユーザーなど) の権限を引き受けることができます。

- 管理ユーザー (dsameuser) は、Access Manager SDK が、特定のユーザーにリンクされていない Directory Server 上で、サービス設定情報の取得などの操作を実行するときバインドするために使用されます。

puser および dsameuser は関連付けられたパスワードを所有し、それぞれのパスワードは serverconfig.xml に暗号化された形式で格納されています。このファイルは次のディレクトリ内にあります。

- Solaris システム: /etc/opt/SUNWam/config
- Linux および HP-UX システム: /etc/opt/sun/identity/config
- Windows システム: *javaes-install-dir*\identity\config

  javaes-install-dir 変数は Java ES 5 のインストールディレクトリを表します。デフォルト値は C:\Program Files\Sun\JavaES5 です。

インストール後に、puser および dsameuser のパスワードを変更することをお勧めします。ただし、amadmin や amldapuser に使用したものと同じパスワードを使用しないでください。puser または dsameuser のパスワードを変更するには、ampassword ユーティリティーを使用します。

- ampassword --admin (または -a) オプションでは、dsameuser のパスワードが変更されます。(このオプションでは、amadmin のパスワードは変更されません。)
- ampassword --proxy (または -p) オプションでは、puser のパスワードが変更されます。

puser と dsameuser のどちらのパスワードを変更するかは、ユーザーの配備によって決まります。

Access Manager が単一のホストサーバー上に配備されている場合は、次の手順を実行します。

1. ampassword ユーティリティーを使用して、Directory Server とローカルの serverconfig.xml ファイル内のそれぞれのパスワードを変更します。
2. Access Manager Web コンテナを再起動します。

Access Manager が複数のホストサーバー上に配備されている場合は、次の手順を実行します。

1. 最初のサーバー上で、ampassword ユーティリティーを使用して、Directory Server とローカルの serverconfig.xml ファイル内のそれぞれのパスワードを変更します。
2. ampassword --encrypt (または -e) オプションを使用して、新しいパスワードを暗号化します。
3. Access Manager の配備されているその他の各サーバー上で、手順 2 で暗号化した新しいパスワードを使用して、serverconfig.xml ファイル内のパスワードを手動で変更します。

4. パスワードを変更した各サーバー上 (最初のサーバーを含む) で、Access Manager Web コンテナを再起動します。

ampassword ユーティリティーについては、『Sun Java System Access Manager 7.1 Administration Reference 』を参照してください。

ポリシーエージェントの管理ユーザー

Access Manager Policy Agent 2.2 ソフトウェアセットのポリシーエージェントは、その AMAgent.properties ファイルに保存されたユーザー名とパスワードを使用して Access Manager への認証を行います。このプロセスは54 ページの「Web エージェント」と54 ページの「J2EE エージェント」では似ている部分もありますが、多少の違いがあります。

Web エージェント

Web エージェントは、AMAgent.properties ファイルの次のプロパティーを使用して、Access Manager への認証に使用する自身のユーザー名とパスワードを保存します。

- com.sun.am.policy.am.username には、Web エージェントが Access Manager にログインするために使用するユーザー名が格納されます。デフォルト名は UrlAccessAgent です。
- com.sun.am.policy.am.password には、Web エージェントが Access Manager にログインするために使用するユーザーの暗号化されたパスワードが格納されます。パスワードは crypt_util ユーティリティーを使用して暗号化する必要があります。

Access Manager インスタンスを最初に設定するとき、Java ES インストーラまたは amconfig スクリプトによって、amldapuser ユーザーと同じパスワードを持つ amService-UrlAccessAgent ユーザーが最上位レベルレルムに作成されます。

デフォルトでは、すべての Web エージェントが同じユーザー名とパスワードを使用して Access Manager インスタンスへの認証を行います。セキュリティーを強化し、各 Web エージェントが一意の名前とパスワードを使用できるようにするために、Web エージェントが認証時に使用するエージェントプロファイルを Access Manager 管理コンソールで作成できます。詳細は、55 ページの「エージェントプロファイルを使用した認証」を参照してください。

J2EE エージェント

J2EE エージェントは、Access Manager 管理コンソールで作成されたエージェントプロファイルによるユーザー名 (エージェント ID) とパスワードを使用して Access Manager と通信します。エージェントプロファイルが作成されると、J2EE エージェントは AMAgent.properties ファイルの次のプロパティーを使用してユーザー名 (エージェント ID) とパスワードを保存します。

- com.sun.identity.agents.app.username には、J2EE エージェントが Access Manager へのログインに使用するユーザー名 (エージェント ID) が格納されます。
- com.iplanet.am.service.secret には、J2EE エージェントが Access Manager へのログインに使用するユーザー名 (エージェント ID) の暗号化されたパスワードが格納されます。パスワードは agentadmin ユーティリティーと --encrypt オプションを使用して暗号化する必要があります。

エージェントプロファイルについては、次の節を参照してください。

エージェントプロファイルを使用した認証

Access Manager への認証を行うには、J2EE エージェントが使用するエージェントプロファイルを Access Manager 管理コンソールで作成する必要があります。Web エージェントもエージェントプロファイルを使用できます。これにより、各 Web エージェントに一意のユーザー名 (エージェント ID) とパスワードを設定できるようになります。エージェントプロファイルの作成手順については、Access Manager コンソールのオンラインヘルプを参照してください。

エージェントプロファイルを使用すると、ポリシーエージェントのパスワードとユーザー名 (エージェント ID) を、配備ごとの必要に応じて変更できるようにもなります。必要な場合にパスワードとユーザー名を変更するには、次の一般的な手順に従います。

1. Access Manager 管理者 (amadmin) として Access Manager コンソールにログインします。
2. ポリシーエージェントのエージェントプロファイルで、必要に応じてパスワードとユーザー名 (エージェント ID) を変更します。プロファイルを保存します。
3. 手順 2 で変更した新しいエージェントパスワードを、Web エージェントの場合は crypt_util ユーティリティーを使用して、J2EE エージェントの場合は agentadmin ユーティリティーと --encrypt オプションを使用して暗号化します。
4. ポリシーエージェントの AMAgent.properties ファイルで、次のプロパティーを設定します。
   - Web エージェントの場合: com.sun.am.policy.am.password プロパティーに、手順 3 で新しく暗号化したパスワードを設定します。ユーザー名 (エージェント ID) も変更した場合は、com.sun.am.policy.am.username プロパティーに、手順 2 で変更した新しいユーザー名 (エージェント ID) を設定します。
   - J2EE エージェントの場合: com.iplanet.am.service.secret プロパティーに、手順 3 で新しく暗号化したパスワードを設定します。ユーザー名 (エージェント ID) も変更した場合は、com.sun.identity.agents.app.username プロパティーに、手順 2 で変更した新しいユーザー名 (エージェント ID) を設定します。
5. 新しいパスワード (および、変更した場合は新しいユーザー名) を有効にするために、Web エージェントの Web コンテナを再起動します。

エージェントプロファイルの作成および設定とパスワードの暗号化については、Access Manager Policy Agent 2.2 のマニュアルコレクションを参照してください。

http://docs.sun.com/coll/1322.1

匿名ユーザー

匿名モジュールが有効な場合、匿名ユーザーはパスワードを提供しなくても Access Manager にログインできます。デフォルトの匿名ユーザーは anonymous ですが、Access Manager 管理コンソールで次のレルム属性を設定することにより、名前の変更や、匿名ユーザーのリストの定義を行うことができます。

- 「有効な匿名ユーザー」では、匿名アクセスを許可するユーザー ID のリストを指定します。
- 「デフォルトの匿名ユーザー名」では、デフォルト値 (anonymous) 以外のユーザー名を指定できます。この名前は「有効な匿名ユーザー」リストが空の場合に使用されます。
- 「ユーザー ID の大文字と小文字を区別する」では、匿名ユーザー ID の大文字と小文字を区別する必要があることを指定します。デフォルトでは、ユーザー ID の大文字と小文字は区別されません。
- 「認証レベル」では、匿名ユーザーのアクセスの種類を、読み取りと検索のみなど、特定のものに制限します。デフォルト値は 0 です。

これらの属性はレルムに適用されるため、レルムごとに異なる匿名アクセス属性を設定できます。

匿名モジュールの有効化と匿名ユーザーの作成については、Access Manager コンソールのオンラインヘルプを参照してください。

Access Manager スキーマ

スキーマとは、データに課されるルールセットのことで、通常はデータの格納方法の定義に利用されます。Sun Java System Directory Server は LDAP (Lightweight Directory Access Protocol) スキーマを使用して、データの格納方法を定義します。オブジェクトクラスは、LDAP スキーマ内の属性を定義します。Directory Server では、各データエントリは、内部の属性セットを記述および定義するオブジェクトのタイプを指定するため、1 つ以上のオブジェクトクラスを保持する必要があります。基本的に、各エントリは、属性セットとその対応する値、およびこれらの属性に対応するオブジェクトクラスのリストになります。

Access Manager は、Sun Java System Directory Server をデータリポジトリとして使用します。このリポジトリには Directory Server スキーマを拡張する Access Manager スキーマが組み込まれています。Access Manager のインストール時に、

ds_remote_schema.ldif と sunone_schema2.ldif のファイルで構成される Access Manager スキーマは、Directory Server スキーマと統合されます。ds_remote_schema.ldif ファイルは、Access Manager が固有に使用する LDAP オブジェクトクラスと属性を定義します。sunone_schema2.ldif ファイルは、Access Manager LDAP スキーマのオブジェクトクラスと属性をロードします。

ds_remote_schema.ldif、sunone_schema2.ldif 、およびその他の Access Manager LDIF ファイルは、以下のディレクトリにあります。

- Solaris システム: /etc/opt/SUNWam/config/ldif
- Linux および HP-UX システム: /etc/opt/sun/identity/config/ldif
- Windows システム: *javaes-install-dir*\identity\config\ldif

  javaes-install-dir 変数は Java ES 5 のインストールディレクトリを表します。デフォルト値は C:\Program Files\Sun\JavaES5 です。

