



Sun Java™ System

Identity Server

配備計画ガイド

2004Q2

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

Part No: 817-7013

Copyright © 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. は、この製品に含まれるテクノロジーに関する知的所有権を保持しています。特に限定されることなく、これらの知的所有権は <http://www.sun.com/patents> に記載されている 1 つ以上の米国特許および米国およびその他の国における 1 つ以上の追加特許または特許出願中のものが含まれている場合があります。

このソフトウェアは SUN MICROSYSTEMS, INC. の機密情報と企業秘密を含んでいます。SUN MICROSYSTEMS, INC. の書面による許諾を受けることなく、このソフトウェアを使用、開示、複製することは禁じられています。

U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

この配布には、第三者が開発したソフトウェアが含まれている可能性があります。

本製品の一部は、カリフォルニア大学からライセンスされている Berkeley BSD システムに基づいていることがあります。UNIX は、X/Open Company, Ltd が独占的にライセンスしている米国およびその他の国における登録商標です。

Sun、Sun Microsystems、Sun のロゴマーク、Java、Solaris、JDK、Java Naming and Directory Interface、JavaMail、JavaHelp、J2SE、iPlanet、Duke のロゴマーク、Java Coffee Cup のロゴ、Solaris のロゴ、SunTone 認定ロゴマークおよび Sun ONE ロゴマークは、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

Legato および Legato のロゴマークは Legato Systems, Inc. の商標であり、Legato NetWorker は同社の商標または登録商標です。

Netscape Communications Corp のロゴマークは Netscape Communications Corporation の商標または登録商標です。

OPEN LOOK および Sun Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカルユーザインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

この製品は、米国の輸出規制に関する法規の適用および管理下にあり、また、米国以外の国の輸出および輸入規制に関する法規の制限を受ける場合があります。核、ミサイル、生物化学兵器もしくは原子力船に関連した使用またはかかる使用者への提供は、直接的にも間接的にも、禁止されています。このソフトウェアを、米国の輸出禁止国へ輸出または再輸出すること、および米国輸出制限対象リスト (輸出が禁止されている個人リスト、特別に指定された国籍者リストを含む) に指定された、法人、または団体に輸出または再輸出することは一切禁止されています。

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われえないものとします。

目次

図目次	9
表目次	11
コード例一覧	13
本書について	15
本書の対象読者	15
Identity Server 2004Q2 のマニュアルセット	16
Identity Server のコアマニュアル	16
Identity Server ポリシーエージェントのマニュアルセット	17
マニュアルへのフィードバック	18
表記上の規則	18
表記上の規則	18
用語	19
関連情報	20
サードパーティーの関連 Web サイト	20
第 1 章 はじめに	21
アイデンティティ管理とは	21
アイデンティティ管理のインフラストラクチャ	22
アイデンティティプロファイルのライフサイクル	24
Sun Java System Identity Server	24
アクセス管理	24
シングルサインオン (SSO)	25
プラグイン可能な認証	25
ポリシーの評価	25
連携管理	26

Liberty Alliance Project	26
SAML (Security Assertion Markup Language)	26
アイデンティティ管理	27
ユーザープロファイル管理	27
ポリシー設定	27
サービス管理	27
監査	28
ポリシーエージェント	28
Identity Server コンソール	28
プログラミング用インタフェース	28
Sun Java System Directory Server	29
Identity Server の配備	29
ポリシーエージェントを使用した Identity Server の統合	30
配備ロードマップ	31
配備計画ガイドの各章	32
Identity Server に関連するマニュアル	33
 第 2 章 配備の計画	35
リソースの定義	35
人事情報	36
経営権を持つスポンサー	36
チームリーダー	37
プロジェクト管理	37
システムアナリスト	37
LOB アプリケーション管理者	37
システム管理者	38
独立系ソフトウェアベンダー	39
提携先のサードパーティ	39
資金の調達	39
目標の設定	40
情報の収集	41
ビジネスプロセス	41
IT インフラストラクチャ	42
仮想データ	43
アプリケーションの評価	44
プラットフォームの情報	45
セキュリティモデル	46
セッションのライフサイクル	46
カスタマイズおよびブランド設定	46
データの分類	47
認証へのマッピング	49
承認へのマッピング	49
スケジュールの作成	50

配備の設計	50
コンセプト証明	50
初期採用	51
一般の参加	51
製品環境	51
配備のチューニング	52
 第 3 章 Identity Server のアーキテクチャ	53
概要	53
統合化ポイント	55
ポリシーエージェント	55
Web およびプロキシサーバーエージェント	55
J2EE エージェント	56
Identity Server SDK	57
アイデンティティ管理 SDK	57
サービス管理 SDK	57
認証 API と認証 SPI	57
ユーティリティ API	57
ログ API とログ SPI	58
クライアントディテクション API	58
SSO API	58
ポリシー API	58
SAML SDK	58
連携管理 API	58
機能プロセス	59
認証とユーザーセッション	59
HTML over HTTP(S) インタフェース	59
XML over HTTP(S) インタフェース	60
ポリシーの統合	61
クライアントディテクションの統合	62
CDSSO、SAML、および連携	63
CDSSO	63
SAML	63
連携	63
Identity Server の拡張	64
Web コンテナ	64
複数の Directory Server インスタンス	64
LDAP ロードバランサ	65
 第 4 章 配備前の考慮事項	67
配備オプション	67
セキュリティ	68

高可用性	68
クラスタリング	69
スケーラビリティ	69
ハードウェア要件	70
ソフトウェア要件	71
オペレーティングシステム要件	71
Solaris 用パッチクラスタ	71
JDK ソフトウェア要件	72
Web コンテナ要件	72
Directory Server 要件	72
Web ブラウザ要件	73
Identity Server スキーマの理解	73
マーカーオブジェクトクラス	78
管理ロール	79
管理者パスワード	81
スキーマの制限	81
組織としてマークできるエントリのタイプは 1 つに限られる	82
ピープルコンテナをユーザーの親エントリにする必要がある	83
Identity Server XML で可能な組織の説明は 1 つに限られる	83
サポートされない DIT の例	84
第 5 章 配備シナリオ	87
複数サーバーのシナリオ	87
複数の Identity Server インスタンスをインストールするには	89
Web 配備	90
Java アプリケーションの配備	91
複数の JVM 環境	92
レプリケーションに関する考慮事項	93
レプリケーション用の設定	94
ロードバランサを使用する場合の設定	97
レプリケーションの警告	100
ファイアウォールを使用した Directory Server	100
グローバルタイムアウト属性の設定	101
個々のクライアント接続のタイムアウトの設定	101
Identity Server のセッションフェイルオーバー	102
セッションフェイルオーバーの概要	102
セッションフェイルオーバーの要件	103
セッションフェイルオーバーの実装	103
Web Server 6.0 SP6 のインストール (ロードバランサプラグイン用)	103
Application Server 7.0.0_01 EE のインストールおよび設定	104
Identity Server インスタンスのインストール	107
Identity Server 2004Q2 の設定	108
Identity Server および Portal Server の配備	111

単一のサーバーへのインストール	111
複数のサーバーへのインストール	112
連携管理	112

付録 A インストールされる製品のレイアウト	115
ベースインストールディレクトリ	116
製品ディレクトリ	116
/agents ディレクトリ	117
/bin ディレクトリ	118
/docs ディレクトリ	119
/dtd ディレクトリ	119
/include ディレクトリ	120
/ldaplib/ ディレクトリ	120
/lib ディレクトリ	120
/locale ディレクトリ	120
/migration ディレクトリ	120
/public_html ディレクトリ	120
/samples ディレクトリ	121
/share ディレクトリ	121
/web-src ディレクトリ	121
/debug、/logs、および /tmp ディレクトリ	122
設定 (/config) ディレクトリ	122

付録 B ユーザーセッションのライフサイクル	125
概要	125
要求	126
認証	127
セッショントークン	129
ポリシー	133
要求されたページ	135
シングルサインオン要求	137
スレッド 1: シングルサインオン	137
スレッド 2: クロスドメインシングルサインオン	141
セッションの終了	144

付録 C Active Directory に対する認証	147
概要	147
既存の LDAP 認証モジュールを指し示す	147
Active Directory 認証モジュールを新規作成する	148
複数の LDAP サブ設定	148
Active Directory 認証の設定	148
トラブルシューティング	151

Identity Server へのクイックアクセス	151
Directory Server を使用した再設定	151
付録 D chroot 環境のインストール	153
付録 E ロードバランサの設定	155
ロードバランサの概要	155
スティッキーセッション	156
Resonate Central Dispatch のインストール	157
ロードバランサの設定	158
Central Dispatch を setcookie 用に設定する	158
Identity Server を setcookie 用に設定する	164
ロードバランサ Cookie の使用に合わせて Central Dispatch を設定する	165
ロードバランサ Cookie に合わせて Identity Server を設定する	167
設定を確認する	168
付録 F RADIUS サーバーに対する認証	169
概要	169
RADIUS サーバーの設定	170
Identity Server の設定	171
用語集	173
索引	175

図目次

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	23
図 2-1	データおよびサービスのセキュリティ要件	48
図 3-1	Identity Server 2004Q2 の機能アーキテクチャ	54
図 5-1	1 つの Directory Server に対する複数の Identity Server インスタンス	88
図 5-2	簡単な Web 配備シナリオ	91
図 5-3	Java アプリケーションの配備	92
図 5-4	シングルサブライヤレプリケーション	94
図 5-5	複数サブライヤ設定 (マルチマスターレプリケーションとも呼ばれる)	95
図 5-6	ロードバランサを使用する複数サブライヤレプリケーション	98
図 E-1	ロードバランサを使用した Identity Server の設定	156
図 E-2	Resonate を使用したノードの作成	159
図 E-3	仮想 IP アドレスを新規作成する	160
図 E-4	HTTP スケジューリングルールを設定する	161
図 E-5	CDMaster でノードを設定する	162
図 E-6	Cookie Persistence Scheduling Rule を設定する	163

表目次

表 4-1	デフォルトおよび動的なロールとそのアクセス権	79
表 A-1	Identity Server のコマンド行ツールおよびユーティリティ	118
表 A-2	Identity Server DTD ファイル	119
表 E-1	Resonate Central Dispatch の定義済みの用語	157

コード例一覧

コード例 4-1	ds_remote_schema.ldif	74
コード例 4-2	sunone_schema2.ldif	77
コード例 5-1	serverconfig.xml レプリケーションの修正	97
コード例 5-2	serverconfig.xml ロードバランサの変更	99
コード例 B-1	GET 要求ヘッダー	126
コード例 B-2	リダイレクト情報に対する GET 応答	126
コード例 B-3	認証サービスにリダイレクトされる GET 要求	127
コード例 B-4	ユーザーに返される認証フォーム	127
コード例 B-5	Identity Server に返される POST 証明情報	128
コード例 B-6	最初に要求されたリソースへのリダイレクト	128
コード例 B-7	トークンとともにリダイレクトされる GET 要求ヘッダー	129
コード例 B-8	ネーミング情報の POST 要求	129
コード例 B-9	ネーミング情報に関する応答	130
コード例 B-10	セッションサービスへのセッション検証用の POST 要求	131
コード例 B-11	セッションの有効性を示すセッションサービス応答	132
コード例 B-12	ポリシー情報を求める POST 要求	133
コード例 B-13	許可ポリシー応答	134
コード例 B-14	ポリシーエージェントからのログ要求	135
コード例 B-15	エージェントにログを通知する応答	136
コード例 B-16	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	136
コード例 B-17	有効なセッショントークンを含む 2 番目の要求	137
コード例 B-18	ネーミングサービスの POST 要求	138
コード例 B-19	セッションサービスへの POST 要求	138
コード例 B-20	ポリシーサービスへの POST 要求	139
コード例 B-21	ポリシーサービスからのアクセス否定応答	139
コード例 B-22	ログサービスへの POST 要求	140
コード例 B-23	アクセス拒否のメッセージを表示する HTML ページ	140

コード例 B-24	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	141
コード例 B-25	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	141
コード例 B-26	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	142
コード例 B-27	ブラウザへの POST 返信	142
コード例 B-28	ポリシーエージェントへのブラウザ POST	143
コード例 B-29	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 . .	144
コード例 B-30	ログアウトサービスの GET 要求	144
コード例 B-31	ユーザーに返される成功を示す HTML ページ	145
コード例 B-32	セッション通知用の POST	145
コード例 F-1	RADIUS ユーザーのエントリ	170
コード例 F-2	RADIUS クライアントのエントリ	170

本書について

本書『Sun Java™ System Identity Server 配備計画ガイド』では、Sun Java System Identity Server (以前の Sun™ ONE Identity Server) を計画し配備するための技術情報と簡単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。『配備計画ガイド』のこの部分では、次の項目について説明します。

- 15 ページの「本書の対象読者」
- 16 ページの「Identity Server 2004Q2 のマニュアルセット」
- 18 ページの「表記上の規則」
- 20 ページの「関連情報」
- 20 ページの「サードパーティーの関連 Web サイト」

本書の対象読者

この『配備計画ガイド』は、Sun Java System 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必要があります。

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

Sun Java System 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 2004Q2 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 Java System Identity Server 2004Q2 のコアアプリケーションマニュアルで、もう 1 つは Sun Java System 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。

- 『Technical Overview』(<http://docs.sun.com/doc/817-5706>) Identity Server コンポーネントの高レベルの概要。コンポーネントを統合してアイデンティティ管理を行い、企業資産と Web ベースアプリケーションを保護する方法について説明しています。また、Identity Server の概念と用語についても説明しています。
- 『Migration Guide』(<http://docs.sun.com/doc/817-5708>) 既存のデータおよび Sun Java System 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 とその他の製品のインストール方法については、『Sun Java Enterprise System 2004Q2 インストールガイド』(<http://docs.sun.com/doc/817-5760>) を参照してください。
- 『管理ガイド』(<http://docs.sun.com/db/prod/entsys?l=ja>) 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。
- 『配備計画ガイド』(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。
- 『Developer's Guide』(<http://docs.sun.com/doc/817-5710>) Identity Server のカスタマイズ方法とその機能を組織の現在の技術インフラストラクチャに統合する方法について説明します。製品とその API のプログラムの側面に関する詳細も含まれています。
- 『Developer's Reference』(<http://docs.sun.com/doc/817-5711>) Identity Server C API を構成するデータタイプ、構造、および機能について説明します。
- 『Federation Management Guide』(<http://docs.sun.com/doc/817-6362>): Liberty Alliance Project に基づく、連携管理について説明します。

- 『リリースノート』(<http://docs.sun.com/db/prod/entsys?l=ja>) は、製品のリリース後、オンラインでご利用になれます。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 Java System 2004Q2 マニュアル Web サイトの Identity Server のページ (<http://docs.sun.com/db/prod/entsys?l=ja>) 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のマニュアルは、この Web サイトにあります。

http://docs.sun.com/coll/S1_IdServPolicyAgent_21

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。『リリースノート』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 Java System 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。ご意見やご提案の提供には、次の Web フォームを使用してください。

