配備ガイド

Sun™ ONE Identity Server

Version 6.1

817-4727-10
2003 年 12 月

目 次

図目次

表目次

コード例一覧

本書について

本書『Sun™ ONE Identity Server 配備ガイド』は、各種の技術情報のほか、組織へ Sun ONE Identity Server の配備を開始するための、シンプルなシナリオを提供しています。また、アイデンティティ管理および配備チームの選択方法に関する一般的な情報も提供します。ここでは、次の項目について説明します。

本書の対象読者

この『配備ガイド』は、Sun ONE のサーバーおよびソフトウェアを使用した統合アイデンティティサービスおよび Web アクセスプラットフォームを実装する IT 管理者向けに書かれています。管理者は次の技術に精通していることが推奨されます。

- LDAP (Lightweight Directory Access Protocol)
- Java™
- JavaServer Pages™ (JSP)
- HTTP (HyperText Transfer Protocol)
- HTML (HyperText Markup Language)
- XML (eXtensible Markup Language)

Sun ONE Directory Server は Identity Server の配備ではデータストアとして使用するため、管理者は、この製品に付属のマニュアルも目を通しておく必要があります。Directory Server の最新のマニュアルには、オンラインでアクセスできます。

Identity Server 6.1 のマニュアルセット

Identity Server のマニュアルセットは、2 つのコアマニュアルセットに分けられます。1 つは Sun ONE Identity Server 6.1 のコアアプリケーションマニュアルで、もう 1 つは Sun ONE Identity Server ポリシーエージェント関連のマニュアルです。

Identity Server のコアマニュアル

Identity Server のマニュアルセットには、次のマニュアルが含まれています。

- 『Product Brief』: Identity Server アプリケーションの概要と機能について説明します。
- 『Migration Guide』: 既存のデータおよび Sun ONE 製品の配備を最新バージョンの Identity Server に移行する方法について説明します。Identity Server のインストール方法については、『Sun Java Enterprise System 2003Q4 インストールガイド』を参照してください。
- 『管理ガイド』: Identity Server コンソールの使用方法と、コマンド行によるユーザー管理およびデータサービスの方法について説明します。
- 『Customization and API Guide』: Identity Server インストールのカスタマイズ方法について説明します。また、公共の API を使ってアプリケーションに新しいサービスを付加する方法についても説明します。
- 『配備ガイド』(本書): 既存の情報技術インフラストラクチャ内に Identity Server を配備する方法について説明します。
- 『リリースノート』は、製品のリリース後、オンラインでのみご利用になれます。このリリースの最新情報、既知の問題、制限事項、インストールに関する注意事項、ソフトウェアまたはマニュアルに関する問題の報告方法などの各種情報を提供します。

『リリースノート』のアップデートおよびコアマニュアルの修正点へのリンクは、Sun ONE マニュアル Web サイトの Identity Server のページを参照してください。更新されたマニュアルには改訂日を記してあります。

Identity Server ポリシーエージェントのマニュアルセット

Identity Server のポリシーエージェントは、入手可能なスケジュールがサーバー製品とは異なります。このため、ポリシーエージェントのマニュアルセットは、Identity Server マニュアルのコアセットとは別個に入手できます。マニュアルセットに含まれるタイトルは、次のとおりです。

- 『Web Policy Agents Guide』: さまざまな Web およびプロキシサーバーで Identity Server ポリシーエージェントをインストールおよび設定する方法を説明します。また、トラブルシューティングや、各エージェントに固有の情報についても説明します。
- 『J2EE Policy Agents Guide』: さまざまなホスト J2EE アプリケーションを保護可能な Identity Server ポリシーエージェントのインストールおよび設定方法について説明します。また、トラブルシューティングや、各エージェントに固有の情報についても説明します。
- 『リリースノート』は、エージェントセットのリリース後、オンラインでのみご利用になれます。『リリースノート』は、各エージェントタイプのリリースにつき 1 ファイル提供されます。このファイルでは、このリリースの最新情報、既知の問題、制限事項、インストールに関する注意事項、ソフトウェアまたはマニュアルに関する問題の報告方法などの各種情報が提供されます。

『リリースノート』のアップデートおよびポリシーエージェントマニュアルの修正点については、Sun ONE マニュアル Web サイトのポリシーエージェントのページを参照してください。更新されたマニュアルには改訂日を記してあります。

マニュアルへのフィードバック

米国 Sun Microsystems, Inc. および Identity Server のマニュアル執筆陣は、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。ご意見は `docfeedback@sun.com` まで電子メールをお送りください。

表記上の規則

Identity Server のマニュアルでは、特定の表記および用語を使用します。これらの規則について、以降の節で説明します。

表記上の規則

このマニュアルでは、次の表記規則を適用します。

- *イタリック体*は、新出用語、強調語句、および文字通りの意味を示すときに使用します。
- モノスペース（等倍）フォントは、サンプルコードとコードのリスト、API およびプログラミング言語の要素 (関数名、クラス名など)、ファイル名、パス名、ディレクトリ名、HTML タグ、および画面に入力する文字を示すときに使用します。
- *イタリック体のセリフフォント*は、コードおよびその一部で可変部分を表すときに使用します。たとえば、次のコマンドの場合、*filename* の位置には `gunzip` コマンドの引数が入ります。

  `gunzip -d` *filename*`.tar.gz`

用語

Identity Server マニュアルセットで共通に使用する用語を次に示します。

- Identity Server は、Identity Server および Identity Server ソフトウェアのインストール済みのインスタンスを示します。
- ポリシーおよび管理サービスは、Web サーバーなど専用の配備コンテナで実行される、インストール済みの Identity Server コンポーネントおよびソフトウェアの集合的なセットを示します。
- ディレクトリサーバーは、Sun ONE Directory Server のインストール済みのインスタンスを示します。
- アプリケーションサーバーは、Sun ONE Application Server のインストール済みのインスタンスを示します。
- Web サーバーは、Sun ONE Web Server のインストール済みのインスタンスを示します。
- *IdentityServer_base* という変数は、Identity Server のインストール先であるホームディレクトリを示します。

- *DirectoryServer_base* という変数は、Sun ONE Directory Server のインストール先であるホームディレクトリを示します。
- *ApplicationServer_base* という変数は、Sun ONE Application Server のインストール先であるホームディレクトリを示します。
- *WebServer_base* という変数は、Sun ONE Web Server のインストール先であるホームディレクトリを示します。
- Identity Server を実行する Web コンテナは、ポリシーおよび管理サービスがインストールされた専用の J2EE コンテナ (Web サーバーやアプリケーションサーバーなど) を示します。

関連情報

Identity Server のマニュアルのほかにも、参考になるマニュアルがあります。表 0-1 に、これらのマニュアルの入手先と関連情報を示します。

表 0-1　関連する Sun ONE リソースの入手先

情報またはリソース	インターネット上の位置
Directory Server のマニュアルセット	http://docs.sun.com/coll/S1_DirectoryServer_52
Web Server のマニュアルセット	http://docs.sun.com/coll/S1_websvr61_en
Web Proxy Server のマニュアルセット	http://docs.sun.com/prod/s1.webproxys#hic
Sun ONE Download Center	http://wwws.sun.com/software/download/
Sun ONE テクニカルサポート	http://www.sun.com/service/sunone/software/index.html
Sun ONE プロフェッショナルサービス情報	http://www.sun.com/service/sunps/sunone/index.html
Sun エンタープライズサービスによる Solaris のパッチとサポート	http://sunsolve.sun.com/
開発者用情報	http://developers.sun.com/prodtech/index.html

Sun は、本書に記載されたサードパーティの 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、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。

第 1 章

はじめに

Sun™ ONE Identity Server は、Web ベースのサービスや Web ベース以外のアプリケーションを利用する顧客、従業員、およびパートナーのデジタルアイデンティティに対し、組織による管理プロセスのインフラストラクチャを提供します。これらのリソースは、内部および外部の広範なコンピューティングネットワーク上で分散される可能性があるため、アクセスを管理する属性、ポリシー、および制御が定義されます。この章では、Identity Server を導入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。

アイデンティティ管理とは

現在、各企業では、日々の作業の管理を容易にするため、高度な情報技術のインフラストラクチャを維持しています。このインフラストラクチャの不可欠な要素には、次のようなものがあります。

- 異なるオペレーティングシステムを実行するネットワークサーバー
- 情報データストア
- 人材、給与、および契約管理システム
- アカウンティング、サプライチェーン管理、およびリソース計画に対応した事業分野別アプリケーション
- 販売、営業、顧客サービス、現場サポート、その他のクライアント関連機能を統合した顧客関係管理 (CRM) システム
- ショッピングおよびセキュリティ保護されたクレジットカード取引に対応した電子商取引用アプリケーション

これらの各要素は個別に配備されるため、システムごとにユーザーを追跡して、実行可能および実行不可能な操作を制御します。通常、この追跡処理には、個人のプロファイル、認証情報、およびアクセス制御などのアイデンティティデータの管理が含まれます。アイデンティティ管理を利用することで、この複製データ (矛盾していることが多い) の管理が簡単になります。

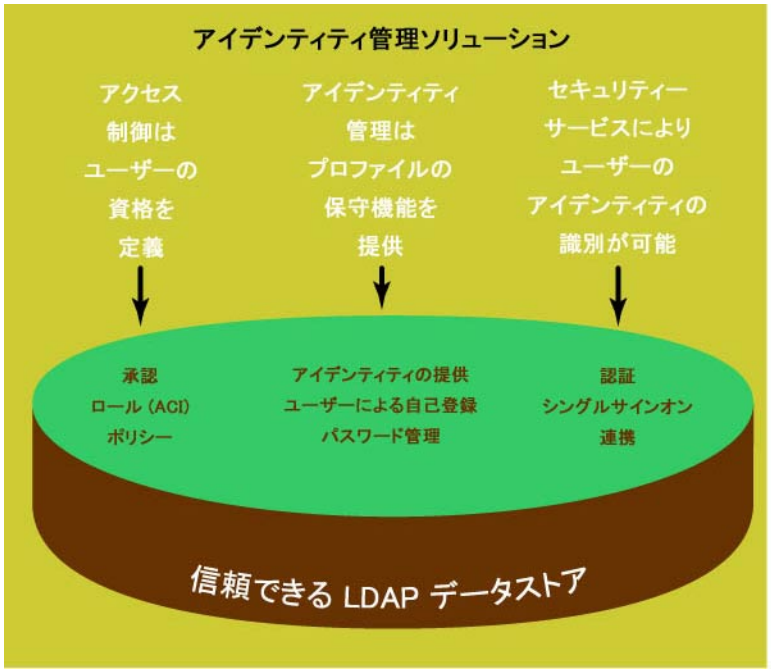
アイデンティティ管理インフラストラクチャ

アイデンティティ管理システムの目的は、企業全体を対象として、単一のインフラストラクチャによるすべてのユーザーの管理を実現することです。単一のアイデンティティ管理システムを使用することで、繰り返しを避けて一貫性を維持することができるため、ユーザープロファイルの管理が簡単になります。さらに単一のシステムにより、アイデンティティ管理プロセスの合理化、簡略化、および自動化が可能になります。以下に、アイデンティティ管理システムの構成要素を示します。

- アイデンティティ管理の提供するインフラストラクチャは、アイデンティティおよび対応する属性、証明情報、資格の作成および管理をサポートします。この機能には、以下が含まれます。
  - アイデンティティの提供 (プロファイルの作成、変更、および削除)
  - ユーザーの自己登録
  - パスワード管理およびパスワードリセット
- セキュリティーサービスを使用することで、ユーザーのアイデンティティがネットワーク上で一貫したものになります。この機能には、以下が含まれます。
  - 認証 (当人のアイデンティティを証明する)
  - シングルサインオン機能 (一度の認証で多数のリソースへのアクセスを可能にする)
- アクセス制御は、ユーザーの資格 (さまざまなリソースをどのようにまたいつ使用できるか) を定義します。多くの場合、これらの制御によりユーザーの組織内でのロールが指定されます。集中化されたポリシーおよびロールを作成して適用することにより、組織は顧客、従業員、およびパートナーの責任を委譲できます。
  - 認証 (アイデンティティが要求されたアクションを実行できるかどうかを決定する)
  - ポリシー (保護されたリソースへのアクセス承認を定義する)
  - アクセス制御命令 (アイデンティティデータへのアクセス承認を定義する)
- 連携サービスは、独立した情報システム間の認証および承認を提供します。
- 企業の LDAP ディレクトリは、すべてのアイデンティティ情報およびアイデンティティ管理システム自体の設定情報の信頼できるデータストアとして動作します。

これらの構成要素の詳細については、図 1-1 を参照してください。

図 1-1　アイデンティティ管理ソリューションの構成要素



アイデンティティプロファイルのライフサイクル

一般的なアイデンティティプロファイルのライフサイクルについて考慮すると、アイデンティティ管理を行う際の課題を概観できます。組織内のアイデンティティには、次の 3 つの段階が存在します。

1. プロファイルの作成

   ユーザーが組織に参加すると、アイデンティティプロファイルが作成されます。プロファイルには、個人情報、雇用データ、パスワード情報、および定義済みのアクセス権が含まれます。

2. プロファイルの管理

   セットアップが終了したら、プロファイルを管理する必要があります。これには、プロファイルデータの変更、リソースアクセスポリシーの管理、アクセス制御命令の更新が含まれます。

3. プロファイルの無効化

   ユーザーがログアウトすると、プロファイルにそのことを示すフラグを付け、システムリソースへのアクセスを無効にする必要があります。

Sun ONE Identity Server

Sun ONE Identity Server は、統合化された標準ベースのミドルウェアのパッケージです。Identity Server の提供する Web サービスにより、アクセス管理、連携、およびアイデンティティ管理がサポートされます。これにより、Identity Server が総合的なアイデンティティ管理ソリューションになり、ユーザープロファイルの作成および管理機能が、セキュリティプロセス、アクセス管理ツール、およびデータ格納用ディレクトリに統合されます。これらの機能を使用すると、リソースおよび情報が保護された総合的なシステムを組織に配備すると共に、Web ベースのアプリケーションをセキュリティ保護された方法で配信できます。

アクセス管理

アクセス管理は、独自およびアプリケーション固有の認証および承認方法に代わる、共通の認証および承認インフラストラクチャを提供します。組織は、集中化された管理位置から、複数のサービスに対してポリシーベースのアクセス制御を行えます。Identity Server 内の複数のアクセス管理サービスが総体として、次の機能を提供します。

シングルサインオン (SSO)

シングルサインオン機能は、一度のユーザー認証で、複数のリソースへのアクセスを可能にします。Identity Server は、Web ベースのアプリケーションで SSO をサポートします。また、Web ベース以外のアプリケーションに SSO 機能を統合するためのプログラミング用インタフェースも提供します。

プラグイン可能な認証

JAAS (Java™ Authentication and Authorization Services) ベースの認証フレームワークは、LDAP、RADIUS (Remote Authentication Dial-In User Service)、X.509 デジタル証明書、SecureID、SafeWord、UNIX (PAM ベース)、Windows NT、HTTP 基本認証、匿名、および自己登録を含む、さまざまなプラグイン可能認証モジュールをサポートします。このフレームワークを利用することで、提供される認証サービスプロバイダインタフェース (SPI) を使用して、カスタム認証モジュールも開発できます。認証を設定することで、同一システム内の多様な組織、ロール、またはユーザーのニーズを同時にサポートしたり、複数の要因が連鎖した設定をサポートしたりできます。マルチレベルの認証を使用すると、データやサービスの特性に基づいて求められるさまざまな認証レベルにリソースを割り当てることができます。認証サービスには、Web ベース、Java、C、および XML インタフェースからアクセスできます。

ポリシーの評価

ポリシーサービスを使用すると、さまざまなロールやグループ化メカニズムにマッピング可能なアクセス管理ルールを集中的に設定および評価できます。IP アドレス、日付と時刻、カスタム条件などのポリシー制約は、実行時にポリシーに適用して評価できます。

連携管理

インターネットは、ビジネス、コミュニティ、および個人のやり取りのための主要な手段に急速になりつつあるため、ユーザーがさまざまなアカウントアイデンティティを集約させるシステムを構築し、単一のネットワークアイデンティティを使用できるようにする必要が生じてきました。このシステムをアイデンティティ連携といいます。アイデンティティ連携を使用すると、複数のインターネットサービスプロバイダのローカルアイデンティティを関連付けたり、連結したり、またはバインドしたりすることができます。ネットワークアイデンティティを使用すると、あるサービスプロバイダのサイトにログインすれば、アイデンティティを再認証または再確立しなくても提携先のサイトに移動できます。Identity Server は、Liberty 1.1 および SAML 1.0 の完全な実装を提供します。これには、完全なプロファイル実装および、カスタム統合化に対応した SDK サポートが含まれます。Liberty アイデンティティおよびサービスプロバイダの複数ホスティングが提供されます。

Liberty Alliance Project

連携管理は、認証ドメインおよびプロバイダに関するメタデータを表示、管理、および設定する方法を備えています。アイデンティティ連携を実現するために策定された Liberty Alliance Project は、20 億を超える顧客と広範な業種から参加した 138 のメンバー企業により構成されています。その使命は、消費者とビジネスユーザーのシングルサインオンを可能にする、連携されたネットワークアイデンティティソリューションを提供およびサポートすることにより、断片化するアイデンティティの問題を解決することです。このため、Liberty ベースのアプリケーションは、ユーザーアカウントを別の Liberty 対応アプリケーションのユーザーアカウントと連携 (リンク) させて、2 つのアプリケーション間のシングルサインオンを実現できます。Identity Server は、Liberty Alliance Project の仕様を実装しています。

SAML (Security Assertion Markup Language)

SAML は、ビジネス間インフラストラクチャの主要な成功要因です。アプリケーションは、Identity Server に統合された SAML API を使用して、セキュリティ情報を交換し、信頼される他のアプリケーションとのビジネス取引を実行できます。エンドユーザーは、Web ブラウザを使用して Identity Server への認証を行い、サイト内転送 URL 経由で信頼できるサイトの外部 URL にシームレスにアクセスできます。開発者は、アプリケーション内で SAML API を使用し、信頼できる外部アプリケーション間で認証、承認、および属性情報を交換できます。

アイデンティティ管理

アイデンティティ管理は、ユーザー提供、ポリシー設定、サービス管理を可能にする拡張可能なブラウザベースのインタフェースを提供します。Identity Server コンソールを使用すると、単一のインタフェースからアイデンティティ管理を集中的に実行できます。また、ローカルグループマネージャ、外部パートナー、さらにエンドユーザーにまでも管理を委任できます。

ユーザープロファイル管理

簡潔にまとめると、ユーザープロファイル管理では、アイデンティティプロファイルの作成と削除を行います。ただし、これには、プロファイルの管理を事情に通じた管理者に委任することや、セルフサービスコンポーネント (ユーザーはこれを使ってサービスやアプリケーションを利用できる) を提供すること、新規ユーザーアカウントを作成してそのプロファイルを管理すること (パスワードの変更、自宅住所の更新など) が含まれます。

ポリシー設定

ポリシー設定は、アクセス承認時に評価されるルールの定義です。委任を実行すると、最上位レベルの管理者がポリシーの設定および管理を組織のあらゆるレベルの個人に分散できます。これにより、ポリシーの設定および管理を、リソースに対する権限を保持するユーザーに確実に託すことができます。

サービス管理

サービス管理を使用すると、Web サービスおよび対応する属性を設定、登録、および管理できます。Identity Server では、独自の管理に使用するサービス用インタフェースも提供されます。

監査

管理者は、高度な設定が可能なログ機能を使用して、ユーザーのアクティビティ、トラフィックパターン、認証および承認違反に関する詳細なレポートを作成できます。これらの機能を使用して、リソースアクセスに対するセキュリティレベルの監査も実行できます。MAC (Message Authentication Code) およびデジタル署名ベースのログセキュリティは、ログまたは監査記録に対するいかなる改ざんも検出します。デバッグ機能も有効にできます。

ポリシーエージェント

Identity Server 内のアクセス制御は、ポリシーエージェントを使用して行われます。ポリシーエージェントは、指定された Web サーバー、アプリケーションサーバー、およびプロキシサーバーを不正な侵入から保護します。Identity Server では、Web およびプロキシサーバーを URL レベルで保護するポリシーエージェント、および Java テクノロジに対応したアプリケーションサーバーへのアクセスを行う Java™ 2 Platform, Enterprise Edition (J2EE) ポリシーエージェントがサポートされます。

Identity Server コンソール

Identity Server コンソールは、Identity Server の配備全体で設定された、アイデンティティ、サービス、およびポリシーの作成、管理、および監視用のブラウザベースインフラストラクチャです。これは、機能的な Web アプリケーションを作成する開発者を支援する J2EE フレームワークである、Sun ONE Application Framework を使用して構築されます。HTML ページの外観を定義するために、XML ファイル、JSP (JavaServer Pages™)、CSS (Cascading Style Sheets) が使用されます。

プログラミング用インタフェース

非グラフィカルインタフェースには、Identity Server の拡張およびカスタマイズに使用される API、SPI、およびコマンド行ツールが含まれます。このインタフェースを使用すると、その他のアプリケーションから各機能にアクセスできるようになります。API および SPI の詳細は、53 ページの「Identity Server SDK」および『Sun ONE Identity Server Customization And API Guide』を参照してください。コマンド行ツールの詳細は、『Sun ONE Identity Server 管理ガイド』を参照してください。

Sun ONE Directory Server

Sun ONE Directory Server は、アイデンティティ、ポリシー、設定およびサービス情報を格納する統合化されたデータリポジトリとして機能します。

配備 Identity Server

Identity Server は、オープンな標準に準拠したプラットフォームに合わせて設計されています。このプラットフォームは、認証、承認、シングルサインオン、ポリシー、アイデンティティ、および管理機能を既存のインフラストラクチャに統合する際に使用できます。その機能は、Web コンテナの Java 仮想マシン (JVM) 内部で動作し、API およびさまざまなサーバーフレームワークにアクセス可能な Java サーブレット、JavaBeans™、および JSP の集合体として提供されます。Identity Server を企業のインフラストラクチャに統合することで、以下のタスクを達成できます。

- 柔軟性に欠ける独自ユーティリティを排除する
- 複数の Web およびアプリケーションサービス間のセキュリティ保護された認証、アクセス制御、および監査を実現する
- アクセス制御命令 (ACI) のレベルを設定して、集中化されたアイデンティティ管理を、委任機能と共に実装する。これにより、以下の処理が組織内で可能になる
  - 内部アイデンティティ管理を IT 担当以外の従業員に委任する
  - 外部アイデンティティ管理をパートナーまたはサプライヤーの従業員に委任する
- 集中化されたポリシーフレームワークを設定して、実行中のアプリケーションや新たに配備されたサービスに認証機能を提供する
- 連携管理を Liberty Alliance 仕様 v.1.1 および SAML のサポートに統合する

Identity Server の統合

Identity Server は総合的なアイデンティティ管理システムですが、大抵の組織では何らかのアイデンティティ管理システムを既に実装していることでしょう。たとえば、ディレクトリサーバーや Web コンテナを既に配備している場合が考えられます。Identity Server と他のシステムとの相互運用を可能にするには、保護されたサーバーにポリシーエージェントをダウンロードおよびインストールする必要があります。Identity Server の開発およびリリースと並行して、新規エージェントの開発およびリリースが行われています。Identity Server 6.1 のリリース時点では、Sun Microsystems

の Web, Portal & Directory Servers Download Center からダウンロードできるのは、つぎのオペレーティングシステムに対応するエージェントです。ダウンロードセンターのページには、リリース済みのポリシーエージェントに関するより詳細で最新のリストが掲載されています。

警告 Identity Server のポリシーエージェントのリリーススケジュールは、製品自体のリリースとは別個に計画されています。このマニュアルの発行以降、このリストにポリシーエージェントが追加されている可能性があります。最新のリストについては、Sun Microsystems の Web, Portal & Directory Servers Download Center を参照してください。

Solaris オペレーティングシステム

- Sun ONE Web Server 6.0
- Sun ONE Web Server 4.1
- Sun ONE Web Proxy 3.6
- Apache 1.3.27
- Lotus Domino 6.0
- Sun ONE Application Server 7.0 (J2EE)
- BEA WebLogic 6.1 SP2 (J2EE)
- BEA WebLogic 7.0 SP2 (J2EE)
- IBM WebSphere 5.0 (J2EE)
- PeopleSoft 8.3, 8.4 and 8.8 (J2EE)

Windows 2000 Server

- Sun ONE Web Server 6.0
- Lotus Domino 6.0
- Microsoft IIS 5.0
- Sun ONE Application Server 7.0 (J2EE)
- BEA WebLogic 6.1 SP2 (J2EE)

Linux

- Apache 1.3.27
- Sun ONE Application Server 7.0 (J2EE)

HPUX 11

- Sun ONE Web Server 6.0
- PeopleSoft 8.3、8.4、および 8.8 (J2EE)
- BEA WebLogic 6.1 SP2 (J2EE)

配備ロードマップ

Identity Server の統合を入念に計画することは、成功を収めるために欠かすことができません。これには、ハードウェア、配備中のアプリケーション、アイデンティティ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。Identity Server の配備作業は、次の段階に分けることができます。

1. ビジネスの目的を識別

   例 :

   - 業務効率を改善
   - データのセキュリティを確保
   - 次の方法で生産性を保証
     - 組織内の範囲および関係を理解
     - ビジネスの目的のサポートに必要な行動変化を分析
2. 次の方法で高度なテクノロジ分析を開発し、ビジネスの目的に適用
   - テクノロジサービスを列挙
   - ビジネスの目的の達成に必要なツールを列挙
3. 次に例示する、テクノロジサービスの具体策を定義

   例 :