マーカーオブジェクトクラス

Access Manager コンソールを使用して作成し、Directory Server 内に格納したアイデンティティーエントリには、マーカーオブジェクトクラスが追加されます。マーカーオブジェクトクラスは、指定されたエントリを Access Manager が管理するエントリとして定義します。オブジェクトクラスは、サーバーやハードウェアなど、ディレクトリツリーのほかの面には影響を与えません。また、既存のアイデンティティーエントリは、これらのオブジェクトクラスを含めるようにエントリを変更しない限り、Access Manager を使用して管理することはできません。マーカーオブジェクトクラスについては、『Access Manager 開発者ガイド』を参照してください。既存の Directory Server データを Access Manager で使用するために移行する方法については、『 Sun Java Access Manager 6 2005Q1 Migration Guide』を参照してください。

スキーマの制限

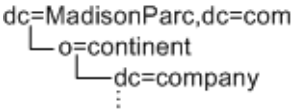
Access Manager は、管理するエントリを抽象的に表現します。したがって、たとえば、Access Manager 内の組織は、Directory Server 内の組織とは必ずしも同じにはなりません。特定の DIT (Directory Infromation Tree) を管理できるかどうかは、ディレクトリエントリを表すまたは管理する方法と、DIT が各 Access Manager タイプの制限に適しているかどうかによります。

以下の項で、Access Manager スキーマに課される制限について説明します。この節の最後には、59 ページの「サポートされない DIT の例」も複数記載しています。

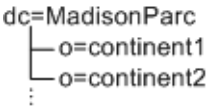
組織としてマークできるエントリのタイプは 1 つに限られる

Access Manager の `sunISManagedOrganization` 予備クラスを任意のエントリに追加することにより、Access Manager では、このエントリを組織とみなして管理できます。ただし、組織としてマークできるエントリのタイプは 1 つに限られます。たとえば、DIT にエントリ `o=sun` と別のエントリ `dc=ibm` がある場合、両方のエントリを組織としてマークすることはできません。

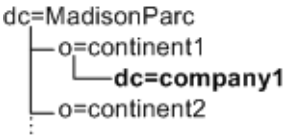
次の例のような DIT 構造に対して、`dc` と `o` の両方のエントリを組織にしようとしても、そのような構造は Access Manager で管理できません。



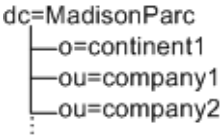
ただし、Access Manager ルートサフィックスでのエントリは、1 つのエントリには数えられません。したがって、次の例のDIT 構造は、Access Manager で管理できます。



`o=continent1` の下に `dc=company1` を追加した場合、`dc` がコンテナとしてマークされている場合にのみ、このDIT の管理が可能になります。コンテナは、Access Manager の別の抽象タイプであり、通常、組織単位にマッピングされます。ほとんどの DIT では、 `iplanet-am-managed-container` エントリをすべての組織単位に追加します。



ただし、このマーカーオブジェクトクラスはどのエントリタイプにも追加できます。次の例の DIT 構造が可能です。



この例の場合は、`o=` と `ou=` の両方のエントリを組織としてマークすることはできませんが、`o=` エントリを組織としてマークし、`ou=` エントリをコンテナとしてマークすることができます。コンソールに表示されるときに、組織とコンテナで利用できるオプションは同じです。ピープルコンテナ、グループ、ロール、およびユーザーなどの従属エントリや下位エントリは両方で作成できます。

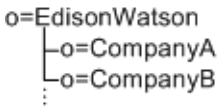
ピープルコンテナをユーザーの親エントリにする必要がある

もう 1 つの抽象エントリタイプはピープルコンテナです。Access Manager タイプは、このエントリがユーザーの親エントリであると想定します。ピープルコンテナとしてエントリに `iplanet-am-managed-people-container` のマークが付けられていると、UI は、このコンテナには下位ピープルコンテナまたはユーザーだけが存在すると見なします。属性 `OrganizationUnit` が通常、ピープルコンテナとして使用されますが、親のタイプが Access Manager で管理可能な組織またはコンテナで、オブジェクトクラスに `iplanet-am-managed-people-container` が含まれているエントリであれば、ピープルコンテナとして扱えます。

Access Manager XML で可能な組織の説明は 1 つに限られる

Access Manager の組織は、`amEntrySpecific.xml` で定義されます。このファイルでは、1 つの組織の説明だけを記述できます。この結果、ディレクトリエントリのプロパティーをカスタマイズしたり、管理ページや検索ページを UI 内に作成すると、カスタム属性は Access Manager 設定全体に適用されます。この Access Manager 要件は、特にホスティングサービスを行う企業など、配備における組織ごとに異なる表示属性を必要とする諸企業のニーズに合わない場合があります。

次の例で、Edison-Watson 社はホスティング企業として、インターネットサービスを多数の企業に提供しています。企業 A では、ユーザーの姓、名、バッジ番号を入力するフィールドを表示する必要があります。企業 B では、ユーザーの姓、名、社員番号を入力するフィールドを表示する必要があります。

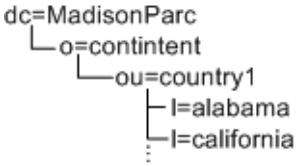


組織の説明は、組織レベルではなくルートレベル (`o=EdisonWatson`) で定義されます。デフォルトでは、企業 A と 企業 B の両方の UI を同一にする必要があります。また、すべてのサービスは、下位スキーマタイプユーザーの属性になるように、属性をグローバルに定義します。したがって、企業 A が、予備クラス `CompanyA-user` にそのユーザー用の属性を保持し、企業 B が、`CompanyB-user` に属性を保持している場合、企業 B の属性は上書きされ、表示されません。

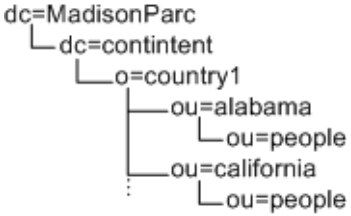
回避策としては、ユーザー表示に対処するように ACI を修正する方法があります。ただしこの回避策は、「検索」および「作成」ウィンドウでの属性には対処しません。

サポートされない DIT の例

次の例では、次の 3 タイプの組織マーカーが必要になります。`o`、`ou`、および `l`。`l=california` および `l=alabama` がピープルコンテナではないと見なされるため、この DIT は Access Manager では動作しません。



次の例では、3 タイプの Access Manager マーカー (dc、o、ou)、およびピープルコンテナ (ou=people) が必要になります。この条件下では、DIT は Access Manager で動作しません。



◆ ◆ ◆ 第 4 章

Access Manager を使用する場合の論理設計

ソリューションライフサイクルの論理設計段階では、ソリューションにおける論理コンポーネントの相互関係を示す論理アーキテクチャーを設計します。論理アーキテクチャーと技術要件段階で作成した使用状況分析とによって配備シナリオが作成され、このシナリオが配備設計段階でのインプットとなります。この章は、Sun Java™ System Access Manager の論理設計に関する次の節で構成されています。

論理アーキテクチャーについて

論理アーキテクチャーによって、ソリューションの実装に必要なソフトウェアコンポーネントが明確にされ、それらコンポーネントの相互関係が示されます。論理アーキテクチャーと技術要件段階で決定した QoS 要件とによって配備シナリオが作成されます。配備シナリオは、次の配備設計段階で行う配備アーキテクチャー設計のための基礎となります。

論理アーキテクチャーの設計

論理アーキテクチャーを設計する際には、技術要件段階で確認したユースケースを使用して、ソリューションに必要なサービスを提供する Java Enterprise System (Java ES) コンポーネントを決定します。最初に指定したコンポーネントに対してサービスを提供するコンポーネントについても、すべて列挙する必要があります。

Java ES コンポーネントは、提供するサービスに応じて、多層アーキテクチャーのコンテキスト内に配置します。コンポーネントが多層アーキテクチャーの一部である

ことを理解すると、コンポーネントによって提供されるサービスを分配する方法を決めたり、スケーラビリティーや可用性などの Quality of Service (QoS) を実現する方針を決めたりするのに役立ちます。

論理アーキテクチャーおよびソリューションライフサイクルについては、次のマニュアルコレクション内の『Sun Java Enterprise System 配備計画ガイド』を参照してください。http://docs.sun.com/coll/1286.2

Access Manager コンポーネント

Access Manager の配備には、次の製品およびコンポーネントが含まれます。

- 62 ページの「Web コンテナ」
- 62 ページの「Directory Server」
- 63 ページの「セッションフェイルオーバー用の Message Queue および Berkeley DB」

Web コンテナ

Access Manager は、次のいずれかの Web コンテナ内で実行する必要があります。

- Sun Java System Web Server
- Sun Java System Application Server
- BEA WebLogic Server
- IBM WebSphere Application Server

サポートされる各 Web コンテナの特定のバージョンについては、『Sun Java System Access Manager 7.1 リリースノート』を参照してください。

Directory Server

Access Manager には、次のエンティティー用に LDAP ディレクトリサーバーが必要です。

- 62 ページの「Access Manager 情報ツリー」
- 63 ページの「アイデンティティーリポジトリ」

Access Manager 情報ツリー

Access Manager 7.1 では、Access Manager 情報ツリーを格納するために Sun Java System Directory Server が必要です。Access Manager が作成および維持する Access Manager 情報ツリーには、システムアクセスに関係する次の情報が保持されます。

- ユーザー認証の方法

- ユーザーがアクセス可能なリソース
- ユーザーがリソースへのアクセス権を取得したあとに、アプリケーションで使用可能になる情報

アイデンティティーリポジトリ

Access Managerには、ユーザーやグループなどのユーザーデータを格納するためのアイデンティティーリポジトリが必要です。以前のバージョンのAccess Managerでは、アイデンティティーリポジトリとしてSun Java System Directory Serverが必要でした。しかし、Access Manager 7.1では、Sun Java System Directory Serverに加えて、LDAP Version 3 (LDAP v3)準拠のディレクトリサーバーもサポートされています。