<http://www.sun.com/hwdocs/feedback/>

該当するフィールドに、マニュアルの完全なタイトルとパート番号を入力してください。マニュアル番号は、マニュアルのタイトルページまたはマニュアルの上部にある 7 桁または 9 桁の番号です。たとえば、『Identity Server 配備計画ガイド』のマニュアル番号は、817-7013 です。

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成的なセットを示します。
- ディレクトリサーバーは、Sun Java System Directory Server のインストール済みのインスタンスを示します。
- Application Server は、Sun Java System Application Server (Sun ONE Application Server と呼ばれる) のインストール済みのインスタンスを示します。
- Web Server は、Sun Java System Web Server (Sun ONE Web Server と呼ばれる) のインストール済みのインスタンスを示します。
- 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。
- *IdentityServer_base* は、Identity Server のベースインストールディレクトリを表します。Identity Server 2004Q2 のデフォルトのベースインストールと製品ディレクトリは、使用するプラットフォームにより異なります。
 - Solaris™ システム /opt/SUNWam
 - Linux システム /opt/sun/identity

製品ディレクトリは、Solaris システムの場合は /SUNWam、Linux システムの場合は /identity です。Identity Server 2004Q2 をインストールする場合は、以下のよう別ディレクトリを指定することもできます。Solaris システムの場合は /opt、Linux システムの場合は /opt/sun となりますが、製品ディレクトリの /SUNWam または /identity は変更しないでください。

以下の製品のベースインストールディレクトリについては、それぞれの製品のマニュアルを参照してください。

- *DirectoryServer_base* は、Sun Java System Directory Server のベースインストールディレクトリを示します。
- *ApplicationServer_base* という変数は、Sun Java System Application Server のベースインストールディレクトリを示します。
- *WebServer_base* という変数は、Sun Java System Web Server のベースインストールディレクトリを示します。

関連情報

関連情報は、次の場所に置かれています。

- Directory Server のマニュアルセット :
<http://docs.sun.com/db/prod/entsys?l=ja>
- Web Server のマニュアルセット :
<http://docs.sun.com/db/prod/entsys?l=ja>
- Application Server のマニュアル
<http://docs.sun.com/db/prod/entsys?l=ja>
- Web Proxy Server のマニュアルセット :
<http://docs.sun.com/prod/s1.webproxys#hic>
- ダウンロードセンター :
<http://www.sun.com/software/download/>
- テクニカルサポート :
<http://www.sun.com/service/sunone/software/index.html>
- プロフェッショナルサービス :
<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>

サードパーティーの関連 Web サイト

Sun は、本書に記載されたサードパーティーの Web サイトの有効性について責任を負いません。Sun は、これらのサイトまたはリソースを通じて入手可能なコンテンツ、広告、製品、その他の内容についていかなる保証もせず、かつ責任や義務を負いません。Sun は、これらのサイトやリソースを通じて入手したコンテンツ、製品、またはサービスを使用または信頼することに起因または関連する、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。

はじめに

Sun Java™ System Identity Server (以前の 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に分散す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ています。

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。

- [21 ページの「アイデンティティ管理とは」](#)
- [24 ページの「Sun Java System Identity Server」](#)
- [29 ページの「Identity Server の配備」](#)

アイデンティティ管理とは

現在、各企業では、日々の作業の管理の効率化のため、高度な情報技術のインフラストラクチャを運用しています。このインフラストラクチャに不可欠な要素には、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
- 人事、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、マーケティング、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、個人のプロフィール、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のインフラストラクチャ

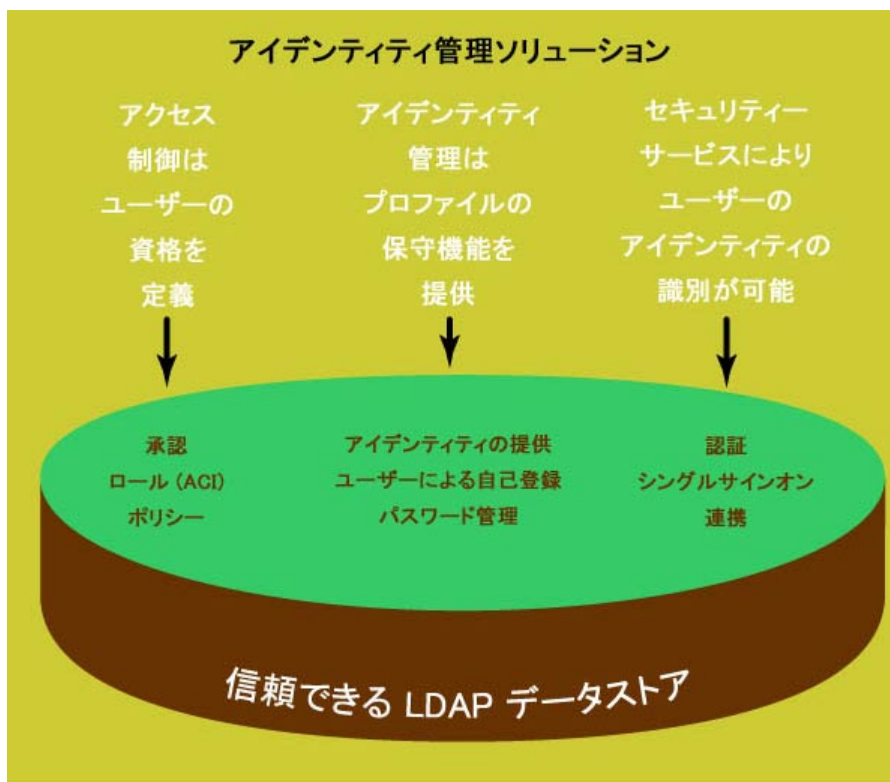
アイデンティティ管理システムの目的は、企業内のすべてのユーザーを単一のインフラストラクチャにより管理することです。単一のアイデンティティ管理システムを使用することで、繰り返しを避けて一貫性を維持することができるため、ユーザープロフィールの管理が簡単になります。さらに単一のシステムにより、アイデンティティ管理プロセスの合理化、簡略化、および自動化が可能になります。以下に、アイデンティティ管理システムの構成要素を示します。

- 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 Java System Identity Server

Sun Java System Identity Server は、アクセス管理、連携、およびアイデンティティ管理をサポートする Web サービスを統合した標準ベースのミドルウェアのパッケージです。このため、Identity Server は総合的なアイデンティティ管理ソリューションとして、ユーザープロファイルの作成および管理機能を、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します。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ともに、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。組織では、中心となる1箇所の管理機能から、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提供できます。Identity Server の一連のアクセス管理サービスは、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一度ログインすれば、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も含まれます。

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 対応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を実現するのに重要です。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。

ユーザープロフィール管理

簡潔にまとめると、ユーザープロフィール管理では、アイデンティティプロフィールの作成と削除を行います。ただし、これには、プロフィー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ィールを管理すること (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 テクノロジー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。詳細については、[30 ページの「ポリシーエージェントを使用した Identity Server の統合」](#)を参照してください。

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 Java System 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、[57 ページの「Identity Server SDK」](#) および『Identity Server 2004Q2 Developer's Guide』を参照してください。コマンド行ツールの詳細は、『Identity Server 2004Q2 管理ガイド』を参照してください。

Sun Java System Directory Server

Java System 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。

Identity Server の配備

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ともに実装する。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

ポリシーエージェントを使用した Identity Server の統合

Identity Server 2004Q2 は総合的なアイデンティティ管理システムですが、それぞれの組織では何らかのアイデンティティ管理システムをすでに実装している場合があります。たとえば、ディレクトリサーバーや Web コンテナをすで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、組織の要件を満たすエージェントを利用できる場合は、ポリシーエージェントをダウンロードして保護されたサーバーにインストールできます。

Identity Server の各リリースと並行して、新規ポリシーエージェントの開発およびリリースが行われています。たとえば、Apache Webserver、BEA WebLogic、IBM HTTP Server、IBM WebSphere、Lotus Domino、Microsoft IIS、および PeopleSoft のさまざまなリリースに対応したポリシーエージェントが利用できます。

ポリシーエージェントに対応したオペレーティングシステムには、Solaris™ Operating System (SPARC® Platform Edition および x86 Platform Edition)、Red Hat™ Linux、HP-UX 11.x、および Windows 2000 があります。

注 上記の各製品用のポリシーエージェントが、これらのすべてのオペレーティングシステムに用意されているわけではありません。現在利用可能なポリシーエージェントの一覧については、次の Web サイトで Sun の「Web, Portal & Directory Servers Download Center」を調べて、要件を満たすポリシーエージェントが利用できるかどうかを確認してください。

http://www.sun.com/software/download/inter_ecom.html

配備ロードマップ

Identity Server の統合の成功には詳細な計画が重要です。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ー分析を開発し、ビジネスの目的に適用
 - テクノロジーサービスを列挙
 - 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、テクノロジー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

配備計画ガイドの各章

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、将来の必要を明確にする方法（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。

『Identity Server 2004Q2 配備計画ガイド』では、付録が追加されて説明が補足されました。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 F「[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 D「[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 に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。

- インストーラー — インストール方法については、『Sun Java Enterprise System 2004Q2 インストールガイド』を参照してください。
- 移行 — 既存のデータの移行と以前のバージョンの Identity Server の詳細は、『Sun Java Enterprise System 2004Q2 インストールガイド』および『Identity Server 2004Q2 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、『Identity Server 2004Q2 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、『Identity Server 2004Q2 Developer's Guide』を参照してください。

配備の計画

Sun™ System 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。

- [35 ページの「リソースの定義」](#)
- [40 ページの「目標の設定」](#)
- [41 ページの「情報の収集」](#)
- [44 ページの「アプリケーションの評価」](#)
- [47 ページの「データの分類」](#)
- [50 ページの「スケジュールの作成」](#)
- [52 ページの「配備のチューニング」](#)

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。

人事情報

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 プロフェッショナルサービス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、ユーザーインター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。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。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ともに、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。これらの担当者プロジェクトの里程碑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。対策チーム (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ともに、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、独自の内部配備とともに考慮してください。この場合、重要なのは、提供するビジネス機能を保持する LOB アプリケーションを含めること、および当事者すべての技術リソースとの通信を管理することです。弁護士も、関係する当事者間の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ともに、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、これらの部門間の情報収集は、特定の機能および目標を定義し、関係するグループのモチベーションの理解を深めるのに役立ちます。収集が済んだら、情報を設計の青写真として、および経営権を持つスポンサーから確実な承認を得るために活用できます。

実地調査の際、以下のグループの支援を得られます。

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

初期調査には、[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業務の実行を定義する手順です。プロセスには、次の手順を含められます。

- 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
- 従業員の雇用終了

通常、これらのプロセスは、業務単位ごとに使用されるアプリケーションによりサポートされるため、これらのプロセスの評価は必須です。考慮すべき内容には、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 Java System Directory Server (または既存のデータストア)、および Sun Java System 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、[44 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、明らかにより広範で機密性の高いプロファイルを保持します。

- アクセス可能なデータ

データには、公開情報、内部情報、機密情報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、[47 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、配備または再構築の初期プロセスにあるか、Identity Server 配備と一致する時系列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、アプリケーション間の統合化の複雑性を比較できます。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

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

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項目です。ユーザーセッションが作成、管理、および破棄される方法について明確に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。この項目の詳細については、[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

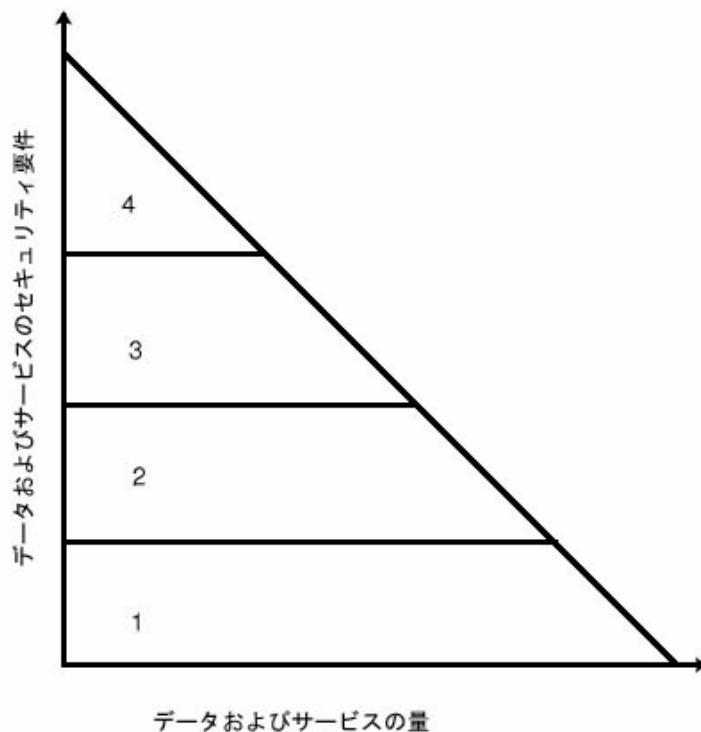
アプリケーションのブランド設定やルックアンドフィールに関する特定の要件を考慮する必要があります。多くの場合、アプリケーション単体のルックアンドフィールを重視するか、ユーザーにとって一貫した使い勝手を重視するかは重要な問題です。カスタマイズおよびブランド設定要件を満たす場合、そのための時間もスケジュールに組み込む必要があります。このため、アプリケーションの評価にカスタマイズおよびブランド設定要件が含まれているかどうかを確認してください。

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ともに、データのアクセス時刻およびネットワーク上の位置などの修飾条件も念頭に置くようにしてください。

認証へのマッピング

データをセキュリティ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、データで分類した場合、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 Java System 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、通常的环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。
---	---

配備のチューニング

Identity Server 2004Q2 のインストール後、amtune と関連するスクリプトを使用して配備のチューニングを行い、パフォーマンスを最適化できます。これらのスクリプトを使用して、管理者は Identity Server 2004Q2、Solaris™ Operating System (OS)、SPARC® Platform Edition、Web コンテナ (Web Server 6.1 SP2 または Application Server 7.0 Update 3)、および Directory Server のチューニングを行うことができます。

Java Enterprise System インストーラにより、チューニングスクリプトと関連ファイルが bin/amtune ディレクトリにインストールされます。

amtune は、インタラクティブなスクリプトではありません。amtune を実行する前に、amtune-env 設定ファイルのパラメータを編集して、特定の環境で実行したい amtune チューニングを指定します。amtune-env 設定ファイルには、次の 2 つの主要なセクションがあります。