   - 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
   - アイデンティティ管理を使用して、パスワード同期およびアイデンティティ管理を実行
   - ロールの戦略を開発して、企業のセキュリティ保護を実現
4. 以下の要素に基づいて、イニシアチブに優先順位を設定
   - 統計的正確性
   - 予測可能性
   - 範囲
   - 費用

- 影響
- 複雑性
- 動作
- インフラストラクチャ
- 利点
- サポート
- 依存関係

配備ガイド の章

配備ロードマップで説明した各段階については、このマニュアルの以下の章で詳しく説明されています。

- 第 2 章「配備の計画」では、組織のアイデンティティ管理ソリューションの現状を評価し、将来の必要を見極める方法 (目標および課題を含む) を定義します。
- 第 3 章「Identity Server のアーキテクチャ」では、Identity Server 製品の全コンポーネントのアーキテクチャに関する高度な概要を示します。
- 第 4 章「配備前の考慮事項」では、ハードウェア、データソース、および専門知識を含む、特定の要件を分析する方法を示します。
- 第 5 章「配備シナリオ」では、トポロジを計画し、アプリケーションを配備する簡単なシナリオについて説明します。

補足的な情報を提供するため、*配備ガイド* に付録が追加されました。以下にその内容を示します。

- 付録 A「インストールされる製品のレイアウト」では、Identity Server のインストール時に作成されるディレクトリおよびファイルについて説明します。
- 付録 B「ユーザーセッションのライフサイクル」では、複数の HTTP (HyperText Transfer Protocol) 要求間で行われる、ユーザーと Web アプリケーションとのやり取りの追跡に使用するセッションオブジェクトについて説明します。
- 付録 C「Active Directory に対する認証」では、Microsoft Active Directory でユーザーを認証する方法を説明します。
- 付録 E「RADIUS サーバーに対する認証」では、RADIUS (Remote Authentication Dial-In User Service) サーバーでユーザーを認証する方法について説明します。
- 付録 F「chroot 環境のインストール」では、chroot 環境に Identity Server をインストールして、悪意のあるプログラムが実際のルートファイルシステムにアクセスすることを防ぐ方法について説明します。

Identity Server の関連するマニュアル

Identity Server に関する追加情報については、以下のマニュアルを参照してください。

- インストール ― インストール関連の情報については、『Sun Java Enterprise System 2003Q4 インストールガイド』を参照してください。
- 移行 ― 既存データの移行および Identity Server の以前のバージョンの更新については、『Sun ONE Identity Server Migration Guide』を参照してください。
- 管理 ― コンソールの使用方法および Identity Server の配備を管理する方法については、『Sun ONE Identity Server 管理ガイド』を参照してください。
- カスタマイズ ― アプリケーションのカスタマイズ方法については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

第 2 章

配備の計画

Sun™ ONE Identity Server は、複雑な分散型アイデンティティ管理システムであり、適切な配備によって、企業の組織境界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
- 目標の設定
- 情報の収集
- アプリケーションの評価
- データの分類
- スケジュールの作成

リソースの定義

アイデンティティ管理ソリューションには組織全体の多種多様なシステムが関係するため、Identity Server を適正に配備するには広範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。

人材

組織内のさまざまな取引関係および政治的枠組みに精通しておくことは重要です。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、Identity Server の配備は、1 人のプロジェクトマネージャおよび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。彼らは、チームリーダーおよび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。チームは、Sun Professional Services

リソースおよび必要に応じて段階的に投入および廃止される LOB アプリケーション管理者で構成される仮想チームメンバーにより、しばしば増強されます。これは、現実のニーズを満たすかどうかは別として、ほぼ一般的な配備チームモデルを表したものです。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、典型的な Identity Server 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

通常、アイデンティティ管理を成功させるには、組織および行政上の境界を超えて配備を行う必要があります。これには、企業の決定権を持つ人物の承認および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、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、現在のアイデンティティ管理コストに関する統計を収集します。具体的な統計を入手することで、配備チームが経営陣のサポートを得るのに役立つ明確な ROI を定義できます。関連する他の問題には、以下が含まれます。

- 誰がアイデンティティ管理の配備からメリットを享受するか
- アイデンティティ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ゆくか

注 アイデンティティ管理の概念および Identity Server を配備することの価値を納得させることがしばしば必要になります。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の利点を経営陣に積極的に語り、統合化に関する需要を喚起したり、インフラストラクチャ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リーダーとして、1 人の人物を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。これは、テクニカルリーダー、プロジェクトマネージャ、および執行責任者などの間で論理的に分散されたロールとすることも可能です。このロールがどのように定義されるとしても、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

この担当者は、スケジュールの調整を行います。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。この担当者は、優れたコミュニケーション能力と政治的手腕を持っていなければなりません。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があります。

注 事業分野別の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

この人物は、Identity Server 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、および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仕方で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。Identity Server のアーキテクチャ、サービス、エージェント、および API に関する詳細な知識が求められます。

LOB アプリケーション管理者

これは、Identity Server ポリシーエージェントまたはポリシー実施ポイントをアプリケーションに統合する責務を担う人物です。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、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、Identity Server 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、Identity Server 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。通常は、最後に 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。このため、LOB アプリケーションに関する詳細な知識および制御能力を備えた、技術面の専門家でなければなりません。

システム管理者

Identity Server 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、Identity Server 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。

Identity Server 管理者

この人物は、Identity Server の配備および保守を担当します。通常、専任のユーザーが割り当てられ、共通サービスの可用性の保証や、必要なインフラストラクチャの拡張、およびポリシーやロールの設定を行います。Identity Server 管理者は、ガイドラインを策定して統合化サポートを支援し、LOB アプリケーション管理者に技術サポートを提供します。Java、XML、LDAP、HTTP、および Web アプリケーションアーキテクチャの理解が必須です。

Directory Server 管理者

Identity Server 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、定義されている LDAP スキーマおよびアイデンティティデータへの追加や変更を受け入れたり統合したりすると共に、ディレクトリサービスの可用性管理を担当します。アイデンティティ管理インフラストラクチャをサポートするために、ディレクトリサービスの変更が必要になることがあります。

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、データセンターや、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、これらのさまざまな管理者間で明快なコミュニケーションを行う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することや、特定のネットワークを確立することが重要な場合があります。これらの担当者にプロジェクトの里程標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 および他の独立系ソフトウェアベンダー (ISV) は、Identity Server 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。

注	独立系ソフトウェアベンダーは、コンピュータの 1 つ以上のハードウェアタイプ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ソフトウェア製品を制作および販売します。プラットフォームを作成する企業 (IBM、Hewlett-Packard、Apple、Microsoft など) は、ISV の支援およびサポートを提供します。Sun Microsystems は、プラットフォームおよびソフトウェア製品を作成します。

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 プロフェッショナルサービスおよび他の ISV と契約を結んで、プロジェクトの立ち上げの支援や以前の Identity Server 配備で得た知識を提供してもらってください。アカウントチーム (Identity Server のエンジニアと配備チームとの仲介役として働くことができる) と率直な討議を行うと共に、プロフェッショナルサービスを利用すると、投資を有効に活用し、配備を成功させる助けが得られます。

提携先のサードパーティ

Identity Server 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、独自の内部配備と共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公平な話し合いの場を設定する助けになります。

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

目標を設定することにより、Identity Server の配備完了後のあるべき状態を定義できます。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、アイデンティティ管理ソリューションでは、セキュリティを拡張し、インフラストラクチャの管理機能を向上させると共に、コストを削減します。より具体的にいえば、Identity Server 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 (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を実装する
- アイデンティティ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
- アイデンティティプロファイルの迅速な終了により、セキュリティを改善する
- セキュリティ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タイムフレームを圧縮する
- 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およびデータストアに関する情報を収集できます。さらに、これらの部門インタビューは、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、日常業務で使用す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、雇用および解雇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、ビジネスプロセス、IT インフラストラクチャ、および仮想データに関する情報収集を含めることができます。

ビジネスプロセス

ビジネスプロセスとは、組織内の異なるグループがそれぞれのジョブの実行を定義する手順です。プロセスには、次の手順を含められます。

- 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
- 従業員の解雇

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。考慮すべき内容には、以下が含まれます。

- 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

IT インフラストラクチャには、Identity Server 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。

- Identity Server 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、または重要性のあまり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、Identity Server 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。

- Identity Server を利用するシステム

  これには、アプリケーションの配備先のハードウェアおよびそのオペレーティングシステムが含まれます。Identity Server 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 ONE Directory Server (または既存のデータストア)、および Sun ONE Identity Server アプリケーション自体が最低限含まれます。また、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。これには、セキュリティ上の理由で Identity Server のポリシーエージェントをインストール可能な企業のリソースが含まれます。

- 各部門が利用する Identity Server のサービス

  これには、Identity Server 内部に統合されたデフォルトおよびカスタムのサービスが含まれます。ロールおよびポリシーの戦略は、部門ごとに割り当ておよび定義を行う必要があります。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内に非互換性が存在するかどうか
- 現在のシステムに速度低下が見られるか。ダウンタイムはどの程度か
- アプリケーションは十分にセキュリティ保護されているか
- 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、42 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

仮想データは、Identity Server にアクセスするプロファイル用の、あらゆる状況で利用可能なデータです。このデータは、Identity Server からアクセス可能な設定でもあり、Identity Server によりセキュリティ保護されるデータでもあります。これには、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、および他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。

- Identity Server が保護する対象

  Identity Server は、あらゆる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保護します。管理者は、Identity Server 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。

- Identity Server 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、外部の販売情報を閲覧するためにサインインする顧客よりも、明らかにより大規模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、機密情報および秘密データが含まれます。具体的には、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、および連携顧客プロファイルが含まれる場合があります。このデータは、Directory Server に格納される場合も格納されない場合もあります。

- 信頼すべきデータの入手元

  多くの場合、異なるデータタイプを定義する複数のスキーマが存在します。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る、または既存のデータが変更される頻度はどれほどか

詳細は、44 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

アイデンティティ管理サービスは、通常、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使用する、集中化された 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配備されています。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行う場合は、次の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、アイデンティティ管理インフラストラクチャに受け入れられるアプリケーション
- 現在、配備または再ファクタリングの初期プロセスにあるか、Identity Server 配備と一致する時刻表を論理的に保持するアプリケーション

スプレッドシートを作成して、最も将来性の高いアプリケーションから取得し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、アプリケーションの値を統合化の複雑性と比較できます。これにより、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、セキュリティ目的で Identity Server 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は、メインフレームに由来する「グリーンスクリーン」アプリケーション (テキストベースのインタフェースを持つアプリケーション) です。この場合、メインフレームアプリケーションの再ファクタリング / アーキテクチャの待機中に、他のアプリケーションを統合するというメリットを享受できます。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。

注　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の候補として適切かどうかを評価できます。収集されたプラットフォーム情報には、以下が含まれます。

- 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される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

注 LOB アプリケーションも、サードパーティ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) を稼働させることができます。これらのアプリケーションは、Identity Server エージェントがサポートするプラットフォーム上に常に配備されるわけではありません。エージェントの可用性に基づいて、これらのアプリケーションの配備基準を評価し、組み込みのスケジュール設定を行ってください。カスタム作業が必要な場合には、通常、Java ベースのアプリケーションの方が統合がより容易です。

セキュリティモデル

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ため、配備の有力な候補になります。セキュリティ情報には、以下を含めることができます。

- 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
- 認証を外部で行うことは可能か。それは意味のあることか
- 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か

セッションのライフサイクル

アイデンティティのセッションライフサイクルは、認証アプリケーションの評価を行う上で重要なトピックです。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理解しているかを確認してください。アプリケーションの統合化を行う際に参照できるよう、このプロセスを正確にドキュメント化してください。このトピックの詳細については、付録 B 「ユーザーセッションのライフサイクル」を参照してください。

カスタマイズおよびブランド設定

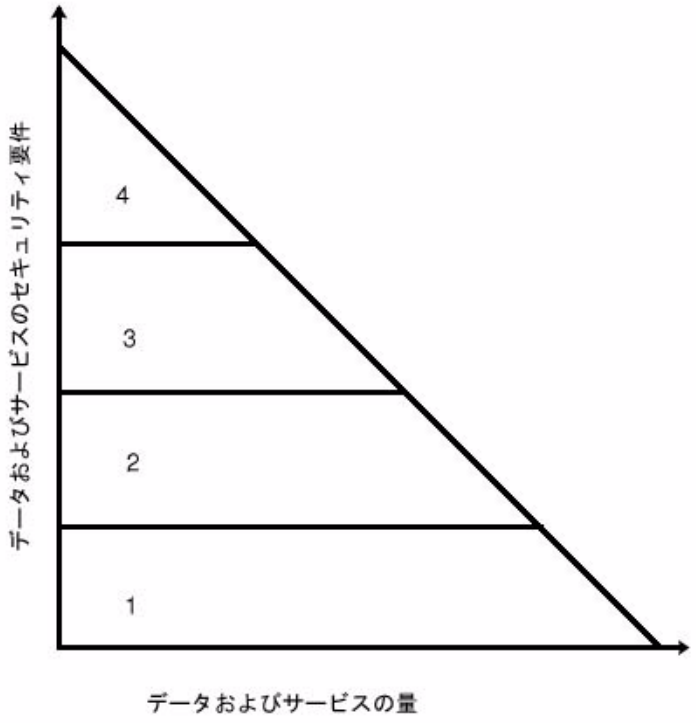
アプリケーションのブランド設定やルックアンドフィールに関する特定の要件を考慮する必要があります。多くの場合、アプリケーション単体のルックアンドフィールを重視するか、ユーザーにとって一貫した使い勝手を重視するかは重要な問題です。カスタマイズおよびブランド設定要件を満たす場合、そのための時間もスケジュールに組み込む必要があります。このため、アプリケーションの評価にカスタマイズおよびブランド設定要件が含まれているかどうかを確認してください。

データの分類

アプリケーションを分析し、適切なレベルに分類したなら、アプリケーションにより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。アプリケーションの評価で収集された情報は、プロセスの前の部分で使用されます。収集した情報をさまざまなセキュリティ層に編成するのは、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。

図 2-1 は、一般的な組織内のデータを表現したものです。データまたはサービスは、4 つのセキュリティ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レベルを表します。層 1 は、セキュリティ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、セキュリティが最大であることを示します。これは財務や HR データなどに適用されます。組織により、これよりも層が多いか少ない場合がありますが、この図から、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要件も低くなることを理解できます。関連するリスクが大きくなるにつれ、セキュリティ要件も大きくなります。(実際のところ、高度なセキュリティ要件を備えたデータはごくわずかであり、大量のデータはセキュリティ要件を必要としません。)

図 2-1　データおよびサービスのセキュリティ要件



認証および承認機能の割り当てが可能な、データおよびサービスタイプの機能グループを構築することが目標であ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。適切であれば、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共に、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レベルに応じて分類できたなら、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層に関連付けます。たとえば、図 2-1 で分類したデータの場合、次のようにできます。

- 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設定するか、ネットワーク上に配置しないようにします。

認証要件とデータ / サービス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、HR や財務などの機能セキュリティ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、万能であるとは言えませんが、セキュリティ要件を理解し、論理的に管理可能なグループにマッピングする助けになります。

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、アプリケーション間で使用する共通のグループ / ロールを見つけることです。これらは、組織内の機能ロールにマッピングされます。これは理想的な方法です。また、これらのロール / グループのソース (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。これは Sun ONE Directory Server に存在するのが理想です。存在しない場合には、カスタムプラグインが必要になります。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アクセスします。作業の完了時点で、以下のものを入手できているはずです。

- 既存のロールおよびグループの明快なマップ
- 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、コスト / 複雑性を低減するために作成する新規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できます。次の節では、一般的な配備スケジュールの手順を説明します。

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、大抵、期待される結果を含む設定済みのテストケースセットである、「テストベッド」データベースが存在します。このテストベッドに、コンセプト証明を適用できます。すべてが順調に進めば、新たに得られた結果がドキュメント化された結果と同じになります。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対する答えを提出し、すべてのニーズを効果的に満たすことができること、およびリスクを最小限に抑えられることを証明することです。これは通常高速に行われるため、限定さ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。コンセプト証明および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、大抵、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予算作成用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、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、重要なハブアプリケーション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、成功が迅速に立証および認識されるように、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分析 (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、非人工的な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。これは、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、テストベッド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後で、徐々に製品環境に追加されてゆきます。

注　スケジュールはプロジェクトの複雑さに応じて変化するため、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。

第 3 章

Identity Server のアーキテクチャ

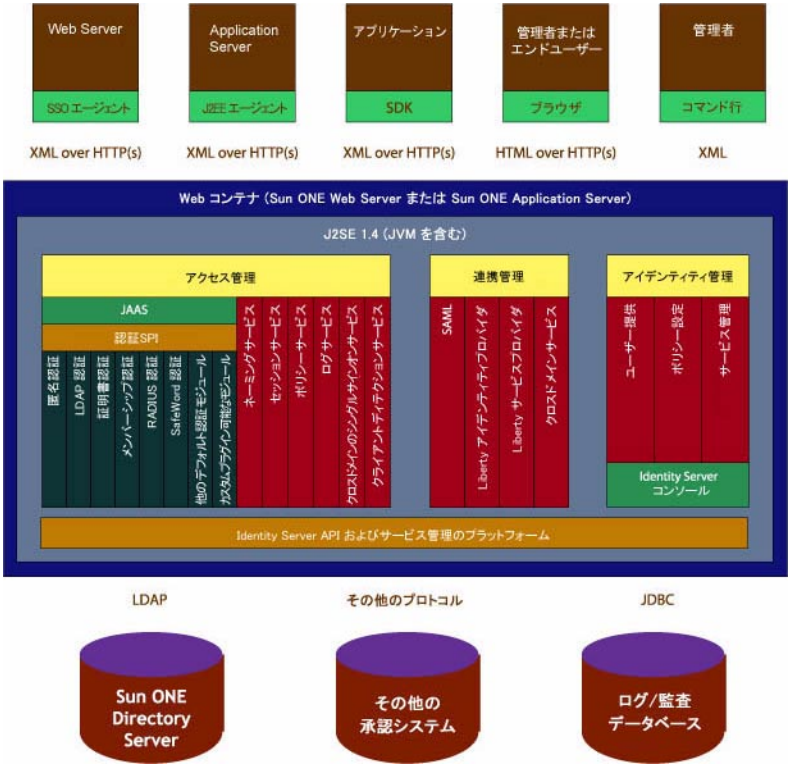
Sun™ ONE Identity Server は、アイデンティティオブジェクト、ポリシー、およびサービスの管理用インタフェースを提供します。この章では、これらのインタフェースについて、製品のコアアーキテクチャとの関連を示しながら説明します。また、必要に応じて、サービスのアーキテクチャ上の詳細についても説明します。次の節で構成されています。

概要

Identity Server は、Java™ 2 Platform, Enterprise Edition (J2EE) プラットフォーム上に構築されており、サービスを提供するサーブレット、Web サービスエンドポイント、および Web アプリケーションを操作するためのインフラストラクチャとして Web コンテナを使用します。提供される各サービスは、Web サービスとして実装されます。Web サービスは、現在注目を集めている標準技術セットを使用して構築され、ネットワーク上で集中的に提供されます。これらのサービスを企業間で実装するため、Identity Server はアイデンティティ関連のオブジェクト、ポリシー、およびサービスの管理用インタフェースを提供します。主なインタフェースには、多様な Web サーバーおよびアプリケーションサーバーにセキュリティを提供するポリシーエージェント、Java™ および C アプリケーションプログラミングインタフェース (API)、XML (eXtensible Markup Language) ファイル、JavaServer Pages™、HTML (HyperText Markup Language) ページ、コマンド行インタフェース (CLI)、およびグラフィカルユーザーインタフェース (Identity Server コンソール) が含まれます。これらのインタフェースを使用することで、Sun ONE Directory Server 内のアイデンティティ情報の

管理が可能になります。また、SSO、認証、ポリシーベースの承認、およびログ機能を企業規模のアプリケーションにまで拡張することも可能になります。図 3-1 に、Identity Server 6.1 の機能アーキテクチャ、および異なるコンポーネントが相互にやり取りを行う方法を示します。

図 3-1 Identity Server 6.1 の機能アーキテクチャ



これらのコンポーネントのアーキテクチャ固有の詳細については、『Sun ONE Identity Server Customization And API Guide』を参照してください。次の「統合化ポイント」では、一部のコンポーネントが相互にやり取りを行う方法について説明します。

統合化ポイント

Identity Server は、さまざまな統合化ポイントを利用してサービスの提供を行う分散型システムです。「統合化ポイント」とは、アプリケーションがサードパーティ製のシステムやソフトウェアとの円滑な対話処理を可能にするコネクタのことです。Identity Server の主要な統合化ポイントは、ポリシーエージェントおよび SDK (ソフトウェア開発キット) です。

ポリシーエージェント

Identity Server 内のアクセス制御は、ポリシーエージェントを使用して行われます。ポリシーエージェントは、指定された Web、アプリケーション、またはプロキシサーバーを不正な侵入から保護します。ポリシーエージェントは、Identity Server とは別個に提供されます。最新の Sun ONE Identity Server ポリシーエージェントは、Sun Microsystems Download Center からダウンロードできます。ポリシーエージェントは、Web およびプロキシサーバー上のリソースを保護するものと、さまざまな配備コンテナ内の J2EE アプリケーションを保護するものの 2 つに分類できます。

Web およびプロキシサーバーエージェント

Web サーバーおよびプロキシサーバーのポリシーエージェントは、対象のサーバーに配備されたコンテンツを不正侵入から保護します。エージェントは、管理者が設定したポリシーに基づいて、多数のサービスおよび Web リソースへのアクセスを制御します。ユーザーが、ブラウザで、保護された Web サーバー上の URL を指定すると、エージェントは要求を遮断し、ユーザーのセッショントークンが存在する場合にはそれを検証します。トークンの認証レベルが不十分であるか存在しない場合、ログインページに適切な認証サービスが呼び出されて、認証を (さらに) 行うようユーザーに求めます。認証サービスは、ユーザーの証明情報の有効性を検証します。(たとえば、LDAP サービスは、ユーザー名およびパスワードが Sun ONE Directory Server に格納されているかどうかを検証します。) ユーザーの証明情報が正しく認証されている場合、ポリシーエージェントはユーザーに割り当てられているすべてのロール (ポリシーを含む) を確認します。ユーザーに割り当てられたすべてのポリシーを集約して、URL へのアクセスを許可するか拒否するかが個別に決定されます。

J2EE エージェント

Identity Server は、さまざまな配備コンテナ内の J2EE アプリケーションを保護するエージェントを提供します。これらのアプリケーションには、IBM Lotus Domino™ などのサーバーや、PeopleSoft などの企業が提供するアプリケーションが含まれます。通常、J2EE エージェントは、エージェントフィルタとエージェントレルムという 2 つのコンポーネントで構成されます (ただし、配備コンテナにより公開およびサポートされるインタフェースに部分的に左右されます)。前者は認証を処理し、後者は承認を処理します。

エージェントフィルタ

エージェントフィルタは、サーバー内部に向かう要求を遮断するサーブレットフィルタです。これは、要求にセッショントークンが含まれるかどうかをチェックします。利用可能なセッショントークンが存在する場合、エージェントフィルタは Identity Server セッションサービスを使用してトークンを検証します。利用可能なトークンが存在しない場合、ユーザーは、通常の SSO 交換の場合のように認証サービスにリダイレクトされます。認証されたユーザーは、サーバーに再度導かれます。エージェントフィルタはそこで要求を再び遮断し、新たに取得したトークンを検証します。検証後に、フィルタはコンテナの主体およびレルムをインスタンス化します。要求は許可され、フィルタを通過してアプリケーションにアクセスすることが可能になります。このメカニズムを通して、エージェントフィルタは、有効な Identity Server トークンを保持する要求だけが、保護されたアプリケーションへのアクセスを許可されることを保証します。

エージェントレルム

エージェントレルムは、承認コンポーネントを提供します。「レルム」とは、J2EE 準拠のアプリケーションサーバーが、ユーザー、グループ、配備済みアプリケーションへのアクセス制御に関する情報を提供する手段です。これは、セキュリティポリシーが定義および施行される範囲を指します。リソースへのアクセスが試みられた場合、サーバーは、特定のレルムを使用してユーザーおよびそのロールを検証するように設定されます。デフォルトでは、多数のアプリケーションサーバーが、出荷時にレルムを実装しています。これには、デフォルトの File Based に加え、LDAP、NT、UNIX、および RDBMS (Relational Database Management System) が含まれます。エージェントレルムは、サーバーのレルムインタフェースを実装します。Identity Server を配備することでユーザーおよびロール情報の管理が可能になります。エージェントレルムを使用すると、エージェントフィルタにより認証されたユーザーに対し、ロールベースの J2EE リソースの承認をきめ細かく提供できます。

注　詳細は、『Sun ONE Identity Server Web Policy Agents Guide』または『Sun ONE Identity Server J2EE Policy Agents Guide』を参照してください。

Identity Server SDK

Identity Server 内部のアプリケーションを統合する別の方法は、SDK を使用してプログラム上で行う方法です。SDK アクセスは、主に Java で提供されます。いくつかの機能は C インタフェースで提供されます。XML over HTTP(s) 経由で Identity Server サービスとのインタフェースを直接提供することも可能です。ただし、ドキュメント形式は以降のリリースで変更される可能性があります。Identity Server の将来のバージョンでは、WSDL (Web Service Defintion Language) により定義されたエンドポイントとの Web サービスインタフェースが正式にサポートされる予定です。現在提供されている SDK は、次のとおりです。

SSO API

Identity Server では、セッショントークンの検証および管理用インタフェース、およびユーザーの認証証明情報の管理用インタフェースが提供されます。SSO ソリューションへの参加を希望するアプリケーションはすべて、この API を使用できます。パッケージ名は `com.iplanet.sso` です。