セッションフェイルオーバー用の**Message Queue**および**Berkeley DB**

セッションフェイルオーバーの実装を計画している場合は、Access Managerに次のコンポーネントが必要です。

- Sun Java System Message Queue。Message Queueブローカクラスタは、Access Managerインスタンスとセッションストアデータベースの間のセッションメッセージを管理します。
- Berkeley DB (`http://www.oracle.com/database/berkeley-db.html`)はデフォルトのセッションストアデータベースです。

Access Managerを使用する**Java ES**コンポーネント

通常、Access Managerは次のJava ESコンポーネント製品とともに配備されます。

- Sun Java System Portal Server、Sun Java System Messaging Server、Sun Java System Calendar Server、Sun Java System Instant Messaging、およびSun Java System Communications Express。シングルサインオン(SSO)を有効にするために必要です。
- Sun Java System Web Server。アクセス制御を任意に設定するために必要です。

Access Manager 論理アーキテクチャーの例

ここでは、Access Manager ソリューションの論理アーキテクチャーの例として、次のシナリオを取り上げます。

Access Manager の Web 配備

Access Manager 配備では一般に Web ブラウザを使用して、Web サーバー上に配備されたアプリケーションまたはリソースにアクセスします。アプリケーションまたはリソースは Access Manager により保護されるため、通信は、Web サーバーにインストールされたポリシーエージェントを使用して行われます。さらに、Web サーバーに Access Manager SDK を配備することもできます。このシナリオの場合、マシン上の Web サーバーの数や、複数のマシンに配備される Access Manager のインスタンスに関して制限はありません。たとえば、1 台のマシンで複数の Web サーバーを稼働させ、それぞれに Access Manager のインスタンスを配備することもできます。同様に、複数の Web サーバーを別々のマシン上で稼働させ、それぞれに Access Manager のインスタンスを配備することもできます。

次の図に、Access Manager の Web 配備シナリオを示します。

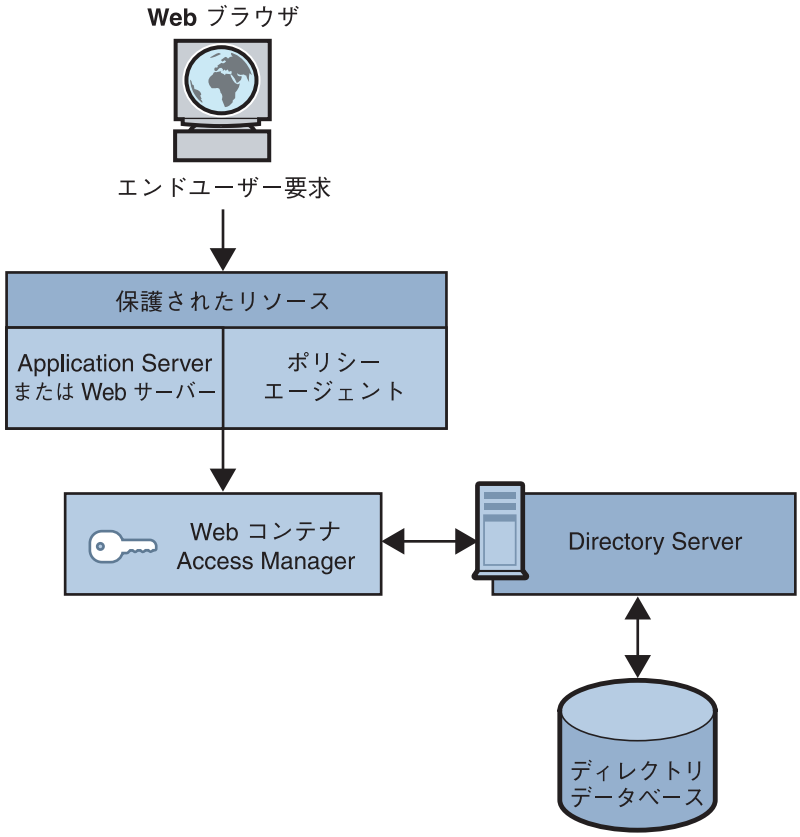


図 4–1　Access Manager の Web 配備

Access Manager の複数サーバー配備

Access Manager を複数のサーバーに配備する場合、2 台以上のホストサーバーに、それぞれ 1 つ以上の Access manager インスタンスを配備します。それぞれの Access Manager インスタンスは同じ Directory Server にアクセスします。配備環境に応じて、Directory Server インスタンスをマルチマスターレプリケーション (MMR) 設定にすることができます。

1 番目のホストサーバーにインストールされた Access Manager インスタンスは、Directory Server のインスタンスを指し示します。Java ES インストーラを使用したインストールでは、実際の配備に応じて既存の Directory Server が、既存のディレクトリ情報ツリー (DIT) を使用するか、使用しないかを選択できます。

Java ES インストーラを実行して、その他のサーバーに Access Manager のそれ以外のインスタンスをインストールし、既存の DIT を使用して、Access Manager インスタン

スが Directory Server を指し示すようにします。このとき Access Manager は、Directory Server がすでに存在しているものと認識しているため、情報を一切 Directory Server に書き込みません。

次の図には、1 つの Directory Server に対して、異なるホストサーバー上に配備された複数の Access Manager インスタンスを示しています。

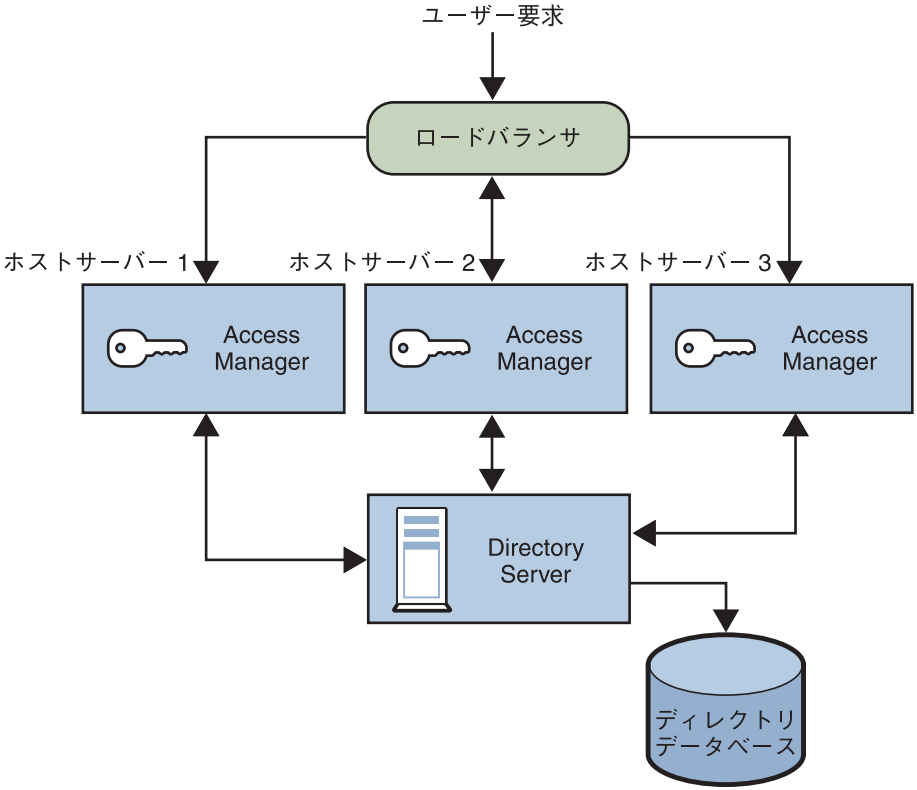


図 4–2　1 つの Directory Server に対する複数の Access Manager インスタンス

詳細は、『Sun Java System Access Manager 7.1 Postinstallation Guide』の第 3 章「Deploying Multiple Access Manager Instances」を参照してください。

Java アプリケーションの配備

別の一般的な Access Manager のシナリオでは、Java アプリケーションは配備先のサーバーに直接インストールされた Access Manager SDK にアクセスできます。このシナリオでは、1 つ以上の Access Manager インスタンスが稼働する Sun Java System Web Server または Sun Java System Application Server など Web コンテナのインスタンスを持

つ追加サーバーが必要になります。 このサーバーは、情報を管理してシングルサインオン (SSO) を提供します。次の図は、Java アプリケーションの配備シナリオを示しています。

