# Sun Java System Access Manager 7.1 管理ガイド

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

Part No: 820–0842

070828@18135

# 目次

# はじめに

『Sun Java System Access Manager 7.1 管理ガイド』では、Sun Java™ System Access Manager コンソールの使用法、およびコマンド行インタフェースでユーザーとサービスのデータを管理する方法について説明します。

Access Manager は、Sun Java Enterprise System (Java ES) のコンポーネントです。Java ES は、ネットワーク環境またはインターネット環境に分散するエンタープライズアプリケーションのサポートに必要なサービスを提供する一連のソフトウェアコンポーネントです。

## 対象読者

本書は、Sun Java System のサーバーとソフトウェアを使用して Web アクセスプラットフォームを実装する IT 管理者およびソフトウェア開発者を対象としています。

## このマニュアルをお読みになる前に

本ガイドを読まれる方は、次の技術に精通していることが必要です。

- 『Sun Java System Access Manager 7.1 Technical Overview』で説明されている Access Manager の技術的概念
- 配備先プラットフォーム: Solaris™ オペレーティングシステムまたは Linux オペレーティングシステム
- Access Manager を実行する Web コンテナ: Sun Java System Application Server、Sun Java System Web Server、BEA WebLogic、または IBM WebSphere Application Server
- 技術的概念: LDAP (Lightweight Directory Access Protocol)、Java テクノロジ、JavaServer Pages™ (JSP) テクノロジ、ハイパーテキスト転送プロトコル (HTTP)、ハイパーテキストマークアップ言語 (HTML)、および XML (eXtensible Markup Language)

# 関連マニュアル

以下の関連マニュアルが用意されています。


- 12 ページの「Access Manager の主要マニュアル」
- 13 ページの「Sun Java Enterprise System 製品のマニュアル」


## Access Manager の主要マニュアル

Access Manager の主要マニュアルセットには、以下のタイトルが含まれます。

- 『Sun Java System Access Manager 7.1 リリースノート』は、製品のリリース後にオンラインで参照できます。このリリースの最新情報の説明、既知の問題と制限事項、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、最新の情報を提供します。
- 『Sun Java System Access Manager 7.1 Technical Overview』は、Access Manager コンポーネントがどのように連携してアクセス制御機能を統合し、企業資産や Web ベースアプリケーションを保護するかについて、その概要を示します。Access Manager の基本的な概念と用語についても説明します。
- 『Sun Java System Access Manager 7.1 配備計画ガイド』は、ソリューションのライフサイクルに基づく Sun Java System Access Manager の計画と配備のソリューションを示します。
- 『Sun Java System Access Manager 7.1 Postinstallation Guide』は、Access Manager のインストール後の 設定に関する情報を提供します。
- 『Sun Java System Access Manager 7.1 Performance Tuning Guide』は、Access Manager とその関連コンポーネントを調整して最適なパフォーマンスを得る方法について説明します。
- 『Sun Java System Access Manager 7.1 管理ガイド』は、Access Manager コンソールの使用法、およびコマンド行インタフェースでユーザーとサービスのデータを管理する方法について説明します。
- 『Sun Java System Access Manager 7.1 Federation and SAML Administration Guide』は、Liberty Alliance Project 仕様に基づく Federation モジュールに関する情報を提供します。Liberty Alliance Project 仕様に基づく統合サービスに関する情報、Liberty ベースの環境を有効にするための手順、フレームワークを拡張するためのアプリケーションプログラミングインタフェース (API) の概要などが含まれます。
- 『Sun Java System Access Manager 7.1 Developer's Guide』は、Access Manager をカスタマイズし、組織の現在の技術インフラストラクチャーにその機能を統合する方法について説明します。製品とその API のプログラミングに関する情報も含まれます。
- 『Sun Java System Access Manager 7.1 C API Reference』は、Access Manager の公開された C API を構成するデータ型、構造体、および関数の概要を示します。

- 『Java API Reference』は、Access Manager の Java パッケージの実装に関する情報を提供します。
- 『Sun Java System Access Manager Policy Agent 2.2 User's Guide』は、Access Manager で利用可能なポリシー機能とポリシーエージェントの概要を示します。

『リリースノート』の更新と主要マニュアルの訂正へのリンクは、Sun Java Enterprise System ドキュメントの Web サイトの Access Manager のページにあります。更新されたマニュアルには、改訂日が記載されています。

### Sun Java Enterprise System 製品のマニュアル

次の製品のマニュアルを参照すると、役に立つ情報が見つかることがあります。

- Directory Server
- Web Server
- Application Server
- Web Proxy Server

## 関連する Sun 以外の Web サイトの引用

このマニュアル内で参照している第三者の URL は、追加の関連情報を提供します。

---

注 – このマニュアルで説明する Sun 以外の Web サイトの利用については、Sun は責任を負いません。こうしたサイトやリソース上の、またはこれらを通じて利用可能な、コンテンツ、広告、製品、その他の素材について、Sun は推奨しているわけではなく、Sun はいかなる責任も負いません。こうしたサイトやリソース上で、またはこれらを経由して利用できるコンテンツ、製品、サービスを利用または信頼したことに伴って発生した実際の、あるいは発生したと主張されるいかなる損害や損失についても、Sun は一切の責任を負いません。