認証 API と認証 SPI

Identity Server では、プログラムによる Identity Server への認証を可能にする JAAS 実装が提供されます。これにより、認証サービスのすべての機能にリモートからアクセスできます。また、JAAS PAM 仕様に準拠した認証 SPI により、新規認証タイプの作成が可能になります。

ポリシー API

ポリシー API を使用することで、Identity Server ポリシーの評価と管理、およびポリシーサービスの追加機能の提供が可能になります。ポリシーストアからポリシーを直接読み込んで解釈するポリシーエバリュエータ、および JAX-RPC (Java API for XML-based Remote Procedure Calls) を使用して Identity Server のクライアントとして動作するリモートポリシーエバリュエータが提供されます。この API は、アイデンティティがアクセスするリソースを判別する機能も提供します。

アイデンティティ管理 SDK

基盤となる Directory Server で使用されるデータモデリングが記述された Identity Server テンプレートと、アイデンティティ管理 SDK を組み合わせて使用することで、ユーザー、ロール、グループ、コンテナ、組織、組織単位、およびサブ組織の作成および管理用フレームワークが提供されます。これにより、LDAP クライアント API よりもはるかに高い抽象化レベルでインタフェースが提供されるため、ディレクトリベースのアイデンティティ情報の管理が簡略化されます。ここで、コアパッケージは、Directory Server に対して直接動作する `com.iplanet.am.sdk`、および JAX-RPC 経由

で Identity Server に対してリモートに動作することで Directory Server 自体に対して機能する com.sun.identity.um です。(これは、am_services.jar の一部です。) 最後に、データの検証、パスワードポリシーの施行などのタスクに対応した、コア SDK のプラグイン作成用の SPI が提供されます。

ログ API とログ SPI

ログサービスの主なタスクは、アクセス許可、アクセス拒否、およびユーザーアクティビティの記録です。ログ API を使用して、外部 Java アプリケーションのログ作成を有効にし、統合化された任意のアプリケーションに共通ログ機能を提供できます。ログ SPI を使用して、カスタマイズされた機能に対応したプラグインを開発できます。

サービス管理 SDK

開発者は、サービス管理インタフェースを使用してサービス定義を登録できます。サービス定義は、アプリケーション設定データの管理に使用されます。API パッケージ名は、com.sun.identity.sm です。

クライアントディテクション API

Identity Server は、リソースにアクセスを試みているクライアントブラウザのタイプを検出して、適切に書式設定されたページを使って応答できます。この目的で使用する API パッケージは、com.iplanet.services.cdm です。(このパッケージは、am_services.jar の一部です。)

SAML SDK

Identity Server は、SAML API を使用して認証動作、認証決定、および属性情報の交換を行います。API パッケージ名は、com.sun.identity.saml で始まります。(このパッケージは、am_services.jar の一部です。)

連携管理 API

Identity Server では、連携管理 API を使用して、Liberty Alliance Project 仕様に基づく機能が追加されます。API パッケージ名は、com.sun.liberty です。(このパッケージは、am_services.jar の一部です。)

機能プロセス

Identity Server には、統合化された多数の機能が含まれます。以下にその機能を示します。

- 認証とユーザーセッション
- 統合化ポリシー
- 統合化されたクライアントディテクション
- CDSSO、SAML、および連携

統合化されたプロセスについては、次の節で説明します。プログラミング関連の詳細については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

認証とユーザーセッション

認証サービスは、Identity Server のエントリポイントです。ユーザーは、Identity Server コンソールおよび対応する管理機能にアクセスする前に、認証プロセスに合格しなければなりません。Identity Server により保護されたサービスまたはアプリケーションへのアクセスを試みるユーザーは、アクセス許可の前にも認証が必要になります。認証サービスは、認証モジュールを呼び出して、必要な証明情報を収集および検証します。Identity Server は、アプリケーションによるシングルサインオン機能への参加を可能にする API も提供します。シングルサインオン機能を利用すると、ユーザーは一度の認証で複数のリソースにアクセスできるようになります。

Identity Server への認証には、2 つの方法があります。1 つは、管理者またはエンドユーザーが管理目的で Identity Server コンソール自体へのアクセスを試み、認証サービスにリダイレクトされる方法です。もう 1 つは、ユーザーが Identity Server により保護されたアプリケーションへのアクセスを試みて認証サービスにリダイレクトされる方法です。前者では HTML over HTTP(S) インタフェースを使用し、後者では C ベースの API または Java API を使用します。

注 3 番目のオプションは、ユーザーがリモートサーバーマシンの保護されたリソースへのアクセスを試みる場合です。このオプションには、Identity Server ポリシーエージェントのインストールが含まれます。詳細は、58 ページの「統合化ポリシー」を参照してください。

HTML over HTTP(S) インタフェース

通常の Web ベースのシナリオでは、管理目的で Identity Server を起動する管理者またはエンドユーザーが、Web ブラウザに認証サービスの URI を入力して、認証ユーザーインタフェース Java サーブレットにアクセスします。

注 *URI* は、よく知られた URL を含む、インターネットオブジェクトの記述法の総称です。

URI は、1 つ以上の認証モジュールを定義する認証プロセスに対して事前に設定されます。サーブレットは URI をパースし、セッションサービスと通信を行い (セッションサービスは無効な状態のセッショントークンを作成する)、トークンを Cookie に埋め込みます。認証 SPI (`com.sun.identity.authentication.spi`) は特定の認証モジュールを呼び出して、ログインページおよびトークン /Cookie を要求元のブラウザに渡します。ユーザーは証明情報を入力し、認証 SPI は情報を認証サービスに返します。認証サービスは、情報を対応するデータストア内の情報と比較して検証します。要求された認証プロセスがすべて完了および成功すると、ユーザーのポリシー (URL アクセスおよび拒否リストを含む)、認証レベル、および設定パラメータがセッションサービスに渡されます。セッションサービスは、トークンを有効な状態に設定します。(セッション情報はトークンに埋め込まれず、サーバーに格納されます。トークンは情報へのポインタに過ぎません。) 最後に、URL パラメータを使用してユーザーが適切なリソースに導かれます。

注 認証サービスの一般的な情報、および URL パラメータの詳細情報については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

XML over HTTP(S) インタフェース

Java と C アプリケーションの両方で、認証 API を使用して Identity Server からユーザー認証を要求できます。Java パッケージ `com.sun.identity.authentication` は、認証プロセスを開始し、未検証の認証証明情報をアプリケーションから認証サービスに送り返すためのインタフェースを提供します。

注 `com.sun.identity.authentication` は、ローカルとリモートのどちらでも実装できます。リモートの場合、開発者はこのパッケージから取得したクラスおよびメソッドを Java アプリケーションに統合します。

Java コールは XML メッセージに変換されて、HTTP(S) 経由で Identity Server に渡されます。受信後に、XML メッセージは Java に再度変換され、認証サービスにより解釈されます。C アプリケーション開発者も同じ手順に従いますが、最初に Identity Server との通信を行います。

注 `http://server_name.domain_name:port/service_deploy_uri/authservice` は、XML over HTTP プロトコルを使用して通信を行うサーブレットです。関連する DTD の `remote-auth.dtd` は、*IdentityServer_base*`/SUNWam/dtd` ディレクトリに存在します。

接続のオープン後に C アプリケーションが有効になり、認証要求を XML メッセージとして HTTP(S) 経由で Identity Server に送信します。受信時に、XML メッセージが再度 Java に変換され、認証サービスによりパースされます。認証サービスは、認証メソッドを判別して、セッションサービスと通信を行います。セッションサービスは、無効な状態のセッショントークンを作成し、トークンを Cookie に埋め込みます。認証 SPI (`com.sun.identity. authentication.spi`) は特定の認証モジュールを呼び出して、ログインページおよびトークン /Cookie を要求元のアプリケーションに渡します。ユーザーは証明情報を入力し、認証 SPI は情報を Identity Server に返します。Identity Server は、情報を対応する認証データストア内の情報と比較して検証します。要求された認証プロセスがすべて完了および成功すると、ユーザーのポリシー (URL アクセスおよび拒否リストを含む)、認証レベル、および設定パラメータがセッションサービスに渡されます。セッションサービスは、情報をセッショントークンに埋め込んで、有効な状態に設定します。最後に、要求元にアプリケーションへのアクセスが許可されます。

統合化ポリシー

Identity Server 配備内部の認証とシングルサインオンはどちらも相互依存の関係にあります。このため、認証が成功した後、ユーザーがセッショントークンを保持していない場合には、ユーザーができることはあまりありません。ポリシーサービスには、同様の相互依存性が存在します。ユーザーのポリシーにより配備内部の保護されたリソースに関する権限が定義されるため、Identity Server なしでは、定義済みのポリシーに関して多くを行うことはできません。この機能は、Identity Server により保護されたリソースを格納するリモートマシンの 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管理されます。

注　最新のポリシーエージェントは、`http://wwws.sun.com/software/download/inter_ecom.html` の Sun Microsystems Download Center 内の「Web And Directory Servers」ページからダウンロードできます。

ユーザーがポリシーエージェントにより保護された Web サーバーに接続すると、エージェントは要求を遮断して Cookie に埋め込まれたセッショントークンをチェックします。トークン /Cookie が見つからない場合、エージェントはユーザーをこのエージェント用に設定されたログイン URL にリダイレクトして認証プロセスを開始し、セッショントークンを作成します。ここから、55 ページの「認証とユーザーセッション」に記載された手順に従って処理が行われます。ユーザーは、有効なセッショントークンを受信すると、最初に要求したリソースに再び導かれます。そこで、エージェントにより要求が再度遮断され、セッショントークンがチェックされます。セッショントークンを検出したエージェントは、新規トークンを検証し、要求がネーミングサービスに渡されてトークンが復号化され、特定のセッションサービスが転送されて検証が行われるようにする必要があります。エージェントは、応答を受信してセッションサービスの URL を抽出し、その URL にアクセスしてトークンの検証を行います。セッションサービスは、応答をエージェントに返してトークンを検証します。エージェントは、何らかの理由でセッションがタイムアウトするか無効になった場合、セッションサービスが通知を受け取ることを要求します。セッションサービスは、肯定的な応答を返します。ユーザーの定義したポリシーを取得するため、エージェントはポリシーサービスと通信できるようになります。ポリシーサービスにより、決定が返されます。次に、エージェントはこの決定を使用して、要求されたリソースへのアクセスを許可または拒否します。また、ログメッセージがログサービスに送信されます。

統合化されたクライアントディテクション

ユーザーを認証する最初のステップは、要求元のクライアントタイプを識別することです。認証サービスは、URL 情報を使用してブラウザタイプの特性を取得します。これらの特性に基づいて、適正な認証ページ (HTML ページなど) がクライアントブラウザに送信されます。ユーザーの検証後に、クライアントタイプがセッショントークンに追加されます。セッショントークンを使用することで、他のサービスからもクライアントタイプを取得できるようになります。クライアントディテクションの詳細は、『Sun ONE Identity Server Customization And API Guide』のクライアントディレクションに関する章を参照してください。

CDSSO、SAML、および連携

CDSSO (Cross-Domain Single Sign-on)、SAML (Security Assertion Markup Language) サービス、および連携管理は、Identity Server の別個の機能です。どの機能も、手法は異なりますが提供する機能は同じです。CDSSO は、複数の DNS ドメイン間でシングルサインオンを可能にする Identity Server 独自の機能です。既存の配備に SAML 仕様が実装されている組織では、同様の目的 (クロスドメインのシングルサインオン) で SAML サービスを使用できます。連携管理では、連携アイデンティティ管理に Liberty Alliance Project バージョン 1.1 仕様が使用されます。連携管理を使用することで、接続されたパートナーのネットワーク経由でのサインオンの簡略化や、個人のアイデンティティの使用および公開などのユーザー制御が可能になります。

注 Liberty 仕様は、SAML 仕様を統合化して作成されています。ただし、この仕様で、SAML は Identity Server SAML サービス本体の一部ではありません。

CDSSO

CDSSO は、Identity Server 認証および承認プロセスの設定可能な部分です。これには、55 ページの「認証とユーザーセッション」で説明したポリシーエージェントのインストールおよび設定が含まれます。詳細は、『Sun ONE Identity Server Customization And API Guide』、『Sun ONE Identity Server Web Policy Agents Guide』、または『Sun ONE Identity Server J2EE Policy Agents Guide』のシングルサインオンに関する章を参照してください。

SAML

SAML サービスは、セキュリティ当局と信頼されるパートナー間のセキュリティ情報の交換に XML フレームワークを使用する、オープンな標準プロトコルに基づいています。Identity Server の内部アーキテクチャの詳細は、『Sun ONE Identity Server Customization And API Guide』の SAML サービスに関する章を参照してください。

連携

連携管理を使用すると、認証ドメイン、ホストプロバイダ、およびリモートプロバイダを Identity Server コンソールから設定および管理できます。これにより、ユーザーが一度サインオンするだけで、他の任意の連携リソースにアクセスできるようになります。連携アーキテクチャの詳細は、『Sun ONE Identity Server Customization And API Guide』の連携管理に関する章を参照してください。

Identity Server の拡張

Identity Server は、リモートの Web またはアプリケーションサーバー、Sun ONE Directory Proxy Server などの LDAP ロードバランサ、およびマルチマスターレプリケーション内に配備できます。インストールする前に、以下の製品が配備に適しているかどうかを確認してください。Identity Server をインストールする前に、これらの製品をインストールおよび設定することが必要な場合があります。

Web コンテナ

一部の Web コンテナは、Identity Server の配備先の Sun ONE Web Server (または他の Web コンテナ) に対してリモートの位置に存在します。リモートの Web コンテナが、組織のコンテンツページ用として配備済みである場合には、Web コンテナを追加する必要があります。コンテンツを保護するためにポリシーエージェントをインストールすると、リモートの Web コンテナは Identity Server 環境に統合されます。Web コンテナのインストールおよび管理の詳細は、サーバーに付属のマニュアルまたは Sun ONE Web Server のマニュアルを参照してください。

複数の Directory Server インスタンス

アップグレード、フェイルオーバーディレクトリの設定、マルチマスターの配備を行うために、Directory Server の複数インスタンスをインストールできます。Identity Server の配備を成功させるためには、Directory Server のインストール、設定、および配備を適切に行う必要があります。Identity Server をインストールする前に、データベースレプリケーションアグリーメントを定義する必要があります。Directory Server のインストールおよび配備に関する詳細は、『Sun ONE Directory Server インストールおよびチューニングガイド』および『Sun ONE Directory Server 配備ガイド』を参照してください。

LDAP ロードバランサ

Sun ONE Directory Proxy Server などのロードバランサと連携して動作するように、Identity Server を設定できます。Identity Server 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サーバー間でロードバランスを行う方法と、ユーザー付近に配置されているレプリケートされたサーバーの位置を検出する方法の 2 つの方法があります。ロードバランサを使用することで、Identity Server の基本機能に、高可用性レイヤーおよびディレクトリフェイルオーバー保護を追加できます。バックエンドの LDAP サーバーが一切利用できない場合、ロードバランサは要求トラフィックを管理し、クライアントのクエリを拒否します。インストールおよび管理の詳細は、Sun ONE Directory Proxy Server 5.2 のマニュアルを参照してください。ロードバランサに関するその他の情報については、製品に付属のマニュアルを参照してください。

第 4 章

配備前の考慮事項

Sun™ ONE Identity Server 6.1 は、異機種システムの混在するハードウェア、ソフトウェア、およびアプリケーションインフラストラクチャを抱える大規模な組織が、従業員、受託業者、顧客、およびサプライヤのアイデンティティ管理ソリューションを成功裏に配備することを可能にします。この章では、このプロセスに関連する高度な技術的概要を説明します。次の節で構成されています。

配備オプション

Identity Server の配備を計画する際、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。例を示します。

- 現在サポートが予定されている環境にどれくらいのユーザーが存在するか。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将来の計画はあるか。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

セキュリティの確保された内部および外部ネットワーク環境を提供する際、考慮する必要のあるオプションが多数存在します。以下にその内容を示します。

- 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、追加セキュリティレイヤーを提供するサーバーベースのファイアウォール。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利用される機会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指します。
- 分割 DNS インフラストラクチャは、1 つのドメイン内に 2 つのゾーンが作成されるインフラストラクチャです。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レベルを実現するために推奨されています。DNS サーバーも、ロードバランスを行うことで、パフォーマンスを向上させることができます。

高可用性

IT の配備では、ユーザーの可用性の継続と同様に、SPOF (Single Point Of Failure) が発生しないよう、努力が求められます。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。期待される高可用性とは、システムやコンポーネントが期待とおりに長期間、連続的に使用可能であることです。このシステムは一般的に、複数のサーバーマシンで構成されますが、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、次の SPOF が含まれます。

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

これらの問題のうち、その大半は事前に予測できるので、高可用性の実現手法は、データストレージのバックアップやアクセスに対するフェイルオーバーの処理を中心に計画することが可能です。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。

注 高可用性シナリオでは、Identity Server のインストールすべてで同一の暗号化キーを指定する必要があります。

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。クラスタリングは、Sun ONE Identity Server では使用できないものの、システムの基盤である Sun ONE Directory Server のデータストアでは、極めて重要な手法です。たとえば、クラスタ化された 1 組の MMR サーバーでは、クラスタでの可用性を保証することにより、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

「水平スケーリング」は、複数のサーバーマシンを接続して 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バーは、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 台のサーバーマシン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、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、協調動作が可能です。通常、環境内のサーバーのインストールにより能力の限界まで使用されることはないため、垂直スケーリングはパフォーマンスを改善するために使用されます。また、特定のマシンが処理能力の限界に近づいた場合も、水平スケーリングは、多数のサーバーによって負荷を分散します。

ハードウェア要件

Identity Server をインストールするハードウェアは、一定の要件を満たす必要があります。Identity Server は、最低限、Sun ONE Directory Server 5.2 (データストアとして使用する) および配備先の Web コンテナと共にインストールする必要があります。Directory Server および Identity Server は、異なるマシンにインストールすることが推奨されています。

詳細な要件は、コンポーネントのデフォルト設定 (Identity Server (Sun ONE Web Server により配備される) の 1 つのインスタンス、および Directory Server の 1 つのインスタンス) に基づいて行われる高度な設定です。Identity Server をインストールする前に、『Directory Server インストールおよびチューニングガイド』、『導入ガイド』、および選択した Web コンテナのマニュアルを参照してください。

注 推奨される手順は、Sun ONE Identity Server を設計および配備する前に Sun ONE プロフェッショナルサービスまたは Sun ONE 認定システムインテグレータに相談することです。

Identity Server は、100M バイト以上の Ethernet ネットワーク上で実行することが推奨されています。Identity Server 配備の最小構成は、Identity Server と Sun ONE Web Server の両方がインストールされたマシンです。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 を搭載しても効果はあまり期待できません。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、256M バイト以上の RAM が必要です。現実の配備シナリオでは、スレッド、SDK、HTTP サーバー、および他の内部処理用に 1G バイトの RAM が推奨されています。各 Identity Server 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います。その後、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 バイトメモリ制限を前提とする)。

注 ディレクトリリソース要件は、顧客固有のデータおよび使用方法に応じて変化しますが、通常は高度な要件が求められます。

ソフトウェア要件

Identity Server のインストール先システムは、最小のソフトウェアおよびオペレーティングシステム要件を満たす必要があります。

オペレーティングシステム要件

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

- Sun Solaris™ 8 オペレーティングシステム (SPARC 32/64 ビットプロセッサ)
- Sun Solaris 9 オペレーティングシステム (SPARC 32/64 ビットプロセッサ)
- Sun Solaris 9 オペレーティングシステム (x86 プロセッサ)

Solaris 用パッチクラスタ

Directory Server を Solaris 8 オペレーティングシステム上で実行する場合、推奨されるパッチクラスタをインストールする必要があります。パッチクラスタは、108827-15 のように 2 つの番号で識別されます。最初の番号 (108827) は、パッチ自体を識別するためのものです。2 番目の番号により、パッチのバージョン (15) を示します。最新の修正の恩恵を受けるには、最新のパッチをインストールする必要があります。全般的なパッチ情報および推奨されるパッチは、SunSolve Patch Support Portal からダウンロードできます。(パッチクラスタは、Sun ONE Directory Server、Sun ONE Web Server、および Sun ONE Application Server に必須です。)

注 コマンド `showrev -p` を使用して、Solaris マシンにインストール済みのパッチのリストを表示できます。

以下に、Identity Server をインストールする前にインストールしておく必要のあるパッチリストを示します。必要パッチに関する詳細な情報については、『Sun Java Enterprise System 2003Q4 インストールガイド』を参照してください。

表 4-1 Identity Server のインストールで推奨されるパッチ

112396-02	111293-04	109888-20
108869-18	109326-10	108725-12
109147-21	112218-01	108981-10
108949-07	108434-10	109885-09

表 4-1　Identity Server のインストールで推奨されるパッチ (続き)

108435-10	110458-02	112237-07
111325-02	109898-05	109234-09
110075-01	110901-01	109134-27
108901-06	112325-01	108968-08
110662-11	110943-01	108975-06
110898-08	109324-05	108875-13
111111-03	110916-04	110460-26
109091-05	110387-03	108727-22
110283-06	109277-03	109318-31
110951-03	111234-01	108985-03
110903-05	112138-01	109238-02
109320-06	109470-02	109793-14
109783-02	109951-01	111299-04
110453-04	110945-07	110934-11
111071-01	111232-01	108997-03
110700-01	110939-01	109007-09
111548-01	111570-02	108974-25
108652-65	110322-02	110386-02
110286-10	111504-01	108987-12
111606-02	109882-06	109805-15
111069-01	110615-08	108827-40
110668-03	110670-01	108993-13
111826-01	111874-06	110723-05
111659-07	111596-03	111327-05
109667-04	110957-02	108977-01
111626-03	111085-02	110380-04
108919-17	110838-06	108528-19
112254-01	112459-01	108989-02
111879-01	112611-01	111881-03
112425-01	112668-01	109657-09

表 4-1　Identity Server のインストールで推奨されるパッチ (続き)

111958-02	112796-01	111883-14
112846-01	112279-02	114152-01
108806-15	110842-10	111310-01
111321-03	109328-03	111098-01
113792-01	109862-03	113650-01
110896-02	108899-04	109223-02

Java™ 要件

Identity Server には、Java バージョン 1.3.1 が必須です。使用が推奨されているのは、Java バージョン 1.4.1 です。

リソース Web サーバー要件

Identity Server でサポートされる Web コンテナは、Sun ONE Web Server 6.1 および Sun ONE Application Server 7.0 です。ポリシーエージェントをこれらの Web サーバーコンテナにインストールする場合、約 10M バイトのディスク容量が使用されます。コンテンツを提供する Web コンテナを設定する場合には、このことを考慮に入れておく必要があります。詳細は、『Sun ONE Identity Server Web Policy Agents Guide』または『Sun ONE Identity Server J2EE Policy Agents Guide』を参照してください。

Web ブラウザ要件

管理者およびエンドユーザーは、Web ブラウザを使用してユーザー管理タスクを実行します。Identity Server は、次の Web ブラウザをサポートします。

- Netscape Communicator 4.79
- Microsoft Internet Explorer 5.5 SP 2
- Microsoft Internet Explorer 6.0

Identity Server スキーマの理解

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。Directory Server には、データの格納方法を定義する LDAP (Lightweight Directory Access Protocol) スキーマが含まれます。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。

Identity Server は、Directory Server を、アイデンティティプロファイル、資格定義、および配備設定情報すべてのデータリポジトリとして使用します。このために、Identity Server は、Directory Server スキーマを拡張する独自のスキーマを保持します。Identity Server のインストール時に、`ds_remote_schema.ldif` および `sunone_schema2.ldif` に記述された Identity Server スキーマは、Directory Server スキーマと統合されます。`ds_remote_schema.ldif` には、特に Identity Server により使用される LDAP オブジェクトクラスおよび属性が記述されます。一般に、これらのオブジェクトクラスおよび属性は、以前のバージョンの Identity Server から受け継いだものです。`sunone_schema2.ldif` は、Sun Microsystems の新しい内部スキーマドキュメントで定義された Identity Server 固有の LDAP 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。参照用に、`ds_remote_schema.ldif` および `sunone_schema2.ldif` の内容を、それぞれ 70 ページのコード例 4-1 および 74 ページのコード例 4-2 に示します。

コード例 4-1 ds_remote_schema.ldif

```
add: objectClasses

objectClasses: ( 2.16.840.1.113730.3.2.175 NAME
'iplanet-am-session-service' DESC 'Session Service OC' SUP top
AUXILIARY MAY ( iplanet-am-session-max-session-time $
iplanet-am-session-max-idle-time $
iplanet-am-session-max-caching-time $
iplanet-am-session-get-valid-sessions $
iplanet-am-session-destroy-sessions $
iplanet-am-session-add-session-listener-on-all-sessions $
iplanet-am-session-service-status ) X-ORIGIN 'Sun Java System
Identity Management' )
```