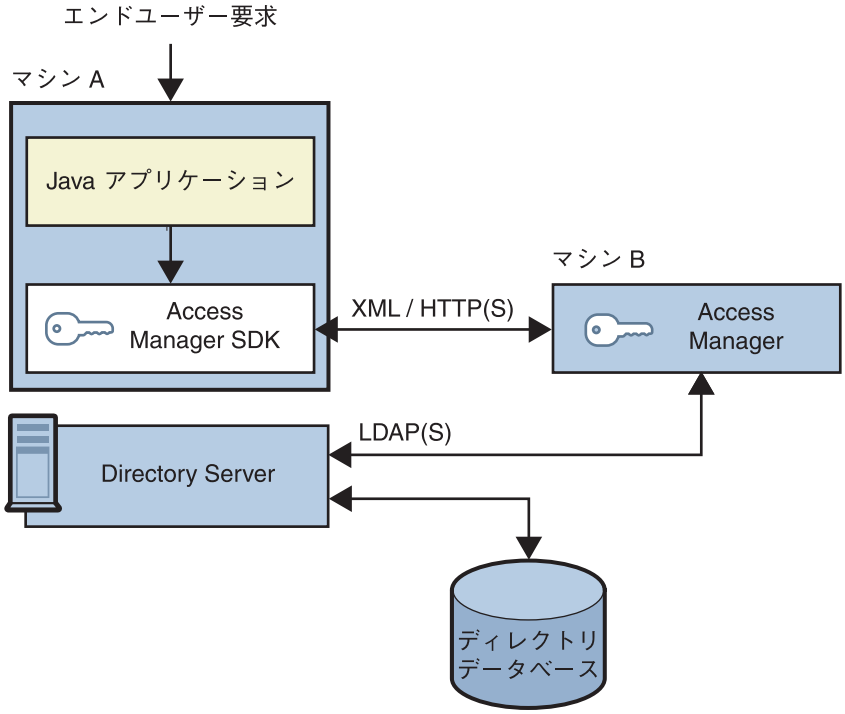


図 4–3　Java アプリケーションの配備

Access Manager セッションフェイルオーバー配備

Access Manager は、Web コンテナから独立したセッションフェイルオーバーの実装を提供しています。その際、Sun Java System Message Queue (Message Queue) を通信ブローカとして使用し、Berkeley DB をデフォルトのセッションストアデータベースとして使用します。Access Manager セッションフェイルオーバーは、単一のハードウェアまたはソフトウェアの障害発生時に、ユーザーの認証セッション状態を維持するため、セッション情報を失ったり、ユーザーに再ログインを要求したりせずに、ユーザーのセッションをセカンダリ Access Manager インスタンスにフェイルオーバーできます。

Access Manager セッションフェイルオーバーの概要

Access Manager 7.1 セッションフェイルオーバーには、次のコンポーネントが含まれます。

- Access Manager 7.1 の複数のインスタンス。各インスタンスは 2 つ以上のホストサーバー上のサポートされる Web コンテナで実行されます。
- Message Queue ブローカクラスタ。Access Manager インスタンスとセッションストアデータベースとの間のセッションメッセージを管理します。
- セッションストアデータベースとして Berkeley DB (`http://www.oracle.com/database/berkeley-db.html`)。Berkeley DB クライアントデーモンは `amsessiondb` です。

Access Manager のセッションフェイルオーバーは、次の Message Queue のパブリッシュ/サブスクライブ (トピック送信先) 配信モデルに従います。

1. ユーザーがセッションの開始、更新、または終了を行うと、Access Manager はセッションの作成、更新、または削除に関するメッセージを Message Queue ブローカクラスタにパブリッシュします。
2. Berkeley DB クライアント (`amsessiondb`) は Message Queue ブローカクラスタにサブスクライブし、セッションメッセージを読み取って、データベースにセッション操作を格納します。

Access Manager インスタンスが、単一のハードウェアまたはソフトウェアの問題によって失敗した場合、そのインスタンスに関連付けられているユーザーのセッションが、次のように、セカンダリ Access Manager インスタンスにフェイルオーバーします。

1. セカンダリ Access Manager インスタンスは、Message Queue ブローカクラスタに、ユーザーのセッション情報に対するクエリー要求をパブリッシュします。
2. Message Queue ブローカクラスタの同じセッション要求トピックにサブスクライブしている Berkeley DB クライアント (`amsessiondb`) は、クエリー要求を受信し、セッションデータベースから対応するエントリを取得して、ユーザーのセッション情報をセッション応答トピックとともに Message Queue ブローカクラスタにパブリッシュします。
3. セッション応答トピックにサブスクライブしているセカンダリ Access Manager インスタンスは、ユーザーのセッションを伴う応答を受信し、セッション情報を失ったり、ユーザーが再度ログオンしたりすることなく続行できます。

Message Queue ブローカが失敗すると、Access Manager は非セッションフェイルオーバーモードで動作します。あとで Message Queue ブローカが再起動すると、Access Manager はセッションフェイルオーバーモードに戻ります。

Message Queue コンポーネントおよびパブリッシュ/サブスクライブ配信モデルについては、『Sun Java System Message Queue 3.7 UR1 技術の概要』を参照してください。

セッションフェイルオーバー配備シナリオ

次の図に、2 台のホストサーバーから構成され、それぞれ Access Manager インスタンス (Web コンテナ上に配備)、Message Queue ブローカークラスタ、および Berkeley DB

クライアント (amsessiondb) を実行している基本的なシナリオを示します。ロードバランサはクライアント要求を Access Manager インスタンスに分配しています。両方の Access Manager インスタンスは同じ Directory Server にアクセスします。これは図に示されていません。

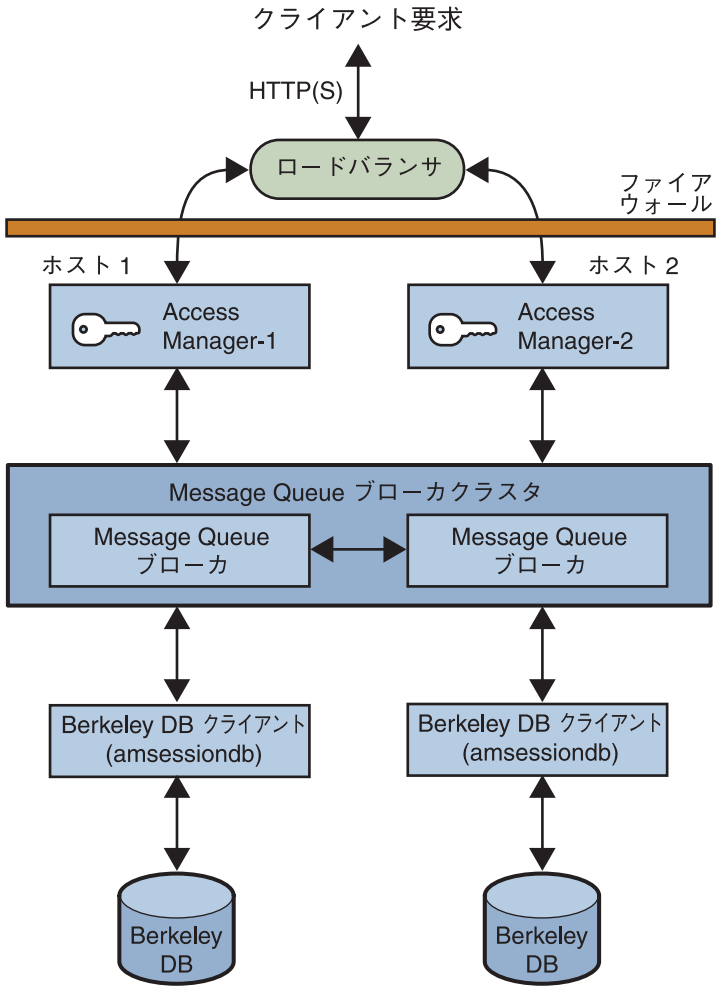


図 4–4　Access Manager セッションフェイルオーバー基本配備シナリオ

図に示すような、それぞれ同じ Directory Server にアクセスするサイトを追加できます。ただし、セッションフェイルオーバーは、サイト内の Access Manager に対してのみ実行され、現在のリリースでは、サイト間のセッションフェイルオーバーはサポートされていません。

詳細は、『Sun Java System Access Manager 7.1 Postinstallation Guide』の第 6 章「Implementing Session Failover」を参照してください。

Access Manager および Portal Server の配備

Java Enterprise System 5 リリースでは、Access Manager とPortal Server を同じ物理サーバーにインストールするか、複数のサーバーにインストールできます。

単一のサーバーへのインストール

このシナリオでは、Access Manager と Portal Server を同じ物理サーバーにインストールします。同じサーバーまたはリモートサーバーに、Directory Server をインストールするか、すでにインストールされている Directory Server にアクセスする必要があります。

これらのコンポーネントをインストールするには、Java Enterprise System インストーラを単独のセッションで実行し、次のように選択します。

- 「コンポーネントの選択」パネルで、次の製品とサブコンポーネントを選択します。
  - 「Communication & Collaboration Services」で、Portal Server を選択します。
  - 「Directory & Identity Services」で、Access Manager 7.1 とそのサブコンポーネントを選択します。
    - アイデンティティー管理およびポリシーサービスコア
    - Access Manager 管理コンソール
    - 連携管理の共有ドメインサービス
    - Access Manager SDK

      デフォルトでは、Portal Server を選択すると、インストーラは Access Manager SDK だけを選択します。したがって、その他のサブコンポーネントには別にチェックを付ける必要があります。

  次の Web コンテナの 1 つをインストールし、設定します。
  - Sun Java System Application Server
  - Sun Java System Web Server

複数のサーバーへのインストール

このシナリオでは、Portal Server は、リモートサーバーからローカルサーバー上の Access Manager にアクセスします。ローカルサーバーまたはリモートサーバーに、Directory Server をインストールするか、すでにインストールされている Directory Server にアクセスする必要があります。

- ローカルサーバーで、Access Manager と Web コンテナをインストールします。リモートサーバーでコンポーネントのインストールと設定を行う前に、このサーバー でコンポーネントのインストールと設定を行う必要があります。
- リモートサーバーで Portal Server と Access Manager SDK をインストールします。リモートサーバーでその他の Access Manager サブコンポーネントを選択する必要はありません。

Access Manager および Portal Server の配備については、『Sun Java System Portal Server 7.1 配備計画ガイド』を参照してください。

連携管理、SAML、および Web サービス

2001 年、Sun Microsystems はほかの企業とともに Liberty Alliance Project に加わりました。このプロジェクトでは、アイデンティティーベースのインフラストラクチャー、ソフトウェア、および Web サービスを開発するための標準を定義しています。

まず Access Manager が実装したのは、アカウント連携およびシングルサインオン (SSO) のためのフレームワークを構成する Liberty Identity Federation Framework (Liberty ID-FF) 仕様です。それ以降のリリースの Access Manager には、Liberty ID-FF 仕様のVersion 1.2 および Liberty Identity Web Services Framework (Liberty ID-WSF) の Version 1.0 仕様で定義されている新機能が追加されました。