---

## マニュアル、サポート、およびトレーニング

| Sun のサービス | URL | 内容 |
|---|---|---|
| マニュアル | http://jp.sun.com/documentation/ | PDF 文書および HTML 文書をダウンロードできます。 |

| Sun のサービス | URL | 内容 |
|---|---|---|
| サポートおよびトレーニング | http://jp.sun.com/supportraining/ | 技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。 |

# 表記上の規則

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

表 P–1 表記上の規則

| 字体または記号 | 意味 | 例 |
|---|---|---|
| AaBbCc123 | コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。 | .login ファイルを編集します。<br>ls -a を使用してすべてのファイルを表示します。<br>machine_name% you have mail. |
| **AaBbCc123** | ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。 | machine_name% **su**<br>Password: |
| *aabbcc123* | 変数を示します。実際に使用する特定の名前または値で置き換えます。 | ファイルを削除するには、rm *filename* と入力します。 |
| 『』 | 参照する書名を示します。 | 『コードマネージャー・ユーザーズガイド』を参照してください。 |
| 「」 | 参照する章、節、ボタンやメニュー名、強調する単語を示します。 | 第 5 章「衝突の回避」を参照してください。<br>この操作ができるのは、「スーパーユーザー」だけです。 |
| \ | 枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。 | sun% **grep '^#define **<br>**XV_VERSION_STRING'** |

コード例は次のように表示されます。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bourne シェルおよび Korn シェル

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

- Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

# このマニュアルに関するコメント

弊社では、マニュアルの改善に努めており、お客様からのコメントおよびご忠告をお受けしております。

http://docs.sun.com にアクセスして「コメントの送信」をクリックしてください。このオンラインフォームでは、マニュアルのタイトルと Part No. もご記入ください。Part No. は、マニュアルのタイトルページか先頭に記述されている 7 桁または 9 桁の番号です。

たとえば、このマニュアルのタイトルは『Sun Java System Access Manager 7.1 管理ガイド』で、Part No. は 820-0842 です。

パート I

# アクセス制御

これは『Sun Java System Access Manager™ 7.1 管理ガイド』の第 1 部です。アクセス制御インタフェースを利用すると、レルムベースのリソースを保護および規制するための認証サービスと承認サービスを作成および管理できます。企業内のユーザーが情報を要求すると、Access Manager はユーザーのアイデンティティーを検証し、ユーザーが要求した特定のリソースにそのユーザーがアクセスすることを承認します。次の章で構成されています。

◆ ◆ ◆ 第 1 章

# 1 Access Manager コンソール

Access Manager コンソールは、さまざまなレベルのアクセス権を持つ管理者がさまざまな操作を行うための Web インタフェースです。たとえば、レルムや組織を作成したり、それらのレルムに対してユーザーの作成または削除を行ったり、レルムのリソースへのアクセスを保護および制限するために適用するポリシーを確立するなどの操作を行います。また、現在のユーザーセッションを表示または終了したり、セッションの連携設定の管理 (認証ドメインやプロバイダの作成、削除、変更) も行います。管理者権限を持たないユーザーの場合は、個人情報 (名前、電子メールアドレス、電話番号など) の管理、パスワードの変更、グループへの加入または加入の解除、ロールの表示などを行うことができます。Access Manager コンソールは、次の 2 つの基本ビューで構成されます。



## 管理ビュー

管理者ロールを持つユーザーが Access Manager から認証されると、デフォルトのビューが管理ビューになります。このビューでは、管理者は Access Manager に関連するほとんどの管理タスクを実行できます。Access Manager は、レルムモードと旧バージョンモードの 2 つのモードでインストールできます。それぞれのモードには、固有のコンソールが使用されます。レルムモードと旧バージョンモードの詳細は、『Sun Java System Access Manager 7.1 Technical Overview』を参照してください。

### レルムモードのコンソール

管理者はレルムモードの管理コンソールを使用して、レルムベースのアクセス制御、デフォルトのサービス設定、Web サービスおよび連携を管理できます。管理者ログイン画面にアクセスするには、ブラウザで次のアドレス構文を使用します。

*protocol*://*servername* /amserver/UI/Login

protocol には、配備方法に応じて http または https を指定します。


図1–1 レルムモードの管理ビュー

# 旧バージョンモードのコンソール

旧バージョンモードのコンソールは、Access Manager 6.3 アーキテクチャーに基づいて設計されています。この旧バージョンの Access Manager アーキテクチャーでは、Sun Java System Directory Server に備えられている LDAP ディレクトリ情報ツリー (directory information tree、DIT) が使用されます。旧バージョンモードでは、ユーザー情報とアクセス制御情報が LDAP 組織に格納されます。旧バージョンモードを選択した場合、LDAP 組織はアクセス制御レルムに相当します。レルム情報は、LDAP 組織に統合されます。旧バージョンモードでは、「ディレクトリ管理」タブを使用して、Access Manager ベースのアイデンティティー管理を行うことができます。