コード例 4-1　　ds_remote_schema.ldif (続き)

```
objectClasses: ( 2.16.840.1.113730.3.2.176 NAME
'iplanet-am-user-service' DESC 'User Service OC' SUP top
AUXILIARY MAY ( iplanet-am-user-auth-modules $
iplanet-am-user-login-status $ iplanet-am-user-admin-start-dn $
iplanet-am-user-auth-config $ iplanet-am-user-alias-list $
iplanet-am-user-success-url $ iplanet-am-user-failure-url $
iplanet-am-user-federation-info-key $
iplanet-am-user-federation-info $
iplanet-am-user-password-reset-options $
iplanet-am-user-password-reset-question-answer $
iplanet-am-user-password-reset-force-reset $
sunIdentityServerDiscoEntries ) X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.177 NAME
'iplanet-am-web-agent-service' DESC 'Web Agent Service OC' SUP
top AUXILIARY MAY ( iplanet-am-web-agent-access-allow-list $
iplanet-am-web-agent-access-deny-list $
iplanet-am-web-agent-access-not-enforced-list $
iplanet-am-web-agent-service-status ) X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.179 NAME
'iplanet-am-managed-role' DESC 'Managed Role OC' SUP top
AUXILIARY MAY ( iplanet-am-role-type $
iplanet-am-role-description $ iplanet-am-role-aci-description $
iplanet-am-role-aci-list $ iplanet-am-role-service-options $
iplanet-am-role-any-options $
iplanet-am-role-managed-container-dn $
iplanet-am-role-display-options) X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.180 NAME
'iplanet-am-managed-group' DESC 'Managed Group OC' SUP top
AUXILIARY MAY ( iplanet-am-group-subscribable $ inetgroupstatus )
X-ORIGIN 'Sun Java System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.181 NAME
'iplanet-am-managed-filtered-group' DESC 'Managed Filter Group
OC' SUP iplanet-am-managed-group AUXILIARY X-ORIGIN 'Sun Java
System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.182 NAME
'iplanet-am-managed-assignable-group' DESC 'Managed Assignable
Group OC' SUP iplanet-am-managed-group AUXILIARY X-ORIGIN 'Sun
Java System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.183 NAME
'iplanet-am-managed-static-group' DESC 'Managed Static Group OC'
SUP iplanet-am-managed-group AUXILIARY X-ORIGIN 'Sun Java System
Identity Management' )
```

コード例 4-1　　ds_remote_schema.ldif (続き)

```
objectClasses: ( 2.16.840.1.113730.3.2.184 NAME
'iplanet-am-managed-person' DESC 'Managed Person OC' SUP top
AUXILIARY MAY ( iplanet-am-modifiable-by $
iplanet-am-static-group-dn $ iplanet-am-user-account-life )
X-ORIGIN 'Sun Java System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.186 NAME
'iplanet-am-managed-org-unit' DESC 'Managed OrganizationalUnit
OC' SUP top AUXILIARY MAY ( sunPreferredDomain $ associatedDomain
$ sunPreferredOrganization $ sunAdditionalTemplates $
sunOverrideTemplates $ iplanet-am-service-status ) X-ORIGIN 'Sun
Java System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.187 NAME
'iplanet-am-managed-people-container' DESC 'Managed People
Container OC' SUP top AUXILIARY X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.189 NAME
'iplanet-am-managed-group-container' DESC 'Managed Group
Container OC' SUP top AUXILIARY X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.166 NAME
'iplanet-am-managed-policy' DESC 'Managed Name Policy OC' SUP top
AUXILIARY MAY iplanet-am-named-policy-dn X-ORIGIN 'Sun Java
System Identity Management' )
objectClasses: ( 2.16.840.1.113730.3.2.167 NAME
'iplanet-am-domain-url-access-service' DESC 'Domain URL Access
Service OC' SUP top AUXILIARY MAY
iplanet-am-domain-url-access-allow X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.22 NAME
'iplanet-am-saml-service' DESC 'SAML Service OC' SUP top
AUXILIARY MAY ( iplanet-am-saml-user $ iplanet-am-saml-password )
X-ORIGIN 'Sun Java System Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.23 NAME
'iplanet-am-auth-configuration-service' DESC 'Authentication
Configuration Service OC' SUP top AUXILIARY MAY (
iplanet-am-auth-configuration $ iplanet-am-auth-login-success-url
$ iplanet-am-auth-login-failure-url $
iplanet-am-auth-post-login-process-class ) X-ORIGIN 'Sun Java
System Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.25 NAME 'sunservice'
DESC 'object containing service information' SUP top MUST ou MAY
( labeleduri $ sunserviceschema $ sunkeyvalue $ sunxmlkeyvalue $
sunpluginschema $ description ) X-ORIGIN 'Sun Java System
Identity Management' )
```

コード例 4-1　　ds_remote_schema.ldif (続き)

```
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.26 NAME 'sunorgservice'
DESC 'Service information specific to organizations' SUP top MUST
ou MAY ( sunkeyvalue $ sunxmlkeyvalue $ description ) X-ORIGIN
'Sun Java System Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.27 NAME
'sunservicecomponent' DESC 'Sub-components of the service' SUP
top MUST ou MAY ( sunserviceid $ sunsmspriority $ sunkeyvalue $
sunxmlkeyvalue $ description ) X-ORIGIN 'Sun Java System Identity
Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.28 NAME
'sunserviceplugin' DESC 'Object that stores information specific
to plugins' SUP top MUST ou MAY ( sunpluginid $ sunkeyvalue $
sunxmlkeyvalue $ sunsmspriority ) X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.74 NAME
'iplanet-am-managed-filtered-role' DESC 'Managed Filtered Role
OC' SUP iplanet-am-managed-role AUXILIARY X-ORIGIN 'Sun Java
System Identity Management' )
objectClasses: ( sunISManagedOrganization-oid NAME
'sunISManagedOrganization' DESC 'Sun Java System objectclass to
identify organizations' SUP top AUXILIARY MAY (
sunOrganizationAlias ) X-ORIGIN 'Sun Java System Identity
Management' )
objectClasses: ( sunIdentityServerDiscoveryService-OID NAME
'sunIdentityServerDiscoveryService' DESC 'Discovery Service OC'
SUP top AUXILIARY MAY ( sunIdentityServerDynamicDiscoEntries )
X-ORIGIN 'Sun Java System Identity Management' )
```

コード例 4-1　　ds_remote_schema.ldif (続き)

```
objectClasses: ( sunIdentityServerLibertyPPService-oid NAME
'sunIdentityServerLibertyPPService' DESC
'sunIdentityServerLibertyPPService OC' SUP top AUXILIARY MAY (
sunIdentityServerPPCommonNameCN $
sunIdentityServerPPCommonNameALTCN $
sunIdentityServerPPCommonNameFN $ sunIdentityServerPPCommonNameSN
$ sunIdentityServerPPCommonNamePT $
sunIdentityServerPPCommonNameMN $ sunIdentityServerPPInformalName
$ sunIdentityServerPPLegalIdentityLegalName $
sunIdentityServerPPLegalIdentityDOB $
sunIdentityServerPPLegalIdentityMaritalStatus $
sunIdentityServerPPLegalIdentityGender $
sunIdentityServerPPLegalIdentityAltIDType $
sunIdentityServerPPLegalIdentityAltIDValue $
sunIdentityServerPPLegalIdentityVATIDType $
sunIdentityServerPPLegalIdentityVATIDValue
$sunIdentityServerPPEmploymentIdentityJobTitle
$sunIdentityServerPPEmploymentIdentityOrg $
sunIdentityServerPPEmploymentIdentityAltO ) X-ORIGIN 'Sun Java
System Identity Management' )
objectClasses: ( sunIdentityServerDevice-OID NAME
'sunIdentityServerDevice' DESC 'Device OC' SUP top AUXILIARY MAY
( cn $ uid $ sunIdentityServerDeviceVersion $
sunIdentityServerDeviceType $ userpassword $
sunIdentityServerDeviceKeyValue $ sunxmlkeyvalue $ description $
sunIdentityServerDeviceStatus ) X-ORIGIN 'Sun Java System
Identity Management' )
```

コード例 4-2　　sunone_schema2.ldif

```
add: objectClasses

objectClasses: ( 2.16.840.1.113730.3.2.185 NAME
'sunManagedOrganization' DESC 'Auxiliary class which must be
present in an organization entry' SUP top AUXILIARY MAY (
inetDomainStatus $ sunPreferredDomain $ associatedDomain $
sunPreferredOrganization $ sunAdditionalTemplates $
sunOverrideTemplates $ sunRegisteredServiceName $
organizationName  ) X-ORIGIN 'Sun Java System Identity
Management' )
```

コード例 4-2　　sunone_schema2.ldif (続き)

```
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.75 NAME
'sunManagedSubOrganization' DESC 'Auxiliary class which must be
present in an sub organization entry' SUP top AUXILIARY MAY (
inetDomainStatus $ parentOrganization ) X-ORIGIN 'Sun Java System
Identity Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.29 NAME 'sunNameSpace'
DESC 'Auxiliary class which must be present at the root of a
subtree representing a namespace' AUXILIARY MAY
sunNameSpaceUniqueAttrs X-ORIGIN 'Sun Java System Identity
Management' )
objectClasses: ( 1.3.6.1.4.1.42.2.27.9.2.27 NAME
'sunservicecomponent' DESC 'Sub-components of the service' SUP
top MUST ou MAY ( sunserviceid $ sunsmspriority $ sunkeyvalue $
sunxmlkeyvalue $ description ) X-ORIGIN 'Sun Java System Identity
Management' )
```

マーカーオブジェクトクラス

Identity Server 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 Identity Server 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他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、Identity Server 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の詳細は、『Sun ONE Identity Server Customization And API Guide』の第 5 章「Identity Management」を参照してください。既存の Directory Server データを Identity Server で使用するために移行する方法については、『Sun ONE Identity Server Migration Guide』を参照してください。

管理ロール

LDAP エントリの委任された管理 (Identity Server 内の各アイデンティティ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、Identity Server のインストール時に作成されます。これは、Identity Server コンソールを使用して表示および管理できます。Identity Server のアイデンティティ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。その後、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、Identity Server 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、そのロールに適したすべてのアクセス権がユーザーに与えられます。表 4-2 に、Identity Server 管理ロールおよび、各ロールに対応する書き込み権限の範囲を示します。

表 4-2 デフォルトおよび動的なロールとそのアクセス権

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	最上位組織内で作成された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。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る

表 4-2 デフォルトおよび動的なロールとそのアクセス権 (続き)

ロール	管理サフィックス	アクセス権
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
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

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 の詳細は、『Sun ONE Identity Server Administration Guide』の第 14 章「管理属性」を参照してください。

スキーマの制限

以降の節では、Identity Server スキーマに課される制限のいくつかを説明します。

ピープルコンテナ

Identity Server では、ピープルコンテナは、ユーザー専用の親エントリです。通常は、組織単位がピープルコンテナとして使用されますが、`iplanet-am-managed-people-container` オブジェクトクラスおよび Identity Server の管理可能な親タイプのオブジェクト、組織、またはコンテナを保持する限り、どのエントリでもピープルコンテナとして使用できます。LDAP エントリに `iplanet-am-managed-people-container` のマークが付けられると、Identity Server は下位ピープルコンテナまたはユーザーのみが含まれると見なします。このため、ピープルコンテナに含めることができるのは、下位ピープルコンテナまたはユーザーのみです。

1 つの Identity Server 組織のみが許可される

Identity Server 組織のマークを付けることができるのは、1 つの LDAP オブジェクトタイプだけです。つまり、1 つの LDAP タイプエントリに追加できるのは、`iplanet-am-managed-org` マーカーオブジェクトクラスだけです。(このオブジェクトクラスが追加されるのは、Identity Server がこのエントリを組織であるかのように管理するためです。)

警告 インストール時に作成されたルートサフィックス組織は、この 1 組織制限の対象にはなりません。

このため、図 4-1 では、`dc` と `o` の両方のエントリを Identity Server コンソールを使用して組織として管理することはできません。

図 4-1 管理不可能なディレクトリツリー

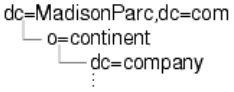
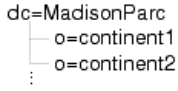


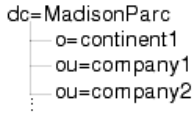
図 4-2 に示すツリーは、Identity Server で管理できます。

図 4-2　1 組織ルールの例外



o=continent1 の下に dc=company1 を追加する場合、dc が組織ではなくコンテナとしてマークされている場合にのみ、このツリーは管理可能になります。通常、iplanet-am-managed-container オブジェクトクラスはすべての組織単位に追加されますが、任意のエントリに追加することが可能です。

図 4-3　2 つの組織単位が許可される

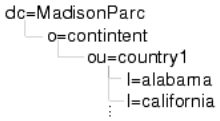


この例では、o= と ou= の両方のエントリを組織としてマークすることはできないため、o= エントリを組織としてマークし、ou= エントリをコンテナとしてマークします。Identity Server を使用して両方の組織およびコンテナを管理する場合、利用可能なオプションは同じになります。ピープルコンテナ、下位組織、グループ、ロール、およびユーザーは、両方の内部で作成できます。

サポートされないディレクトリツリー

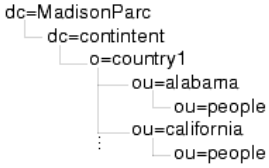
既存のディレクトリツリーの大半は、Identity Server で動作するように再設定可能ですが、再設定が推奨されない場合もあります。一般に、既存のツリーが、組織の定義に複数タイプの LDAP エントリ (例 : dc、o、および ou) を使用する場合、ユーザーデータは、一定の条件下でのみ Identity Server により認識されます。次の例では、3 タイプの組織マーカー、o、ou、および l が必要になります。l=california および l=alabama がピープルコンテナではないと見なされるため、この DIT は Identity Server では動作しません。

図 4-4 シナリオ 1: 許可されない 3 つの LDAP 組織属性



次の例では、3 タイプの Identity Server 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 Identity Server で動作しません。

図 4-5 シナリオ 2: 許可されない 3 つの LDAP 組織属性



第 5 章

配備シナリオ

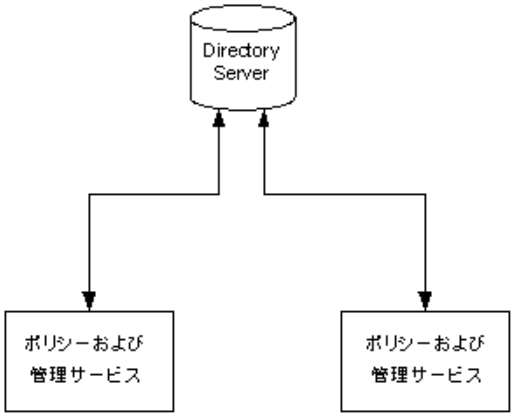
この章では、Identity Server の簡単な配備シナリオについて説明します。次の節で構成されています。

複数サーバーのシナリオ

推奨される配備シナリオには、それぞれに Identity Server と Directory Server の両方がインストールされた 2 つの物理サーバーが含まれます。Directory Server のインスタンスは、複数のマスター設定内で設定されます。この配備を適切な手順で実行するには、まず、最初のマシン上で Directory Server の最初のインスタンスをインストールします。次に Directory Server の 2 番目のインスタンスを 2 番目のマシンにインストールし、マルチマスター関係を設定します。ここで、Identity Server を最初のマシンにインストールして、Directory Server のいずれかのインスタンスを指し示すようにできます。インストール方法は、現在の設定に応じて既存の DIT を使用するか、または使用せずに既存のディレクトリを選択することです。Identity Server の 2 番目のインスタンスを 2 番目のマシンにインストールし、既存の DIT を使用して既存の Directory Server を指し示すようにします。Identity Server は、既存のものと認識する Directory Server にいかなる情報も記述しません。このため、既存のデータを上書きしても危険はありません。Identity Server は、2 つの属性を追加して、2 番目のインスタンスを動作可能にします。

パフォーマンスの拡張、ディレクトリのレプリケーションのサポート、またはエージェントのフェイルオーバーを目的として、Identity Server の複数インスタンスを Directory Server に対してインストールできます。同一の Directory Server に対し、Identity Server の複数インスタンスが異なるハードウェア上に存在するようにするため、83 ページの「ammultiserverinstall を使用して複数の Identity Server をインストールするには」の手順に従って操作を行います。2 番目の方法は、2 番目の Identity Server をインストールし、設定済みの Directory Server に対してそれを指し示すことです。図 5-1 に、単一の Directory Server に対してインストールされた 2 つの Identity Server インスタンスを示します。

図 5-1　1 つの Directory Server に対する複数の Identity Server



ammultiserverinstall を使用して複数の Identity Server をインストールするには

1 つのマスター Directory Server に対して複数の Identity Server インストールを実行するには、`ammultiserverinstall` スクリプトを実行する必要があります。複数の Identity Server インスタンスを作成およびインストールするために、管理者は root 権限を保持する必要があります。

1. 次のディレクトリに移動します。

   ```
   cd Identity_Server_root/SUNWam/bin
   ```

2. コマンド行で、次のコマンドを入力します。

   ```
   ./ammultiserverinstall instance_name port_number
   ```

   instance_name には作成する新規 Identity Server インスタンスを、*port_number* には新規 Identity Server インスタンスのポート番号を指定します。

新規インスタンスの作成時に、以下が作成されます。

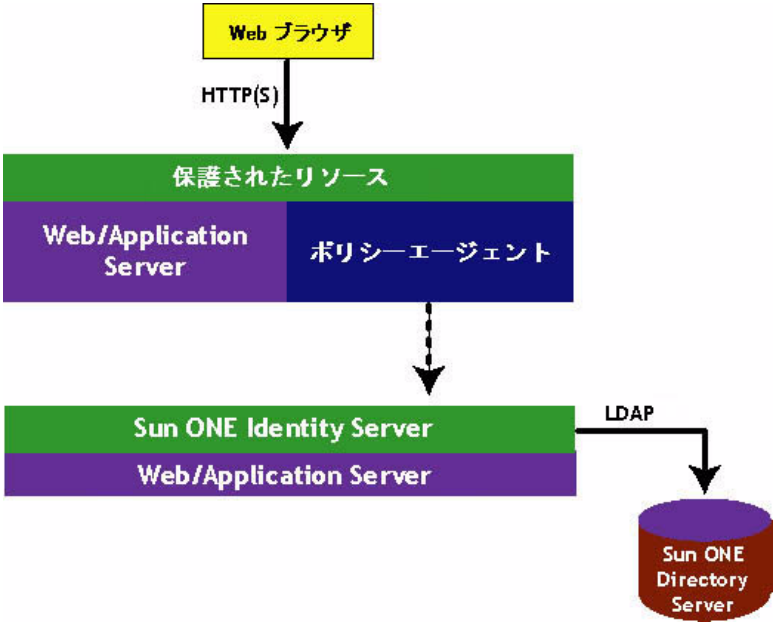
- `/IdentityServer_base/SUNWam/bin/amserver.`*instance_name* に新規 `amserver` スクリプトファイル
- `/IdentityServer_base/SUNWam/lib/AMConfig-`*instance_name*`.properties` に新規 `AMConfig.properties` ファイル
- `/IdentityServer_base/SUNWam/servers/https-`*instance_name* に新規 Web サーバーインスタンスディレクトリ

`ammultiserverinstall` スクリプトの詳細は、『Sun ONE Identity Server 管理ガイド』を参照してください。

Web 配備

Identity Server 配備の最も一般的な使用法は、Web ブラウザが 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する場合です。アプリケーションおよびリソースは Identity Server 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 Identity Server SDK が配備されている場合があります。このアーキテクチャにより、マシン内の Web サーバーの数や、複数マシンにまたがる Identity Server のインスタンスに関して制限が課されることはありません。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 Identity Server を配備できます。同様に、複数の Web サーバーを異なるマシン上で稼働させ、それぞれに Identity Server を配備することもできます。84 ページの図 5-2 に、このサンプル配備シナリオを示します。

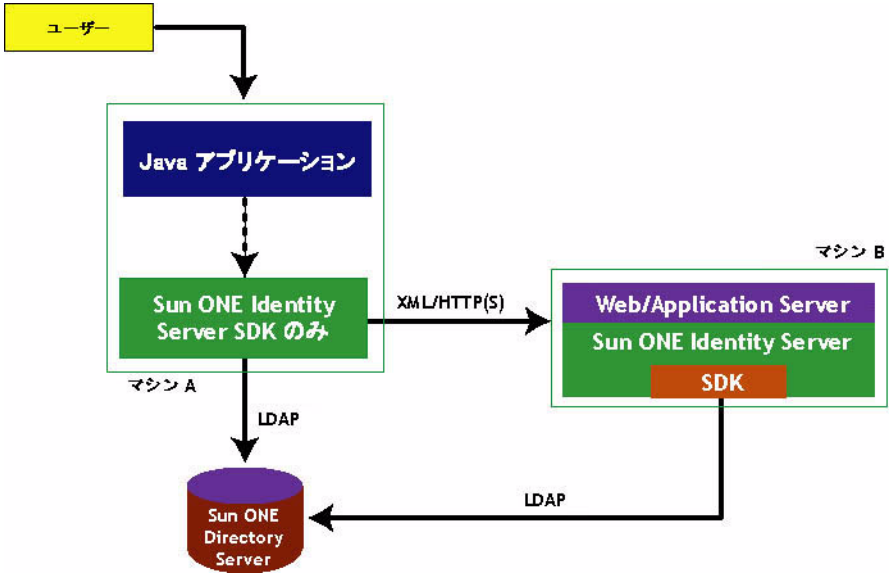
図 5-2　簡単な Web 配備シナリオ



Java アプリケーションの配備

Identity Server の別の一般的なシナリオは、Java™ アプリケーションが配備先のマシンに直接インストールされた Identity Server SDK にアクセスする場合です。このシナリオでは、1 つ以上の Identity Server インスタンスが稼働する Sun ONE Web Server または Sun ONE Application Server のインスタンスが存在する追加マシンが必要になります。このマシンは、状態情報を管理してシングルサインオンを提供します。85 ページの図 5-3 に、この Java アプリケーションの配備シナリオを示します。

図 5-3　Java アプリケーションの配備



複数の JVM 環境

Identity Server 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 ONE 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ること、そのすべてで Identity Server サービスが稼働可能であることを意味します。Identity Server のアーキテクチャは、マシン内の Application Server インスタンスの数、複数マシンをまたがる Identity Server サービスの数、単一の Application Server が保持できる JVM の数について制限を課しません。複数の JVM 環境の詳細は、Sun ONE Application Server のマニュアルを参照してください。

レプリケーションに関する考慮事項

Identity Server 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行う方法と、ユーザー付近に配置されているレプリケートされたサーバーの位置を検出する方法の 2 つの方法があります。Directory Server の設定は、単一サプライヤおよび複数サプライヤ設定内で行うことができます。Sun ONE 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 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。

注 Identity Server をインストールする前に、ディレクトリレプリケーションアグリーメントを定義する必要があります。これにより、参照を検証する時間が許可されて、サプライヤデータベースとコンシューマデータベースが適正に同期するようになるため、更新の同期が正しく実行されます。

Identity Server 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 Identity Server 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
- ディレクトリマネージャのパスワード
- 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

Identity Server は、単一サプライヤまたは複数サプライヤレプリケーションで動作するように設定できます。図 5-4 に、コンシューマが読み取り専用データベースである単一サプライヤ設定を示します。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 5-4　シングルサプライヤレプリケーション

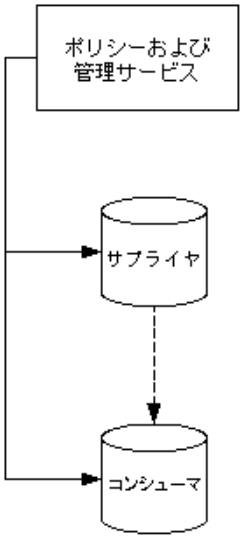
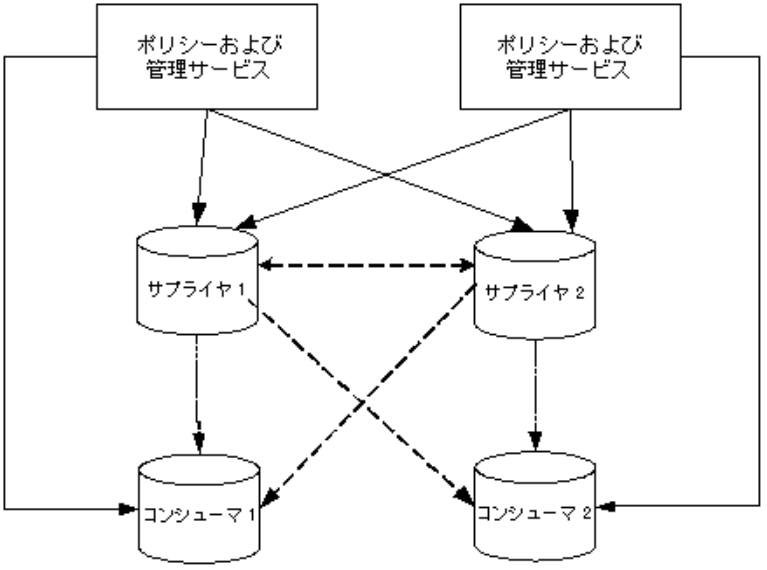


図 5-5 に、Identity Server の複数インスタンスを使用した複数サプライヤ設定を示します。この設定によりフェイルオーバー保護および高可用性が提供されるため、サーバーのパフォーマンスはさらに向上します。

図 5-5　複数サプライヤ設定 (マルチマスターレプリケーションとも呼ばれる)



以下の手順を使用すると、Identity Server がまだインストールされていない場合に、Identity Server 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できます。また、デフォルトの組織レベルでレプリケーションを設定する場合にも、以下の手順を使用できます。

1. サプライヤおよびコンシューマ Directory Server をインストールします。

   手順の詳細は、『Sun ONE Directory Server インストールガイド』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。

   手順の詳細は、『Sun ONE Directory Server 管理者ガイド』を参照してください。

注　このバージョンの Identity Server で機能するように、既存の Directory Server データを移行することが必要な場合があります。手順の詳細は、『Sun ONE Identity Server Migration Guide』を参照してください。

3. Identity Server および Directory Server を初めて配備する場合、または既存のユーザーデータの使用を計画していない場合、Sun Java Enterprise System 2003Q4 インストールプログラムを実行して Identity Server 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、手順 1 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. Identity Server 管理ポリシーサービスをインストールしたサーバーで、次のファイルを修正します。