Liberty ID-WSF フレームワークは、Liberty Alliance Project のビジネスモデルをサポートするために使用できる Web サービススタックを定義します。サービスの例には、個人プロファイルサービス、ディスカバリサービス、認証サービス、SOAP バインディングサービスなどがあります。これらの Web サービスでは、主体認証、連携、およびプライバシ保護のために Liberty ID-FF を利用します。

Access Manager は、セキュリティー情報を交換するための Security Assertion Markup Language (SAML) サービスも実装します。SAML 1.0 および 1.1 の両方の仕様がサポートされています。

詳細は、『Sun Java System Access Manager 7.1 Federation and SAML Administration Guide 』を参照してください。このガイドには、仕様の概要と、Access Manager でこれらの仕様がどのように実装されているかについての情報が記載されています。さらに、アプリケーションプログラミングインタフェース (API) の設定情報、ユースケース、および要約も記載されています。

Sun Java System Federation Manager

Sun Java System Federation Manager 7.0 2005Q4 は軽量のサーバーアプリケーションで、企業はこれを利用することで、相互運用性のあるアイデンティティーサービスおよび認証サービスを、Liberty Alliance Project 仕様に基づいて迅速に構築できます。これらのサービスは、Web アクセス管理ソリューションや認証局など、連携に関する既存の技術または新しく配備された技術と連動し、またそれらの技術を補完します。

Federation Manager を使用すると、再利用可能な標準ベースのフレームワークを構築して、セキュリティー表明、ユーザー属性、およびポリシーをパートナーの分散ネットワーク上で交換することができます。Federation Manager は、任意の Liberty または SAML 準拠製品と組み合わせて使用できるスタンドアロン製品です。Federation Manager を使用するために Access Manager をインストールする必要はありません。詳細は、次のマニュアルコレクションを参照してください。

http://docs.sun.com/coll/1321.1

Sun Java System Application Server 9.0 または Web サービス用の Sun Java System Access Manager Policy Agent 2.2

Sun Java System Application Server 9.0 または Web サービス用の Sun Java System Access Manager Policy Agent 2.2 は、Sun Java System Application Server Platform Edition 9.0 のプラグインとして動作し、Liberty Alliance Project トークンプロファイルおよび Web Services-Interoperability Basic Security Profiles (WS-I BSP) の両方に対して、メッセージレベルのセキュリティーとサポートを提供します。このエージェントは HTTP 認証エージェントと SOAP 認証エージェントの両方を提供し、すべての認証決定のために Access Manager 7.1 を使用します。

エージェントのインストール手順などの詳細は、『Sun Java System Access Manager Policy Agent 2.2 Guide for Sun Java System Application Server 9.0/Web Services 』へのリンクを参照してください。

注 – Sun Java System Application Server Platform Edition 9.0 は Java Enterprise System 5 コンポーネントではありません。詳細は、次のマニュアルコレクションを参照してください。

http://docs.sun.com/coll/1343.3

◆ ◆ ◆ 第 5 章

Access Manager での配備設計

ソリューションライフサイクルの配備設計段階では、ハイレベルの配備アーキテクチャーとローレベルの実装仕様を設計し、ソリューションを実装するために必要な一連の計画および仕様を準備します。この配備設計段階でプロジェクトの承認が行われます。この章は、Sun Java™ System Access Manager での配備設計に関する次の節で構成されています。

ロードバランサの使用法

ほとんどの配備では、ユーザー要求を 2 つ以上の Access Manager インスタンスに分散するために、Access Manager はロードバランサを使用して設定されます。ロードバランサは、ハードウェア、ソフトウェア、またはその両方の組み合わせを使用して実装できます。次の図は、ロードバランサを使用した Access Manager 配備を示しています。

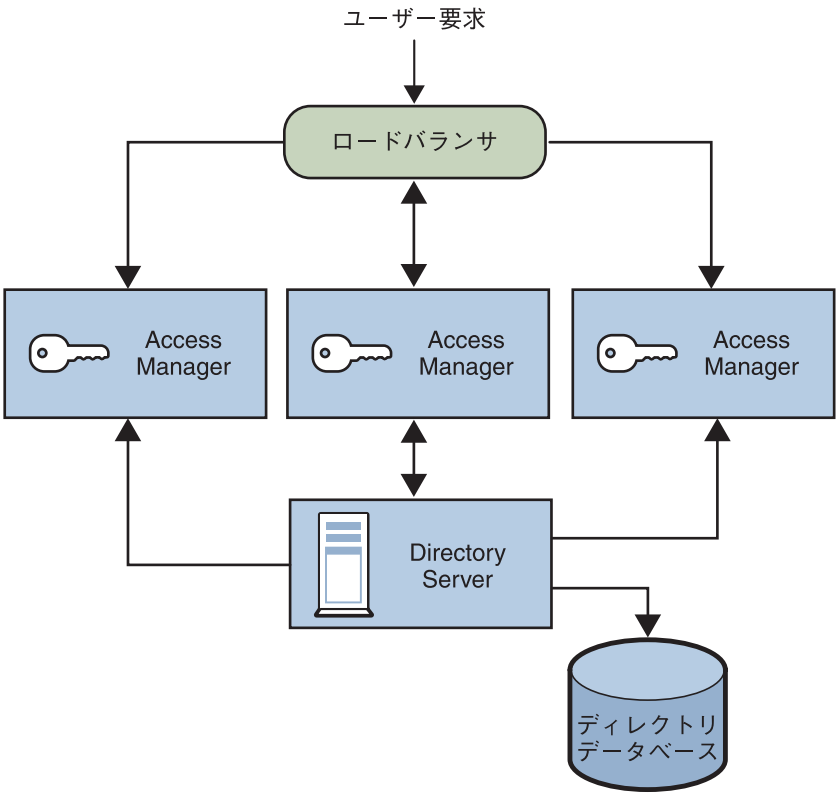


図 5–1　ロードバランサを使用した Access Manager の設定

Cookie ベースのスティッキー要求ルーティング

Access Manager とともに配備するロードバランサは、スティッキーセッションをサポートしている必要があります。スティッキーセッションでは、セッションが特定の Access Manager インスタンスによって作成されると、セッション情報を保持するために、ユーザーからのそれ以降の要求は引き続きその同じインスタンスに配信されます。Access Manager は Cookie を使用してセッション情報を中継するため、ロードバランサは、そのセッションを作成した Access Manager インスタンスに要求をリダイレクトする必要があります。

そのため、Access Manager では Cookie ベースのスティッキー要求ルーティングを実装しました。これにより、クライアント要求が間違った Access Manager サーバー (セッションをホストしていないサーバー) に誤ってルーティングされることが防止されます。この機能により、サーバー間でのバックチャネル通信が不要になり、Access Manager のパフォーマンスが向上します。

詳細は、『Sun Java System Access Manager 7.1 Postinstallation Guide』の「Configuring Cookie-Based Sticky Request Routing」を参照してください。

複数の JVM 環境

Access Manager サービスは、複数の Java 仮想マシン (JVM) 環境でサポートされています。これは、Sun Java System Application Server のインスタンスは複数の JVM を保持するように設定でき、そのすべてで Access Manager サービスが稼働可能であることを意味します。Access Manager のアーキテクチャーでは、マシン内の Sun Java System Application Server インスタンスの数、複数マシンをまたがる Access Manager サービスの数、単一の Application Server が保持できる JVM の数などに関する配備に制限を課しません。

複数の JVM 環境については、次の Sun Java System Application Server のマニュアルを参照してください。http://docs.sun.com/coll/1310.3

Directory Server レプリケーションに関する考慮事項

Access Manager のパフォーマンスと応答時間を改善するには、レプリケートされた Directory Server 間でロードバランスを使用する方法と、レプリケートされたサーバーをユーザーの近くに配置する方法の 2 つがあります。Directory Server は、シングルサプライヤ構成またはマルチサプライヤ構成でセットアップできます。Sun Java System Directory Proxy Server などのロードバランスアプリケーションも使用できます。Directory Proxy Server は、設定された Directory Server セット間の LDAP 操作のプロポーショナルなロードバランスを動的に実行します。1 つ以上の Directory Server インスタンスが利用できない場合、負荷は残りのサーバー間でバランス良く再配分されます。サーバーが復帰すると、負荷がバランス良くかつ動的に再配分されます。

Access Manager をインストールする前に、Directory Server レプリケーションを設定する必要があります。この設定によって、サプライヤとコンシューマのデータベースが正しく同期されるため、参照や更新が適切に同期されていることを確認する時間が取れるようになります。

Access Manager をレプリケーション目的でインストールした場合、Directory Server の各インスタンスおよび Access Manager の各インスタンスは、以下に対して同じ値を使用して設定する必要があります。

- ディレクトリマネージャー
- ディレクトリマネージャーのパスワード
- Directory Server の管理者 ID
- サーバー管理者のパスワード
- ベースサフィックス
- デフォルトの組織

レプリケーション用の設定

Access Manager は、シングルサプライヤまたはマルチサプライヤのレプリケーションで動作するように設定できます。次の図は、コンシューマが読み取り専用のデータベースであるシングルサプライヤ構成を示しています。書き込み操作要求の参照は、サプライヤデータベースに対して行われます。この設定により、負荷が複数のディレクトリに分散させられるため、サーバーのパフォーマンスを向上させる手段として利用できます。

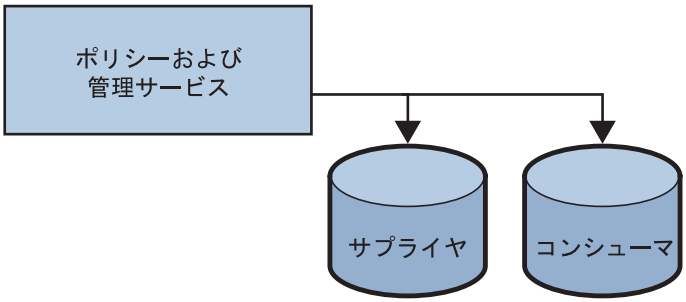


図 5–2　シングルサプライヤの Directory Server レプリケーション

次の図は、Access Manager の複数インスタンスを使用したマルチサプライヤ構成、またはマルチマスターレプリケーション (MMR) を示しています。この設定によりフェイルオーバー保護および高可用性が実現されるため、サーバーのパフォーマンスはさらに向上します。