管理者ログイン画面にアクセスするには、ブラウザで次のアドレス構文を使用します。

*protocol*://*servername* /amserver/console

protocol には、配備方法に応じて http または https を指定します。


図 1–2 旧バージョンモードの管理ビュー

## 旧バージョンモード 6.3 コンソール

Access Manager 6.3 の一部の機能は、Access Manager 7.1 コンソールでは使用できません。このため、管理者は 7.1 の旧バージョン配備から 6.3 コンソールにログインできるようになっています。Access Manager を構築する Sun Java System Portal Server またはその他の Sun Java System 通信製品の 主アイデンティティーリポジトリとして、Sun Java System Directory Server を使用しなければならない場合には、通常はこのコンソールを使用します。委任管理やサービスクラスなどのほかの機能には、このコンソールだけからアクセスできます。

---

注 – 旧バージョンモード 6.3 コンソールと旧バージョンモード 7.1 コンソールを交互に使用しないでください。

---

6.3 コンソールにアクセスするには、ブラウザで次のアドレス構文を使用します。

*protocol*://*servername* /amconsole

protocol には、配備方法に応じて http または https を指定します。

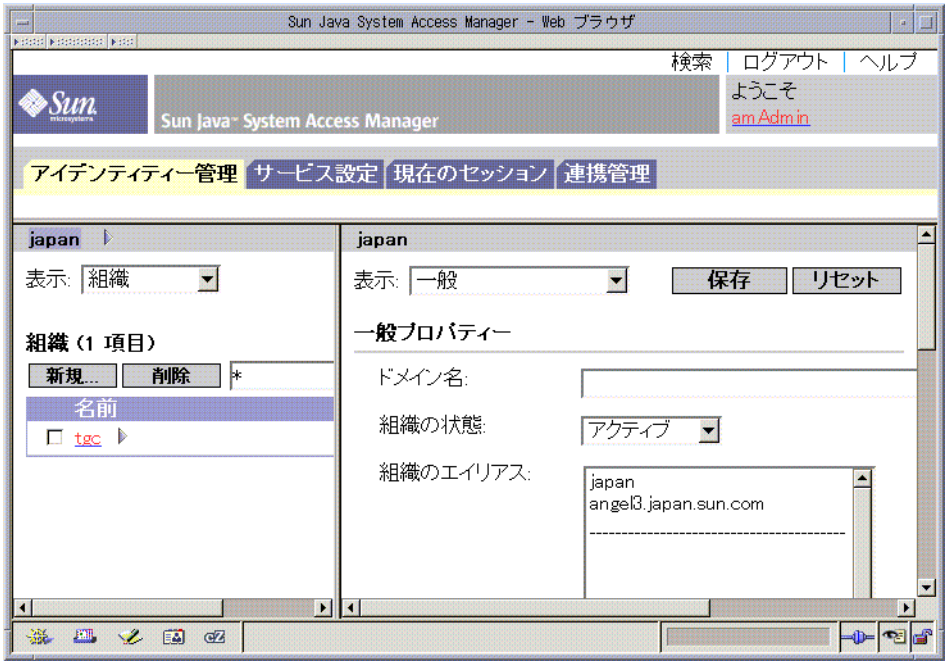


図 1–3　6.3 ベースのコンソールの管理ビュー

# ユーザープロファイルビュー

管理者ロールが割り当てられていないユーザーが Access Manager から認証されると、それぞれのユーザープロファイルビューがデフォルトビューになります。ユーザープロファイルビューには、レルムモードまたは旧バージョンモードでアクセスできます。このビューにアクセスするには、「ログイン」ページでそれぞれのユーザー名とパスワードを入力する必要があります。

ユーザープロファイルビューでは、それぞれのユーザープロファイル固有の属性の値を修正できます。たとえば、名前、ホームアドレス、パスワードなどの属性を修正できます。ユーザープロファイルビューに表示された属性は、展開することができます。