   `/IdentityServer_base/SUNWam/lib/AMConfig.properties`

   a. 次のプロパティを修正して、手順 1 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

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

      com.iplanet.am.replica.retries

   c. 次のプロパティを修正して、Identity Server 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

5. 有効な Identity Server 認証モジュールごとに、手順 1 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。次の手順では、例として LDAP 認証モジュールを使用します。

   a. Identity Server コンソールで、「サービス設定」を選択します。

   b. 再設定する認証モジュールを見つけて、「プロパティ」の矢印をクリックします。

   c. 右側の区画で、以下の操作を行います。

      - 「LDAP サーバーとポート」という名前の最初のフィールドに、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入力します (例 : `consumer1.example.com:389`)。
      - 「LDAP サーバーとポート」という名前の 2 番目のフィールドに、セカンダリ (サプライヤ) Directory Server のホスト名とポート番号を入力します (例 : `supplier1.example.com:389`)。

   d. 「実行」をクリックします。

6. `/Identity_Server_root/SUNWam/config/ums/serverconfig.xml` ファイルで、コード例 5-1 に示すように、手順 1 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。

コード例 5-1　　serverconfig.xml レプリケーションの修正

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

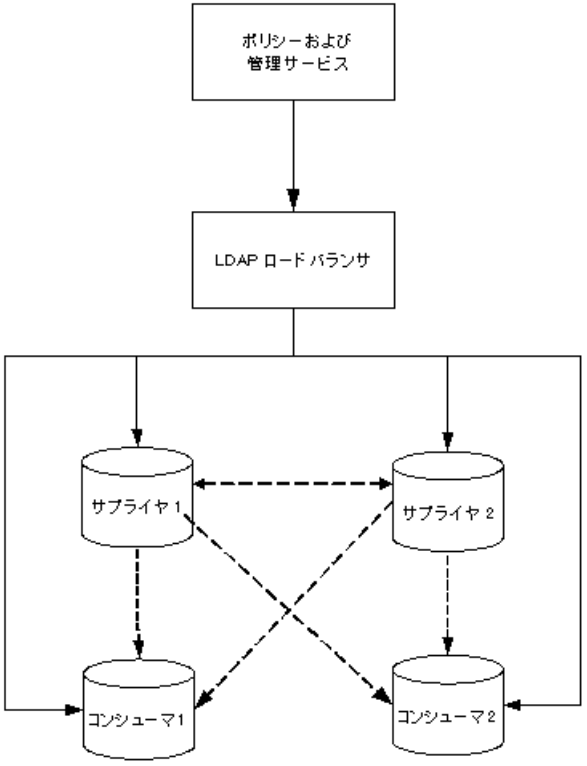
7. 次のコマンドを実行して、Identity Server を再起動します。

```
/Identity_Server_root/SUNWam/bin/amserver start
```

ロードバランサを使用する設定

図 5-6 に、Directory Proxy Server を含む複数サプライヤ設定を示します。この設定により、Identity Server によ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のサポートが最大限に活用されます。

図 5-6　ロードバランサを使用する複数サプライヤレプリケーション



LDAP ロードバランサを使用することで、Identity Server の基本機能に、高可用性レイヤーおよびディレクトリフェイルオーバー保護を追加できます。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 サーバーが利用できなくなった場合には、要求トラフィックの管理を続行し、クライアントクエリの拒否を開始します。ロードバランサのインストールを選択した場合、このアプリケーションを認識するように Identity Server を設定する必要があります。

1. 設定前に、以下の操作を行います。

   a. Directory Server をレプリケーション用に設定します。ディレクトリのレプリケーションに関する総合的な情報、および設定方法に関する詳細情報については、『Sun ONE Directory Server 管理者ガイド』の「レプリケーションの管理」を参照してください。

   b. LDAP ロードバランサをインストールおよび設定します。製品に同梱されているマニュアルの指示に従ってください。

2. /IdentityServer_base/SUNWam/lib/AMconfig.properties ファイルで、次のプロパティを修正し、手順 a 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させます。

   com.iplanet.am.directory.host

   com.iplanet.am.directory.port

3. 有効な Identity Server 認証モジュールごとに、手順 a でインストールしたコンシューマ 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。

   a. Identity Server コンソールで、「サービス設定」を選択します。

   b. 再設定する認証モジュールを見つけて、「プロパティ」の矢印をクリックします。

   c. 右側の区画で、以下の操作を行います。

      - 「LDAP サーバーとポート」という名前の最初のフィールドに、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。
      - 「LDAP サーバーとポート」という名前の 2 番目のフィールドには、何も入力しません。

   d. 「実行」をクリックします。

4. /IdentityServer_base/SUNWam/config/ums/serverconfig.xml ファイルで、コード例 5-2 に示すように、手順 a 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。

コード例 5-2 serverconfig.xml ロードバランサの変更

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

5. 次のコマンドを実行して、Identity Server を再起動します。

   /IdentityServer_base/SUNWam/bin/amserver start

注 ロードバランサを使用する Identity Server の設定方法の詳細は、付録 D 「ロードバランサの設定」を参照してください。

レプリケーションの警告

ディレクトリのレプリケーションを Identity Server 配備内に実装できない場合があります。たとえば、認証サーバーのホスト名または IP アドレスは同じでなければなりません。これにより、地理的に離れている、レプリカ Identity Server を使用することはできなくなります。リモートサーバーは、それぞれの LAN に対して単にローカルであるサーバーへの認証を実行することはできません。Directory Server のレプリケーションの計画および実装に関する総合的な情報については、『Sun ONE Directory Server 配備ガイド』を参照してください。

連携管理の実装

ここでは、Identity Server の連携管理機能を使用して配備設定を行う方法について説明します。このシナリオでは、ドメイン A、ドメイン B、およびドメイン C のそれぞれに 2 つの Identity Server インスタンス、および 1 つの Directory Server インスタンスが含まれるものとします。ドメイン A およびドメイン B は、それぞれサービスプロバイダ A およびサービスプロバイダ B を保持します。ドメイン C は、アイデンティティプロバイダ C を保持します。

注 Identity Server では、1 つのサーバーインスタンス内に複数のプロバイダを受け入れることが可能です。

この連携シナリオを適正に管理するため、理解する必要のある概念が 2 つあります。

- 「ホストプロバイダ」は、特定の Identity Server インスタンスを、サービスプロバイダとして動作するものと定義します。
- 「リモートプロバイダ」には、Identity Server のインスタンスが設定されているマシンとは別のマシンをホストとする任意のタイプのプロバイダに関連するデータが含まれます。これは、Identity Server のインスタンスの場合もあれば、そうでない場合もあります。また、対応する任意のサービスプロバイダまたはアイデンティティプロバイダである可能性があります。

このため、上記の定義済みシナリオでは、サービスプロバイダ A は、自らをホストプロバイダとして、サービスプロバイダ B およびサービスプロバイダ C をリモートプロバイダとして、それぞれ設定します。サービスプロバイダ B は、自らをホストプロバイダとして、サービスプロバイダ A およびサービスプロバイダ C をリモートプロバイダとして、それぞれ設定します。サービスプロバイダ C は、自らをホストプロバイダとして、サービスプロバイダ A およびサービスプロバイダ B をリモートプロバイダとして、それぞれ設定します。これらの設定が完了したら、各ドメイン内に認証ドメインを作成できます。

インストールされる製品のレイアウト

この章では、標準的なインストールを実行した場合の製品ソフトウェアのレイアウトについて説明します。次の節で構成されています。

Sun Java Enterprise System 2003Q4 ベースディレクトリ

Identity Server のインストール時にディレクトリが指定されていない場合、デフォルトの `/opt` が使用されます。Identity Server のインストールは Sun Java Enterprise System 2003Q4 インストーラを使用して行われるようになりました。このため、インストール用に選択したすべてまたはいずれかの製品を、このディレクトリにインストールすることが可能です。Identity Server を選択すると、デフォルトディレクトリに `SUNWam` という名前のサブディレクトリが含まれます。`SUNWam` には、Identity Server の共有バイナリファイルおよびコマンド行ツールがすべて含まれます。インストールされるサブディレクトリの完全なリストについては、『Sun Java Enterprise System 2003Q4 インストールガイド』を参照してください。

SUNWam ディレクトリ

Identity Server パッケージは、このディレクトリにインストールされます。この節では、各パッケージ名およびその簡潔な説明を示します。

注 ここでは、製品を Solaris オペレーティング環境にインストールした場合のディレクトリ構造を示します。その他のプラットフォームにインストールした場合には、ファイル名や拡張子が異なることもあります。Identity Server のインストール後に、特定のパッケージ用のインストール済みパス名の完全なリストを取得するには、pkgchk(1M) ユーティリティの pkgchk -v *package-name* を使用します。

SUNWam ディレクトリには、次のディレクトリおよびファイルが含まれます。

- agents/
- bin/
- config -> /etc/opt/SUNWam/config/
- console.war
- docs/
- dtd/
- ldaplib/
- ldif/
- lib/
- locale/
- migration/
- password.war*
- public_html/
- samples/
- services.war
- share/
- web-apps/

/opt/SUNWam/agents/

このディレクトリには、Identity Server ポリシーエージェントに固有のツール、ヘッダファイル、および設定ファイルが含まれます。このディレクトリに含まれるファイルの詳細については、『Web Policy Agents Guide』または『J2EE Policy Agents Guide』を参照してください。

/opt/SUNWam/bin/

このディレクトリには、Identity Server に同梱のコマンド行ツールが含まれます。これらのツールの詳細については、表 A-1 を参照してください。詳細は、『Sun ONE Identity Server 管理ガイド』の「コマンド行リファレンスガイド」の節を参照してください。

表 A-1 Identity Server のコマンド行ユーティリティ

ユーティリティ	説明
`VerifyArchive*`	下位互換性を維持するために 6.1 にインストールされる、amverifyarchive の推奨されないバージョン
`am2bak*`	Identity Server コンポーネントのバックアップを実行する場合に、このユーティリティを使用する
`amadmin*`	amadmin は、Directory Server への XML サービスファイルのロード、および DIT 上でのバッチ管理タスクの実行に使用される
`ampassword*`	このユーティリティは、Identity Server 管理者およびユーザーのパスワードセットを変更する場合に使用される
`amserver*`	amserver コマンド行ユーティリティは、Identity Server インスタンスの作成、起動、停止、および削除に使用される
`amverifyarchive*`	このユーティリティにより、ログアーカイブの検証が行われる。アーカイブの検証では、可能性のある改ざんや、アーカイブ内のファイルの削除が検出される
`bak2am*`	このユーティリティにより、am2back ユーティリティを使用してバックアップが作成された Identity Server コンポーネントが復元される
`ldapmodify*`	ldapmodify は、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する場合に使用される
`ldapsearch*`	ldapsearch は LDAP ディレクトリに検索要求を発行し、結果を LDIF テキストで表示する

/config ---> /etc/opt/SUNWam/config/

config ディレクトリは、/etc/opt/SUNWam/config へのシンボリックリンクです。このディレクトリには、表 A-2 で説明する XML サービスおよび設定ファイルが含まれます。通常、XML ファイルは設定には使用されません。これらのファイルを変更した場合、Directory Server データストアにファイルを手動で再ロードする必要があります (サーバーで行われた変更がこれらのファイルと同期されることはありません)。このディレクトリ内のすべての XML ファイルに関する情報については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

表 A-2　　XML サービスおよび設定ファイル

サブディレクトリ	内容
ums	• serverconfig.xml には、データストアとして使用される Sun ONE Directory Server に関する Sun™ ONE Identity Server の設定情報が含まれる • ums.xml の提供するテンプレートセットには、Identity Server を使用して管理されるアイデンティティ関連オブジェクトの LDAP 設定情報が含まれる

表 A-2　XML サービスおよび設定ファイル (続き)

サブディレクトリ	内容
xml	• amAdminConsole.xml • amAuth.xml • amAuthAnonymous.xml • amAuthCert.xml • amAuthConfig.xml • amAuthHTTPBasic.xml • amAuthLDAP.xml • amAuthMembership.xml • amAuthNT.xml • amAuthRadius.xml • amAuthSafeWord.xml • amAuthSecurID.xml • amAuthUnix.xml • amAuthenticationDomainConfig.xml • amClientData.xml • amClientDetection.xml • amEntrySpecific.xml • amDSS.xml • amG11NSettings.xml • amLogging.xml • amNaming.xml • amPasswordReset.xml • amPlatform.xml • amPolicy.xml • amPolicyConfig.xml • amProviderConfig.xml • amSAML.xml • amSession.xml • amUser.xml • amWebAgent.xml

/opt/SUNWam/console.war

console.war は、Identity Server コンソールアプリケーション関連のファイルを含む Web アプリケーションアーカイブ (WAR) です。このディレクトリ内のすべての WAR ファイルに関する情報については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

/opt/SUNWam/docs

docs ディレクトリには、API Javadoc に使用される HTML ファイルおよび関連するファイルが含まれます。以下に、このディレクトリに含まれるファイルのリストを示します。

- META-INF/
- allclasses-frame.html
- com/
- deprecated-list.html
- en_US/
- help-doc.html
- index-all.html
- index.html
- overview-frame.html
- overview-summary.html
- overview-tree.html
- package-list
- packages.html
- serialized-form.html
- stylesheet.css

/opt/SUNWam/dtd

dtd ディレクトリには、Identity Server で使用される DTD (Document Type Definition) ファイルがすべて含まれます。DTD は、Identity Server がアクセスする XML ファイルの構造を定義します。DTD ファイルの詳細は、『Sun ONE Identity Server Customization And API Guide』のサービス管理に関する章を参照してください。表 A-3 に、DTD ファイルのリストを示します。

表 A-3　Identity Server の DTD ファイル

ファイル	目的
Auth_Module_Properties.dtd	認証モジュールがプロパティの指定に使用する XML ファイルの構造を定義する
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する
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する。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ある
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する
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する
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 の構造を定義する
sms.dtd	XML サービスファイルの構造を定義する
web-app_2_2.dtd	Identity Server 導入コンテナが J2EE アプリケーションを導入する際に使用する XML ファイルの構造を定義する

/opt/SUNWam/ldaplib

ldaplib には、Identity Server に同梱の LDAP ユーティリティ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。表 A-4 に、これらのファイルのリストを示します。

表 A-4　共有オブジェクトファイル

サブディレクトリ	含まれるファイル
ldapsdk	• libicudata.so.2 • libicui18n.so.2 • libicuuc.so.2 • libldap50.so • libprldap50.so • libsasl.so • libssldap50.so

/opt/SUNWam/ldif

ldif ディレクトリには、Identity Server のインストール時に Directory Server データストアを生成するのに必要な LDIF ファイルが含まれます。表 A-5 に、このディレクトリに含まれる LDIF ファイルのリストを示します。

表 A-5　LDIF ファイル

ファイル	目的
ds_remote_schema.ldif	インストール時に、このファイルは、Identity Server のデータを Directory Server に格納するのに必要な Identity Server 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) をロードする
ds_remote_schema_uninstall.ldif	このファイルは、Directory Server から Identity Server 固有の LDAP スキーマオブジェクトクラスおよび属性を削除するのに使用される
sunone_schema2.ldif	インストール時に、このファイルは、Sun Microsystems 内部の Schema 2 ドキュメントで定義された Identity Server 固有の LDAP スキーマオブジェクトクラスおよび属性をロードする

/opt/SUNWam/lib

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。以下に、このディレクトリに含まれるファイルおよびサブディレクトリのリストを示します。

- AMConfig.properties
- AMConfig.properties.template
- LogConfig.properties
- SSOConfig.properties
- acmecrypt.jar*
- am_logging.jar
- am_sdk.jar
- am_services.jar
- am_sso_provider.jar
- commons-logging.jar
- crimsondeb.jar*
- dom.jar
- dom4j.jar
- endorsed サブディレクトリ。以下のファイルを含む
  - dom.jar
  - sax.jar
  - xalan.jar
  - xercesImpl.jar
  - xslyc.jar
- iaik_ssl.jar*
- jakarta-log4j-1.2.6.jar
- jax-qname.jar
- jaxm-api.jar
- jaxm-runtime.jar
- jaxp-api.jar
- jaxrpc-api.jar

- jaxrpc-ri.jar
- jdk_logging.jar
- jss サブディレクトリ
- libaceclnt.so
- libamsdk.so
- libamsdk.so.2
- libamutils.so
- libxm12.so
- libxm12.so.2
- mail.jar
- saaj-api.jar
- saaj-ri.jar
- sax.jar
- servlet.jar
- swec.jar*
- xalan.jar
- xercesImpl.jar
- xmlsec.jar
- xsltc.jar
- AMConfig.properties
- AMConfig.properties.template

/opt/SUNWam/locale

locale ディレクトリには、地域対応化プロパティファイルが含まれます。以下に、このディレクトリに含まれるファイルのリストを示します。各プロパティファイルには、それに対応する英語の地域対応化ファイルも含まれます (例 : amAdminCLI_en.properties)。

- amAdminCLI.properties
- amAdminConsole.properties
- amAdminModuleMsgs.properties
- amAuth.properties
- amAuthAnonymous.properties
- amAuthApplication.properties
- amAuthCert.properties
- amAuthConfig.properties
- amAuthContext.properties
- amAuthContextLocal.properties
- amAuthenticationDomainConfig.properties
- amAuthHTTPBasic.properties
- amAuthLDAP.properties
- amAuthMembership.properties
- amAuthNT.properties
- amAuthRadius.properties
- amAuthSafeWord.properties
- amAuthSecurID.properties
- amAuthUI.properties
- amAuthUnix.properties
- amClientData.properties
- amClientDetection.properties
- amEntrySpecific.properties
- amFederation.properties
- amG11NSettings.properties

- amLogging.properties
- amNaming.properties
- amPasswordReset.properties
- amPasswordResetModuleMsgs.properties
- amPlatform.properties
- amPll.properties
- amPolicy.properties
- amPolicyConfig.properties
- amProfile.properties
- amProviderConfig.properties
- amSAML.properties
- amSDK.properties
- amSession.properties
- amSSOProvider.properties
- amUser.properties
- amUtilMsgs.properties
- amWebAgent.properties
- getEncoding.class
- LC_MESSAGES/
- netscape/

/opt/SUNWam/migration

migration ディレクトリには、以前のバージョンの Identity Server からの移行に使用される PERL スクリプトが含まれます。以下に、Identity Server に含まれるファイルのリストを示します。移行データの詳細については、『Sun ONE Identity Server Migration Guide』を参照してください。

- `60to61/`
- `update-assignable-dynamic-groups.pl*`
- `update-filtered-groups.pl*`
- `update-groups.pl*`
- `update-o.pl*`
- `update-ou.pl*`
- `update-people.pl*`
- `update-static-groups.pl*`
- `update-users.pl*`

/opt/SUNWam/password.war

`password.war` は、Identity Server パスワードリセットアプリケーション関連のファイルを含む Web アプリケーションアーカイブ (WAR) です。このディレクトリ内のすべての WAR ファイルに関する情報については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

/opt/SUNWam/public_html

`public_html` ディレクトリには、オンラインヘルプに使用される HTML ファイルおよび関連するファイルが含まれます。

/opt/SUNWam/samples

samples ディレクトリには、表 A-6 に示すサブディレクトリが含まれます。これらのサブディレクトリには、対応する機能のサンプルが存在します。各サンプルの詳細については、『Sun ONE Identity Server Customization And API Guide』の関連する章を参照してください。

表 A-6　機能サンプル用のディレクトリ

サブディレクトリ名
Readme.html
admin/
appserver/
authentication/
console/
liberty/
policy/
saml/
sample.css
sso/
sunLogo.gif
um/

/opt/SUNWam/services.war

services.war は、以前に言及した WAR ファイルに含まれないすべての Identity Server サービス関連のファイルを含む Web アプリケーションアーカイブ (WAR) です。このディレクトリ内のすべての WAR ファイルに関する情報については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

/opt/SUNWam/share

share ディレクトリには、Identity Server 内部で使用される追加ユーティリティを含む bin/ サブディレクトリが存在します。

- /bin/ammultiserverinstall*
- /bin/amsecuridd*
- /bin/amserver.instance_template*
- /bin/amunixd*/bin/checkport*
- /bin/configds*
- /bin/unconfigds*
- /bin/wsutils.ksh*

/opt/SUNWam/web-apps

web-apps ディレクトリには、Web コンテナ上での Identity Server J2EE Web アプリケーションの導入先ディレクトリが含まれます。次のディレクトリおよびファイルが含まれます。

- applications/ は、Identity Server Console の導入先ディレクトリです。次のディレクトリおよびファイルが含まれます。
  - META-INF/
  - WEB-INF/
  - console
    - auth/
    - base/
    - css/
    - federation/
    - html/
    - images/
    - index.html
    - js/
    - policy/
    - service/

    - session/
    - user/
  - index.html
- introduction/ は、Identity Server Liberty Common Domain コンポーネントの導入先ディレクトリです。次のディレクトリおよびファイルが含まれます。
  - META-INF
  - WEB-INF
- introduction.war
- password/ は、Identity Server Password Synchronization コンポーネントの導入先ディレクトリです。
  - META-INF/
  - WEB-INF/
  - index.html
  - password/
- services/ は、Identity Server Core Service の導入先ディレクトリです。次のディレクトリおよびファイルが含まれます。
  - META-INF/
  - WEB-INF/
  - admin/
  - config/
  - css/
  - docs/
  - fed_css/
  - fed_images/
  - images/
  - index.html
  - js/
  - login_images/

ユーザーセッションのライフサイクル

Sun™ ONE Identity Server では、アクセス管理サービスを提供する際、複数の HTTP (HyperText Transfer Protocol) 要求にまたがる、ユーザーと Web アプリケーションとの対話処理の追跡に使用するセッションオブジェクトを作成できます。この章では、Identity Server コンポーネントのプロトコルのやり取りを追跡して、セッションのライフサイクルについて説明します。次の節で構成されています。

概要

以降のセクションでは、Web ブラウザを使用して保護されたリソースへのアクセスを要求するユーザーに認証および承認サービスを提供する際の、Identity Server コンポーネントのプロトコルのやり取りを追跡します。これにより、セッションのライフサイクルの様子を理解できます。

要求

最初に、認証されていないユーザーが保護されたリソースに対する要求を作成します。この要求はサーバーに送信されます。リソースは、ポリシーエージェントにより保護されています。コード例 5-3 に、ブラウザから送信される GET 要求を示します。

コード例 5-3 GET 要求ヘッダー

```
GET / HTTP/1.1
Host:application.sun.com:8089
```

Identity Server では、すべてのアクセス要求は、有効なセッショントークン (プログラムでは SSOToken) が存在することで明示的に許可されない限り、暗黙的に拒否されます。

注 この動作は、導入時の条件に基づいて逆にできます。

この場合、セッショントークンは提供されないため、ポリシーエージェントは認証サービスに要求をリダイレクトします。コード例 5-4 に、要求元のブラウザに返されたリダイレクト情報を示します。これには、認証サービスへの URI、および元の要求の URL を含む goto パラメータが含まれます。

コード例 5-4 リダイレクト情報に対する GET 応答

```
HTTP/1.1 302 Moved Temporarily
Location:
http://identityserver.sun.com:58081/amserver/UI/Login?goto=http%
3A%2F%2Fapplication.sun.com%3A8089%2Findex.html
```

HTTP で待機中のユーザーのブラウザによりリダイレクトが許可され、認証サービス URI への要求が実行されます。コード例 5-5 に、認証サービスに送信される GET 要求を示します。

コード例 5-5 認証サービスにリダイレクトされる GET 要求

```
GET
/amserver/UI/Login?goto=http%3A%2F%2Fapplication.sun.com%3A8089%
2Findex.html HTTP/1.1
Host: identityserver.sun.com:58081
```

認証

認証要求の受信時に、認証サービスは、Identity Server の設定および要求パラメータに基づいて、ユーザーに提供する認証モジュールを決定します。新規の認証要求すべてと同様、ユーザーのやり取りを追跡するため、セッションサービスにより無効なセッショントークンが作成されます。(このセッショントークンには、ユーザーを表すランダムに生成された文字列である暗号化されたセッション ID も含まれます。) セッショントークンは、Cookie (デフォルトでは `iPlanetDirectoryPro`) 内で設定されます。認証サービスは、認証要求に応答して、適切な証明情報をユーザーに求めるフォームと共にこれを送信します。

注 フォームのプロトコル (HTML、WML など) は、認証を要求するクライアントに基づいて、クライアントディテクションサービスにより決定されます。このサービスの詳細については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

コード例 5-6 に、ユーザーから認証証明情報を要求する HTML 認証フォームのヘッダーを示します。(HTML 自体は、簡略化のために削除されています。)

コード例 5-6 ユーザーに返される認証フォーム

```
HTTP/1.1 200 OK
Server: Sun-ONE-Web-Server/6.1
X-dsameversion: 6.1
Set-cookie:
JSESSIONID=DE271E3F2D52473B409DD8A7C58C24A5;Path=/amserver
Set-cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D;Domain=.sun.com;Path=/

<html>
Login Form...
</html>
```

ユーザーは、受信したフォームに認証証明情報を入力して、Identity Server に送信します。通常、この処理は、パスワード属性を保護するために SSL 経由で行われます。理解を容易にするため、コード例 5-7 で送信される証明情報はクリア HTTP 経由で送信されています。

コード例 5-7 Identity Server に返される POST 証明情報

```
POST /amserver/UI/Login HTTP/1.1
Host: identityserver.sun.com:58081
Cookie: JSESSIONID=DE271E3F2D52473B409DD8A7C58C24A5;
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D
Content-Type: application/x-www-form-urlencoded

IDToken1=user1&IDToken2=password
```