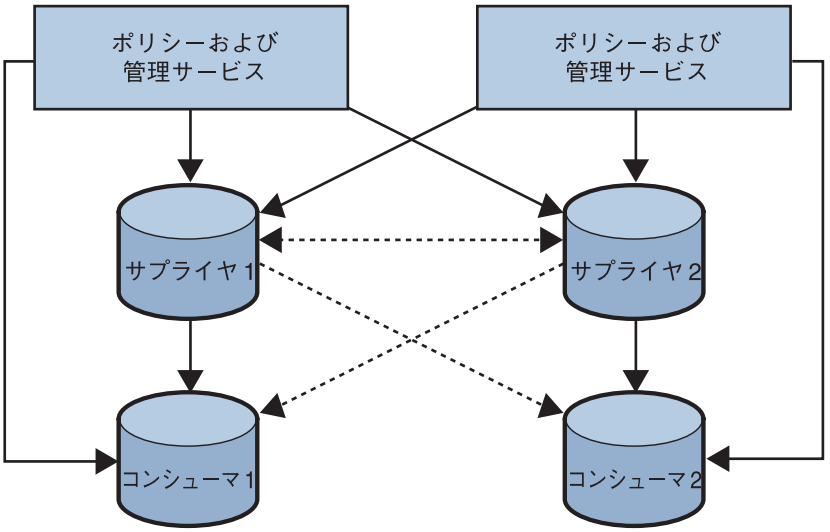


図 5–3　マルチサプライヤの Directory Server 構成

以下の手順を使用すると、Access Manager がまだインストールされていない場合に、Access Manager ディレクトリツリーのルートまたは最上位レベルでレプリケーションを設定したり、デフォルトの組織レベルでレプリケーションを設定したりすることができます。

1. サプライヤおよびコンシューマとして使う Directory Server インスタンスをインストールします。

   手順については、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』を参照してください。

2. サプライヤおよびコンシューマ間のレプリケーションアグリーメントを設定し、ディレクトリ参照および更新が正しく機能することを確認します。

   このバージョンの Access Manager で機能するように、既存の Directory Server データの移行が必要になる場合があります。詳細は、『Sun Java System Access Manager 6 2005Q1 Migration Guide』を参照してください。

3. Access Manager および Directory Server を初めて配備する場合や、既存のユーザーデータを使用する予定がない場合は、Java ES インストールプログラムを実行して Access Manager をインストールしてください。

   インストール時に、既存の Directory Server が存在するかどうか尋ねられたら「はい」を選択し、76 ページの「レプリケーション用の設定」でインストールしたサプライヤ Directory Server のホスト名とポート番号を指定します。

4. Access Manager がインストールされているホストサーバーで、使用しているプラットフォームに応じて、次のディレクトリの `AMConfig.properties` ファイルを修正します。

   - Solaris システム: `/etc/opt/SUNWam/config`
   - Linux および HP-UX システム: `/etc/opt/sun/identity/config`
   - Windows システム: *javaes-install-dir*`\identity\config`

     javaes-install-dir 変数は Java ES 5 のインストールディレクトリを表します。デフォルト値は `C:\Program Files\Sun\JavaES5` です。

5. 次のプロパティーを修正して、76 ページの「レプリケーション用の設定」でインストールしたコンシューマ Directory Server のホストおよびポート番号を反映します。

   - `com.iplanet.am.directory.host`
   - `com.iplanet.am.directory.port`

6. 次のプロパティーを修正して、要求されたエントリが見つからない場合に Access Manager が同じ要求を繰り返す回数を指定します。

   `com.iplanet.am.replica.retries`

7. 次のプロパティーを修正して、Access Manager が再試行を行うまでの時間をミリ秒単位で指定します。

   `com.iplanet.am.replica.delay.between.retries`

8. 有効になっている Access Manager 認証モジュールごとに、Access Manager コンソールを使用して、76 ページの「レプリケーション用の設定」でインストールしたコンシューマディレクトリを指定します。
   - 最初の LDAP サーバーとポートには、プライマリ (コンシューマ) Directory Server のホスト名とポート番号を指定します。たとえば、`consumer1.example.com:389` と指定します。
   - 2 番目の LDAP サーバーとポートには、セカンダリ (サプライヤ) Directory Server のホスト名とポート番号を指定します。たとえば、`supplier1.example.com:389` と指定します。
9. `serverconfig.xml` ファイルで、76 ページの「レプリケーション用の設定」でインストールしたコンシューマディレクトリのホスト名とポート番号を指定します。`serverconfig.xml` ファイルの例を次に示します。
10. Web コンテナを再起動して Access Manager を再起動します。

`serverconfig.xml` ファイルの例

次の例は、`serverconfig.xml` のレプリケーションの変更を示しています。

```
<iPlanetDataAccessLayer>
<ServerGroup name="default" minConnPool="1"
maxConnPool="10">
<Server name="Server1"
host="consumer1.example.com" port="389"
type="SIMPLE" />
```

ロードバランサを使用する場合の設定

次の図は、Directory Proxy Server またはハードウェアロードバランサを含むマルチサプライヤ構成を示しています。この設定により、Access Manager によりサポートされているフェイルオーバー、高可用性、および管理されたロードバランスをうまく活用することができます。

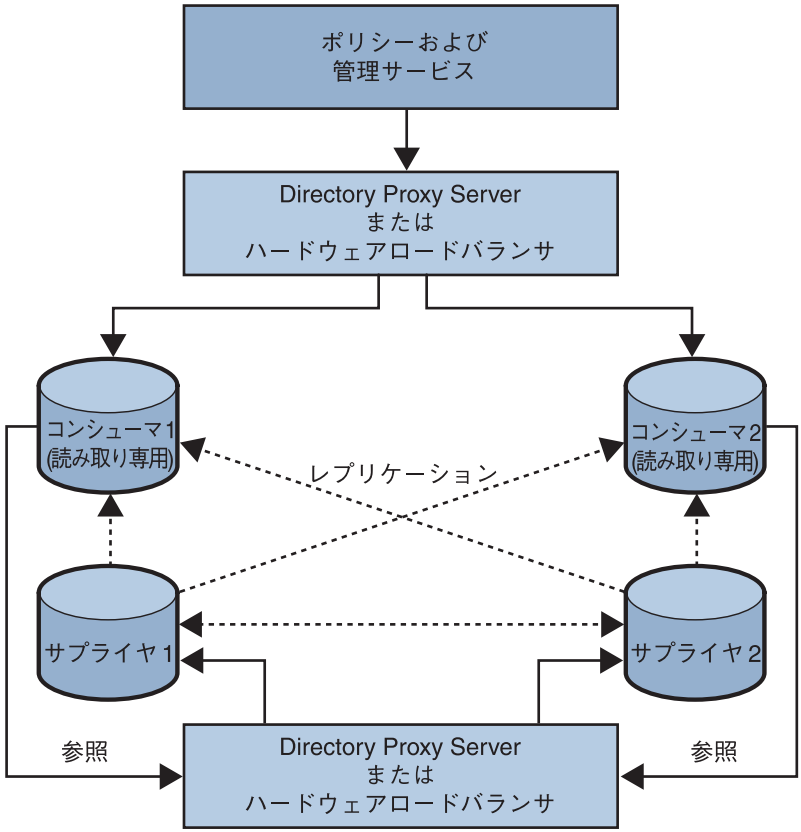


図 5–4　ロードバランサを使用したマルチサプライヤ構成

LDAP ロードバランサを使用することにより、Access Manager で提供されるレベルを上回る高可用性とディレクトリフェイルオーバー保護の機能が追加されます。たとえば、Directory Proxy Server は、各サーバーに再配分される負荷の割合を指定できます。また、すべてのバックエンド LDAP サーバーが使用不可になった場合は、Directory Proxy Server が引き続き要求を管理し、クライアントのクエリーを拒否します。ロードバランサをインストールする場合は、このアプリケーションを認識するように Access Manager を設定する必要があります。

1. Access Manager を設定する前に、Directory Server をレプリケーション用にセットアップします。ディレクトリレプリケーションおよびセットアップ手順については、Sun Java System Directory Server のマニュアルを参照してください。 http://docs.sun.com/coll/1224.1
2. LDAP ロードバランサをインストールおよび設定します。使用しているロードバランサに同梱されているマニュアルの指示に従ってください。

3. AMConfig.properties ファイルで、 com.iplanet.am.directory.host および com.iplanet.am.directory.port プロパティーを、コンシューマ Directory Server のロードバランサのホストおよびポート番号を指すように修正します。
4. 有効になっている Access Manager 認証モジュールごとに、Access Manager コンソールを使用して、コンシューマ Directory Server を指定します。次の手順では、例としてLDAP 認証モジュールを使用します。
   - 最初の LDAP サーバーとポートには、プライマリ(コンシューマ) Directory Server のホスト名とポート番号を、proxyhostname:port の形式で入力します。
   - 2 番目の LDAP サーバーとポートには何も入力しないでください。
5. serverconfig.xml ファイルで、コンシューマ Directory Server のホスト名とポート番号を指定します。serverconfig.xml ファイルの例を次に示します。
6. Web コンテナを再起動して Access Manager を再起動します。

serverconfig.xml ファイルに対するロードバランサの変更

次の例は、serverconfig.xml ファイルに対するロードバランサの変更を示しています。

```
<iPlanetDataAccessLayer>
<ServerGroup name="default" minConnPool="1"
maxConnPool="10">
<Server name="Server1"
host="idar.example.com" port="389"
type="SIMPLE"
```