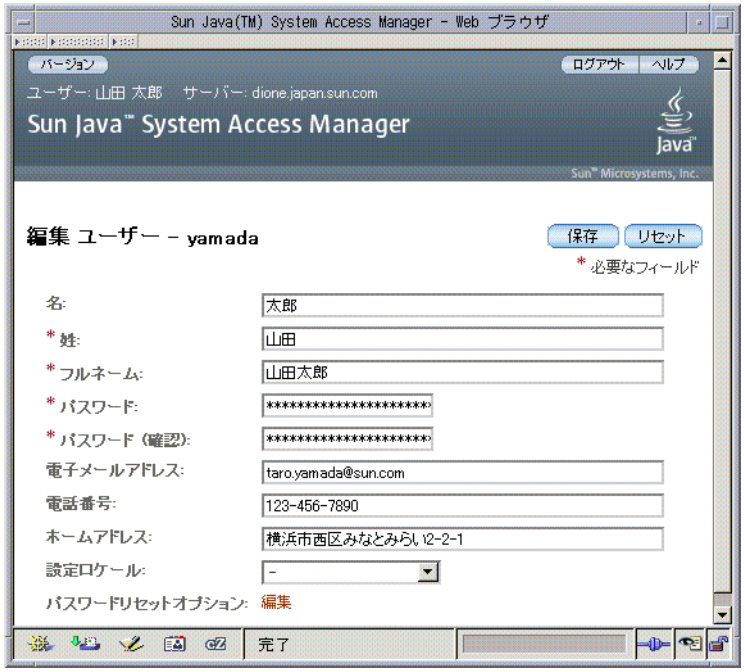


図 1–4　ユーザープロファイルビュー

◆ ◆ ◆ 第 2 章

# 2 レルムの管理

アクセス制御レルムは、ユーザーまたはユーザーのグループと関連付けることのできる認証プロパティーおよび承認ポリシーのグループです。レルムデータは、指定されたデータストアの内部に Access Manager が作成する独自形式の情報ツリーに格納されます。Access Manager フレームワークは、各レルムに含まれるポリシーおよびプロパティーを Access Manager 情報ツリーの内部に集約します。デフォルトでは、Access Manager は自動的に、ユーザーデータとは別の、Sun Java Enterprise System Directory Server 内の特別なブランチとして Access Manager 情報ツリーを挿入します。アクセス制御レルムは、任意の LDAPv3 データベースの使用中に使用できます。

レルムの詳細については、『Sun Java System Access Manager 7.1 Technical Overview』を参照してください。

「レルム」タブで、アクセス制御に関する次のプロパティーを設定できます。



## レルムの作成と管理

ここでは、レルムを作成および管理する方法について説明します。

### ▼ 新しいレルムを作成する

1 「アクセス制御」タブの下の「レルム」リストから「新規」を選択します。

2 次の一般属性を定義します。
名前　　レルムの名前を入力します。

親　　レルムを作成する位置を定義します。その下に新しいレルムが作成される親のレルムを選択します。

3 次のレルム属性を定義します。

| | |
|---|---|
| レルムの状態 | 「アクティブ」または「非アクティブ」の状態を選択します。デフォルトは「アクティブ」です。この属性は、レルムの存続期間中であればいつでも「プロパティー」アイコンを選択して変更できます。「非アクティブ」を選択すると、ログイン時のユーザーアクセスが無効になります。 |
| レルムまたはDNSのエイリアス | レルムのDNS名に対するエイリアス名を追加できます。この属性では、実際のドメインエイリアスだけを使用できます。ランダムな文字列は使用 できません。 |

4 「了解」をクリックして保存するか、「取消し」をクリックして前のページに戻ります。

# 一般プロパティー

「一般プロパティー」ページには、レルムの基本属性が表示されます。これらのプロパティーを変更するには、「アクセス制御」タブの下の「レルム名」リストからレルムをクリックします。その後、次のプロパティーを編集します。

| | |
|---|---|
| レルムの状態 | 「アクティブ」または「非アクティブ」の状態を選択します。デフォルトは「アクティブ」です。この属性は、レルムの存続期間中であればいつでも「プロパティー」アイコンを選択して変更できます。「非アクティブ」を選択すると、ログイン時のユーザーアクセスが無効になります。 |
| レルムまたはDNSのエイリアス | レルムのDNS名に対するエイリアス名を追加できます。この属性では、実際のドメインエイリアスだけを使用できます。ランダムな文字列は使用 できません。 |

プロパティーを編集したら、「保存」をクリックします。

注 – レルムモードで AMAdmin.dtd に「recursive=true」フラグを設定しても、サブレルム内のオブジェクトの検索では機能しません。このフラグは、サブ組織がすべて同じルートサフィックスに配置される旧バージョンモードでのみ機能します。レルムモードでは、各サブレルムは別のルートサフィックスを持つことができ、別のサーバーに配置される場合もあります。サブレルムでグループなどのオブジェクトを検索する場合は、XML データファイルで検索するサブレルムを指定する必要があります。

# 認証

ユーザーがほかの認証モジュールを使ってログインできるようにするには、事前に一般認証サービスをサービスとしてレルムに登録する必要があります。コア認証サービスでは、Access Manager 管理者がレルムの認証パラメータのデフォルト値を定義できます。その後、指定された認証モジュールでオーバーライド値が定義されない場合にこれらの値を使用できます。コア認証サービスに対するデフォルト値は amAuth.xml ファイルで定義され、インストール後に Directory Server に格納されます。