認証サービスにより受信された証明情報は、適切な認証モジュールにより検証されます。証明情報は検査に合格し、セッショントークンの状態は有効に変更され、セッション情報 (ログイン時刻、認証方式、認証レベルなど) は保存されるものとします。`iPlanetDirectoryPro` Cookie には、有効なセッショントークンが含まれるようになります。サーバーはブラウザに対し最初に要求されたリソースへのリダイレクトで応答します。コード例 5-8 に、このリダイレクト応答を示します。

コード例 5-8 最初に要求されたリソースへのリダイレクト

```
HTTP/1.1 302 Moved Temporarily
Server: Sun-ONE-Web-Server/6.1
X-autherrorcode: 0
X-dsameversion: 6.1
Location: http://application.sun.com:8089/index.html
```

HTTP で待機中のユーザーのブラウザにより、元のリソースへのリダイレクトが許可され、その後再度アクセスが要求されます。今回は、認証プロセス中に作成されたセッショントークンが、要求に含まれます。

コード例 5-9 トークンと共にリダイレクトされる GET 要求ヘッダー

```
GET /index.html HTTP/1.1
Host: application.sun.com:8089
Referer:
http://identityserver.sun.com:58081/amserver/UI/Login?goto=http%
3A%2F%2Fapplication.sun.com%3A8089%2Findex.html
Cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D
```

セッショントークン

ポリシーエージェントは、要求を再び遮断します。要求には Identity Server と同じ DNS ドメイン内のセッショントークンが含まれるようになったため、エージェントはこのトークンおよび関連するセッションの有効性の判定を試みます。最初に、セッションの出所を確認する必要があります。このため、ネーミングサービスとの通信が行われます。ネーミングサービスは、Identity Server が使用する内部サービスのサービス URL をクライアントが検索することを許可します。この情報は、セッションに関する通信に使用できます。ネームサービスは、セッションを暗号化して、対応する URL を返します。この URL を使用して、適用可能なサービスからセッションに関する情報が取得されます。コード例 5-10 に、ネーミング情報の POST 要求を示します。

コード例 5-10 ネーミング情報の POST 要求

```
POST /amserver/namingservice HTTP/1.0
Content-Type: text/xml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RequestSet vers="1.0" svcid="com.iplanet.am.naming" reqid="9">
<Request><![CDATA[
<NamingRequest vers="1.0" reqid="2"
sessid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=">
<GetNamingProfile>
</GetNamingProfile>
</NamingRequest>]]>
</Request>
</RequestSet>
```

コード例 5-11 に、ネーミング情報の POST 要求に対する応答を示します。属性名および対応する URL 値に注目してください。

コード例 5-11 ネーミング情報に関する応答

```
HTTP/1.1 200 OK
Content-type: text/html

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResponseSet vers="1.0" svcid="com.iplanet.am.naming" reqid="9">
<Response><![CDATA[<NamingResponse vers="1.0" reqid="2">
<GetNamingProfile>
<Attribute name="iplanet-am-naming-policy-url"
value="http://identityserver.sun.com:58081/amserver/policyservic
e"></Attribute>
<Attribute name="iplanet-am-naming-session-class"
value="com.iplanet.dpro.session.service.SessionRequestHandler"><
/Attribute>
<Attribute name="iplanet-am-naming-session-url"
value="http://identityserver.sun.com:58081/amserver/sessionservi
ce"></Attribute>
<Attribute name="iplanet-am-naming-samlawareservlet-url"
value="http://identityserver.sun.com:58081/amserver/SAMLAwareSer
vlet"></Attribute>
<Attribute name="serviceObjectClasses"
value="iplanet-am-naming-service"></Attribute>
<Attribute name="iplanet-am-naming-auth-url"
value="http://identityserver.sun.com:58081/amserver/authservice"
></Attribute>
<Attribute name="iplanet-am-naming-profile-class"
value="com.iplanet.dpro.profile.agent.ProfileService"></Attribut
e>
<Attribute name="iplanet-am-naming-samlassertionmanager-url"
value="http://identityserver.sun.com:58081/amserver/AssertionMan
agerServlet/AssertionManagerIF"></Attribute>
<Attribute name="iplanet-am-naming-umservice-url"
value="http://identityserver.sun.com:58081/amserver/UserManageme
ntServlet/"></Attribute>
<Attribute name="01"
value="http://identityserver.sun.com:58081"></Attribute>
<Attribute name="iplanet-am-naming-policy-class"
value="com.sun.identity.policy.remote.PolicyRequestHandler"></At
tribute>
<Attribute name="iplanet-am-naming-logging-class"
value="com.sun.identity.log.service.LogService"></Attribute>
<Attribute name="iplanet-am-naming-profile-url"
value="http://identityserver.sun.com:58081/amserver/profileservi
ce"></Attribute>
<Attribute name="iplanet-am-naming-samlsoapreceiver-url"
value="http://identityserver.sun.com:58081/amserver/SAMLSOAPRece
iver"></Attribute>
```

コード例 5-11 ネーミング情報に関する応答 (続き)

```
<Attribute name="iplanet-am-naming-logging-url"
value="http://identityserver.sun.com:58081/amserver/loggingservi
ce"></Attribute>
<Attribute name="iplanet-am-naming-fsassertionmanager-url"
value="http://identityserver.sun.com:58081/amserver/FSAssertionM
anagerServlet/FSAssertionManagerIF"></Attribute>
<Attribute name="iplanet-am-platform-server-list"
value="http://identityserver.sun.com:58081"></Attribute>
<Attribute name="iplanet-am-naming-samlpostservlet-url"
value="http://identityserver.sun.com:58081/amserver/SAMLPOSTProf
ileServlet"></Attribute>
<Attribute name="iplanet-am-naming-auth-class"
value="com.sun.identity.authentication.server.AuthXMLHandler"></
Attribute>
</GetNamingProfile>
</NamingResponse>]]></Response>
</ResponseSet>
```

ネーミングサービスにより提供される情報を使用して、ポリシーエージェントはセッションサービスへの POST 要求を作成して、含まれるセッショントークンを検証します。コード例 5-12 に、セッションサービスへの POST 要求を示します。

コード例 5-12 セッションサービスへのセッション検証用の POST 要求

```
POST /amserver/sessionservice HTTP/1.0
Host: identityserver.sun.com
Content-Type: text/xml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RequestSet vers="1.0" svcid="Session" reqid="10">
<Request><![CDATA[
<SessionRequest vers="1.0" reqid="4">
<GetSession reset="true">
<SessionID>AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=</Ses
sionID>
</GetSession>
</SessionRequest>]]>
</Request>
<Request><![CDATA[
<SessionRequest vers="1.0" reqid="5">
<AddSessionListener>
<URL>http://application.sun.com:8089/amagent/UpdateAgentCacheSer
vlet?shortcircuit=false</URL>
<SessionID>AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=</Ses
sionID>
</AddSessionListener>
```

コード例 5-12 セッションサービスへのセッション検証用の POST 要求 (続き)

```
</SessionRequest>]]>
</Request>
</RequestSet>
```

セッションサービスは、要求を受信し、セッショントークンの有効性をチェックします。セッションがタイムアウトしていることも、その他の理由で無効であることもないと仮定し、セッションサービスはセッションが有効であると応答します。このアサーションは、セッション自体のサポート情報と一体化しています。

注 セッションリスナも、セッションに対して登録されます。これにより、セッションの状態または有効性が変更されていることを、ポリシーエージェントが通知することが可能になります。

コード例 5-13 に、セッションサービスの応答を示します。

コード例 5-13 セッションの有効性を示すセッションサービス応答

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResponseSet vers="1.0" svcid="session" reqid="10">
<Response><![CDATA[<SessionResponse vers="1.0" reqid="4">
<GetSession>
<Session sid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
stype="user" cid="uid=user1,ou=people,dc=sun,dc=org"
cdomain="dc=sun,dc=org" maxtime="300" maxidle="120"
maxcaching="3" timeidle="0" timeleft="17993" state="valid">
<Property name="authInstant"
value="2003-10-02T01:38:23Z"></Property>
<Property name="clientType" value="genericHTML"></Property>
<Property name="CharSet" value="UTF-8"></Property>
<Property name="Locale" value="en_US"></Property>
<Property name="UserToken" value="user1"></Property>
<Property name="loginURL"
value="http://identityserver.sun.com:58081/amserver/UI/Login"></
Property>
<Property name="SessionHandle"
value="shandle:AQIC5wM2LY4SfcxlvOiI7LJwWcusZAdkM3CHmIoeehu6urc="
></Property>
<Property name="Host" value="192.168.1.100"></Property>
<Property name="AuthType" value="LDAP"></Property>
```

コード例 5-13　セッションの有効性を示すセッションサービス応答 (続き)

```
<Property name="Principals"
value="uid=user1,ou=People,dc=sun,dc=org|AQIC5wM2LY4SfcywFlTefDR
mqlthG54qrg27LiyS8LnHAj4="></Property>
<Property name="cookieSupport" value="true"></Property>
<Property name="Organization" value="dc=sun,dc=org"></Property>
<Property name="USERS_DN"
value="uid=user1,ou=people,dc=sun,dc=org"></Property>
<Property name="AuthLevel" value="0"></Property>
<Property name="UserId" value="user1"></Property>
<Property name="HostName" value="192.168.1.100"></Property>
<Property name="Principal"
value="uid=user1,ou=people,dc=sun,dc=org"></Property>
</Session></GetSession>
</SessionResponse>]]></Response>
<Response><![CDATA[<SessionResponse vers="1.0" reqid="5">
<AddSessionListener>
<OK></OK>
</AddSessionListener>
</SessionResponse>]]></Response>
</ResponseSet>
```

ポリシー

ユーザーの提供したセッショントークンが有効であることを示す応答を受け取ると、ポリシーエージェントは要求されたリソースへのアクセスをユーザーに許可できるかどうかを判定する必要があります。ポリシーエージェントは、HTTP ネームスペース内の一部のリソースに関する決定を求める、ポリシーサービスへの要求を作成します。この要求と共に、IP アドレスや DNS 名など、設定済みポリシーの条件セットに影響を与える可能性のある追加環境情報も含まれます。コード例 5-14 に、情報を求めてポリシーサービス URI に送信される POST 要求を示します。

コード例 5-14　ポリシー情報を求める POST 要求

```
POST /amserver/policyservice HTTP/1.0
Host: identityserver.sun.com
Content-Type: text/xml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RequestSet vers="1.0" svcid="Policy" reqid="11">
<Request><![CDATA[
<PolicyService version="1.0">
<PolicyRequest requestId="3"
appSSOToken="AQIC5wM2LY4SfczlhkwdQHfKgzxYu0qWY+DFCB9VGmknzvM=">
```

コード例 5-14 ポリシー情報を求める POST 要求 (続き)

```
<GetResourceResults
userSSOToken="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
serviceName="iPlanetAMWebAgentService"
resourceName="http://application.sun.com:8089/"
resourceScope="subtree">
<EnvParameters>
<AttributeValuePair>
<Attribute name="requestDnsName"/>
<Value>drnick</Value>
<Value>drnick.sun.com</Value>
</AttributeValuePair>
<AttributeValuePair>
<Attribute name="requestIp"/>
<Value>192.168.1.100</Value>
</AttributeValuePair>
</EnvParameters>
</GetResourceResults>
</PolicyRequest>
</PolicyService>]]>
</Request>
</RequestSet>
```

ポリシーサービスは、要求を受信した後で、要求に適用されるリソース定義を含むポリシーをチェックします。

注 ポリシーは、Identity Server 内にキャッシュされます。キャッシュされていない場合、ポリシーが最初に Directory Server からロードされます。

要求に適用されるポリシーが検出されると、ポリシーサービスは、セッショントークンにより識別されたユーザーがいずれかのポリシーサブジェクトのメンバーであるかどうかを確認します。リソースに適合するポリシーが検出され、かつユーザーが有効なサブジェクトである場合、ポリシーの追加条件 (日時は正確かどうか、適切なネットワークから送信されたものかどうかなど) が評価されます。すべての条件が満たされる場合、ポリシーサービスは、ユーザーにアクセスを許可できると判断し、ポリシーエージェントに許可の決定を伝えます。コード例 5-15 に、ユーザーが保護されたリソースにアクセス可能であることを確認する、ポリシーサービスからの応答を示します。

コード例 5-15 許可ポリシー応答

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

コード例 5-15　　許可ポリシー応答 (続き)

```
<ResponseSet vers="1.0" svcid="policy" reqid="11">
<Response><![CDATA[<PolicyService version="1.0">
<PolicyResponse requestId="3">
<ResourceResult name="http://application.sun.com:8089/">
<PolicyDecision>
<ActionDecision timeToLive="1065121515571">
<AttributeValuePair>
<Attribute name="POST"/>
<Value>allow</Value>
</AttributeValuePair>
<Advices>
</Advices>
</ActionDecision>
<ActionDecision timeToLive="1065121515571">
<AttributeValuePair>
<Attribute name="GET"/>
<Value>allow</Value>
</AttributeValuePair>
<Advices>
</Advices>
</ActionDecision>
</PolicyDecision>
</ResourceResult>
</PolicyResponse>
</PolicyService>]]>
</Response>
```

要求されたページ

ポリシーエージェントは、ポリシーサービスからの決定を受信したら、このアクセス情報に基づいて動作する必要があります。この場合、GET および POST 操作に対して許可決定が発行されます。これはユーザーが要求した操作と一致するため、ポリシーエージェントはアクセスを許可します。決定はセッショントークンと共にキャッシュされるため、後続の要求はキャッシュを使用してチェックできます。Identity Server との通信は必要ありません。管理者により定義された時間を過ぎるか、ポリシーまたはセッションの状態変更が明示的に通知されると、キャッシュは期限切れになります。ただし、ユーザーにアクセスが許可される前にアクションをログに記録して、監査トレールを維持する必要があります。ポリシーエージェントは、ログ要求をログサービスに発行します。コード例 5-16 にログ要求を示します。

コード例 5-16 ポリシーエージェントからのログ要求

```
POST /amserver/loggingservice HTTP/1.0
Host: identityserver.sun.com
Content-Length: 416
Accept: text/xml
Content-Type: text/xml; charset=UTF-8

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RequestSet vers="1.0" svcid="Logging" reqid="12">
<Request><![CDATA[
<logRecWrite reqid="2"><log logName="amAuthLog"
sid="AQIC5wM2LY4SfczlhkwdQHfKgzxYu0qWY+DFCB9VGmknzvM="></log><lo
gRecord><recType>Agent</recType><recMsg>User user1 was allowed
access to
http://application.sun.com:8089/index.html.</recMsg></logRecord>
</logRecWrite>]]></Request>
</RequestSet>
```

ログサービスは要求を受信し、設定に基づいて、要求を署名済みのファイル (オプション) または JDBC ストアに記録します。その後、応答はポリシーエージェントに返されて、ログが通知されます。コード例 5-17 に応答を示します。

コード例 5-17 エージェントにログを通知する応答

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResponseSet vers="1.0" svcid="iplanet.webtop.service.logging"
reqid="12">
```

コード例 5-17 エージェントにログを通知する応答 (続き)

```
<Response><![CDATA[OK]]></Response>
</ResponseSet>
```

これで、要求されたリソースへのアクセスがポリシーエージェントにより許可されました。コード例 5-18 に、要求されたリソースの HTML ページを示します。(HTML 自体は、簡略化のために削除されています。)

コード例 5-18 エージェントによる要求されたページへのアクセス許可

```
HTTP/1.1 200 Ok
Content-type: text/html

<HTML>
The requested page....
</HTML>
```

シングルサインオン要求

この節では、2 つのスレッドについて説明します。最初のスレッドは、認証されたユーザーが、同じ DNS ドメイン内の異なるサーバー上の、セッショントークンにより検証済みの保護されたリソースを要求した時点で発生します。この 1 番目のスレッドはシングルサインオン機能を使用します。2 番目のスレッドは、認証されたユーザーが、異なる DNS ドメイン内の異なるサーバー上の保護されたリソースを要求した時点で発生します。2 番目のスレッドは、クロスドメインのシングルサインオン機能を使用します。

スレッド 1: シングルサインオン

要求したページの受信後に、ユーザーは異なるサーバー上の保護されたリソースを要求します。コード例 5-19 は、この 2 番目の要求を示します。セッショントークンが要求に含まれている点に注目してください。これにより、シングルサインオンが可能になります。

コード例 5-19 有効なセッショントークンを含む 2 番目の要求

```
GET / HTTP/1.1
Host: webservice.sun.com:8090
Cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D
```

セッショントークンが存在するため、ポリシーエージェントはユーザー認証を取得する必要はありません。このため、この場合は、113 ページの「認証」で説明した手順は省略されます。ただし、エージェントは、以前のセッショントークンを (キャッシュされたエントリが存在しないため) 表示できません。このため、115 ページの「セッショントークン」、119 ページの「ポリシー」および 122 ページの「要求されたページ」に記載された手順を実行します。

注 以下に示す手順は、115 ページの「セッショントークン」、119 ページの「ポリシー」および 122 ページの「要求されたページ」に記載されている手順を大幅に修正したものです。

1. ネーミングサービスとの通信が行われます。コード例 5-20 に、Identity Server が使用する内部サービスの URL を求めるネーミングサービスへの要求を示します。ネーミングサービスは、セッションを暗号化し、対応する URL を返します。この URL を使用して、セッションデータの取得が行われます。

コード例 5-20 ネーミングサービスの POST 要求

```
POST /amserver/namingservice HTTP/1.0
Host: identityserver.sun.com

...
<NamingRequest vers="1.0" reqid="2"
sessid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=">
....

HTTP/1.1 200 OK

<ResponseSet vers="1.0" svcid="com.iplanet.am.naming" reqid="9">
....
</ResponseSet>
```

2. ネーミングサービスからの応答に基づき、適切なセッションサービスとの通信が行われます。コード例 5-21 に、セッション ID を含むセッションサービスに送信される要求を示します。

コード例 5-21 セッションサービスへの POST 要求

```
POST /amserver/sessionservice HTTP/1.0

...
<SessionID>AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=</Ses
sionID>
....

HTTP/1.1 200 OK

....
<Session sid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
stype="user" cid="uid=user1,ou=people,dc=sun,dc=org"
cdomain="dc=sun,dc=org" maxtime="300" maxidle="120"
maxcaching="3" timeidle="0" timeleft="17991" state="valid">
....
```

3. セッションサービスからの有効なセッション応答を想定し、ポリシーサービスに POST 要求が行われます。コード例 5-22 に、認証済みのユーザーに関するポリシー情報を求める、ポリシーサービスへの要求を示します。

コード例 5-22 ポリシーサービスへの POST 要求

```
POST /amserver/policyservice HTTP/1.0

...
<GetResourceResults
userSSOToken="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
serviceName="iPlanetAMWebAgentService"
resourceName="http://webservice.sun.com:8090/"
resourceScope="subtree">
...
```

4. この場合、保護されたリソースへのアクセスを許可するポリシーは検出されず、ポリシーサービスは適切な否定応答を返します。コード例 5-23 に、この応答を示します。

コード例 5-23　　ポリシーサービスからのアクセス否定応答

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResponseSet vers="1.0" svcid="policy" reqid="11">
<Response><![CDATA[<PolicyService version="1.0">
<PolicyResponse requestId="3">
<ResourceResult name="http://webservice.sun.com:8090/">
<PolicyDecision>
</PolicyDecision>
</ResourceResult>
</PolicyResponse>
</PolicyService>]]>
</Response>
</ResponseSet>
```

5. ポリシーエージェントでログが設定されているため、このアクセス拒否がポリシーエージェントによりログに記録されます。コード例 5-24 に、要求したリソースへのアクセスがユーザーに許可されなかったことを示すログの記録を求める、ログサービスへの要求を示します。

コード例 5-24　　ログサービスへの POST 要求

```
POST /amserver/loggingservice HTTP/1.0

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RequestSet vers="1.0" svcid="Logging" reqid="12">
<Request><![CDATA[
<logRecWrite reqid="2"><log logName="amAuthLog"
sid="AQIC5wM2LY4SfcyueSeNMEFDFRRLub8BfjaxeyDvTPXlVnA="></log><lo
gRecord><recType>Agent</recType><recMsg>User user1 was denied
access to
http://webservice.sun.com:8090/index.html.</recMsg></logRecord><
/logRecWrite>]]></Request>
</RequestSet>

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResponseSet vers="1.0" svcid="iplanet.webtop.service.logging"
reqid="12">
<Response><![CDATA[OK]]></Response>
</ResponseSet>
```

6. 次に、ポリシーエージェントは *Forbidden* メッセージをユーザーに発行します。コード例 5-25 に、このメッセージを表示する HTML ページを示します。ここで、アクセスが拒否されたことを示す、管理者の指定したページにユーザーをリダイレクトすることもできます。

コード例 5-25 アクセス拒否のメッセージを表示する HTML ページ

```
HTTP/1.1 403 Forbidden

<HTML><HEAD><TITLE>Forbidden</TITLE></HEAD>
<BODY><H1>Forbidden</H1>
Your client is not allowed to access the requested object.
</BODY></HTML>
```

スレッド 2: クロスドメインシングルサインオン

2 番目の要求ページへのアクセスが拒否されたため、ユーザーは別の DNS ドメイン内のサーバーに存在する保護されたリソースを要求します。Identity Server は、HTTP ドメイン Cookie (`http://www.ietf.org/rfc/rfc2965.txt`) を使用してアプリケーション間でトークンを転送するため、「スレッド 1: シングルサインオン」のように、要求を使ってポリシーエージェントにトークンが自動的に渡されることはありません。

1. トークンを新規 DNS ドメインに転送して、ドメイン内のアプリケーションから使用可能にするのは、Identity Server 内の CDSSO (Cross Domain Single Sign-On) プロトコルの役割です。コード例 5-26 に、別の DNS ドメイン内の保護されたアプリケーションへのアクセスをセッショントークンなしで求める要求を示します。

コード例 5-26 別の DNS ドメイン内の保護されたアプリケーションの GET 要求

```
GET /index.html?sunwMethod=GET HTTP/1.1
Host: webservice.java.com:8088
```

2. ポリシーエージェントは、セッショントークンが存在しないことを認識します。ただし、この場合、エージェントは CDSSO 用に設定されているため、リダイレクトは認証サービスではなく、セッションの転送に Liberty プロトコルを使用する CDSSO コントローラサービスに対して行われます。このため、リダイレクトには関連する Liberty パラメータが含まれます。コード例 5-27 に、ブラウザ経由で行われる、Liberty パラメータを含むリダイレクトを示します。

コード例 5-27 ブラウザ経由で行われる CDSSO コントローラサービスへのリダイレクト

```
HTTP/1.1 302 Moved Temporarily
Location:
http://identityserver.sun.com:58081/amserver/cdcservlet?goto=htt
p%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3
DGET&refererservlet=http%3A%2F%2Fwebservice.java.com%3A8088%2Fin
dex.html%3FsunwMethod%3DGET&RequestID=29980&MajorVersion=1&Minor
Version=0&ProviderID=http%3A%2F%2Fwebservice.java.com%3A8088%2Fa
magent&IssueInstant=2003-10-02T04%3A25%3A06Z&ForceAuthn=false&Is
Passive=false&Federate=false
```

3. HTTP で待機中のユーザーのブラウザにより、リダイレクトが許可されます。今回、プライマリドメイン内の Cookie であるため、要求にはセッショントークンが含まれます。コード例 5-28 に、ブラウザから CDSSO コントローラサービスへの、セッショントークンを含む HTTP リダイレクトを示します。

コード例 5-28 セッショントークンを含む、ブラウザからの HTTP リダイレクト

```
GET
/amserver/cdcservlet?goto=http%3A%2F%2Fwebservice.java.com%3A808
8%2Findex.html%3FsunwMethod%3DGET&refererservlet=http%3A%2F%2Fwe
bservice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&Request
ID=29980&MajorVersion=1&MinorVersion=0&ProviderID=http%3A%2F%2Fw
ebservice.java.com%3A8088%2Famagent&IssueInstant=2003-10-02T04%3
A25%3A06Z&ForceAuthn=false&IsPassive=false&Federate=false
HTTP/1.1
Host: identityserver.sun.com:58081
Cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D
```

4. CDSSO コントローラサービス内の CDC サーブレットはセッショントークンを受け取り、セッション情報の詳細を定めた Liberty Post プロファイル応答を作成して、ブラウザに返信します。コード例 5-29 に、この返信を示します。

コード例 5-29　　ブラウザへの POST 返信

```
HTTP/1.1 200 OK
Server: Sun-ONE-Web-Server/6.1
Date: Thu, 02 Oct 2003 01:38:28 GMT
Content-type: text/html
Pragma: no-cache
Transfer-encoding: chunked

<HTML>
<BODY Onload="document.Response.submit()">
<FORM NAME="Response" METHOD="POST"
ACTION="http://webservice.java.com:8088/index.html?sunwMethod=GE
T">
<INPUT TYPE="HIDDEN" NAME="LARES" VALUE="<ENCODED_LIBERTY_DOC>"/>
</FORM>
</BODY></HTML>
```

5. ユーザーのブラウザは、Body タグ onLoad に含まれる Action および Javascript に基づいて、Liberty ドキュメントを含むフォームをポリシーエージェントに自動的に送信します。コード例 5-30 に、ポリシーエージェントへのブラウザ POST を示します。

コード例 5-30　　ポリシーエージェントへのブラウザ POST