- チューニングを制御するパフォーマンス関連のパラメータ
- Identity Server エンジニアリングにより管理される内部セクションで、変更不可
amtune は次の 2 つのモードで実行できます。
- レビューモード : amtune が推奨チューニングのレポートを行いますが、環境に対して実際の変更は行いません。
- 変更モード : amtune-env 設定ファイルのパラメータに基づいて、amtune が Directory Server を除いて実際の変更を行います。

amtune が Directory Server のチューニングを自動的に行うことはありません。ほとんどの配備には、Identity Server 以外にも Directory Server にアクセスするアプリケーションがあります。したがって、他のアプリケーションに与える影響を考慮せずにチューニングによる変更を行うことは好ましくありません。

注	Directory Server のチューニングを行う前に、db2bak を使用して Directory Server のデータのバックアップを取ります。
----------	---

amtune を実行すると、スクリプトファイルにより、Directory Server チューニングスクリプト amtune-directory を含む tar ファイルが作成されます。このファイルを一時ディレクトリで解凍し、スクリプトをレビューモードで実行します。変更が、配備で使用するすべてのアプリケーションで受け入れられるものであることを確認して、amtune-directory を変更モードで実行します。

チューニングスクリプトの実行と、amtune-env 設定ファイルでのチューニングパラメータの設定に詳細は、『Sun Java System Identity Server 2004Q2 管理ガイド』を参照してください。

Identity Server のアーキテクチャ

Sun Java™ System Identity Server には、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が用意されています。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。

- [53 ページの「概要」](#)
- [55 ページの「統合化ポイント」](#)
- [59 ページの「機能プロセス」](#)
- [64 ページの「Identity Server の拡張」](#)

概要

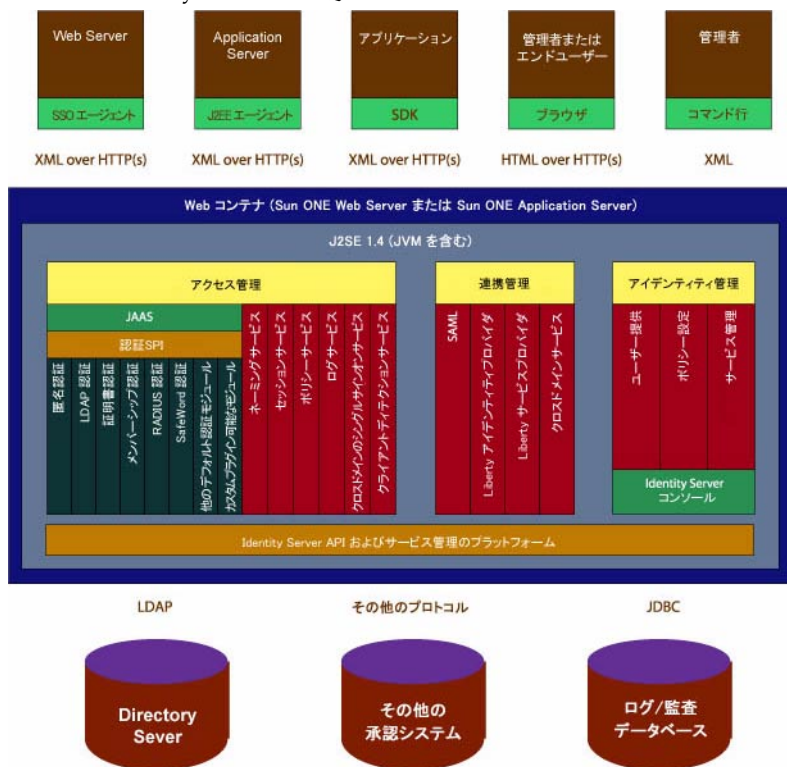
Identity Server は、Java™ 2 Platform, Enterprise Edition (J2EE) プラットフォーム上に構築されており、サービスを提供するサーブレット、Web サービスエンドポイント、および Web アプリケーションを操作するためのインフラストラクチャとして Web コンテナを使用します。提供される各サービスは、Web サービスとして実装されます。Web サービスは、現在注目を集めている標準技術セットを使用して構築され、ネットワーク上で集中的に提供されます。

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 Java System 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 2004Q2 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 2004Q2 の機能アーキテクチャ



これらのコンポーネントに固有の個別サービスの詳細は、『Identity Server 2004Q2 Developer's 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 とは個別に提供されます。最新の Sun Java System 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 Java System Directory Server に格納されているかどうかを検証する)。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る)。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 Java System Identity Server Web Policy Agents Guide』または『Sun Java System 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今後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 Definition 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。

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.ipplanet.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)。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。

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。

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。

ユーティリティ API

この API は、システムリソースの管理に使用できるいくつかの Java クラスを提供します。これには、スレッド管理やデバッグデータ形式などがあります。Java パッケージ名は `com.ipplanet.am.util` です。現在 C インタフェースとの互換性はありません。

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。

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。

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。

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。

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。

詳細については、『Identity Server 2004Q2 Federation Management Guide』を参照してください。

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。以下にその内容を示します。

- [認証とユーザーセッション](#)
- [ポリシーの統合](#)
- [クライアントディテクションの統合](#)
- [CDSO、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。プログラムに関連するポイントの詳細は、『Identity Server 2004Q2 Developer's 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、 61 ページの「ポリシーの統合」 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る。トークンは情報へのポインタに過ぎない)。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、『`Identity Server 2004Q2 Developer's 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ブリットです。関連する DTD の `remote-auth.dtd` は、Solaris システムでは `IdentityServer_base/SUNWam/dtd` ディレクトリ、Linux システムでは `IdentityServer_base/identity/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。

ポリシーの統合

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://www.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、[59 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せん。

クライアントディテクションの統合

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 Java System Identity Server Developer's 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、59 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 Java System Identity Server Developer's Guide』、『Sun Java System Identity Server Web Policy Agents Guide』、または『Sun Java System 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 Java System Identity Server Developer's 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 Java System Identity Server Developer's 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 Java System 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 Java System 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 Java System 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、『Directory Server 5 2004Q2 Installation and Migration Guide』および『Directory Server 5 2004Q2 配備計画ガイド』を参照してください。

LDAP ロードバランサ

Sun Java System 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 Java System 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。

配備前の考慮事項

Sun Java™ Identity Server 2004Q2 を使用すると、異機種のハードウェア、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で、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うまく配備できます。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。

- [67 ページの「配備オプション」](#)
- [70 ページの「ハードウェア要件」](#)
- [71 ページの「ソフトウェア要件」](#)
- [73 ページの「Identity Server スキーマの理解」](#)

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する
- 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。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る。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る。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る

高可用性

IT の配備では、ユーザーに対して可用性を継続するとともに、SPOF (*Single Point Of Failure*) を発生させないことが重要です。可用性を高めるための手法は、クラスタリングやマルチマスターレプリケーションなど、製品ごとに異なります。望ましい高可用性とは、システムやコンポーネントが期待どおりに長期間、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 Java System Identity Server では使用できないものの、システムの基盤である Sun Java System 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、データストアとして使用する Sun Java System Directory Server 2004Q2 (5.2) および配備先の Web コンテナとともにインストールする必要があります。さらに、Directory Server と 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 (Web Server によって配備される 1 つの Identity Server インスタンスと、1 つの Directory Server インスタンス) に基づく高度な内容です。Identity Server をインストールする前に、『Directory Server 5 2004Q2 Installation and Migration Guide』、『Directory Server 5 2004Q2 配備計画ガイド』、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 Java System Identity Server を設計および配備する前に Sun Java System プロフェッショナルサービスまたは Sun Java System 認定システムインテグレータに相談することです。
---	--

パフォーマンスを最適化するために、Identity Server は、100M バイト以上の Ethernet ネットワーク上で実行してください。Identity Server 配備の最小構成は、Identity Server と Sun Java System 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、基本操作およびオブジェクト割り当て領域に 2G バイトの RAM、さらに 10,000 並行セッションごとに 100M バイトの RAM が推奨されています。各 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 2004Q2 は、次のプラットフォームでサポートされています。

- Solaris™ Operating System (OS)、SPARC® Platform Edition、バージョン 8 および 9
- Solaris™ 9 OS、x86 Platform Edition
- Red Hat™ Linux、Advanced Server 2.1 Update 2

プラットフォームの詳細については、『Sun Java Enterprise System 2004Q2 リリースノート』を参照してください。

Solaris 用パッチクラスタ

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。最初の番号はパッチ自体を識別するためのもので、2 番目の番号はパッチのバージョンを識別するためのものです。全般的なパッチ情報および推奨されるパッチは、次の SunSolve Patch Portal からダウンロードできます。

<http://sunsolve.sun.com/>

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

必要なパッチのリストについては、『Identity Server 2004Q2 リリースノート』および『Java Enterprise System 2004Q2 リリースノート』を参照してください。

JDK ソフトウェア要件

Identity Server 2004Q2 では、次の JDK ソフトウェアが必要になります。

- Version 1.4.2_*x*。ただし、*x* は 04 以上
- Version 1.4.1_*x*。ただし、*x* は 05 以上

Web コンテナ要件

Identity Server 2004Q2 では、完全インストールを行うか、SDK だけをインストールするかに応じて、さまざまな Web コンテナをサポートしています。完全インストールの場合は、次の Web コンテナのいずれかを使用します。

- Sun Java System Web Server 6.1 Service Pack (SP) 2
- Sun Java System Application Server 7.0 Update 3

SDK だけをインストールする場合は、次の Web コンテナのいずれかを使用します。

- Web Server 6.1 SP2
- Application Server 7.0 Update 3
- Application Server 7.0.0_01 EE (セッションフェイルオーバーの実装専用)
- BEA WebLogic 6.1 SP4、6.1 SP5、8.1
- IBM WebSphere Application Server 5.1

ポリシーエージェントを Web コンテナにインストールする場合は、約 10M バイトのディスク容量が使用されます。Web コンテナを設定するときには、この容量を考慮に入れる必要があります。詳細は、『Sun Java System Identity Server Web Policy Agents Guide』または『Sun Java System Identity Server J2EE Policy Agents Guide』を参照してください。

Directory Server 要件

Identity Server 2004Q2 には、次のバージョンのいずれかが必要になります。

- Directory Server 5 2004Q2
- Directory Server 5.1 SP 1 (またはこれ以上)

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 2004Q2 では、次の Web ブラウザをサポートしています。

- Solaris OS: Netscape Navigator™ バージョン 4.79、6.2.1、および 7.0
- Linux: Netscape Navigator™ バージョン 6.2.1、および 7.0
- Windows: Internet Explorer バージョン 5.5 および 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 [74 ページのコード例 4-1](#) および [77 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' )
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' )

```

コード例 4-1 ds_remote_schema.ldif (続き)

```

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' )
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' )

```

コード例 4-1 ds_remote_schema.ldif (続き)

```

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' )
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、『Identity Server 2004Q2 Developer's 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、『Identity Server 2004Q2 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、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 4-1 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 4-1 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
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

表 4-1 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 Java System Identity Server 管理ガイド』の第 14 章「管理属性」を参照してください。

管理者パスワード

Identity Server のインストール中には、管理者ユーザー ID (amadmin) と LDAP ユーザー ID (amldapuser) のパスワードを入力する必要があります。

serverconfig.xml ファイルには、Identity SDK が Directory Server への LDAP 接続プールを確立するために使用するパラメータが含まれます。たとえば、次のユーザーです。

- プロキシユーザー (ユーザー名 ="User 1"、cn=puser) は、任意のユーザー (組織管理者またはエンドユーザーなど) の権限を引き受ける
- 管理ユーザー (ユーザー名 ="User 2"、cn=dsameuser) は、Identity Server SDK が、特定のユーザーにリンクされていない Directory Server 上で操作を実行するとき (たとえば、サービス設定情報の取得)、バインドするために使用される

これらのユーザーにはそれぞれのパスワードが伴い、serverconfig.xml ファイルに暗号化された形式で格納されています。もちろん、これらのパスワード (特に puser パスワード) を常に保護する必要がありますが、次の点にも注意してください。

- Identity Server をインストールした後に puser と dsameuser のパスワードを変更する
- puser および dsameuser には、amadmin または amldapuser に使用したパスワードと同じパスワードを使用しない

puser および dsameuser のパスワードの変更については、『Identity Server 2004Q2 管理ガイド』の amPassword コマンド行ツールを参照してください。

スキーマの制限

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

以下の項で、Identity Server スキーマに課される制限について説明します。

- [82 ページ](#)の「組織としてマークできるエントリのタイプは1つに限られる」
- [83 ページ](#)の「ピープルコンテナをユーザーの親エントリにする必要がある」
- [83 ページ](#)の「Identity Server XML で可能な組織の説明は1つに限られる」

この節の最後には、[84 ページ](#)の「サポートされない DIT の例」も複数記載しています。

組織としてマークできるエントリのタイプは1つに限られる

Identity Server `iplanet-am-managed-org` 予備クラスを、任意のエントリに追加することにより、Identity Server は、このエントリを組織であるように管理します。ただし、Identity Server では、組織としてマークできるエントリのタイプは1つに限られます。たとえば、DIT にエントリ `o=sun` と別のエントリ `dc=ibm` がある場合、両方のエントリを組織としてマークすることはできません。

次の例では、`dc` と `o` の両方のエントリを組織にしようとしています、この DIT 構造は Identity Server で管理できません。

```
dc=MadisonParc,dc=com
├── o=continent
│   └── dc=company
│       ⋮
```

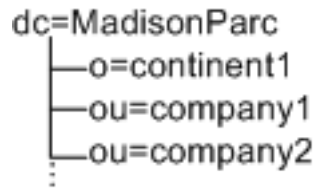
この規則には1つの例外があります。Identity Server ルートサフィックスでのエントリは、1つのエントリには数えられません。したがって、次の例の DIT 構造は、Identity Server で管理できます。

```
dc=MadisonParc
├── o=continent1
├── o=continent2
│   ⋮
```

`o=continent1` の下に `dc=company1` を追加したとすると、`dc` がコンテナとしてマークされている場合にのみ、この DIT は管理できます。コンテナは、Identity Server の別の抽象タイプであり、通常、`OrganizationalUnit` にマッピングされます。ほとんどの DIT では、`iplanet-am-managed-container` エントリをすべての `OrganizationUnits` に追加します。

```
dc=MadisonParc
├── o=continent1
│   └── dc=company1
├── o=continent2
│   ⋮
```

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



この例では、`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 `organization` としてマークし、`ou=` エントリを `containers` としてマークしています。UI に表示されるときに、組織とコンテナで利用できるオプションは同じです。従属または下位区分、ピープルコンテナ、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。

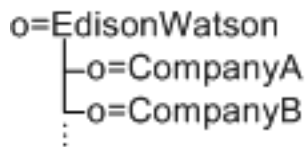
ピープルコンテナをユーザーの親エントリにする必要がある

もう 1 つの抽象エントリタイプはピープルコンテナです。Identity Server タイプは、このエントリがユーザーの親エントリであると想定します。ピープルコンテナとしてエントリに `iplanet-am-managed-people-container` のマークが付けられていると、UI は、このコンテナには下位ピープルコンテナまたはユーザーだけが存在すると見なします。属性 `OrganizationUnit` が通常、ピープルコンテナとして使用されますが、`iplanet-am-managed-people-container` オブジェクトクラスを保有し、Identity Server の管理可能な親タイプの `organization` または `container` を保持する限り、Identity Server の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。