ファイアウォールを使用した Directory Server

Access Manager とDirectory Server の間にファイアウォールが設定されている場合、ファイアウォールのアイドル接続タイムアウト値が、Directory Server のアイドル接続タイムアウト値 (nsslapd-idletimeout 属性) を下回ると、Access Manager 接続がタイムアウトすることがあります。この問題は通常、Access Manager の負荷が低く、使用率がピークに達していないときに発生します。

Directory Server 接続がファイアウォールによって切断されると、Access Manager は、この接続が切断されていることを認識せず、LDAP 接続プールで使用可能なすべての接続を使い果たします。LDAP 接続プールを新規に作成するには、Access Manager を再起動する必要があります。この問題を回避するには、次の解決策を検討してください。

- 81 ページの「グローバルタイムアウト属性の設定」
- 81 ページの「個々のクライアント接続のタイムアウトの設定」

グローバルタイムアウト属性の設定

Directory Server の nsslapd-idletimeout グローバル属性を、ファイアウォールのアイドル接続タイムアウト値より小さい値に設定できます。ただし、nsslapd-idletimeout は Access Manager 以外のアプリケーションにも影響するグローバル設定属性であるため、この解決策を採用できない場合もあります。

個々のクライアント接続のタイムアウトの設定

Directory Server では、個々のクライアント接続に対して個別の属性を設定できます。nsIdleTimeout 属性では、個々のクライアントのアイドル接続タイムアウト値を指定します。この値は、Directory Server のグローバル設定で指定した nsslapd-idletimeout 値よりも優先されます。

LDAP ディレクトリにバインドした Access Manager ユーザーについて、nsIdleTimeout 属性を設定します。このユーザーは、デフォルトでは amldapuser になっています。この属性は、dsameuser および puser ユーザーにも適用されます。

amldapuser に nsIdleTimeout 属性を追加するには、Directory Server コンソールまたは ldapmodify ツールのどちらかを使用します。次に例を示します。

```
ldapmodify -h host-name -p port
-D "cn=Directory Manager" -w password
dn: cn=amldapuser,ou=DSAME Users, dc=example,dc=com
changetype: modify
add: nsIdleTimeout
nsIdleTimeout: timeout-value
```

timeout-value には、ファイアウォールについて設定したアイドル接続タイムアウト値よりも低い値を指定します。このように設定すると、amldapuser ユーザーの Access Manager 接続は、ファイアウォールによって切断される前に、Directory Server によって切断されます。

dsameuser または puser にタイムアウトを追加する場合にも、上の構文を使用しますが、dn オプションを dsameuser または puser ユーザーに設定します。

AMConfig.properties ファイルの com.sun.am.event.connection.idle.timeout プロパティーでは、持続検索が再起動するまでのタイムアウト値 (分) を指定します。このプロパティーを指定しておけば、接続が切断されたときに持続検索が確実に再起動します。この値は、ロードバランサまたはファイアウォール TCP のタイムアウト値よりも低い値にしておくことが理想的です。そうすれば、接続が切断する前に持続検索が再起動するようになるからです。デフォルト値はゼロ (0) で、持続検索は再起動されません。

Directory Server 属性と `ldapmodify` ツールについては、次の Sun Java System Directory Server のマニュアルを参照してください。`http://docs.sun.com/coll/1224.1`

◆ ◆ ◆ 第 6 章

Access Manager 設計の実装

ソリューションライフサイクルの実装段階では、配備設計段階で作成した仕様および計画をもとに作業を始め、配備を構築およびテストし、最終的に配備を本稼働環境にロールアウトします。

注 – 『Sun Java System Access Manager 7.1 Postinstallation Guide 』では現在、『Access Manager 7 2005Q4 配備計画ガイド』の以前のバージョンからの実装情報について掲載しています。

実装段階のタスク

ソリューションライフサイクルの実装段階に含まれるタスクには、次のものがあります。

- ネットワークおよびハードウェアのインフラストラクチャーを決定し、構築する
- インストール計画に従って、Access Manager および関連ソフトウェアをインストールし、設定する
- Access Manager のデータを既存のアプリケーションから現在の配備に移行する
- Access Manager のユーザー管理計画を実装する
- テスト計画に従って、Access Manager のパイロットまたはプロトタイプを設計し、テスト環境に配備する
- テスト計画に従って、機能テストおよびストレステストを設計し、実施する
- ロールアウト計画に従って、Access Manager のテスト環境を本稼働環境にロールアウトする
- トレーニング計画に従って、配備環境の Access Manager 管理者およびユーザーに対しトレーニングを実施する

◆ ◆ ◆ 付 録 A

インストールされる製品のレイアウト

この付録では、Sun Java™ System Enterprise System (Java ES) インストーラを使用して Sun Java™ System Access Manager 7.1 をインストールしたあとのディレクトリレイアウトについて説明します。

Access Manager 7.1 WAR ファイルを配備している場合は、『Sun Java System Access Manager 7.1 Postinstallation Guide』の第 12 章「Deploying Access Manager as a Single WAR File」を参照してください。

次の表は、インストール後の Access Manager のデフォルトディレクトリの概要を示しています。

Access Manager ディレクトリの概要

表 A–1 Access Manager ディレクトリの概要

説明	デフォルトディレクトリ
ベースインストールディレクトリ 86 ページの「ベースインストールディレクトリ」を参照してください。	Solaris システム: `/opt/SUNWam` Linux および HP-UX システム: `/opt/sun/identity` Windows システム: `C:\Program Files\Sun\JavaES5\identity` インストール中に、必要に応じて `/opt`、`/opt/sun`、または `C:\Program Files\Sun\JavaES5` と異なるベースインストールディレクトリを指定できます。 ただし、製品ディレクトリ名 (`/SUNWam`、`/identity` 、または `\identity`) は変更しないでください。

表 A–1 Access Manager ディレクトリの概要 (続き)

説明	デフォルトディレクトリ
設定ディレクトリ 92 ページの「設定 (/config) ディレクトリ」を参照してください。	Solaris システム: /etc/opt/SUNWam/config Linux および HP-UX システム: /etc/opt/sun/identity/config Windows システム: C:\Program Files\Sun\JavaES5\identity\config
一時ファイルディレクトリ	Solaris システム: /var/opt/SUNWam/tmp Linux および HP-UX システム: /var/opt/sun/identity/tmp Windows システム: C:\Program Files\Sun\JavaES5\identity\tmp
デバッグファイルディレクトリ	Solaris システム: /var/opt/SUNWam/debug Linux および HP-UX システム: /var/opt/sun/identity/debug Windows システム: C:\Program Files\Sun\JavaES5\identity\debug
ログファイルディレクトリ	Solaris システム: /var/opt/SUNWam/logs Linux および HP-UX システム: /var/opt/sun/identity/logs Windows システム: C:\Program Files\Sun\JavaES5\identity\logs

ベースインストールディレクトリ

デフォルトのベースインストールディレクトリは、Access Manager をインストールするプラットフォームによって異なります。

- Solaris システム: /opt
- Linux および HP-UX システム: /opt/sun
- Windows システム: C:\Program Files\Sun\JavaES5

Access Manager のマニュアルでは、*AccessManager-base* 変数は Solaris、Linux、および HP-UX システムのベースインストールディレクトリを表します。Windows システムの場合は、*javaes-install-dir* 変数が Java ES 5 のインストールディレクトリを表します。

ベースインストールディレクトリ内で、Access Manager パッケージ、共有バイナリファイル、コマンド行ツール、およびその他のファイルは、Solaris システムでは /SUNWam ディレクトリ、Linux および HP-UX システムでは /identity ディレクトリ、Windows システムでは \identity ディレクトリにインストールされます。したがって、デフォルトのベースおよび製品ディレクトリも、プラットフォームによって異なります。

- Solaris システム: /opt/SUNWam
- Linux および HP-UX システム: /opt/sun/identity
- Windows システム: C:\Program Files\Sun\JavaES5\identity

注– インストール中に、必要に応じて別のベースインストールディレクトリを指定できますが、製品ディレクトリ名 (/SUNWam、/identity 、または \identity) は変更しないでください。

Windows システムでは、Access Manager を設定するために使用できる次のファイルが \setup に格納されます。

- amconfig.bat は、Access Manager の配備、設定、および再設定に使用されるバッチファイルです。このファイルは、UNIX および Linux プラットフォームでの amconfig スクリプトに相当します。
- AMConfigurator.properties は、Access Manager の設定プロパティーが格納される設定入力ファイルです。このファイルは、UNIX および Linux プラットフォームでの amsamplesilent ファイルに相当します。AMConfigurator.properties の値にはバックスラッシュ (\) を含めないでください。

/SUNWam、/identity、または \identity ディレクトリには、次のファイルとディレクトリが含まれます。

- Web アプリケーションアーカイブ (WAR) ファイル (amcommon.war、amconsole.war、ampassword.war、および amserver.war)

  WAR ファイルについては、『Sun Java System Access Manager 7.1 Developer's Guide』および『Sun Java System Access Manager 7.1 Postinstallation Guide』の第 12 章「Deploying Access Manager as a Single WAR File」を参照してください。

  サブディレクトリ:

  - 88 ページの「/bin ディレクトリ」
  - 89 ページの「/docs ディレクトリ」
  - 89 ページの「/dtd ディレクトリ」
  - 90 ページの「/include ディレクトリ」
  - 90 ページの「/ldaplib ディレクトリ」
  - 90 ページの「/lib ディレクトリ」
  - 90 ページの「/locale ディレクトリ」
  - 91 ページの「/migration ディレクトリ」
  - 91 ページの「/public_html ディレクトリ」
  - 91 ページの「/samples ディレクトリ」
  - 91 ページの「/share ディレクトリ」
  - 91 ページの「/upgrade ディレクトリ」
  - 92 ページの「/web-src ディレクトリ」