```
POST /index.html?sunwMethod=GET HTTP/1.1
Host: webservice.java.com:8088
Referer:
http://identityserver.sun.com:58081/amserver/cdcservlet?goto=htt
p%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3
DGET&refererservlet=http%3A%2F%2Fwebservice.java.com%3A8088%2Fin
dex.html%3FsunwMethod%3DGET&RequestID=29980&MajorVersion=1&Minor
Version=0&ProviderID=http%3A%2F%2Fwebservice.java.com%3A8088%2Fa
magent&IssueInstant=2003-10-02T04%3A25%3A06Z&ForceAuthn=false&Is
Passive=false&Federate=false
Content-Type: application/x-www-form-urlencoded

LARES=<ENCODED_LIBERTY_DOC>
```

6. ポリシーエージェントは、Liberty ドキュメントを受信して、ユーザーのセッション情報を抽出します。この時点で、ポリシーエージェントが通常のエージェントと同様の方法でセッションを検証する必要があります。このため、リソースへのアクセスを許可または拒否する前に、115 ページの「セッショントークン」、119 ページの「ポリシー」、および 122 ページの「要求されたページ」に記載された手順に従って処理が実行されます。

注 説明を簡略化するため、115 ページの「セッショントークン」、119 ページの「ポリシー」および 122 ページの「要求されたページ」の内容はここでは繰り返しません。

この場合、セッショントークンは有効であると判定され、ユーザーはアクセスが許可されます。ポリシーエージェントは、要求されたドキュメントをユーザーに渡し、新規 DNS ドメインの Cookie 内にセッショントークンを設定します。これで、新規ドメイン内のすべてのエージェントが Cookie を使用できるようになります。コード例 5-31 に、新規 DNS ドメイン Cookie セットを含む、返される HTML ページを示します。(HTML 自体は、簡略化のために削除されています。)

コード例 5-31 ユーザーにアクセスが許可された、新規 Cookie を含む HTML ページ

```
TTP/1.1 200 Ok
Set-cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcwwteЗpeDKZILScZ40uK%2FsU0I0WQQ
P6xXA%3D;Domain=.java.com;Path=/

<HTML>
...
</HTML>
```

セッションの終了

1. 認証、および SSO と CDSSO の実行が完了したため、ユーザーはセッションを終了できます。セッションは、アイドルや最大タイムアウト、またはユーザーの明示的なログアウトを条件に、管理者が終了できます。この場合は、ユーザーがサービスへのリンクをクリックしてログアウトします。コード例 5-32 に、ログアウトサービスへのアクセス要求を示します。

コード例 5-32 ログアウトサービスの GET 要求

```
GET /amserver/UI/Logout HTTP/1.1
Host: identityserver.sun.com:58081
Cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnH
Aj4%3D
```

2. ログアウトサービスは、ログアウト要求を受信し、ユーザーのセッションを破棄されたものとしてマークし、セッショントークンに無効な値を新たに設定し、成功を示すログアウトページをユーザーに返します。コード例 5-33 に、ログアウト成功後にユーザーに送信される HTML ページの詳細を示します。(HTML 自体は、簡略化のために削除されています。)

コード例 5-33 ユーザーに返される成功を示す HTML ページ

```
HTTP/1.1 200 OK
Server: Sun-ONE-Web-Server/6.1
Set-cookie:
iPlanetDirectoryPro=AQICv6c3Z1VfvgCgpLlaEqkqM70TLV24OTB1XkPa%2BL
tA8bXvC7c5XABU95Ta8UJi6dQZhXEUSgTRBWHXQ6kcxe8qgw%3D%3D;Domain=.s
un.com;Path=/

<title>Sun ONE Identity Server (Logout)</title>
...
```

3. ユーザーのセッション状態が変更されたため、セッションに関心を示したアプリケーションに通知するのはセッションサービスの役割になります。この場合は、各ポリシーエージェントはセッション通知用に設定されており、セッションが無効であることをエージェントに通知するドキュメントがポリシーエージェントごとに送信されます。これにより、セッションがキャッシュからフラッシュされます。

コード例 5-34　　セッション通知用の POST

```
POST /amagent/UpdateAgentCacheServlet?shortcircuit=false HTTP/1.1
Host: application.sun.com:8089

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NotificationSet vers="1.0" svcid="session" notid="8">
<Notification><![CDATA[<SessionNotification vers="1.0" notid="8">
<Session sid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
stype="user" cid="uid=user1,ou=people,dc=sun,dc=org"
cdomain="dc=sun,dc=org" maxtime="300" maxidle="120"
maxcaching="3" timeidle="3" timeleft="17983" state="destroyed">
<Property name="authInstant"
value="2003-10-02T01:38:23Z"></Property>
<Property name="clientType" value="genericHTML"></Property>
<Property name="CharSet" value="UTF-8"></Property>
<Property name="Locale" value="en_US"></Property>
<Property name="UserToken" value="user1"></Property>
<Property name="loginURL"
value="http://identityserver.sun.com:58081/amserver/UI/Login"></
Property>
<Property name="SessionHandle"
value="shandle:AQIC5wM2LY4SfcxlvOiI7LJwWcusZAdkM3CHmIoeehu6urc="
></Property>
<Property name="Host" value="192.168.1.100"></Property>
<Property name="AuthType" value="LDAP"></Property>
<Property name="Principals"
value="uid=user1,ou=People,dc=sun,dc=org|AQIC5wM2LY4SfcywFlTefDR
mqlthG54qrg27LiyS8LnHAj4="></Property>
<Property name="cookieSupport" value="true"></Property>
<Property name="Organization" value="dc=sun,dc=org"></Property>
<Property name="USERS_DN"
value="uid=user1,ou=people,dc=sun,dc=org"></Property>
<Property name="AuthLevel" value="0"></Property>
<Property name="UserId" value="user1"></Property>
<Property name="HostName" value="192.168.1.100"></Property>
<Property name="Principal"
value="uid=user1,ou=people,dc=sun,dc=org"></Property>
</Session>
<Type>5</Type>
<Time>1065058714469</Time>
</SessionNotification>]]></Notification>
</NotificationSet>


POST /amagent/UpdateAgentCacheServlet?shortcircuit=false HTTP/1.1
Host: webservice.sun.com:8090

....
```

コード例 5-34 セッション通知用の POST (続き)

```
<Session sid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
stype="user" cid="uid=user1,ou=people,dc=sun,dc=org"
cdomain="dc=sun,dc=org" maxtime="300" maxidle="120"
maxcaching="3" timeidle="3" timeleft="17983" state="destroyed">
...

POST /amagent/UpdateAgentCacheServlet?shortcircuit=false HTTP/1.1
Host: webservice.java.com:8088

...
<Session sid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4="
stype="user" cid="uid=user1,ou=people,dc=sun,dc=org"
cdomain="dc=sun,dc=org" maxtime="300" maxidle="120"
maxcaching="3" timeidle="3" timeleft="17983" state="destroyed">
...
```

4. セッション通知の受信後に、ポリシーエージェントはキャッシュからセッションをフラッシュします。これで、セッションのライフサイクルは終了します。

Active Directory に対する認証

Sun™ ONE Identity Server では、さまざまなバックエンドソースに対して認証を実行することが可能です。主に使用されるのは Sun ONE Directory Server および LDAP ですが、LDAP 認証モジュールを設定することで Microsoft Active Directory を認証ソースとして使用できます。この付録は、次の節で構成されています。

概要

Active Directory は LDAPv3 に準拠したディレクトリサーバーであるため、Identity Server を設定するだけで Active Directory を認証ソースとして認識させることが可能です。この設定を定義する際、管理者は、すべての LDAP 認証が Active Directory を指し示すようにするか、この目的に使用する認証モジュールを新規に作成して登録するかを選択できます。

注	LDAP v3 に準拠するすべてのサーバーに対し、LDAP 認証モジュールをここで説明する方法で使用できます。

既存の LDAP 認証モジュールを指し示す

この方法では、LDAP 認証モジュールを使用して認証を試みるすべてのユーザーが Active Directory で認証されます。Sun ONE Directory Server で検証が行われるデフォルトの LDAP 認証とは、この点が異なります。この付録では、このシナリオについて主に説明します。

Active Directory 認証モジュールを新規作成する

別のオプションは、Active Directory に対して LDAP 認証を実行する、専用の認証モジュールを作成および登録する方法です。Identity Server では、1 つの認証モジュールの複数インスタンスを異なる複数のサーバ設定で使用することはできません。この手法を採用する場合は、新規クラスが必要になります。このオプションは、Active Directory サフィックスと Directory Server サフィックスが同じでない場合に使用します。このオプションについて、ここでは扱いません。カスタム認証モジュールの作成方法については『Sun ONE Identity Server Customization And API Guide』を参照してください。

複数の LDAP サブ設定

管理者は、1 つの組織内に複数の LDAP 認証モジュール設定を定義できます。これらの追加設定はコンソールには表示されませんが、要求元のユーザーの認証が初期検索で見つからない場合に、プライマリ設定と連動して動作します。たとえば、ある組織で、2 つの異なるドメイン内で認証を検索できるように LDAP サーバーの検索を定義することも、1 つのドメイン内で複数のユーザーネーミング属性を設定することも可能です。後者の場合、コンソールにテキストフィールドが 1 つだけ表示されます。プライマリ検索条件を使用してユーザーが検出されない場合、LDAP モジュールは 2 番目のスコープを使用して検索を実行します。このオプションの詳細については、『Sun ONE Identity Server Customization And API Guide』を参照してください。

Active Directory 認証の設定

以下に、LDAP 認証モジュールを使用して Active Directory に対して証明情報を確認する手順を示します。

1. Identity Server をインストールおよび設定します。

   以下に、LDAP 認証モジュールのデフォルト設定を変更する手順を示します。これは、潜在的な危険性の伴う操作であり、ロックアウトが発生する可能性があります。Identity Server へのアクセスが拒否された場合は、「トラブルシューティング」を参照してください。

2. 組織のコア認証サービスの「ユーザープロファイル」で、「ダイナミックに作成」を選択します。

   Identity Server では、外部ソースに認証が委任されていても、Directory Server 内部に認証用のアカウントが存在する必要があります。これには、次のオプションが存在します。

   - メタディレクトリを使用して、アカウントを同期させる
   - ダイナミックプロファイル作成を有効にする。これにより、Identity Server は該当するユーザーのアカウントを検索できるようになる。該当するアカウントが存在しない場合は、Active Directory 内に同名のアカウントが自動的に作成される

注 この属性の詳細および有効化の方法については、『Sun ONE Identity Server 管理ガイド』を参照してください。

3. LDAP 認証モジュール内のプライマリ LDAP サーバーおよびポート属性を、Active Directory のホスト名およびポート番号に変更します。書式は、`hostname.domain.com:389` になります。

4. 「ユーザー検索の開始 DN」を、Active Directory 用の適切なベースに変更します。

   ベースサフィックスを指定するだけでは、この属性は機能しません。通常、`cn=Users` に `cn=Users,dc=domain,dc=com` 内のサフィックスを追加したものになります。

5. 「root ユーザーバインド DN」属性を、Active Directory の読み取り権限を持つユーザーに変更して、パスワードを更新します。

   Active Directory への匿名アクセスは許可されないため、バインドアカウントが必要になります。これは、ユーザーが認証する属性に対する読み取り機能を持つ Active Directory の単なるアカウントです。たとえば、cn=administrator,cn=Users,dc=domain,dc=com のようになります。このエントリのパスワードは、更新が必要です。

6. ユーザーネーミング属性を変更します。

   LDAP 認証モジュールは、この属性を Active Directory 内で検索し、検出した属性を使って Directory Server 内の UID を一致させます。たとえば、メタディレクトリにより 2 つのシステムの同期が行われ、Directory Server 内のエントリがメールを使用して RDN として格納される場合、Active Directory 内へのメールアドレスの格納時に userPrinicipalName をここに指定できます。この場合、`sAMAccountName` が Active Directory 内で最も UID に類似した属性であるため、これを指定するのが最善です。

7. 「ユーザーエントリ検索属性」を変更します。

   この属性は、Active Directory 内でのアカウントの検出に使用されます。これは、ユーザーがログインに使用する属性に対応している必要があります。たとえば、ユーザーが共通名を使用してログインする場合、この属性の値は cn になります。この場合、`sAMAccountName` が Active Directory 内で最も UID に類似した属性であるため、これを指定するのが最善です。

注 この属性には複数の値を含めることができ、それぞれの値を使用して実行が試みられます。

---

8. 「認証において ユーザー DN を返す」を選択解除します。

   この属性により、LDAP 認証モジュールは認証した DN を Identity Server に返すため、DN を再度検索して承認する手間を省くことができます。Active Directory の DN は、Directory Server の DN と同じではないため、オフにする必要があります。これにより、Identity Server が Active Directory から返される値を手順 6 の「ユーザーネーミング属性」で指定された属性として取得し、コア認証で指定された属性 (デフォルトでは `uid`) を使用して Directory Server を検索することが可能になります。この例では、LDAP 認証は Active Directory で `sAMAccountName=`*UID_entered_by_user* を検索し、認証後に Directory Server で Active Directory から返された `uid=sAMAccountName` を検索します。

9. `amAdmin` アカウントを Active Directory に追加します。

   `amAdmin` が Identity Server へのログインを続行できるように、`amAdmin` の `sAMAccountName` を使用してアカウントを作成する必要があります。このユーザーは、次のいずれかの方法で Active Directory 内で作成されます。

   a. 「スタート」->「プログラム」->「管理ツール」で、「Active Directory ユーザーとコンピュータ」を選択します。

   b. 左の区画で「ユーザー」ノードを右クリックして、「新規」->「ユーザー」を選択します。

   c. 適切な情報を入力し、アカウント名に `amAdmin` を指定します。

10. `amAdmin` が Identity Server にログインできることを確認します。

11. Active Directory で定義されたユーザーが Identity Server にログインできることを確認します。

これで、Identity Server が Active Directory での認証用に設定されました。

トラブルシューティング

LDAP 認証情報の変更により、ロックアウトが発生する可能性があります。以下に、この問題に対処する方法を示します。

Identity Server へのクイックアクセス

`AMConfig.properties` の `com.iplanet.authentication.super.user` プロパティ内で検出された DN を使用すると、`amAdmin` が LDAP 認証モジュールを介して Identity Server にログインできます。`amAdmin` は、Directory Server の Identity Server マネージャです。このプロパティの値は、`amAdmin` アカウントの完全な DN である、`uid=amAdmin,ou=People,`*root-suffix* です。完全な DN は、「ユーザー名」フィールドに入力します。この場合、`AMConfig.properties` で設定された Directory Server のインスタンスが使用されます。LDAP 認証モジュールパラメータは使用されません。

Directory Server を使用した再設定

認証設定情報は Directory Server に格納されるため、エントリは簡単に変更できます。`ou=default,ou=OrganizationConfig,ou=1.0,ou=iPlanetAMAuthLDAPService,ou=service,`*root-suffix* は、認証設定サービスを定義するオブジェクトです。iplanetKeyValue 属性を変更して、発生したすべてのエラーを訂正します。該当する値は次のとおりです。

- iplanet-am-auth-ldap-server
- iplanet-am-auth-ldap-base-dn
- iplanet-am-auth-ldap-user-naming-attribute
- iplanet-am-auth-ldap-user-search-attributes
- iplanet-am-auth-ldap-return-user-dn
- iplanet-am-auth-ldap-bind-dn
- iplanet-am-auth-ldap-bind-passwd

注	このプロパティの値は、ampassword ユーティリティを使用して暗号化されます。

付録 D

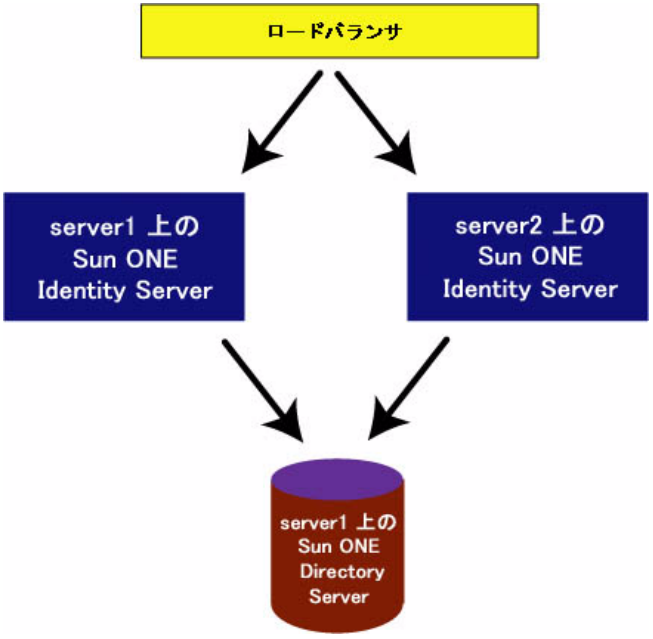
ロードバランサの設定

Sun ONE Identity Server は、ロードバランサで使用できるように設定できます。この章では、ロードバランスの機能およびロードバランスを実現する方法について説明します。次の節で構成されています。

ロードバランサの概要

ロードバランスを使用すると、通常は 1 つのサーバーで実行される作業を複数のサーバー間で配分することで、同一時間内により多くの作業を処理できるようになります。一般に、これはすべてのユーザーの要求がより高速に処理されることを意味します。ロードバランスは、ハードウェア、ソフトウェア、またはその組み合わせで実現されます。図 5-7 に、ロードバランサで使用できるように Identity Server の配備を設定する方法を示します。この設定では、Identity Server のすべてのインスタンスが同一の Directory Server を共有することが重要です。設定が完了すると、ロードバランサ (およびすべての Identity Server サービス) が URL `http://`*loadbalancer_host.domain*`:`*port*`/amconsole` 経由でアクセスされます。

図 5-7　Identity Server をロードバランサで使用する場合の設定



スティッキーセッション

ロードバランサを Identity Server に配備する場合、ロードバランサが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により、指定されたサーバーによるセッションの作成後に、セッション情報を保持するため、ユーザーからの後続の要求が同一のサーバーに引き続き配信されます。Identity Server は Cookie を使用してセッション情報を中継するため、ロードバランサはセッションを作成したサーバーへのリダイレクトを実行する必要があります。スティッキーセッションが存在しない場合、すべてのサーバーを信頼する必要があるため、パフォーマンスが低下します。

Resonate Central Dispatch のインストール

Resonate Central Dispatch は、ソフトウェアベースのロードバランサです。Identity Server をロードバランサで使用できるように設定する最初のステップは、ロードバランサのインストールです。2 つの物理サーバーが存在することを前提として、マシンが同一のサブネットに存在することを確認します。machine1 に Sun ONE Web Server、Sun ONE Directory Server、および Identity Server をこの順番でインストールし、Identity Server のインスタンスがインストール済みの Directory Server インスタンスを指し示すようにします。machine2 に Sun ONE Web Server および Identity Server をインストールし、Identity Server のインスタンスが server1 にインストール済みの Directory Server インスタンスを指し示すようにします。Central Dispatch ソフトウェアは、次のようにインストールします。

- machine1 に CDNode、CDMaster、および CDAction が含まれる
- machine2 に CDNode、CDAdaptor、および CDAction が含まれる

インストール処理中に、Reporter Agent が両方のマシンに自動的にインストールされます。表 5-1 に、設定手順で使用される可能性のある Central Dispatch に固有の用語を示します。

表 5-1 Resonate Central Dispatch の定義済みの用語

Central Dispatch の用語	定義
CDMaster	Central Dispatch Master は、単一 (または複数) の Central Dispatch サイトの管理および監視に使用されるグラフィカルユーザーインタフェース。Central Dispatch の設定はすべて、このコンソールを使用して適用される
ノード	CDMaster コンソールを介してノードとして設定された Identity Server のインスタンス。ノードは、スケジューラまたはサーバーとして設定可能
CDAdapter	Central Dispatch Adapter は、単一の Central Dispatch サイトと CDMaster 間のリンクを提供するプロキシ
CDAction	CDAction は、Central Dispatch サイトの設定、監視、および管理に使用されるコマンド行ユーティリティ

Central Dispatch のインストールおよび製品全般について詳しくは、ソフトウェアに同梱のドキュメントセットを参照してください。

ロードバランサの設定

「スティッキーセッション」は、setcookie 関数またはロードバランサ Cookie を使用して実装できます。以降の節で、両方のオプションに合わせてロードバランサを設定する手順を示します。説明する手順は Resonate Central Dispatch ロードバランサに適用されますが、任意のロードバランサソフトウェアで動作するように修正することも可能です。

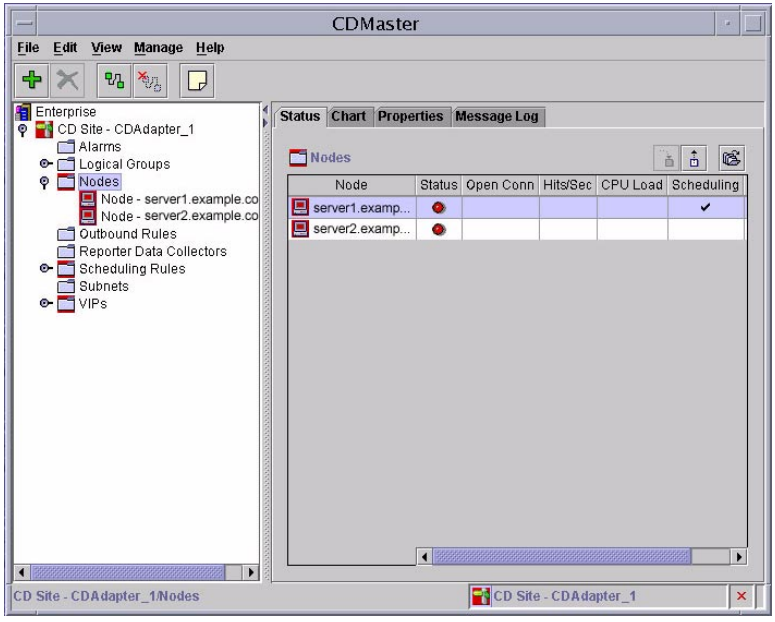
Central Dispatch を setcookie 用に設定する

1. admintool を使用して、2 つの Solaris ユーザー (cdadmin および cdmon) を作成します。
2. server1 上で CDMaster コンソールを起動します。

   デフォルトディレクトリ (/usr/local/resonate/cd/cdmaster/bin) に移動して、./cdmaster を実行します。指示に従って、machine2 にインストールされた CDAdapter に接続します。
3. CDMaster の左側のフレーム内の「Nodes」をクリックし、インストール済みの 2 つの Identity Server インスタンスのそれぞれに対して 1 つのノードを作成します。

   図 5-8 に、この手順の実行時の CDMaster コンソールのスクリーンショットを示します。

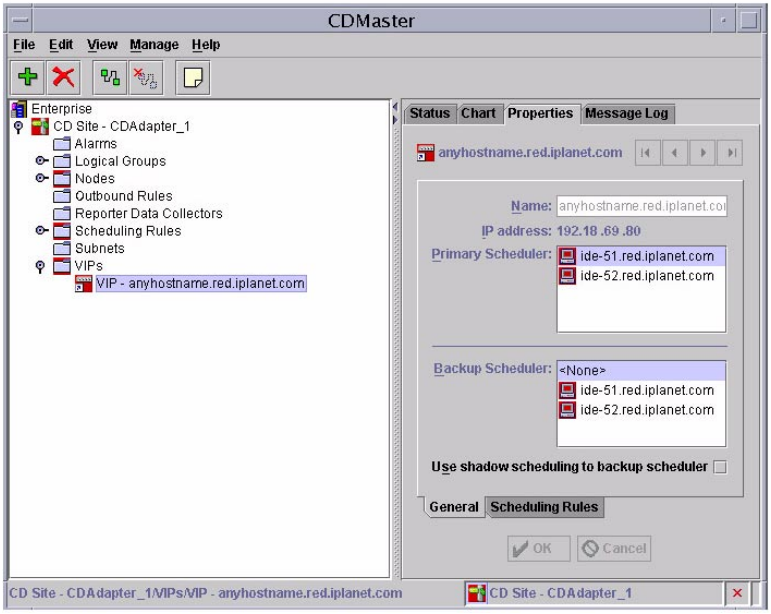
図 5-8　　Resonate を使用したノードの作成



4. CDMaster の左側のフレーム内で「VIPs」をクリックし、ロードバランサのインストール先ホストの新規仮想 IP アドレスを作成します。

   図 5-9 に、この手順および後続の手順を実行する際の CDMaster コンソールのスクリーンショットを示します。Primary Scheduler および Backup Scheduler が正しく設定されていることを確認してください。

図 5-9　　仮想 IP アドレスを新規作成する



5. 「VIPs」の右側のフレーム内の「Scheduling Rules」をクリックして「HTTP」を選択し、次のように HTTP スケジューリングルールを設定します。

   a. 「Properties」タブで、ロードバランサのインストール先ホストが仮想 IP として表示されていることを確認します。

   b. VIP ポートが、仮想 IP として定義されているものと同じであることを確認します。

   c. 「Resource」として「URL」を選択します。

6. CDMaster の左側のフレームの「Scheduling Rules」で、「HTTP」をクリックします。

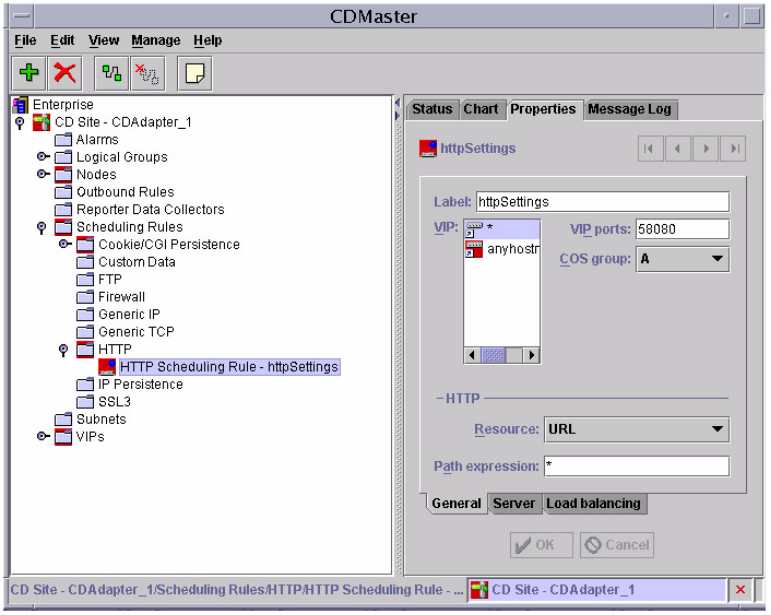
   147 ページの図 5-10 に、この手順および後続の手順を実行する際の CDMaster コンソールのスクリーンショットを示します。

7. 右側のフレーム下部の「Server」タブを選択します。

   サーバーが選択されていることを確認します。

8. 右側のフレーム下部の「Load Balancing」タブをクリックし、「Round Robin (Basic)」を選択します。

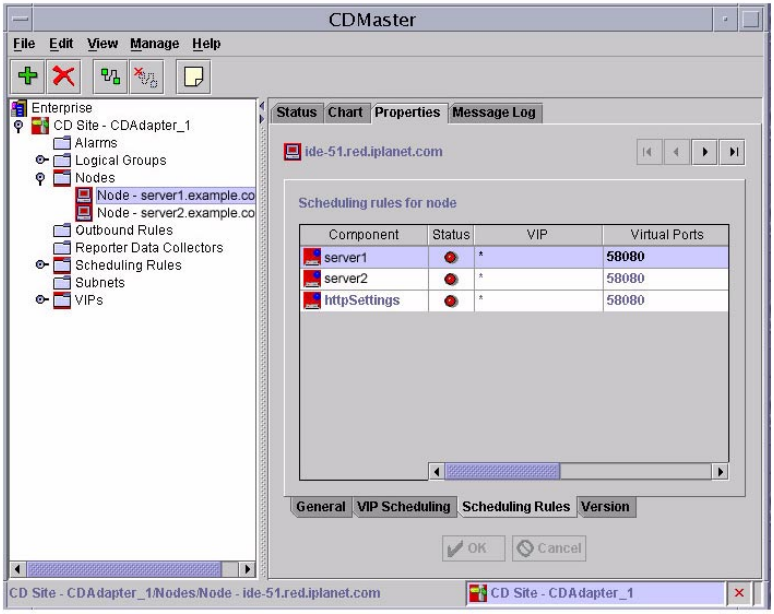
図 5-10　　HTTP スケジューリングルールを設定する



9. CDMaster の左側のフレームの「Nodes」をクリックし、Identity Server の 2 番目のインスタンスである server2 の設定済みノードを選択します。

   図 5-11 に、この手順および後続の手順を実行する際の CDMaster コンソールのスクリーンショットを示します。

図 5-11　CDMaster でノードを設定する

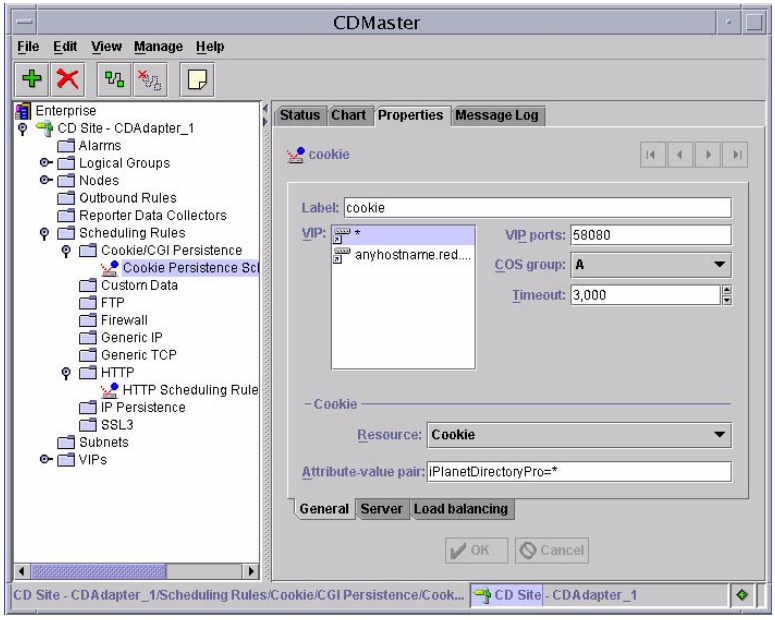


10. 右側のフレーム上部の「Properties」をクリックし、エイリアスが `server2.example.com` であり、`Server Enabled` および `Server auto enabled` が選択されていることを確認します。
11. 右側のフレーム下部の「VIP Scheduling」タブをクリックして、プライマリ仮想 IP がロードバランサのインストール先ホストに設定されていることを確認します。
12. 右側のフレーム下部の「Scheduling Rules」タブを選択して、すべてのサーバーおよびポートが「Component」に表示されることを確認します。
13. server1 に対して、147 ページの手順 9 から上記の手順 12 までを繰り返します。

    説明した手順に従って Identity Server の最初のインスタンス (server1) を設定します。ただし、手順 11 は、server2 を「Scheduling Rules」で「Primary」として設定する手順であるため、省略します。
14. CDMaster の左側のフレームの「Scheduling Rules」をクリックします。

    図 5-12 に、この手順および後続の手順を実行する際の CDMaster コンソールのスクリーンショットを示します。

図 5-12　Cookie Persistence Scheduling Rule を設定する



15. 「Cookie/CGI Persistence」を選択して、Cookie Persistence Scheduling Rule を作成します。

    属性と値のペアが、`iPlanetDirectoryPro=*` として定義されます。

16. ルールにラベルを設定し、仮想 IP 用の適切なポートが定義されていることを確認します。

    VIP リストには、ロードバランサがインストールされた、設定済みのホストが含まれていなければなりません。

17. 右側のフレーム下部の「Server」タブを選択し、両方のサーバーにチェックが付けられていることを確認します。

18. 右側のフレーム下部の「Load Balancing」タブを選択し、「Round Robin (Basic)」を選択します。

これで、`setcookie` 用の Central Dispatch の設定が完了しました。引き続いて次の節「Identity Server を setcookie 用に設定する」に進み、配備を完成させてください。

Identity Server を setcookie 用に設定する

setcookie の使用時にロードバランサを認識できるように、server1 および server2 用の Identity Server の設定を更新する必要があります。

1. server1 にインストールされた Identity Server インスタンスに、amadmin としてログインします。
2. ロードバランサがインストールされたホストマシンの値を、「組織のエイリアス」属性に追加します。

   「アイデンティティ管理」タブで最上位の組織を表示して、「組織のエイリアス」属性を見つけます。
3. server2 を、「サービス設定」タブの「プラットフォーム」内にある「サーバーリスト」属性に追加します。
4. `AMConfig.properties` で fqdnMap プロパティを設定します。

警告 この手順を、Identity Server コンソールを使用して実行することはできません。

   デフォルトでは、fqdnMap プロパティはコメントアウトされています。# を削除し、このプロパティを `com.sun.identity.server.fqdnMap[`*loadbalancer_host.domain*`]=`*loadbalancer_host.domain* として設定してください。
5. server1 および server2 を再起動します。

設定が正しく行われたことを確認する方法については、154 ページの「設定を確認する」を参照してください。

ロードバランサ Cookie の使用に合わせて Central Dispatch を設定する

1. admintool を使用して、2 つの Solaris ユーザー (cdadmin および cdmon) を作成します。
2. server1 上で CDMaster コンソールを起動します。

   デフォルトディレクトリ (/usr/local/resonate/cd/cdmaster/bin) に移動して、./cdmaster を実行します。指示に従って、server2 にインストールされた CDAdapter に接続します。
3. CDMaster の左側のフレーム内の「Nodes」をクリックし、インストール済みの 2 つの Identity Server インスタンスのそれぞれに対して 1 つのノードを作成します。
4. CDMaster の左側のフレーム内で「VIPs」をクリックし、ロードバランサのインストール先ホスト新規仮想 IP アドレスを作成します。

   Primary Scheduler および Backup Scheduler が設定されていることを確認してください。
5. 「VIPs」の右側のフレーム内の「Scheduling Rules」をクリックして「HTTP」を選択し、次のように HTTP スケジューリングルールを設定します。
   a. 「Properties」タブで、ロードバランサのインストール先ホストが仮想 IP として表示されていることを確認します。
   b. VIP ポートが、仮想 IP として定義されているものと同じであることを確認します。
   c. 「Resource」として「URL」を選択します。
6. CDMaster の左側のフレームの「Scheduling Rules」で、「HTTP」をクリックします。
7. 右側のフレーム下部の「Server」タブを選択します。

   サーバーが選択されていることを確認します。
8. 右側のフレーム下部の「Load Balancing」タブをクリックし、「Round Robin (Basic)」を選択します。
9. CDMaster の左側のフレームの「Nodes」をクリックし、Identity Server の 2 番目のインスタンスである server2 の設定済みノードを選択します。
10. 右側のフレーム上部の「Properties」をクリックし、エイリアスが server2.example.com であり、Server Enabled および Server auto enabled が選択されていることを確認します。
11. 右側のフレーム下部の「VIP Scheduling」タブをクリックして、プライマリ仮想 IP がロードバランサのインストール先ホストに設定されていることを確認します。

12. 右側のフレーム下部の「Scheduling Rules」タブを選択して、すべてのサーバーおよびポートが「Component」に表示されることを確認します。
13. server1 に対して、147 ページの手順 9 から上記の手順 12 までを繰り返します。

    説明した手順に従って Identity Server の最初のインスタンス (server1) を設定します。ただし、手順 11 は、server2 を「Scheduling Rules」で「Primary」として設定する手順であるため、省略します。
14. CDMaster の左側のフレームの「Scheduling Rules」をクリックします。
15. 「Cookie/CGI Persistence」を選択して、Cookie Persistence Scheduling Rule を server1 用に 1 つ、server2 用に 1 つ、合計 2 つ作成します。
16. server1 にラベルを設定し、仮想 IP 用の適切なポートが定義されていることを確認します。

    VIP リストには、ロードバランサがインストールされた、設定済みのホストも含まれていなければなりません。
17. 「Resource」として `cookie` を選択し、`Attribute-value pair` を `server1=server1` として定義します。
18. 右側のフレーム下部の「Server」タブを選択し、両方のサーバーが選択されていることを確認します。
19. 右側のフレーム下部の「Load Balancing」タブを選択し、「Round Robin (Basic)」を選択します。
20. server2 に対して、手順 14 から手順 19 までを繰り返します。

これで、ロードバランサ Cookie 用の Central Dispatch の設定が完了しました。引き続いて次の節「ロードバランサ Cookie に合わせて Identity Server を設定する」に進み、配備を完成させてください。

ロードバランサ Cookie に合わせて Identity Server を設定する

ロードバランサを認識できるように、server1 および server2 用の Identity Server の設定を更新する必要があります。

1. server1 にインストールされた Identity Server インスタンスに、amadmin としてログインします。
2. ロードバランサがインストールされたホストマシンの値を、「組織のエイリアス」属性に追加します。

   「アイデンティティ管理」タブで最上位の組織を表示して、「組織のエイリアス」属性を見つけます。
3. server2 を、「サービス設定」タブの「プラットフォーム」内にある「サーバーリスト」属性に追加します。
4. `AMConfig.properties` で fqdnMap プロパティを設定します。

警告 この手順を、Identity Server コンソールを使用して実行することはできません。

   デフォルトでは、fqdnMap プロパティはコメントアウトされています。# を削除し、このプロパティを `com.sun.identity.server.fqdnMap[`*loadbalancer_host.domain*`]=`*loadbalancer_host.domain* として設定してください。
5. 次のプロパティを、server1 および server2 上の `AMConfig.properties` にそれぞれ追加します。
   a. server1 上の Cookie の名前および値を、次のように設定します。

      ```
      com.iplanet.am.lbcookie.name=server1
      com.iplanet.am.lbcookie.value=server1
      ```

   b. server2 上の Cookie の名前および値を、次のように設定します。

      ```
      com.iplanet.am.lbcookie.name=server2
      com.iplanet.am.lbcookie.value=server2
      ```
6. server1 および server2 を再起動します。

設定を確認する

次の手順で、設定が適正であることを確認します。

警告 以下の手順を実行する前に、Sun ONE Web Server の Web コンテナの `keepAliveTimeout` オプションを無効に設定してください。

1. コンソールの「Manage」タブ内の「Start」を選択して、CDMaster を起動します。
2. 複数の新規ユーザーを作成し、作成したユーザーでログインします。
3. Web ブラウザの「Location」バーに、`http://`*loadbalancer_host*`.`*domain*`:`*port*`/amconsole` と入力します。
4. Identity Server に amadmin でログインし、「現在のセッション」タブを選択します。

   amadmin として、作成したユーザーおよび対応するサーバーが表示可能になります。ユーザーは、セッションを開始したサーバーにすべてをリダイレクトする必要があります。これは、Web サーバーのアクセスログを使用しても確認できます。

RADIUS サーバーに対する認証

Sun™ ONE Identity Server は、RADIUS (Remote Authentication Dial-In User Service) サーバーに対してユーザーを認証できます。ここでは、その導入方法を説明します。次の節で構成されています。

概要

RADIUS は、認証および承認サービスの提供に使用される業界標準のプロトコルです。この種の認証では、クライアントである Identity Server は RADIUS 形式のメッセージを RADIUS サーバーに送信します。RADIUS サーバーは要求を認証および承認し、応答を RADIUS 形式で返します。

RADIUS サーバーの設定

管理者は、次の手順を実行し、RADIUS サーバーに対して Identity Server の認証をテストできます。

1. 認証のテストに使用するユーザーのエントリを RADIUS サーバーに追加します。

   次のユーザー情報を *RADIUS_install*/etc/ raddb/users に追加する必要があります。*Login-Host* は、Identity Server が稼働中のマシンのホストおよびドメインです。

コード例 5-35 RADIUS ユーザーのエントリ