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

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

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

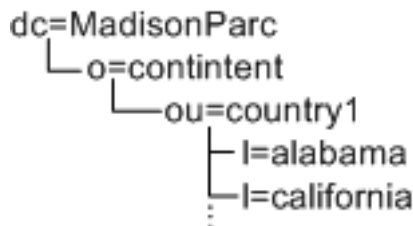


組織の説明は、組織レベルではなくルートレベル (o=EdisonWatson) で定義されます。デフォルトでは、企業 A と 企業 B の両方の UI を同一にする必要があります。また、すべてのサービスは、下位スキーマタイプユーザーの属性になるように、属性をグローバルに定義します。したがって、企業 A が、予備クラス CompanyA-user にそのユーザー用の属性を保持し、企業 B が、CompanyB-user に属性を保持している場合、企業 B の属性は上書きされ、表示されません。

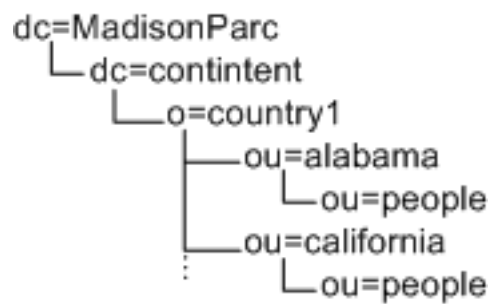
回避策としては、ユーザー表示に対処するように ACI を修正する方法があります。ただしこの回避策は、「検索」および「作成」ウィンドウでの属性には対処しません。

サポートされない DIT の例

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



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。



配備シナリオ

この章では、Sun Java™ System Identity Server 2004Q2 のさまざまな配備シナリオについて説明します。以下の節で構成されています。

- [87 ページの「複数サーバーのシナリオ」](#)
- [90 ページの「Web 配備」](#)
- [91 ページの「Java アプリケーションの配備」](#)
- [92 ページの「複数の JVM 環境」](#)
- [93 ページの「レプリケーションに関する考慮事項」](#)
- [100 ページの「ファイアウォールを使用した Directory Server」](#)
- [102 ページの「Identity Server のセッションフェイルオーバー」](#)
- [111 ページの「Identity Server および Portal Server の配備」](#)
- [112 ページの「連携管理」](#)

複数サーバーのシナリオ

一般的な配備シナリオは 2 台のサーバーから構成され、各サーバーには、Identity Server と Directory Server の両方がインストールされています。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。

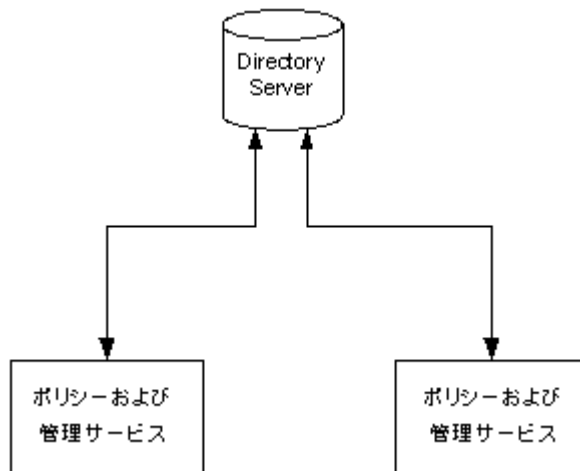
1 番目のサーバーにインストールされた Identity Server は、Directory Server のいずれかのインスタンスを指し示します。Java Enterprise System インストーラを使用したインストールでは、ユーザー自身の設定に応じて既存の DIT を使用するかまたは使用せずに、既存の Directory Server を選択できます。

Identity Server インストールスクリプトを実行して、2 番目のサーバーに Identity Server の 2 番目のインスタンスをインストールし、既存の DIT を使用して、このインスタンスが Directory Server を指し示すようにします。Identity Server は、すでに存在していることを認識した情報は、Directory Server に一切書き込みません。したがって、既存のデータが上書きされる危険はありません。

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のために、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。同じ Directory Server に対して、Identity Server の複数インスタンスを別々のサーバーにインストールするには、[89 ページの「複数の Identity Server インスタンスをインストールするには」](#)を参照してください。

図 5-1 には、1 つの Directory Server に対する 2 つの Identity Server インスタンスを示しています。

図 5-1 1 つの Directory Server に対する複数の Identity Server インスタンス



複数の Identity Server インスタンスをインストールするには

Identity Server 2004Q2 の複数のインスタンスをインストールするには、次の手順に従ってください。

1. Java Enterprise System インストーラを実行して、Identity Server の 1 番目のインスタンスをインストールします。このインストーラの詳細については、『Sun Java Enterprise System 2004Q2 インストールガイド』を参照してください。
2. 作成しようとする Identity Server インスタンスごとに、新しい Web コンテナインスタンスを、Web コンテナ用の管理ツールを使用して作成および起動します。

Web Server 6.1 SP2 については、以下を参照してください。

http://docs.sun.com/coll/s1_websvr61_en

Application Server 7.0 Update 3 については、以下を参照してください。

http://docs.sun.com/coll/s1_asse_en

3. *IdentityServer_base/SUNWam/bin/amsamplesilent* ファイルを、書き込み可能なディレクトリにコピーし、このディレクトリを現在のディレクトリにします。Linux システムでは、このファイルは *IdentityServer_base/identity/bin* ディレクトリにあります。たとえば、*/newinstances* というディレクトリを作成します。
4. 配備する新しいインスタンスを表すように *amsamplesilent* ファイルのコピーの名前を変更します。たとえば、Web Server 6.1 用の新しい Identity Server インスタンスを作成する場合は、ファイル名を *amnews6instance* に変更します。

後でこのインスタンスをアンインストールする必要がある場合は、新しい *amnews6instance* ファイルを保存します。

5. *amnews6instance* ファイルで、次の変数を設定します。

```
DEPLOY_LEVEL=1
NEW_INSTANCE=true
WEB_CONTAINER=WS6
```

amnews6instance ファイルで、作成する新しいインスタンスの必要に応じて他の変数を設定します。これらの変数の説明については、『Identity Server 2004Q2 管理ガイド』を参照してください。

重要 すべての Identity Server インスタンスでは、パスワード暗号化キーに同じ値を使用する必要があります。新しいインスタンスについて *AM_ENC_PWD* 変数を設定するには、最初のインスタンスの *AMConfig.properties* ファイルの *am.encrypted.pwd* プロパティから値をコピーします。

6. 新しい *amnews6instance* ファイルを入力値に指定して、*amconfig* スクリプトを実行します。Solaris システムの場合は次のようになります。

```
cd IdentityServer_base/SUNWam/bin/  
./amconfig -s /newinstances/amnews6instance
```

-s オプションを付けると、スクリプトはサイレントモードで実行されます。

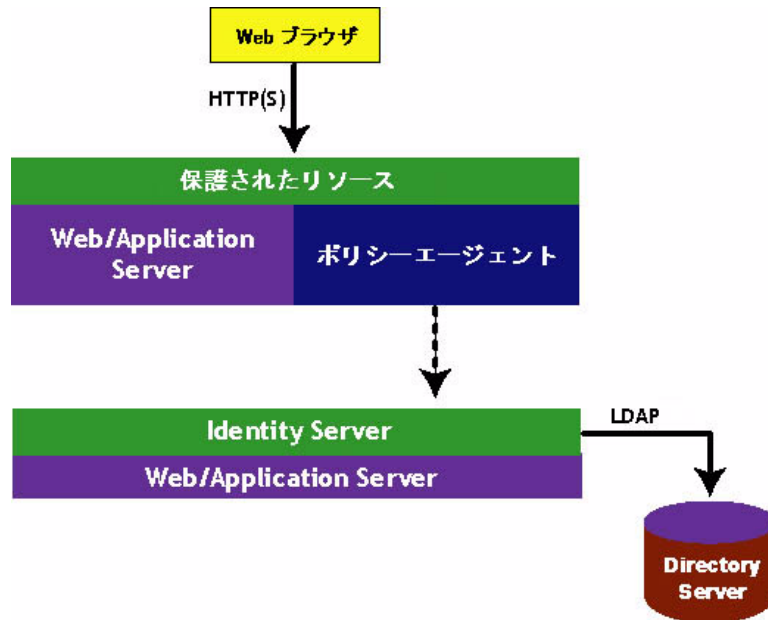
スクリプトは、amnews6instance ファイルの変数を読み込み、サイレントモードで実行され、新しい Web コンテナを設定します。

新しい Identity Server インスタンスの作成の詳細については、『Identity Server 2004Q2 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。図 5-2 に、この簡単な配備シナリオを示します。

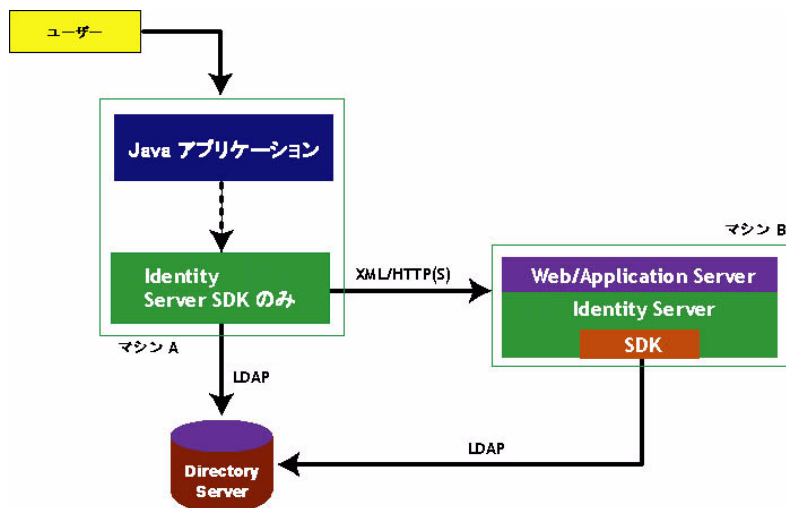
図 5-2 簡単な Web 配備シナリオ



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 Java System Web Server または Sun Java System 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。92 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 Java System Application Server のインスタンスは複数の JVM を保持するように設定できること、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、Java System 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 は、シングルスブライヤ設定またはマルチサブライヤ設定を行うことができます。Sun Java System Directory Proxy Server などのロードバランスアプリケーションも使用できます。Directory Proxy Server は、設定された Directory Server セット間の LDAP 操作のプロポーショナルなロードバランスを動的に実行します。1 つ以上の Directory Server 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、サブ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、シングル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シングル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

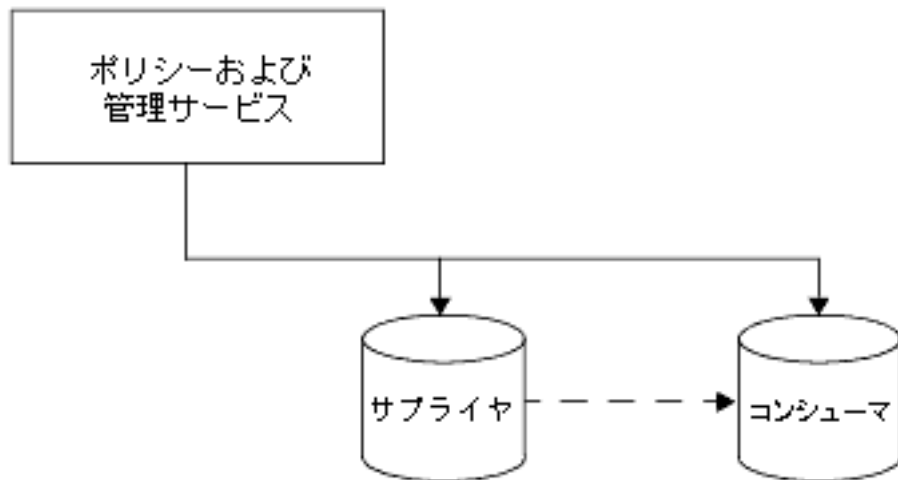
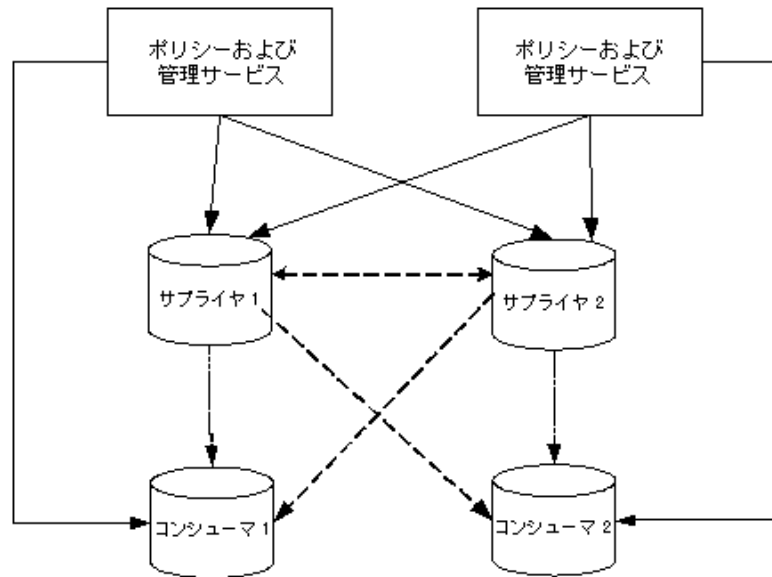


図 5-5 に、Identity Server の複数インスタンスを使用した複数サプライヤ設定を示します。この設定によりフェイルオーバー保護および高可用性が提供されるため、サーバーのパフォーマンスはさらに向上します。

図 5-5 複数サブライヤ設定 (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. サブ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 Java Enterprise System 2004Q2 インストールガイド』を参照してください。

2. サブ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、『Directory Server 5 2004Q2 管理ガイド』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、『Identity Server 2004Q2 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は、Sun Java Enterprise System 2004Q2 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ブライヤ **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、`AMConfig.properties` ファイルを修正します。
 - Solaris システム `IdentityServer_base/SUNWam/lib/AMConfig.properties`
 - Linux システム `IdentityServer_base/identity/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.ipplanet.am.directory.host`
 - `com.ipplanet.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.ipplanet.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.ipplanet.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 (サブライヤ) **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. 次のファイルで、以下の操作を行います。
`Identity_Server_base/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。
7. 次のコマンドを実行して、**Identity Server** を再起動します。

```
Identity_Server_base/SUNWam/bin/amserver start
```