詳細については、認証の管理を参照してください。

# サービス

Access Manager におけるサービスとは、Access Manager コンソールでひとまとめに管理される属性のグループのことです。社員の氏名、役職、電子メールアドレスなど、関連性のある情報を属性として扱うことができます。ただし、属性は一般に、メールアプリケーションや給与支払サービスのようなソフトウェアモジュール用の設定パラメータとして使用されます。

「サービス」タブでは、多数の Access Manager デフォルトサービスをレルムに追加し、設定できます。次のサービスを追加できます。

- 管理
- ディスカバリサービス
- 国際化設定
- パスワードリセット
- セッション
- ユーザー

注 – Access Manager は、サービスの .xml ファイルの必須属性に一部のデフォルト値が含まれるようにします。値のない必須属性のあるサービスがある場合は、デフォルト値を追加してサービスを再読み込みします。

### ▼ サービスをレルムに追加する

1. 新しいサービスを追加するレルムの名前をクリックします。
2. 「サービス」タブを選択します。
3. 「サービス」リスト内の「追加」をクリックします。
4. レルムに追加するサービスを選択します。
5. 「次へ」をクリックします。
6. レルム属性を定義して、サービスを設定します。サービス属性の詳細については、オンラインヘルプの「設定」を参照してください。
7. 「終了」をクリックします。
8. サービスのプロパティーを編集するには、「サービス」リスト内の名前をクリックします。

## 権限

Access Manager の委任モデルは、管理者に割り当てられている権限または権利に基づきます。権限は、リソース上で実行できる操作またはアクションです。たとえば、ポリシーオブジェクトでの読み取り操作です。定義されている操作のセットは、読み取り、変更、および委任です。リソースは、アクションを実行できるオブジェクトであり、設定オブジェクトとアイデンティティーオブジェクトのいずれかにすることができます。

設定オブジェクトの例は、認証設定、ポリシー、データストアなどです。アイデンティティーオブジェクトの例は、ユーザー、グループ、ロール、およびエージェントです。権限のセットは動的に作成し、Access Manager に動的に追加できます。ただし、インストール中は Access Manager を正常に実行するために、小さいセットの権限が追加されます。権限がロードされたら、その権限をロールおよびグループに割り当てることができます。これらのロールおよびグループに属するユーザーは、委任

管理者となり、割り当てられた操作を実行できるようになります。基本的に、管理者は、1 つ以上の権限のセットが割り当てられているロールおよびグループのメンバーであるユーザーです。

Access Manager 7.1 では、次の管理者タイプのアクセス権を設定できます。

- レルム管理者 — レルム管理者は、すべてのオブジェクト (設定オブジェクトとアイデンティティーオブジェクト) の読み取り、変更、および委任の操作に対して、すべてのアクセス権を持っています。レルム管理者は、Unix システムにおける「root」と考えることができます。レルム管理者は、サブレルムを作成し、すべてのサービスの設定を変更できます。また、ユーザー、グループ、ロール、およびエージェントを作成、変更、および削除することもできます。
- ポリシー管理者 — ポリシー管理者は、ポリシーおよびポリシーサービスの設定のみを管理するアクセス権を持っています。ルール、対象、条件、および応答の属性で構成されるポリシーを作成、変更、および削除できます。ただし、ポリシーを管理するために、ポリシー管理者には、アイデンティティーリポジトリの対象および認証設定に対する読み取り権が必要です。ポリシー管理者は、アイデンティティーおよび認証設定を参照できます。
- ログ管理者 — ログ管理者は、ログレコードへの読み取り権または書き込み権、あるいはその両方を持っています。ログレコードを使用すると、悪意を持ったアプリケーションによって監査ログが悪用されるのを防ぐことができます。ログインタフェースは公開されているため、認証されたユーザーはすべてログレコードを読み取り、書き込むことができます。この権限を追加すると、このような悪用を防ぐことができます。ログインタフェースの主なユーザーは、J2EE および Web エージェントです。これらのユーザーは、変更権限しか必要ないため、読み取り権限を持つべきではありません。同様に、ログを参照する管理者は読み取り権限のみを持つべきであり、変更権限は持つべきではありません。このように使用するためには、次のようにログ権限をさらに分割します。
  - 書き込みアクセスを持つログ管理者 – これらの管理者は、すべてのログファイルを書き込むアクセス権を持っています。
  - 読み取りアクセスを持つログ管理者 – これらの管理者は、すべてのログファイルを読み取るアクセス権を持っています。
  - 読み取りアクセスおよび書き込みアクセスを持つログ管理者 – これらの管理者は、すべてのログファイルを読み取るアクセス権と書き込むアクセス権を持っています。

## Access Manager 7.1 の権限の定義

新しい Access Manager 7.1 のインストールインスタンスでは、ポリシー管理者、レルム管理者 (または旧バージョンモードでの組織管理者)、およびログ管理者に対して