Access Manager をインストールしたあと、pkgchk (1M) ユーティリティーを使用して、パッケージのインストールが正しく行われたことを確認してください。次に例を示します。

```
pkgchk -l -p /opt/SUNWam
```

/bin ディレクトリ

次の表では、/bin ディレクトリのコマンド行ツールおよびユーティリティーについて説明しています。これらのツールおよびユーティリティーについては、『Sun Java System Access Manager 7.1 Administration Reference 』を参照してください。

表 A–2　Access Manager のコマンド行ツールおよびユーティリティー

ユーティリティー	説明
am2bak am2bak.bat (Windows)	Access Manager コンポーネントをバックアップします。
amadmin amadmin.bat (Windows)	XML サービスを Directory Server にロードし、DIT でバッチ管理タスクを実行します。
amsfo、amsfoconfig、amsfopassword amsfo.pl、amsfoconfig.bat、amsfopassword.bat (Windows)	Access Manager セッションフェイルオーバースクリプト。
ampassword ampassword.bat (Windows)	Access Manager 管理者またはユーザーのパスワードを変更します。
amsamplesilent	インストールスクリプトおよび設定スクリプトで使用するサンプルのサイレントインストールファイル。
amconfig、amutils、amdsconfig 、amsdkconfig、amsvcconfig、amas70config、amwas51config、amwl81config 、amws61config	Access Manager インスタンスのインストール、設定、およびアンインストールに使用する、インストール、および設定スクリプト。これらのスクリプトについては、『Sun Java System Access Manager 7.1 Postinstallation Guide』の第 2 章「Running the Access Manager amconfig Script」を参照してください。
amserver	amunixd デーモンと amsecuridd デーモンを起動および停止します。
amtune ディレクトリ	Access Manager チューニングスクリプトが含まれます。これらのスクリプトを使用して、オペレーティングシステム、Access Manager、Web コンテナ、および Directory Server のパラメータを設定し、パフォーマンスを向上させることができます。
amverifyarchive amverifyarchive.bat (Windows)	ログアーカイブを調べて、アーカイブ内のすべてのファイルの改ざんや削除を検出します。
bak2am bak2am.bat (Windows)	am2back または am2back.bat ユーティリティーによってバックアップされた Access Manager コンポーネントを復元します。

表 A–2 Access Manager のコマンド行ツールおよびユーティリティー (続き)

ユーティリティー	説明
ldapmodify	新規エントリを追加するか、既存のエントリを変更して、LDAP ディレクトリの内容を編集します。
ldapsearch	LDAP ディレクトリに検索要求を発行し、結果を LDIF テキストで表示します。
amGenerateLDIF.pl、amGenerateNI.pl	Access Manager の一括連携スクリプト。
am2bak.template、amserver.template、amadmin.template、amverifyarchive.template、ampassword.template、bak2am.template	Access Manager のテンプレートファイル。

/docs ディレクトリ

/docs ディレクトリには、Java API リファレンス (Javadoc) 用に使用される HTML、JAR、CSS、および関連ファイルが含まれます。

/dtd ディレクトリ

/dtd ディレクトリには、Access Manager で使用される DTD (Document Type Definition) ファイルが含まれます。DTD は、Access Manager がアクセスする XML ファイルの構造を定義します。詳細は、『Sun Java System Access Manager 7.1 Developer's Guide 』を参照してください。

次の表では /dtd ディレクトリの Access Manager DTD ファイルについて説明しています。

表 A–3 Access Manager DTD ファイル

ファイル	説明
Auth_Module_Properties.dtd	認証モジュールがプロパティーの指定に使用するXML ファイルの構造を定義します。
amAdmin.dtd	amAdmin コマンド行ツールを使用してディレクトリツリー上でバッチ LDAP 操作を実行する際に使用する XML ファイルの構造を定義します。

表 A–3 Access Manager DTD ファイル (続き)

ファイル	説明
amWebAgent.dtd	Web エージェントからの要求を処理し、応答を Web エージェントに送信する際に使用する XML ファイルの構造を定義します。これは、下位互換性を維持する目的で残されている非推奨のファイルです。
policy.dtd	ポリシーを Directory Server に格納する際に使用する XML ファイルの構造を定義します。
remote-auth.dtd	認証サービスのリモート認証 API により使用される XML ファイルの構造を定義します。
server-config.dtd	すべてのサーバーおよびユーザータイプの ID、ホスト、およびポート情報を記述する serverconfig.xml の構造を定義します。
sms.dtd	XML サービスファイルの構造を定義します。
web-app_2_2.dtd	Access Manager 配備コンテナが J2EE アプリケーションを配備する際に使用する XML ファイルの構造を定義します。

/include ディレクトリ

/include ディレクトリにはヘッダー (.h) ファイルが含まれます。

/ldaplib ディレクトリ

/ldaplib/ldapsdk サブディレクトリには、Access Manager に含まれる LDAP ユーティリティーの実行に必要な共有オブジェクト (.so) ファイルが含まれます。

/lib ディレクトリ

/lib ディレクトリには、JAR ファイルおよび追加の共有オブジェクト (.so) ファイルが含まれます。また、AMConfig.properties ファイルへのリンクも含まれます。

/locale ディレクトリ

/locale ディレクトリには、ローカリゼーションプロパティーファイルが含まれます。各プロパティーファイルには、対応する英語版のローカリゼーションファイルが含まれます。たとえば、amAdminCLI_en.properties は amAdminCLI.properties に対応するファイルです。

/migrationディレクトリ

/migrationディレクトリには、以前のバージョンのAccess Managerからのデータの移行に使用するスクリプトとサポートファイルが含まれます。

移行については、次のマニュアルコレクション内の『Sun Java Enterprise System 5 アップグレードガイド』を参照してください。http://docs.sun.com/coll/1286.2

/public_htmlディレクトリ

/public_htmlディレクトリとそのサブディレクトリには、Access Managerコンソールのオンラインヘルプで使用されるHTMLファイルおよび関連するファイルが含まれます。

/samplesディレクトリ

/samplesディレクトリには、次のサブディレクトリが含まれます。 /admin、/appserver、/authentication、/console、/csdk、/liberty、/logging、/phase2、/policy、/saml、/sso、および/um。

各サブディレクトリには、サブディレクトリ名で示されたそれぞれの機能に応じたサンプルが含まれます。これらのサンプル別の詳細については、Readme.htmlファイルを参照してください。

/shareディレクトリ

/share/binサブディレクトリには、amsecuridd、amunixd、amwar、checkport、wsutils.kshなど、Access Managerによって内部的に使用される追加ユーティリティーが含まれます。

/upgradeディレクトリ

/upgradeディレクトリには、次のディレクトリが含まれます。

- /scriptsには、アップグレードスクリプトおよびファイルが含まれます。
- /servicesディレクトリには、Access Managerサービスで使用されるディレクトリが含まれます。

/web-src ディレクトリ

/web-src ディレクトリには、Web コンテナ上での Access Manager J2EE Web アプリケーションの配備先サブディレクトリが含まれます。次のサブディレクトリが含まれます。

- applications/ ディレクトリ。Access Manager コンソールを配備します。index.html ファイルと各種サブディレクトリが含まれます。/console ディレクトリには、各種コンソール関連のサブディレクトリが含まれます。
- /common ディレクトリ (およびサブディレクトリ)。Access Manager Liberty Common Domain コンポーネントを配備します。
- /password ディレクトリ (およびサブディレクトリ) 。Access Manager Password Synchronization コンポーネントを配備します。index.html ファイルと、各種サブディレクトリが含まれます。
- /services ディレクトリ (およびサブディレクトリ) 。Access Manager Core Service を配備します。index.html ファイルと、各種サブディレクトリが含まれます。

設定 (/config) ディレクトリ

設定 (/config) ディレクトリのデフォルトの場所は、Access Manager がインストールされたプラットフォームによって異なります。

- Solaris システム: /etc/opt/SUNWam/config
- Linux および HP-UX システム: /etc/opt/sun/identity/config
- Windows システム: C:\Program Files\Sun\JavaES5\identity\config

/config ディレクトリには、次のような設定ファイル、XML ファイル、および LDIF ファイルが含まれます。

- .version ファイルには、Access Manager の現在のバージョンが記述されています。
- AMConfig.properties および SSOConfig.properties ファイルには、Access Manager の設定属性が含まれます。
- serverconfig.xml ファイルは、Directory Server 用の Access Manager の設定情報を提供します。
- /ldif サブディレクトリには、Access Manager のインストール時に Directory Server データストアを生成するために必要な LDIF ファイルが含まれます。次に例を示します。
  - インストール時に、ds_remote_schema.ldif ファイルは、Access Manager のデータを Directory Server に格納するために必要な Access Manager 固有の LDAP スキーマオブジェクトクラスおよび属性 (iplanet-am-managed-people-container など) をロードします。sunone_schema2.ldif ファイルは、Access Manager LDAP 固有のスキーマオブジェクトクラスおよび属性をロードします。

- アンインストール時に、ds_remote_schema_uninstall.ldif ファイルは、Access Manager の LDAP スキーマオブジェクトおよび属性を Directory Server から削除します。

- /xml サブディレクトリには、XML ファイルが含まれます。
- /ums サブディレクトリには、次のような XML ファイルが含まれます。
  - amserveradmin スクリプトは、Access Manager サービスをロードします。
  - ums.xml ファイルは、Access Manager によって管理されるオブジェクトの LDAP 設定情報を含むテンプレートセットを提供します。
  - 通常、XML ファイルは設定に使用されません。これらの XML ファイルを修正した場合は、Directory Server データストアに手動で再ロードする必要があります。サーバーでの変更は一切、これらのファイルと同期しません。このディレクトリの XML ファイルについては、『Sun Java System Access Manager 7.1 Developer's Guide』を参照してください。

索引

ろ