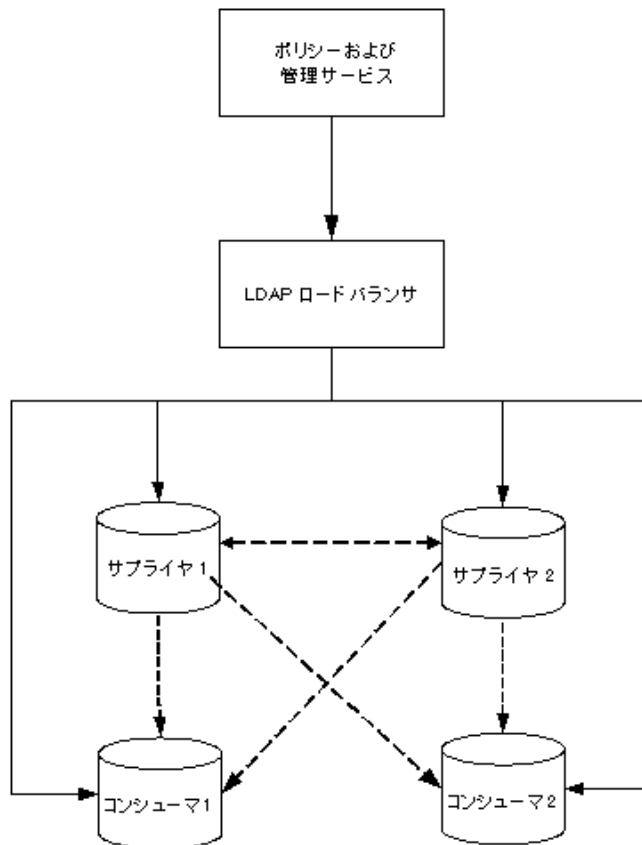
コード例 5-1 serverconfig.xml レプリケーションの修正

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

ロードバランサを使用する場合の設定

図 5-6 に、Directory Proxy Server を含む複数サブ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ブ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、各サーバーに再分配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、『Directory Server 5 2004Q2 管理ガイド』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 (Solaris システム) 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.ipplanet.am.directory.host
```

```
com.ipplanet.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 (Solaris システム) 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。
5. `amserver start` コマンドを実行して、Identity Server を再起動します。Solaris システムの場合は次のようになります。

```
IdentityServer_base/SUNWam/bin/amserver start
```

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" />
  </ServerGroup>
</iPlanetDataAccessLayer>
```

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 E](#)
「[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 Java System Directory Server 配備計画ガイド』を参照してください。

ファイアウォールを使用した Directory Server

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

Directory Server 接続がファイアウォールによって切断されると、Identity Server は、この接続が切断されていることを認識せず、LDAP 接続プールのすべての接続を使い果たします。LDAP 接続プールを新規に作成するには、Identity Server を再起動する必要があります。

この問題を回避するには、次の解決策を参考にしてください。

- [グローバルタイムアウト属性の設定](#)
- [個々のクライアント接続のタイムアウトの設定](#)

グローバルタイムアウト属性の設定

ファイアウォールのアイドル接続タイムアウト値を下回る値に対して、Directory Server の `nsslapd-idletimeout` グローバル属性を設定できます。ただし、この解決策は、`nsslapd-idletimeout` が、Identity Server 以外のアプリケーションにも影響するグローバル設定属性であるため、採用できない場合もあります。

個々のクライアント接続のタイムアウトの設定

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

LDAP ディレクトリにバインドした Identity Server ユーザーについて、`nsIdleTimeout` 属性を設定します。このユーザーは、デフォルトでは、`amldapuser` になっています。

`amldapuser` に `nsIdleTimeout` 属性を追加するには、Directory Server コンソールか、`ldapmodify` ツールを使用します。`ldapmodify` を使用するには、たとえば次のように指定します。

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

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

Directory Server 属性と `ldapmodify` ツールの詳細については、次の Web サイトにアクセスして、『Directory Server 5 2004Q2 管理ガイド』を参照してください。

<http://docs.sun.com/db/prod/entsys?l=ja>

Identity Server のセッションフェイルオーバー

この節では、Identity Server 2004Q2 のセッションフェイルオーバーについて説明します。以下の項から構成されています。

- [セッションフェイルオーバーの概要](#)
- [セッションフェイルオーバーの要件](#)
- [セッションフェイルオーバーの実装](#)

セッションフェイルオーバーの概要

Identity Server 2004Q2 では、Sun Java System Application Server 7.0.0_01 Enterprise Edition (EE) を Web コンテナとして使用した、セッションフェイルオーバーを提供しています。

注	Application Server 7.0.0_01 EE は、Sun Java Enterprise System 2004Q2 リリースのコンポーネントではありません。このリリースのコピーを入手するには、ご購入先の顧客サービスの担当者に問い合わせてください。
---	--

ハードウェアまたはソフトウェアの問題によりプライマリサーバーが機能しなくなった場合、または一時的に停止している場合に、セッションフェイルオーバーは、Identity Server 要求をセカンダリサーバーへ自動的かつ透過的にリダイレクトします。

たとえば、ユーザーが Identity Server を設定しているときに、プライマリサーバーが機能しなくなることがあります。セッションフェイルオーバーは、ユーザーのセッションをセカンダリサーバーへ自動的に転送し、ユーザーは再度認証することなく、セッションを継続できます。

Identity Server 2004Q2 では、次のようにセッションフェイルオーバーをサポートします。

- 認証セッションの状態を保存する
- 持続データ層に LDAP レプリケーションを使用する
- Identity Server セッション状態のレプリケーションに、Application Server コンテナの HTTP セッションフェイルオーバー機能を使用する
- ロードバランス機能を使用して、要求を分散し、クライアントを個々のサーバー障害から保護する

セッションフェイルオーバーの要件

この章で説明するセッションフェイルオーバーの実装には、Identity Server 2004Q2 が必要です。それ以外に次の要件があります。

- サポートするプラットフォーム
 - Solaris 8 または 9 オペレーティングシステム (SPARC® Platform Edition)
 - Solaris 9 オペレーティングシステム (x86 Platform Edition)

- Application Server 7.0.0_01 EE

Application Server 7 EE のマニュアルは、次の Web サイトで入手できます。

<http://docs.sun.com/db/prod/entsys?l=ja>

- サポートする Web Server のバージョン
 - Web Server 6.1 SP2、ロードバランサプラグインを使用しない場合
 - Web Server 6.0 SP 6、ロードバランサプラグインを使用する場合

Web Server 6.0 SP6 は、Sun Java Enterprise System 2004Q2 リリースのコンポーネントではありません。この製品の詳細については、次の Web サイトの説明を参照してください。

http://docs.sun.com/coll/S1_ipwebsrvree60_en

セッションフェイルオーバーの実装

この項では、Identity Server 2004Q2 でのセッションフェイルオーバーの実装方法について説明します。

1. [Web Server 6.0 SP6 のインストール \(ロードバランサプラグイン用\)](#)
2. [Application Server 7.0.0_01 EE のインストールおよび設定](#)
3. [Identity Server インスタンスのインストール](#)
4. [Identity Server 2004Q2 の設定](#)

Web Server 6.0 SP6 のインストール (ロードバランサプラグイン用)

Application Server 7.0.0_01 EE のインストール時に、ロードバランサプラグインをインストールする予定がある場合は、プラグインのインストール先のサーバーに、Web Server 6.0 SP 6 をインストールする必要があります。Sun ONE Web Server のインストールおよび設定については、次の Web サイトにアクセスして、『Web Server Enterprise Edition Installation Guide』を参照してください。

http://docs.sun.com/coll/S1_ipwebsrvree60_en

Application Server 7.0.0_01 EE のインストールおよび設定

最低 2 台の管理されたサーバーを使用して、Application Server 7.0.0_01 EE をクラスター環境にインストールして設定します。インストールの詳細については、次の Web サイトにアクセスして、『Application Server 7, Enterprise Edition インストールガイド』を参照してください。

<http://docs.sun.com/db/prod/entsys?l=ja>

Application Server 7.0.0_01 EE のインストールでの考慮事項は、次のとおりです。

- [以前にインストールした Application Server 7 のバージョン](#)
- [インストールディレクトリ](#)
- [コンポーネント選択ページ](#)
- [Java 設定パネル](#)
- [設定ディレクトリ](#)
- [Web Server](#)
- [Application Server コンポーネントオプション](#)
- [インストール後のタスク](#)

以前にインストールした Application Server 7 のバージョン

以前のバージョンの Application Server 7 を対象のサーバーにインストールしている場合は、Application Server 7.0.0_01 EE をインストールする前に、アンインストールプログラムを使用して以前のバージョンを削除する必要があります。

Solaris 9 OS のバンドル版またはパッケージベースでない評価版をインストールしていても、Application Server 7.0.0_01 EE のインストールプログラムに影響しません。したがって、これらを削除する必要はありません。ただし、ポート番号が競合している場合は、すべて解決する必要があります。また、バンドル版の Application Server 7 インスタンスも、インストールする前に停止させる必要があります。

インストールディレクトリ

デフォルトのインストールディレクトリは、/opt/SUNWappserver7 ですが、必要に応じて別のディレクトリも指定できます。

コンポーネント選択ページ

ロードバランサプラグイン: コンポーネント選択ページで、ロードバランサプラグインをインストールするには、Web Server 6.0 SP 6 を前もってインストールする必要があります。詳細については、[103 ページの「Web Server 6.0 SP6 のインストール \(ロードバランサプラグイン用\)」](#)を参照してください。

最初の Application Server インスタンス用にロードバランサプラグインをインストールした場合、追加インスタンスをインストールするときに、このプラグインを選択する必要はありません。

高可用性データベースサーバー：コンポーネント選択ページでは、ODBC ドライバをインストールするために、「高可用性データベース」を選択し、「高可用性データベース管理クライアント」は選択しません。「高可用性データベース管理クライアント」を選択すると、ODBC ドライバがインストールされず、後にアプリケーションを実行したときに問題が生じます。

Java 設定パネル

J2SE バージョン：Java 設定パネルでは、Java 2 SDK 1.4.2_03 のバージョンを選択します。

設定ディレクトリ

製品の設定ディレクトリは、デフォルトで /etc/opt/SUNWappserver7 になっています。サーバーの設定ディレクトリは、デフォルトで /var/opt/SUNWappserver7 になっています。

デフォルトの設定を使用することも、必要に応じて別のディレクトリを指定することもできます。

Web Server

Web Server：ロードバランサプラグインを選択した場合、Web Server Directory ページが表示されます。Web Server 6.0 SP 6 を選択してから、Web Server インスタンスのパスを指定します。

Application Server コンポーネントオプション

Application Server のオプションは次のとおりです。

- 管理ユーザーの名前とパスワード。デフォルトのユーザー名は admin
- 管理サーバーポート。デフォルトは 4848
- HTTP サーバーポート。デフォルトは 80

インストールプログラムは、管理サーバーポートと HTTP サーバーポートのデフォルトのポートが使用されていない場合に、そのポート番号を表示します。必要に応じてポート番号を変更してください。パネルの「次へ」をクリックすると、インストールプログラムは、ポート番号が有効で使用可能かどうかを調べます。

インストール後のタスク

Application Server 7.0.0_01 EE をインストールした後、『Application Server 7, Enterprise Edition インストールガイド』(<http://docs.sun.com/db/prod/entsys?l=ja>) の第 3 章「HADB の設定準備」を参照してください。

第3章では、「共有メモリーとセマフォの設定」、「ホスト通信の設定」、「ユーザー環境の設定」、および「`clsetup` コマンドの使用」など、Application Server HADB を使用して実行する必要のあるタスクについて説明しています。

Application Server クラスタ、ロードバランサプラグイン、および高可用性データベース (HADB) の設定と管理に関する情報については、『Application Server 7, Enterprise Edition 管理者ガイド』(<http://docs.sun.com/db/prod/entsys?l=ja>) を参照してください。

特に、第16章「ロードバランスの設定」が参考になります。

sun-web.xml ファイルの編集

次の各ディレクトリの `sun-web.xml` ファイルを編集します。

- `/var/opt/SUNWappserver7/domains/domain1/server1/applications/j2ee-modules/amserver_1/WEB-INF`
- `/var/opt/SUNWappserver7/domains/domain1/server1/applications/j2ee-modules/amconsole_1/WEB-INF`
- `/var/opt/SUNWappserver7/domains/domain1/server1/applications/j2ee-modules/ampassword_1/WEB-INF`

プロパティの変更: `sun-web.xml` ファイルの以下のプロパティを、ボールド書体で示すように変更します。

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright 2004 Sun Microsystems, Inc. All rights reserved.-->

<!DOCTYPE sun-web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Sun
Java System Application Server 7.0 Servlet 2.3//EN"
'http://www.sun.com/software/sunone/appserver/dtds/sun-web-app_2_3-
0.dtd'>

<?xml version="1.0" encoding="UTF-8"?>
...
<sun-web-app>
<property name="encodeCookies" value="false" />
  <locale-charset-info>
    <parameter-encoding form-hint-field="gx_charset" />
  </locale-charset-info>
  <session-config>
    <cookie-properties>
      <property name="cookiePath" value="/" />
    </cookie-properties>
  </session-config>
  <resource-env-ref>
    <resource-env-ref-name>jdbc/SessionRepository</resource-env-r
ef-name>
```

```

        <jndi-name>jdbc/hastore</jndi-name>
    </resource-env-ref>
<sun-web-app>
...