アクセス権が付与されます。権限の割り当てまたは変更を行うには、編集するロールまたはグループの名前をクリックします。次の中から、いずれか 1 つを選択します。

すべてのログファイルに対する読み取りおよび書き込みのアクセス
　ログ管理者に対して読み取りアクセス権限と書き込みアクセス権限の両方を定義します。

すべてのログファイルに対する書き込みのアクセス
　ログ管理者に対して書き込みアクセス権限のみを定義します。

すべてのログファイルに対する読み取りのアクセス
　ログ管理者に対して読み取りアクセス権限のみを定義します。

ポリシープロパティーのみに対する読み取りおよび書き込みのアクセス
　ポリシー管理者に対して読み取りアクセス権限および書き込みアクセス権限を定義します。

すべてのレルムプロパティーおよびポリシープロパティーに対する読み取りおよび書き込みのアクセス
　レルム管理者に対して読み取りアクセス権限および書き込みアクセス権限を定義します。

## Access Manager 7.0 から 7.1 へのアップグレードに対する権限の定義

Access Manager を Version 7.0 から 7.1 にアップグレードした場合は、権限設定が新しい Access Manager 7.1 のインストールとは異なります。ただし、ポリシー管理者、レルム管理者、およびログ管理者に対する権限はサポートされています。権限の割り当てまたは変更を行うには、編集するロールまたはグループの名前をクリックします。次の中から、いずれか 1 つを選択します。

データストアに対する読み取り専用のアクセス
　ポリシー管理者についてデータストアに対する読み取りアクセス権限を定義します。

すべてのログファイルに対する読み取りおよび書き込みのアクセス
　ログ管理者に対して読み取りアクセス権限と書き込みアクセス権限の両方を定義します。

すべてのログファイルに対する書き込みのアクセス
　ログ管理者に対して書き込みアクセス権限のみを定義します。

すべてのログファイルに対する読み取りのアクセス
　ログ管理者に対して読み取りアクセス権限のみを定義します。

ポリシープロパティーのみに対する読み取りおよび書き込みのアクセス
　ポリシー管理者に対して読み取りアクセス権限および書き込みアクセス権限を定義します。

すべてのレルムプロパティーおよびポリシープロパティーに対する読み取りおよび書き込みのアクセス
　レルム管理者に対して読み取りアクセス権限および書き込みアクセス権限を定義します。

すべてのプロパティーおよびサービスに対する読み取り専用のアクセス
　ポリシー管理者についてすべてのプロパティーおよびサービスに対する読み取りアクセス権限を定義します。

Access Manager では、次の定義を別々または一緒に使用することはサポートされていません。

- データストアに対する読み取り専用のアクセス
- すべてのプロパティーおよびサービスに対する読み取り専用のアクセス

ポリシー管理者に対して委任制御を定義するには、これらの権限定義を「ポリシープロパティーのみに対する読み取りおよび書き込みのアクセス」の定義とともに使用する必要があります。

◆ ◆ ◆ 第 3 章

# 3 データストア

データストアは、ユーザー属性およびユーザー設定データを格納できるデータベースです。Access Manager は、LDAPv3 アイデンティティーリポジトリフレームワークに接続するアイデンティティーリポジトリプラグインを提供します。これらのプラグインにより、既存のユーザーデータベースに変更を加える必要なしに、Access Manager ユーザー情報を表示および取得できます。Access Manager フレームワークは、アイデンティティーリポジトリプラグインからのデータをほかの Access Manager プラグインからのデータと統合し、各ユーザーの仮想アイデンティティーを形成します。Access Manager はその後、複数のアイデンティティーリポジトリ間で、認証と承認のプロセスにユニバーサルアイデンティティーを使用できます。仮想ユーザーアイデンティティーは、ユーザーのセッション終了時に破棄されます。

## Access Manager データストアのタイプ

この節では、設定可能なデータストアのタイプについて説明します。また、新しいデータストアタイプを作成する手順および設定する方法についても説明します。

次のデータストアタイプのいずれかの場合に、新しいデータストアインスタンスを作成できます。

### Access Manager リポジトリプラグイン

このデータストアタイプは、Sun Java System Directory Server インスタンスに属し、Access Manager 情報ツリーを保持します。このデータストアタイプでは LDAP Version 3 の仕様の一部ではない Directory Server 機能 (ロールやサービスのクラスなど) が使用され、以前のバージョンの Access Manager との互換性があります。

### Active Directory

このデータストアタイプでは、LDAP Version 3 の仕様を使用して、アイデンティティーデータを Microsoft Active Directory のインスタンスに書き込みます。

### フラットファイルリポジトリ

このリポジトリを使用すると、別々のデータストアを作成することなく、Access Manager のローカルインストールインスタンス上のフラットな DIT 構造にデータとアイデンティティーを格納できます。一般に、このリポジトリはコンセプト配備のテストまたは検証に使用されます。

### 汎用 LDAPv3

このデータストアタイプを使用すると、アイデンティティーデータを LDAPv3 に準拠した任意のデータベースに書き込むことができます。使用中の LDAPv3 データベースが持続検索をサポートしていない場合は、キャッシュ機能を使用できません。