```
"Sample_User1" Password == "Password"
User-Service-Type = Login-User,
Login-Host = identity_server_host.domain_name,
Login-Service = PortMaster
```

2. Identity Server の完全修飾ドメイン名 (FQDN) または IP アドレスを RADIUS サーバーに追加します。

   このクライアント情報は、*RADIUS_install*/etc/raddb/clients に追加されます。定義済みの共有「シークレット」も追加されることを確認してください。

コード例 5-36 RADIUS クライアントのエントリ

```
191.18.18.111           <secret>
ms.red.example.com       <secret>
```

3. *RADIUS_install/sbin* ディレクトリに移動し、次のコマンドを使用して RADIUS サーバーを再起動します。

   ./radiusd &.

Identity Server の設定

1. Identity Server に `amAdmin` としてログインします。
2. 最上位レベルの組織に移動します。
3. ナビゲーションフレームの「表示」ドロップダウンメニューから「サービス」を選択します。
4. RADIUS が登録された認証サービスではない場合、「登録」をクリックします。

   RADIUS が登録済みの場合、手順 6 に進みます。
5. データフレームから「RADIUS」を選択して、「登録」をクリックします。
6. ナビゲーションフレームで RADIUS プロパティの矢印をクリックします。

   テンプレートが作成されていない場合、作成します。
7. RADIUS サーバーの完全修飾ドメイン名または IP アドレスを、「RADIUS サーバー 1」フィールドに追加します。
8. 156 ページの「RADIUS サーバーの設定」の手順 2 で使用した共有シークレットを入力します。
9. RADIUS サーバーのポート番号を入力し、テンプレートの変更を保存します。

   デフォルトは 1645 です。
10. ナビゲーションフレームで、「コア」プロパティの矢印をクリックします。
11. 「組織認証モジュール」リストで「RADIUS」を選択し、変更を保存します。

警告 手順 11 で RADIUS を選択する際、LDAP の選択を解除しないようにしてください。

12. Identity Server コンソールからログアウトします。
13. URL `http://identity_server_host.domain_name:port/service_deploy_uri/UI/Login?module=RADIUS` で、`Sample_User1` としてログインします。

chroot 環境のインストール

この付録では、chroot 環境で Sun™ ONE Identity Server をインストールおよび実行する方法について説明します。この付録は、次の節で構成されています。

概要

Chroot は、UNIX の change root 関数に由来します。Identity Server を chroot 環境にインストールすると、指定されたディレクトリがルートディレクトリになります。新規 chroot ディレクトリが、Identity Server のあらゆる検索で使用されるパスの基点になります。chroot は、悪質なプログラムが実際のルートファイルシステムにアクセスすることを防ぐことで、Identity Server を稼働させる Web Server や Directory Server プロセスへのセキュリティを確保する付加的な手段を提供します。

chroot を作成する前に

chroot 作成処理を開始する前に、以下の点に留意してください。

- chroot 環境に Identity Server をインストールするには、UNIX では root の権限が、Windows では Administrator の権限が必要です。
- chroot 環境にパッチを自動的にインストールすることはできません。システムにパッチがインストールされている場合、以下の適切な場所に手動でコピーする必要があります。
  - Sun ONE Directory Server
  - Sun ONE Web Server
  - Sun ONE Identity Server
  - Solaris

chroot 環境の作成

要求に対応して動作する Identity Server プロセスは、chroot ディレクトリの外部にあるファイルを検出およびアクセスすることはできません。このため、適切な Identity Server ファイルを chroot ディレクトリ構造内にコピーする必要があります。最初に実行する手順は、Identity Server の *user root* 環境の作成です。この新規 root には、Identity Server が本来のルート (/) に対して期待するすべてのものを含める必要があります。

注　　*$CHROOT* は、chroot ディレクトリの変数名です。

---

1. Identity Server のホストとなるコンピュータシステムに、次のシステムディレクトリを作成します。

   ```
   /dev
   /etc
   /sbin
   /usr
   /var
   /proc
   /opt
   ```

```
/etc/lib
/usr/platform
/usr/bin
/usr/sbin
/usr/lib
/usr/openwin/lib
/var/opt
/var/tmp
/usr/share
/usr/share/lib
```

2. `ln -s` コマンドを使用して、システムソフトリンクを作成します。

   作成する必要のあるソフトリンクは、次のとおりです。

   - /usr/bin -> bin
   - /usr/lib -> lib
   - /var/tmp -> tmp

3. `mknod` コマンドを使用して、*$CHROOT* 内にデバイスを作成します。

ヒント たとえば、*$CHROOT* ディレクトリ内にデバイス /dev/tcp を作成するには、次のように入力します。

```
mknod  dev/tcp c 11 42
```

$CHROOT ディレクトリ内に作成する必要のあるデバイスは、以下のとおりです。

- /dev/null
- /dev/tcp
- /dev/ticots
- /dev/ticlts
- /dev/ticotsord
- /dev/tty
- /dev/udp
- /dev/zero
- /dev/conslog

4. ディレクトリ /etc から、*$CHROOT* ディレクトリ内の対応する場所にファイルをコピーします (たとえば、/etc/hosts を *$CHROOT*/etc/hosts にコピーします)。

   コピーする必要のあるファイルは、次のとおりです。

   - /etc/hosts
   - /etc/nsswitch.conf
   - /etc/vfstab
   - /etc/group
   - /etc/passwd
   - /etc/shadow
   - /etc/hosts
   - /etc/resolv.conf
   - /etc/nsswitch.conf

5. バイナリを *$CHROOT* ディレクトリ内の対応する場所にコピーします (たとえば、/usr/bin/sh を *$CHROOT*/usr/bin/sh にコピーします)。

   コピーする必要のあるファイルは、次のとおりです。

   - /usr/bin/sh
   - /usr/bin/ls
   - /usr/bin/cat
   - /usr/bin/date
   - /usr/bin/dirname
   - /usr/bin/mv
   - /usr/bin/expr
   - /usr/bin/file
   - /usr/bin/awk
   - /usr/bin/grep
   - /usr/bin/gettext
   - /usr/bin/echo
   - /usr/bin/rm
   - /usr/bin/sed
   - /usr/bin/sleep
   - /usr/bin/hostname

- /usr/bin/domainname
- /usr/bin/cut
- /usr/bin/uname
- /usr/bin/ksh
- /usr/bin/basename
- /usr/bin/id
- /usr/bin/chmod
- /usr/bin/nohup
- /usr/bin/pkginfo
- /usr/bin/su
- /usr/bin/chown
- /usr/bin/ftp
- /usr/bin/isainfo
- /usr/bin/ldd
- /usr/bin/truss
- /usr/bin/rm
- /usr/xpg4/bin/id

6. ライブラリを *$CHROOT* ディレクトリ内の対応する場所にコピーします (たとえば、`/usr/lib/libc.so*` を *$CHROOT*`/usr/lib/` にコピーします)。

注	Identity Server がインストールされたコンピュータシステムが 64 ビットマシンの場合、`/usr/lib/64` および `/usr/lib/lwp/64` 内のライブラリをそれぞれ `$CHROOT/usr/lib/64` および `$CHROOT/usr/lib/lwp/64` にコピーしてください。

コピーする必要のあるファイルは、次のとおりです。

- /usr/lib/libc.so*
- /usr/lib/ld.so*
- /usr/lib/libdl.so*
- /usr/lib/libintl.so*
- /usr/lib/libnsl.so*
- /usr/lib/libsocket.so*

- /usr/lib/librpcsvc.so*
- /usr/lib/libw.so*
- /usr/lib/libproject*
- /usr/lib/libproc*
- /usr/lib/libsecdb*
- /usr/lib/libcmd*
- /usr/lib/libmp.so*
- /usr/lib/libm.so*
- /usr/lib/libresolv.so*
- /usr/lib/nss_dns.so*
- /usr/lib/nss_files.so*
- /usr/lib/nss_nis.so*
- /usr/lib/nss_nisplus.so*
- /usr/lib/straddr.so*
- /usr/lib/libthread.so*
- /usr/lib/libposix4.so*
- /usr/lib/libC.so*
- /usr/lib/librt.so*
- /usr/lib/libaio.so*
- /usr/lib/libelf.so*
- /usr/lib/libgen.so*
- /usr/lib/libpthread.so*
- /usr/lib/libadm.so*
- /usr/lib/libX*
- /usr/lib/libCrun.so*
- /usr/lib/libatomic.so*
- /usr/lib/libdhcpagent.so*
- /usr/lib/libproject.so*
- /usr/lib/libpam.so*
- /usr/lib/libbsm.so*
- /usr/lib/libsldap.so*

- /usr/lib/libldap.so*
- /usr/lib/libdoor.so*
- /usr/lib/libz.so*
- /usr/lib/libkstat*
- /usr/lib/libld.so*
- /usr/lib/liblddbg.so*
- /usr/lib/librtld.so*
- /usr/lib/libsh.so*
- /usr/lib/lddstub*
- /usr/lib/librtld_db.so*
- /usr/lib/libmd5.so*
- /usr/lib/locale
- /usr/lib/libCstd.so*

7. `zoneinfo` ファイルを、*$CHROOT* ディレクトリ内の対応する場所にコピーします (たとえば、`/usr/share/lib/zoneinfo/*` を *$CHROOT*`/usr/share/lib/` にコピーします)。
8. `locale` ファイルを、*$CHROOT* ディレクトリ内の対応する場所にコピーします。(たとえば、`/usr/lib/locale/*` を *$CHROOT*`/usr/lib/locale` にコピーします)。

chroot 内に Identity Server をインストールする

1. Identity Server 6.1 を、*$CHROOT* 以外のディレクトリにインストールします。

   インストールディレクトリを指定しない場合、Identity Server はデフォルトのインストールディレクトリである /opt/*IdentityServer_base* にインストールされます。

2. Sun ONE Web Server のインスタンスを停止します。

   ```
   ./WebServer_base/https-instanceName/https-stop
   ```

3. Sun ONE Directory Server のインスタンスを停止します。

   ```
   ./DirectoryServer_base/slapd-instanceName/stop-slapd
   ```

4. Identity Server のディレクトリ全体を、同一のディレクトリ構造を保持したまま *IdentityServer_base* から *$CHROOT* ディレクトリにコピーします。

5. 次の場所にあるファイルを、*$CHROOT* 内の対応するディレクトリにコピーします。

   - /etc/opt/SUNWam
   - /var/opt/SUNWam
   - /etc/init.d/amserver

6. Java 用のループバックマウントを作成します。

   次のコマンドを使用して、chroot 環境内で Identity Server を実行する準備を整えます。

   ```
   mount  -F lofs /usr/j2se $CHROOT/usr/j2se
   ```

7. /usr/share/lib および /usr/lib/mps のループバックマウントを作成します。

   例 :

   ```
   mount  -F lofs /usr/share/lib $CHROOT/usr/share/lib
   mount  -F lofs /usr/lib/mps $CHROOT/usr/lib/mps
   ```

chroot 内で Identity Server を起動する

chroot 環境で Identity Server を起動するには、Directory Server と Web Server の両方を起動する必要があります。

1. chroot コマンドを使用して、Directory Server を起動します。

   ```
   chroot $ROOT $IS_ROOT/DSServers/slapd-hostName/start-slapd
   ```

2. chroot コマンドを使用して、Web Server を起動します。

   ```
   chroot $ROOT $IS_ROOT/SUNWam/servers/https-instanceName/https-start
   ```

chroot 内の Identity Server ログファイル

通常、Identity Server は次のディレクトリ内にログファイル (Error Log、Debug など) を作成します。

```
/var/opt/SUNWam/debug and /var/opt/SUNWam/logs
```

chroot では、これらのログファイルは次の場所に作成されます。

$CHROOT/var/opt/SUNWam/log

$CHROOT/var/opt/SUNWam/debug

索引

L

R

S

W

あ

か

き

こ

さ

す

せ

そ

と

に

は

へ

ま

ら

れ

ろ