```

ボールド書体で示すように、各 sun-web.xml ファイルに、<distributable/> 行を追加します。

```

<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.3//EN" "file:/etc/opt/SUNWam/dtd/web-app_2_3.dtd">
<web-app>
    <display-name>iDSAME Services</display-name>
    <bistributable/>
    <context-param>
        <param-name>Webmaster</param-name>
        <param-value>webmaster@sun.com</param-value>
    </context-param>

```

各 sun-web.xml ファイルの次の行をコメント解除します。

```

<resource-env-ref>
<description>DataSource Reference</description>
    <resource-env-ref-name>jdbc/SessionRepository</resource-env-ref-n
ame>
    <resource-env-ref-type>javax.sql.DataSource</resource-env-ref-type>
</resource-env-ref>

```

Identity Server インスタンスのインストール

Application Server Web コンテナごとに、Identity Server 2004Q2 のインスタンスをインストールし設定します。

- Sun Java Enterprise System インストーラを実行して、Identity Server 2004Q2 の最初のインスタンスをインストールします。インストーラを実行するときに、Java Enterprise System 2004Q2 の他のコンポーネント製品も必要に応じてインストールできます。

このインストーラの詳細については、『Sun Java Enterprise System 2004Q2 インストールガイド』を参照してください。

- Identity Server インストールおよび設定スクリプトとサイレントインストールファイルを使用して、Identity Server 2004Q2 の追加インスタンスを作成します。

Identity Server 2004Q2 インストールおよび設定スクリプトと `amsamplesilent` ファイルは、`IdentityServer_base/SUNWam/bin` ディレクトリにあります。Linux システムでは、このファイルは `IdentityServer_base/identity/bin` ディレクトリにあります。Identity Server 2004Q2 のインスタンスをインストールするには、次の手順に従ってください。

1. スーパーユーザー (root) としてログインするか、スーパーユーザーに変更します。
2. `amsamplesilent` ファイルを書き込み可能なディレクトリにコピーし、このディレクトリを現在のディレクトリにします。たとえば、`/newinstances` というディレクトリに、ファイルをコピーします。
3. 配備する新しいインスタンスがわかるように、`amsamplesilent` ファイルのコピーの名前を変更します。たとえば、Application Server 用の新しいインスタンスを作成するので、ファイルの名前を `amnewAS7instance` に変更します。
4. `amnewAS7instance` ファイルを編集し、変数を設定して、Application Server 用の新しい Identity Server インスタンスを作成します。例を示します。

```
DEPLOY_LEVEL=1
NEW_INSTANCE=true
WEB_CONTAINER=AS7
DIRECTORY_MODE=1 (新規インストールの場合)
```

`amnewAS7instance` ファイルで、作成する新しい Application Server インスタンスの必要に応じて他の変数を設定します。これらの変数の説明については、『Identity Server 2004Q2 管理ガイド』を参照してください。

重要 すべての Identity Server インスタンスでは、パスワード暗号化キーに同じ値を使用する必要があります。新しいインスタンスについて `AM_ENC_PWD` 変数を設定するには、最初のインスタンスの `/etc/opt/SUNWam/config/AMConfig.properties` ファイルの `am.encryption.pwd` プロパティから値をコピーします。

5. 新しい `amnewAS7instance` ファイルを入力ファイルに指定して、`amconfig` スクリプトを実行します。Solaris システムの場合は次のようになります。

```
cd IdentityServer_base/SUNWam/bin/
./amconfig -s /newinstances/amnewAS7instance
```

`-s` オプションを付けると、スクリプトはサイレントモードで実行します。

スクリプトは、`amnewAS7instance` ファイルの変数を読み込み、サイレントモードで実行し、新しい Web コンテナを設定します。

作成する追加 Identity Server インスタンスごとに、以上の手順を繰り返します。

Identity Server 2004Q2 の設定

Identity Server を設定するには、次のタスクを順番どおりに実行します。

1. 最初のインスタンスのコンソールでサーバーの名前と URL を追加する
2. 各インスタンスの `AMConfig.properties` ファイルを修正する

最初のインスタンスのコンソールでサーバーの名前と URL を追加する

最初の Identity Server インスタンスにログインして、次の手順を行います。

1. ブラウザから、最初の Identity Server インスタンスのコンソールにアクセスし、`amadmin` としてログインします。例を示します。

```
http://host-name.domain-name:port/amconsole
```

ただし、`host-name.domain-name:port` は、最初の Identity Server インスタンスの完全修飾されたホスト名とポートです。

2. 「アイデンティティ管理」タブで、ロードバランサのサーバー名など、クラスタ内のサーバー名をすべて「組織のエイリアス」に追加します。
3. 「サービス設定」タブで、「プラットフォーム」をクリックし、ロードバランサの URL など、サーバーの URL をすべて「サーバーリスト」に追加します。各サーバー URL の形式は次のようになります。

```
http://host-name.port | ServerID
```

ただし、

`host-name.port` は、完全修飾されたホスト名とポート番号です。

`ServerID` は、インスタンスを一意に識別する 2 バイト値です。

デフォルトは、最初のインスタンスの値になります。

たとえば、4 つの Identity Server インスタンスと 1 つのロードバランサがある場合は次のようになります。

```
http://IS1.example.com:8080|01
http://IS2.example.com:8080|02
http://IS3.example.com:8080|03
http://IS4.example.com:8080|04
http://lbhost.example.com:80|05
```

各インスタンスの `AMConfig.properties` ファイルを修正する

次のように、Identity Server インスタンスごとに、

`/etc/opt/SUNWam/config/AMConfig.properties` ファイルを修正します。

1. セッションフェイルオーバーを有効にするには、次のプロパティを設定します。

```
com.ipplanet.am.session.failover.enabled=true
com.ipplanet.am.cookie.encode=false
```

2. サーバーインスタンスを識別するために、次のプロパティを追加します。

```
com.ipplanet.am.session.failover.cluster.serverList=01,02,03,04
```

ただし、01,02,03,04 は、Identity Sever インスタンスを一意に識別する 2 バイトの値です。ロードバランスに組み入れられる各サーバーには、重複しない識別名を付ける必要があります。

ただし、ロードバランサ自体の識別名は追加しません。

3. ロードバランサを指し示すように、サーバーホスト、ポート、プロトコル、他の URL を設定します。例を示します。

```
com.ipplanet.am.server.host=lbhost.example.com
com.ipplanet.am.server.port=lbport
com.ipplanet.am.console.host=lbhost.example.com
com.ipplanet.am.console.port=lbport
com.ipplanet.am.profile.host=lbhost.example.com
com.ipplanet.am.profile.port=lbport
com.ipplanet.am.naming.url=http://lbhost.example.com:lbport/amserver/naming
service
```

注 次のプロパティには、ロードバランサを指し示すような設定 (または修正) を行わないでください。

```
com.ipplanet.am.notification.url=http://IS1.example.com:8080/amserver/
notification
service
com.ipplanet.am.localserver.protocol=http
com.ipplanet.am.localserver.host=IS.example.com
com.ipplanet.am.localserver.port=8080
com.ipplanet.am.admin.cli.certdb.prefix=https-IS1.example.com-IS1
com.ipplanet.am.directory.host
com.ipplanet.am.directory.port
```

Identity Server および Portal Server の配備

Java Enterprise System 2004Q2 リリースでは、Identity Server 2004Q2 と Portal Server 6 2004Q2 を同じ物理サーバーにインストールするか、複数のサーバーにインストールできます。

単一のサーバーへのインストール

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

これらのコンポーネントをインストールするには、Java Enterprise System インストーラを単独のバージョンで実行するか、以下を選択します。

- コンポーネントの選択パネルで、これらの製品とサブコンポーネントを選択します。
 - Communication & Collaboration Services で、Portal Server 6.3 を選択します。
 - Directory & Identity Services で、Identity Server 6.2 とそのサブコンポーネントを選択します。
 - アイデンティティ管理およびポリシーサービスコア
 - Identity Server 管理コンソール
 - 連携管理の共有ドメインサービス
 - Identity Server SDK

重要 デフォルトでは、Portal Server 6.3 を選択するとインストーラは Identity Server SDK だけを選択します。したがって、その他のサブコンポーネントをチェックする必要があります。

- 次の Web コンポーネントの 1 つをインストールし、設定します。
 - Application Server 7.0 Update 3
 - Web Server 6.1 SP2

複数のサーバーへのインストール

このシナリオでは、Portal Server は、リモートサーバーからローカルサーバー上の Identity Server にアクセスします。ローカルサーバーまたはリモートサーバーに、Directory Server をインストールするか、インストールされた Directory Server のバージョンにアクセスする必要があります。

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

詳細については、次のマニュアルを参照してください。

- インストーラの実行：『Sun Java Enterprise System 2004Q2 インストールガイド』(<http://docs.sun.com/db/prod/entsys?l=ja>)
- Portal Server の配備：『Sun Java System Portal Server 6 2004Q2 Deployment Guide』(<http://docs.sun.com/doc/817-5321>)
- Identity Server の設定：『Sun Java System Identity Server 2004Q2 管理ガイド』(<http://docs.sun.com/db/prod/entsys?l=ja>)

連携管理

ここでは、Identity Server の連携管理機能を使用した配備設定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。

連携管理の詳細については、『**Identity Server 2004Q2 Federation Management Guide**』を参照してください。

インストールされる製品のレイアウト

この付録では、Sun Java System Enterprise インストーラを使用して Identity Server 2004Q2 をインストールした後のディレクトリレイアウトについて説明します。次のディレクトリがあります。

- 116 ページの「ベースインストールディレクトリ」
- 116 ページの「製品ディレクトリ」
 - /agents ディレクトリ
 - /bin ディレクトリ
 - /dtd ディレクトリ
 - /include ディレクトリ
 - /ldaplib/ ディレクトリ
 - /lib ディレクトリ
 - /locale ディレクトリ
 - /migration ディレクトリ
 - /public_html ディレクトリ
 - /samples ディレクトリ
 - /share ディレクトリ
 - /web-src ディレクトリ
- 122 ページの「/debug、/logs、および /tmp ディレクトリ」
- 122 ページの「設定 (/config) ディレクトリ」

ベースインストールディレクトリ

デフォルトのベースインストールディレクトリは、Identity Server 2004Q2 をインストールしたプラットフォームにより異なります。

- Solaris システム /opt
- Linux システム /opt/sun

Identity Server のマニュアルの中では、*IdentityServer_base* 変数を使用してベースインストールディレクトリを示しています。

製品ディレクトリ

ベースインストールディレクトリ内で、Identity Server 2004Q2 パッケージ、共有バイナリファイル、コマンド行ツール、およびさまざまなその他のファイルは、Solaris システムの /SUNWam ディレクトリ、および Linux システムの /identity ディレクトリにインストールされます。したがって、デフォルトの製品ディレクトリは、プラットフォームによって異なります。

- Solaris システム /opt/SUNWam
- Linux システム /opt/sun/identity

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

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

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

WAR ファイルの詳細については、『Identity Server 2004Q2 Developer's Guide』を参照してください。

- サブディレクトリ：

- [/agents ディレクトリ](#)
- [/bin ディレクトリ](#)
- [/docs ディレクトリ](#)
- [/dtd ディレクトリ](#)
- [/include ディレクトリ](#)
- [/ldaplib/ ディレクトリ](#)
- [/lib ディレクトリ](#)
- [/locale ディレクトリ](#)
- [/migration ディレクトリ](#)
- [/public_html ディレクトリ](#)
- [/samples ディレクトリ](#)
- [/share ディレクトリ](#)
- [/web-src ディレクトリ](#)

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

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

/agents ディレクトリ

/agents 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。これらの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。

/bin ディレクトリ

表 A-1 では、/bin ディレクトリのコマンド行ツールおよびユーティリティについて説明しています。詳細については、『Identity Server 2004Q2 管理ガイド』を参照してください。

表 A-1 Identity Server のコマンド行ツールおよびユーティリティ

ユーティリティ	説明
am2bak	Identity Server コンポーネントをバックアップする
amadmin	Directory Server への XML サービスファイルをロードし、DIT 上でのバッチ管理タスクを実行する
ampassword	Identity Server 管理者またはユーザーのパスワードを変更する
amsamplesilent	インストールスクリプトおよび設定スクリプトで使用するサンプルのサイレントインストールファイル
amconfig、amutils、amdsconfig、amsdkconfig、amsvconfig、amas70config、amwas51config、amwl81config、amws61config	Identity Sever インスタンスのインストール、設定、およびアンインストールに使用する、インストールおよび設定スクリプト。これらのスクリプトの詳細については、『Identity Server 2004Q2 管理ガイド』を参照
amserver	Identity Server インスタンスを起動および停止する
amtune	パフォーマンスを改善するように、オペレーティングシステム、Identity Server、Web コンテナ、および Directory Server パラメータを設定する
amverifyarchive	ログアーカイブを調べて、アーカイブ内のすべてのファイルの改ざんや削除を検出する
bak2am	am2back ユーティリティでバックアップした Identity Server コンポーネントを復元する
ldapmodify	新規エントリを追加するか、既存のエントリを変更して、LDAP ディレクトリの内容を編集する
ldapsearch	LDAP ディレクトリに検索要求を発行し、結果を LDIF テキストで表示する

/docs ディレクトリ

/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ファイルが含まれます。これらのファイルには、allclasses-frame.html、am_public_javadocs.jar、com ディレクトリ、deprecated-list.html、help-doc.html、index-all.html、index.html、META-INF ディレクトリ、overview-frame.html、overview-summary.html、overview-tree.html、package-list、packages.html、serialized-form.html、および stylesheet.css があります。

/dtd ディレクトリ

/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。詳細は、『Sun Java System Identity Server Developer's Guide』を参照してください。表 A-2 では /dtd ディレクトリの DTD ファイルについて説明しています。

表 A-2 Identity Server DTD ファイル

ファイル	説明
Auth_Module_Properties.dtd	認証モジュールがプロパティの指定に使用する 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 配備コンテナが J2EE アプリケーションを配備する際に使用する XML ファイルの構造を定義する

/include ディレクトリ

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

/ldaplib/ ディレクトリ

/ldaplib/ldapsdk サブディレクトリには、Identity Server に含まれる 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。

/lib ディレクトリ

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

/locale ディレクトリ

/locale ディレクトリには、地域対応化プロパティファイルが含まれます。各プロパティファイルには、英語版の対応するファイルが含まれます。たとえば、amAdminCLI_en.properties は amAdminCLI.properties に対応するファイルです。

/migration ディレクトリ

/migration ディレクトリには、以前のバージョンの Identity Server からのデータの移行に使用するスクリプトとサポートファイルが含まれます。たとえば、/opt/SUNWam/migration/61to62/scripts サブディレクトリには、DIT を Identity Server 2004Q2 へ移行する場合に使用する Upgrade61DitTo62 スクリプトが含まれます。

移行の詳細については、『Java System Identity Server Migration Guide』を参照してください。

/public_html ディレクトリ

/public_html ディレクトリとサブディレクトリには、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。

/samples ディレクトリ

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

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

/share ディレクトリ

/share ディレクトリには、Identity Server 内部で使用する次の追加ユーティリティを収録した、bin/ サブディレクトリが含まれます。

- amtune/amtune-utils
- amsecuridd、amunixd、amwar、checkport、および wsutils.ksh

/web-src ディレクトリ

web-src ディレクトリには、Web コンテナ上での 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。次のサブディレクトリが含まれます。

- applications/ は、Identity Server Console の導入先ディレクトリ。index.html と、/META-INF、/WEB-INF、および /console のサブディレクトリが含まれる
/console ディレクトリには、/auth、/base、/federation、/images、/js directory、/policy directory、/service、/session、および /user のサブディレクトリが含まれる
- /common は、Identity Server Liberty Common Domain コンポーネントの導入先ディレクトリ。次のサブディレクトリが含まれる。/META-INF および /WEB-INF
- /password は、Identity Server Password Synchronization コンポーネントの導入先ディレクトリ。index.html と、/META-INF、/WEB-INF、および /password のサブディレクトリが含まれる
- /services は、Identity Server Core Service の導入先ディレクトリ。index.html と、/META-INF、/WEB-INF、/admin、/config、/css、/docs、/fed_css、/fed_images、/images、/js、および /login_images のサブディレクトリが含まれる

/debug、/logs、および /tmp ディレクトリ

/debug、/logs、および /tmp のデフォルトの場所は、Identity Server 2004Q2 をインストールしたプラットフォームによって異なります。