### Access Manager スキーマを含んだ Sun Directory Server

このデータストアタイプは、Sun Java System Directory Server インスタンスに属し、Access Manager 情報ツリーを保持します。これは、より多くの設定属性によってデータストアをより適切にカスタマイズできるという点で、Access Manager リポジトリプラグインと異なります。

## ▼ 新しいデータストアを作成する

次の節では、データストアを接続する手順について説明します。

1. 新しいデータストアを追加するレルムを選択します。
2. 「データストア」タブをクリックします。
3. 「データストア」リストから「新規」をクリックします。
4. データストアの名前を入力します。
5. 作成するデータストアのタイプを選択します。

6 「次へ」をクリックします。

7 適切な属性値を入力して、データストアを設定します。

8 「終了」をクリックします。

# データストア属性

この節では、新しい Access Manager データストアごとに設定する属性を定義します。データストア属性は、次のとおりです。


- 35 ページの「Access Manager リポジトリ属性」
- 37 ページの「フラットファイルリポジトリの属性」
- 39 ページの「LDAPv3 属性」


---

注 – Active Directory、汎用 LDAPv3、および Access Manager スキーマを含んだ Sun Directory Server の各データストアタイプは、同じ基本プラグインを共有しているため、設定属性も同じです。ただし、一部の属性のデフォルト値はデータストアタイプごとに異なり、Access Manager コンソールにそれぞれ表示されます。

---

## Access Manager リポジトリ属性

Access Manager リポジトリプラグインを設定するために、次の属性を使用できます。

### クラス名

Access Manager リポジトリプラグインを実装するクラスファイルの場所を指定します。

### Access Manager がサポートするタイプと操作

この LDAP サーバー上で許可されている、または実行可能な操作を指定します。この LDAPv3 リポジトリプラグインでサポートされている操作はデフォルト操作だけです。LDAPv3 リポジトリプラグインでサポートされている操作は次のとおりです。

- グループ — 読み取り、作成、編集、削除
- ユーザー — 読み取り、作成、編集、削除、サービス
- エージェント — 読み取り、作成、編集、削除

LDAP サーバー設定とタスクに基づいて、これらのリストからアクセス権を削除できますが、アクセス権の追加はできません。

設定された LDAPv3 リポジトリプラグインが Sun Java Systems Directory Server のインスタンスを指している場合は、「ロール」タイプのアクセス権を追加できます。これ

以外の場合は、ほかのデータストアではロールがサポートされていない場合があるため、このアクセス権は追加できません。「ロール」タイプのアクセス権は、次のとおりです。

- ロール — 読み取り、作成、編集、削除

サポートする LDAPv3 リポジトリのタイプとして「ユーザー」がある場合は、そのユーザーに対して読み取り、作成、編集、および削除のサービス操作が可能です。言い換えると、「ユーザー」タイプがサポートされている場合は、読み取り、編集、作成、および削除の操作を使用して、アイデンティティーリポジトリからユーザーエントリの読み取り、編集、作成、および削除を実行できます。`user=service` 操作を使用すると、Access Manager サービスがユーザーエントリ内の属性にアクセスできます。さらに、ユーザーが属するレルムまたはロールにサービスが割り当てられている場合、ユーザーは動的サービス属性にアクセスできます。

また、ユーザーは、割り当てられたすべてのサービスのユーザー属性を管理できます。ユーザーに操作として「サービス」(`user=service`) がある場合は、サービス関連の操作がすべてサポートされるように指定されます。それらの操作には、`assignService`、`unassignService`、`getAssignedServices`、`getServiceAttributes`、`removeServiceAttributes`、および `modifyService` があります。

## 組織 DN 値

Access Manager によって管理される、Directory Server 内の組織を指す DN を定義します。これは、データストア内で実行されるすべての操作のベース DN となります。

## ピープルコンテナネーミング属性

ユーザーがピープルコンテナ内に位置する場合に、ピープルコンテナのネーミング属性を指定します。ユーザーがピープルコンテナ内に位置しない場合、このフィールドは空のままとされます。

## ピープルコンテナ値

ピープルコンテナの値を指定します。デフォルトは `people` です。

## エージェントコンテナネーミング属性

エージェントがエージェントコンテナ内に位置する場合の、エージェントコンテナのネーミング属性。エージェントがエージェントコンテナ内に位置しない場合、このフィールドは空のままとされます。

## エージェントコンテナ値

エージェントコンテナの値を指定します。デフォルトは `agents` です。

### 再帰検索

これを有効にした場合、Access Manager リポジトリでは、指定されたアイデンティティーに対して再帰検索が実行されます。たとえば、次のデータ構造で再帰検索を実行します。

```
root
realm1
  subrealm11
    user5
  subrealm12
    user6
realm2
  user1
  user2
  subrealm21
    user3
    user4
```