- Solaris システム /var/opt/SUNWam
- Linux システム /var/opt/sun/identity

これらのディレクトリの詳細については、『Identity Server 2004Q2 管理ガイド』を参照してください。

設定 (/config) ディレクトリ

デフォルトの設定ディレクトリ (/config) の場所は、Identity Server 2004Q2 をインストールしているプラットフォームによって異なります。

- Solaris システム /etc/opt/SUNWam
- Linux システム /etc/opt/sun/identity

/config ディレクトリには、次のような設定ファイル、XML ファイル、および LDIF ファイルが収められています。

- .version ファイルには、Identity Server の現在のバージョンが記述されています。
- AMConfig.properties ファイル、SSOConfig.properties、および LogConfig.properties には、Identity Server 設定属性が含まれます。
- /ldif サブディレクトリには、Identity Server のインストール時に、Directory Server データストアを生成するために必要な LDIF ファイルが含まれます。例を示します。
 - インストール時に、ds_remote_schema.ldif ファイルは、Identity Server のデータを Directory Server に格納するために必要な 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) をロードします。sunone_schema2.ldif 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します。
 - アンインストール時に、ds_remote_schema_uninstall.ldif ファイルは、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、Directory Server から削除します。
- /ums サブディレクトリには、次のような XML ファイルが含まれます。
 - serverconfig.xml ファイルは、Identity Server の設定情報を Directory Server に提供します。

- 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 設定情報が含まれます。
- /xml サブディレクトリには、XML ファイルが含まれます。通常これらの XML ファイルは、設定には使用されません。これらを修正した場合は、Directory Server データストアに手動で再ロードする必要があります (サーバーでの変更は一切、これらのファイルと同期しない)。このディレクトリの XML ファイルの詳細については、『Identity Server 2004Q2 Developer's Guide』を参照してください。

設定 (/config) ディレクトリ

ユーザーセッションのライフサイクル

Sun Java™ System 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。

- [125 ページの「概要」](#)
- [126 ページの「要求」](#)
- [127 ページの「認証」](#)
- [129 ページの「セッショントークン」](#)
- [133 ページの「ポリシー」](#)
- [135 ページの「要求されたページ」](#)
- [137 ページの「シングルサインオン要求」](#)
- [144 ページの「セッションの終了」](#)

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 B-1](#) に、ブラウザから送信される GET 要求を示します。

コード例 B-1 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 B-2](#) 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 B-2 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 B-3](#) 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 B-3 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 も含まれる)。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とも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は、『**Identity Server 2004Q2 Developer's Guide**』を参照してください。

コード例 B-4 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、省略されています。

コード例 B-4 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-dsnameversion: 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 B-5 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 B-5 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 B-6 に、このリダイレクト応答を示します。

コード例 B-6 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-dsnameversion: 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 B-7

トークンととも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 B-8](#) に、ネーミング情報の POST 要求を示します。

コード例 B-8

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.ipplanet.am.naming" reqid="9">
<Request><![CDATA[
<NamingRequest vers="1.0" reqid="2"
sessid="AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=">
<GetNamingProfile>
</GetNamingProfile>
</NamingRequest>]]>
</Request>
</RequestSet>
```

[コード例 B-9](#) 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 B-9

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.ipplanet.am.naming" reqid="9">
<Response><![CDATA[<NamingResponse vers="1.0" reqid="2">
<GetNamingProfile>
<Attribute name="ipplanet-am-naming-policy-url"
value="http://identityserver.sun.com:58081/amserver/policyservic
e"></Attribute>
<Attribute name="ipplanet-am-naming-session-class"
value="com.ipplanet.dpro.session.service.SessionRequestHandler"><
/Attribute>
<Attribute name="ipplanet-am-naming-session-url"
value="http://identityserver.sun.com:58081/amserver/sessionservi
ce"></Attribute>
<Attribute name="ipplanet-am-naming-samlawareervlet-url"
value="http://identityserver.sun.com:58081/amserver/SAMLAwareSer
vlet"></Attribute>
<Attribute name="serviceObjectClasses"
value="ipplanet-am-naming-service"></Attribute>
<Attribute name="ipplanet-am-naming-auth-url"
value="http://identityserver.sun.com:58081/amserver/authservice"
></Attribute>
<Attribute name="ipplanet-am-naming-profile-class"
value="com.ipplanet.dpro.profile.agent.ProfileService"></Attribut
e>
<Attribute name="ipplanet-am-naming-samlassertionmanager-url"
value="http://identityserver.sun.com:58081/amserver/AssertionMan
agerServlet/AssertionManagerIF"></Attribute>
<Attribute name="ipplanet-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="ipplanet-am-naming-policy-class"
value="com.sun.identity.policy.remote.PolicyRequestHandler"></At
tribute>
<Attribute name="ipplanet-am-naming-logging-class"
value="com.sun.identity.log.service.LogService"></Attribute>
<Attribute name="ipplanet-am-naming-profile-url"
value="http://identityserver.sun.com:58081/amserver/profileservi
ce"></Attribute>
<Attribute name="ipplanet-am-naming-samlsoapreceiver-url"
value="http://identityserver.sun.com:58081/amserver/SAMLSOAPRece
iver"></Attribute>

```

コード例 B-9 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-samlpostervlet-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 B-10 に、セッションサービスへの POST 要求を示します。

コード例 B-10 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/sessionsservice 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>AQIC5wM2LY4SfcywF1TefDRmqlthG54qrg27LiyS8LnHAj4=</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>AQIC5wM2LY4SfcywF1TefDRmqlthG54qrg27LiyS8LnHAj4=</Ses
sionID>
</AddSessionListener>

```

コード例 B-10 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。このアプリケーションは、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 B-11 に、セッションサービスの応答を示します。

コード例 B-11 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 B-11 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。この要求とともに、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 B-12 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 B-12 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 B-12 ポリシー情報を求める POST 要求 (続き)

```

<GetResourceResults
userSSOToken="AQIC5wM2LY4SfcywFlTefDRmq1thG54qrg27LiyS8LnHAj4="
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。ポリシーがキャッシュに書き込まれていない場合は、Directory Server から読み込まれます。

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。リソースに一致するポリシーが検出され、ユーザーが有効なサブジェクトである場合は、ポリシーの追加条件が評価されます (たとえば、正しい日時であるか、正しいネットワークからのものであるかなど)。すべての条件が満たされた場合は、ポリシーサービスがそのユーザーにアクセス許可を与えることを決定し、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 B-13 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 B-13 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 B-13

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とも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 B-14 にログ要求を示します。

コード例 B-14

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

```

コード例 B-14 ポリシーエージェントからのログ要求 (続き)

```
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><logRecord><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 B-15 に応答を示します。

コード例 B-15 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">
<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 B-16 に、要求されたリソースの HTML ページを示します。HTML 自体は、省略されています。

コード例 B-16 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 B-17](#) 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 B-17

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、[127 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、[129 ページの「セッショントークン」](#)、[133 ページの「ポリシー」](#) および [135 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、[129 ページの「セッショントークン」](#)、[133 ページの「ポリシー」](#) および [135 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 B-18](#)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 B-18 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 B-19](#)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 B-19 セッションサービスへの POST 要求

```
POST /amserver/sessionsservice HTTP/1.0

...
<SessionID>AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4=</SessionID>
....

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">
....
```

- 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 B-20 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 B-20 ポリシーサービスへの POST 要求

```
POST /amserver/policyservice HTTP/1.0

...
<GetResourceResults
userSSOToken="AQIC5wM2LY4SfcywFlTefDRmq1thG54qrg27LiyS8LnHAj4="
serviceName="iPlanetAMWebAgentService"
resourceName="http://webservice.sun.com:8090/"
resourceScope="subtree">
...
```

- 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 B-21 に、この応答を示します。

コード例 B-21 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>
```

- 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 B-22 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 B-22

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><logRecord><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 B-23 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 B-23

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 CDSO (Cross Domain Single Sign-On) プロトコルの役割です。[コード例 B-24](#) 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 B-24 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 CDSO 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 CDSO 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 B-25](#) 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 B-25 ブラウザ経由で行われる CDSO 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=http%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&refererservlet=http%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&RequestID=29980&MajorVersion=1&MinorVersion=0&ProviderID=http%3A%2F%2Fwebservice.java.com%3A8088%2Famagent&IssueInstant=2003-10-02T04%3A25%3A06Z&ForceAuthn=false&IsPassive=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 B-26 に、ブラウザから CDSO 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 B-26 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/cdcervlet?goto=http%3A%2F%2Fwebsevice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&refererservlet=http%3A%2F%2Fwebsevice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&RequestID=29980&MajorVersion=1&MinorVersion=0&ProviderID=http%3A%2F%2Fwebsevice.java.com%3A8088%2Famagent&IssueInstant=2003-10-02T04%3A25%3A06Z&ForceAuthn=false&IsPassive=false&Federate=false
HTTP/1.1
Host: identityserver.sun.com:58081
Cookie:
iPlanetDirectoryPro=AQIC5wM2LY4SfcywFlTefDRmqlthG54qrg27LiyS8LnHAj4%3D
```

4. CDSO 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 B-27 に、この返信を示します。

コード例 B-27 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://websevice.java.com:8088/index.html?sunwMethod=GET">
<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 B-28 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 B-28 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=http%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&refererservlet=http%3A%2F%2Fwebservice.java.com%3A8088%2Findex.html%3FsunwMethod%3DGET&RequestID=29980&MajorVersion=1&MinorVersion=0&ProviderID=http%3A%2F%2Fwebservice.java.com%3A8088%2Famagent&IssueInstant=2003-10-02T04%3A25%3A06Z&ForceAuthn=false&IsPassive=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、129 ページの「セッショントークン」、133 ページの「ポリシー」、および 135 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 129 ページの「セッショントークン」、説明を簡略化するため、133 ページの「ポリシー」および 135 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 B-29 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る)。

コード例 B-29

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=AQIC5wM2LY4Sfcwwte3peDKZILScZ40uK%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 B-30](#) 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 B-30

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=AQIC5wM2LY4SfcywF1TefDRmqlthG54qrg27LiyS8LnH
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 B-31](#) 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る)。

コード例 B-31

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 B-32

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="AQIC5wM2LY4SfcywFlTefDRmq1thG54qrg27LiyS8LnHAj4="
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>

```

コード例 B-32

セッション通知用の POST (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>
<Type>5</Type>
<Time>1065058714469</Time>
</SessionNotification>]]></Notification>
</NotificationSet>
```

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

```
....
<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 Java™ System 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 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。

- [147 ページの「概要」](#)
- [148 ページの「Active Directory 認証の設定」](#)
- [151 ページの「トラブルシューティング」](#)

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場合には、2 つの選択肢を考慮します。管理者は、この目的のために Active Directory に対するすべての LDAP 認証か、新規の認証モデルのいずれかを作成して、登録できます。

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 Java System 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 サフィックスが異なる場合に使用します。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『Identity Server 2004Q2 Developer's 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。このサービスの詳細は、『Identity Server 2004Q2 Developer's 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ィー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 Java System 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 `userPrincipalName` 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.ipplanet.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ount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。ipplanetKeyValue 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。

- ipplanet-am-auth-ldap-server
- ipplanet-am-auth-ldap-base-dn
- ipplanet-am-auth-ldap-user-naming-attribute
- ipplanet-am-auth-ldap-user-search-attributes
- ipplanet-am-auth-ldap-return-user-dn
- ipplanet-am-auth-ldap-bind-dn
- ipplanet-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。
---	--

chroot 環境のインストール

Sun Java™ System Identity Server の chroot (changed root) 環境へのインストールは、Solaris™ Operating System の chroot 機能に基づいています。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新規のルートディレクトリになります。新規の chroot ディレクトリが、Identity Server のあらゆるインストールで使用されるパスの基点になります。

chroot 環境は、悪意のあるプログラムによる実際の root ファイルシステムへのアクセスを防ぎ、より安全な Identity Server 環境を実現する手段となります。

実際の配備の特別な要件によっては、chroot 環境の設定は複雑になります。chroot 環境に Identity Server をインストールして実行する必要がある場合には、Sun Microsystems の技術担当者に連絡してサポートを受けてください。

ロードバランサの設定

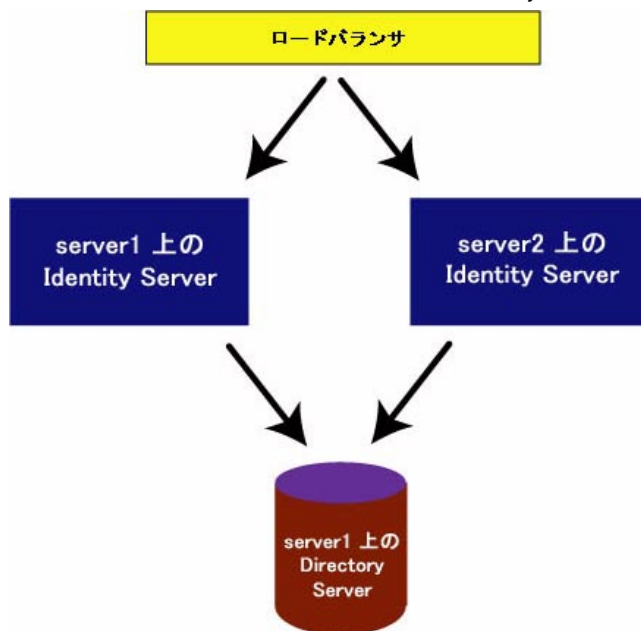
Sun Java™ System Identity Server は、ロードバランサを使用するよう設定できます。この章では、ロードバランスの機能およびロードバランスを実現する方法について説明します。次の節で構成されています。

- [155 ページの「ロードバランサの概要」](#)
- [158 ページの「ロードバランサの設定」](#)
- [168 ページの「設定を確認する」](#)

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 E-1](#) 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 E-1 ロードバランサを使用した Identity Server 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え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 Java System Web Server、Sun Java System 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 Java System 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、CDAaptor、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 E-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 E-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。

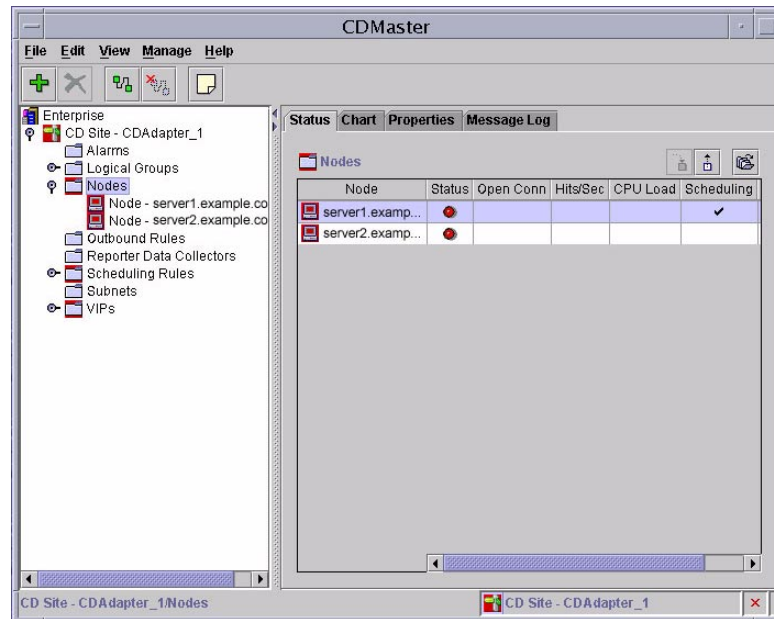
ロードバランサの設定

「[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 E-2](#) 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。

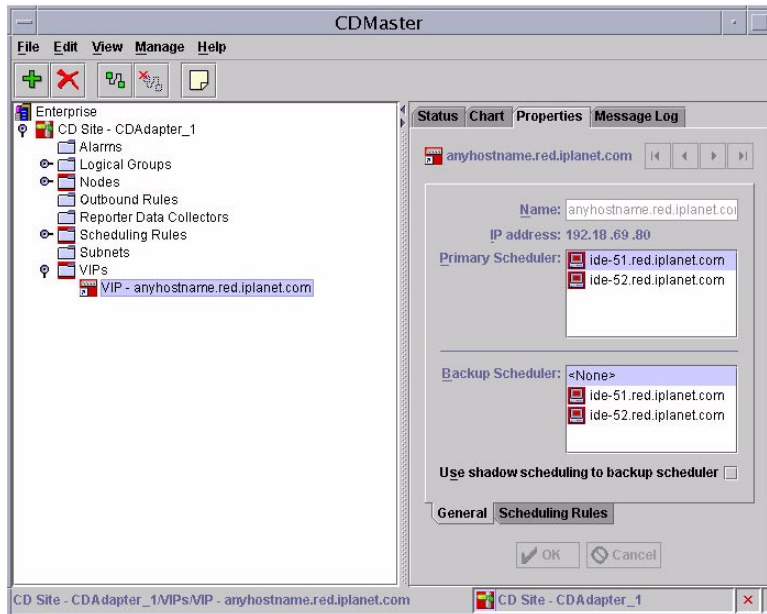
図 E-2 Resonate を使用したノードの作成



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新規仮想 IP アドレスを作成します。

図 E-3 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 E-3 仮想 IP アドレスを新規作成する

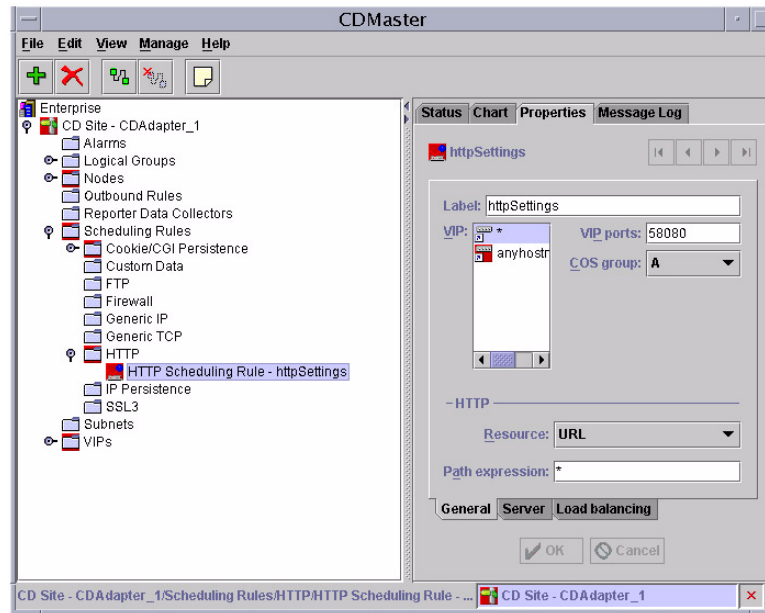


5. 「VIPs」の右側のフレーム内の「Scheduling Rules」をクリックして「HTTP」を選択し、次のように HTTP スケジューリングルールを設定します。
 - a. 「Properties」タブで、ロードバランサのインストール先ホストが仮想 IP として表示されていることを確認します。
 - b. VIP ポートが、仮想 IP として定義されているものと同じであることを確認します。
 - c. 「Resource」として「URL」を選択します。
6. CDMaster の左側のフレームの「Scheduling Rules」で、「HTTP」をクリックします。

161 ページの図 E-4 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。

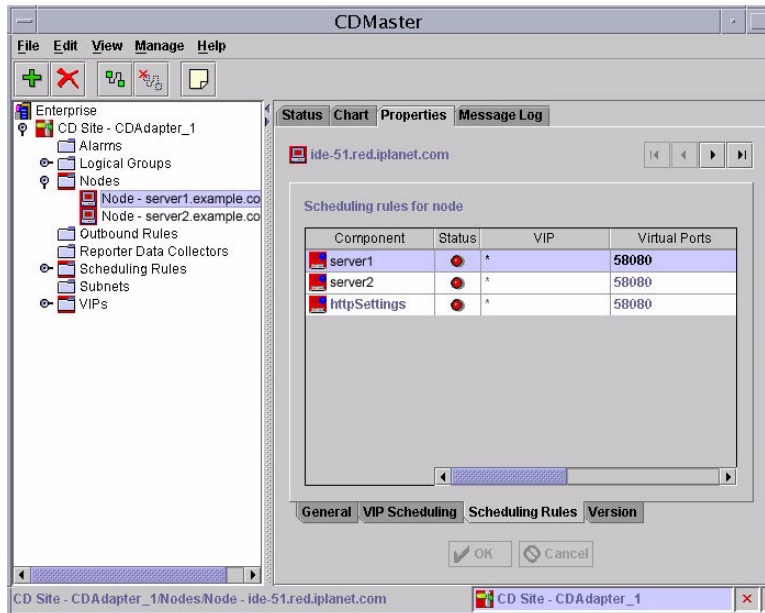
図 E-4 HTTP スケジューリングルールを設定する



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 E-5 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。

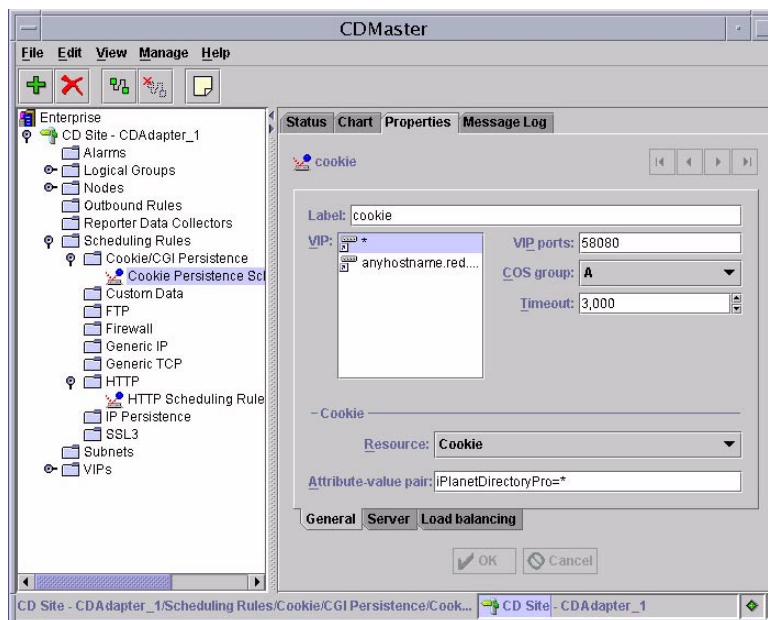
図 E-5 CDMaster でノードを設定する



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、161 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 E-6 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 E-6 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。引き続いて次の項「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」属性を見つけてます。
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。# を削除して、次のようにプロパティを設定します。

```
com.sun.identity.server.fqdnMap[loadbalancer_host.domain]=loadbalancer_host.domain
```

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、[168 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、[161 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。引き続いて次の項「[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。# を削除して、次のようにプロパティを設定します。

```
com.sun.identity.server.fqdnMap[loadbalancer_host.domain]=loadbalancer_host.domain
```

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.ipplanet.am.lbcookie.name=server1
com.ipplanet.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.ipplanet.am.lbcookie.name=server2
com.ipplanet.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 Java System Web Server の Web コンテナの <code>keepAliveTimeout</code> 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 Java™ System 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。

- [169 ページの「概要」](#)
- [170 ページの「RADIUS サーバーの設定」](#)
- [171 ページの「Identity Server の設定」](#)

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 F-1 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 F-2 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 が認証サービスに登録されていない場合、「追加」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. サービスフレームから「RADIUS」を選択して、「了解」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. [170 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。

用語集

このマニュアルセットで使用する用語の一覧については、最新の『Sun Java™ Enterprise System Glossary』を参照してください。

<http://docs.sun.com/doc/816-6873>

索引

A

Active Directory

認証する, 147

AM_ENC_PWD 変数, 89, 108

AMConfig.properties ファイル、セッションフェイルオーバー用, 109

amconfig スクリプト, 89

amsamplesilent ファイル, 89, 108

Application Server 7.0.0_01 Enterprise Edition (EE), 102

Application Server 7.0 Update 3, 89

C

CDSSO、SAML、および連携, 63

chroot 環境, 153

chroot 環境によるセキュリティ, 153

D

DEPLOY_LEVEL 変数, 89, 108

Directory Server

概要, 169

複数インスタンスの配備, 93

複数のインスタンス, 64

ds_remote_schema.ldif, 74

H

HTTP サーバーポート、Application Server, 105

HTTP セッションフェイルオーバー、Application Server, 102

I

Identity Server

chroot 環境, 153

SDK, 57

アイデンティティ管理の概要, 27

アクセス管理の概要, 24

拡張, 64

監査の概要, 28

関連製品情報, 20

技術上の考慮事項, 67

高可用性, 68

コンソールの概要, 28

スキーマの概要, 73

管理ロール, 79

制限, 81

マーカーオブジェクトクラス, 78

スケーラビリティ, 69

製品の概要, 24

セキュリティ, 68

ソフトウェア要件, 71

ハードウェア要件, 70

配備, 29

章の概要, 32

統合, 30

ロードマップ, 31

複数インスタンスの配備, 88

複数のインスタンス, 89

プログラミング用インタフェースの概要, 28

ポリシーエージェントの概要, 28

連携管理の概要, 26

Identity Server の統合, 30

J

Java Enterprise System インストーラ, 89, 107

Java アプリケーションの配備, 91

JVM の配備, 92

L

ldapmodify ツール, 101

LDAP 接続プール, 100

LDAP レプリケーション, 102

LDAP ロードバランサ, 65

N

NEW_INSTANCE 変数, 89, 108

nsslapd-idletimeout グローバル属性, 101

nsslapd-idletimeout 属性, 100

O

ODBC ドライバ, 105

R

RADIUS サーバー

設定, 169

認証する, 169

S

Solaris

サポート, 20

パッチ, 20

Solaris 8 オペレーティングシステム, 103

Solaris 9 オペレーティングシステム, 103

sunone_schema2.ldif, 77

SUNWam ディレクトリ, 116

sun-web.xml ファイル、セッションフェイルオーバー用, 106

W

WEB_CONTAINER 変数, 89, 108

Web Server 6.0 SP6, 103

Web Server 6.1 SP2, 89, 103

Web コンテナ, 64

あ

アーキテクチャ

CDSSO、SAML、および連携, 63

Identity Server SDK, 57

概要, 53

機能プロセス, 59

統合化されたクライアントディテクション, 62

統合化ポイント, 55

統合化ポリシー, 61

認証とユーザーセッション, 59

ポリシーエージェント, 55

アイデンティティ管理

アイデンティティプロファイル , 24
インフラストラクチャ , 22
定義 , 21
アイデンティティプロファイル , 24

い

インストーラ、Java Enterprise System インストーラ , 89, 107

か

概要

Directory Server , 169
アイデンティティ管理 , 27
アクセス管理 , 24
監査 , 28
コンソール , 28
スキーマ , 73
 管理ロール , 79
 制限 , 81
 マーカーオブジェクトクラス , 78
 プログラミング用インタフェース , 28
 ポリシーエージェント , 28
 連携管理 , 26
管理サーバーポート、Application Server , 105
管理ユーザー名、Application Server , 105
管理ロール , 79

き

技術上の考慮事項 , 67
 高可用性 , 68
 スケーラビリティ , 69
 セキュリティ , 68
機能プロセス , 59
 CDSSO、SAML、および連携 , 63
 統合化されたクライアントディテクション , 62

統合化ポリシー , 61
認証とユーザーセッション , 59

く

クライアント接続属性 , 101

こ

高可用性 , 68
高可用性データベース管理クライアント , 105
高可用性データベースサーバー , 105
コンソール、Directory Server , 101
コンソール、Identity Server , 109

さ

サイレントモード、amconfig スクリプト , 90
サポート
 Solaris , 20

す

スキーマの概要 , 73
 管理ロール , 79
 制限 , 81
 マーカーオブジェクトクラス , 78
スケーラビリティ , 69

せ

セキュリティ , 68
セッションのライフサイクル , 125
セッションフェイルオーバー

概要, 102

実装, 103

設定

RADIUS サーバー, 169

そ

ソフトウェア要件, 71

た

タイムアウト、Directory Server アイドル接続, 100

と

統合化されたクライアントディテクション, 62

統合化ポイント, 55

Identity Server SDK, 57

ポリシーエージェント, 55

統合化ポリシー, 61

導入ガイド

章の概要, 32

に

認証

Active Directory, 147

RADIUS, 169

RADIUS サーバー, 169

認証セッションの状態, 102

認証とユーザーセッション, 59

は

ハードウェア要件, 70

配備計画

アプリケーションの評価, 44

情報の収集, 41

スケジュールの作成, 50

データの分類, 47

目標の設定, 40

リソースの定義, 35

配備シナリオ

Directory Server, 93

Identity Server, 88

Java アプリケーション, 91

複数の JVM 環境, 92

配備ロードマップ, 31

ふ

ファイアウォール、Identity Server と Directory Server で使用, 100

複数インスタンス、Identity Server, 89

へ

ベースディレクトリ, 116

変数、Identity Server 設定, 89

ま

マーカオブジェクトクラス, 78

マニュアル

概要, 16

表記上の規則, 18

用語, 19

ら

ライフサイクル

ユーザーセッション, 125

れ

レプリケーション , 93

設定 , 94

ロードバランサの使用 , 97

連携

実装 , 112

ろ

ロードバランサ , 65

ロードバランサプラグイン、セッションフェイル
オーバー , 104

ロードバランサレプリケーション , 97

ロードバランス、セッションフェイルオーバー ,
102