結果は、次のようになります。

- 検索が root から実行され、このレベルでユーザーが定義されていない場合 (`amadmin` と `anonymous` は除く)、検索の結果で user1 から user6 が返されます。
- 検索が `realm1` から実行され、ユーザーが定義されていない場合、検索の結果で `user5` と `user6` が返されます。
- 検索が `realm2` (2 つのユーザーが定義されている) から実行される場合、検索の結果で user1 から user4 が返されます。

### レルム設定をコピー

この属性をレルムモードのインストールで有効にした場合、Access Manager は、リポジトリに存在するレルムおよびサブレルムごとに対応する組織およびサブ組織を作成します。さらに、新しく作成された組織およびサブ組織に、レルムおよびサブレルムに登録されたサービスが登録されます。レルム DIT と組織 DIT の両方が、データストアに存在します。

## フラットファイルリポジトリの属性

フラットファイルリポジトリを設定するために、次の属性を使用できます。

### ファイルリポジトリプラグインのクラス名

この属性は、フラットファイルの実装を提供する Java クラスファイルを指定します。この属性は変更できません。

## ファイルリポジトリディレクトリ

アイデンティティーおよびその属性を格納するベースディレクトリを定義します。

## キャッシュ

これを有効にした場合 (デフォルト)、アイデンティティーとその属性がキャッシュされます。後続の要求は、ファイルシステムにアクセスしません。

## キャッシュを更新する時間

キャッシュを有効にした場合、この属性は、ファイルシステムへの変更がないかどうか、キャッシュ内のエントリをチェックするまでの間隔を分単位で指定します。このチェックのメカニズムは、タイムスタンプに基づきます。

## ファイルに含めるユーザーオブジェクトクラス

作成されたときに、自動的にユーザーに追加されるオブジェクトクラスを定義します。

## パスワード属性

認証に使用されるパスワードを含む属性名を指定します。この属性は、データストアの認証モジュールを有効にしたときに、ユーザーを認証するために使用されます。

## 状態属性

アイデンティティーの状態を格納する属性名を指定します。状態属性の値は、「アクティブ」と「非アクティブ」のいずれかです。これは、アイデンティティーの認証時に使用されます。アイデンティティーが「非アクティブ」の場合は、ユーザーは認証されません。

## ハッシュ属性

ハッシュ化され、ファイルに格納される値を含む属性のリストを指定します。ハッシュ化されると、元の値は取得できません。ハッシュ化された値のみが取得されます。これは、特定の属性を永続的に格納すべきではないにもかかわらず検証に使用する際、機密性を確保するために使用します。このタイプの属性の例には、アイデンティティーのパスワード属性があります。

## 暗号化された属性

暗号化され、ファイルに格納される値を含む属性のリストを指定します。値は暗号化され、格納されますが、アイデンティティーリポジトリ API を呼び出すと、元の復号化された値が返されます。これにより、ユーザーがファイルシステムに直接アクセスし、機密性の高い属性を読み取ることを防ぎます。

# LDAPv3 属性

LDAPv3 リポジトリプラグインを設定するために、次の属性を使用できます。

## LDAP サーバー

接続先 LDAP サーバーの名前を入力します。「ホスト名.ドメイン名:ポート番号」の形式を使用することをお勧めします。

複数の「ホスト:ポート番号」エントリが入力された場合、リスト内の最初のホストへの接続が試みられます。リスト内の次のエントリは、現在のホストへの接続試行が失敗した場合にのみ試行されます。

## LDAP バインド DN

現在接続している LDAP サーバーに対して認証を行うために Access Manager が使用する DN 名を指定します。バインドに使用される DN 名を持つユーザーには、40 ページの「LDAPv3 プラグインでサポートされるタイプおよび操作」属性で設定した、正しい追加/変更/削除権限を付与することをお勧めします。

## LDAP バインドパスワード

現在接続している LDAP サーバーに対して認証を行うために Access Manager が使用する DN パスワードを指定します。

## LDAP バインドパスワード (確認)

パスワードを確認します。

## LDAP 組織 DN

このデータストアリポジトリのマッピング先となる DN。これは、このデータストア内で実行されるすべての操作のベース DN となります。

## LDAP SSL

有効にすると、Access Manager は HTTPS プロトコルを使用してプライマリサーバーに接続します。

## LDAP 接続プールの最小サイズ

接続プール内の接続の初期数を指定します。接続プールを利用すると、新しい接続を毎回作成する必要がなくなります。

## LDAP 接続プールの最大サイズ

許容される接続数の上限を指定します。

## 検索で返される結果の最大数

検索操作で返されるエントリ数の上限を指定します。この制限に達すると、Directory Server は検索要求に一致するあらゆるエントリを返します。

## 検索タイムアウト

検索要求に割り当てられる最大の秒数を指定します。この制限に達すると、Directory Server は検索要求に一致するあらゆる検索エントリを返します。

## 参照先も **LDAP** 検索する

このオプションを有効にすると、ある LDAP サーバーからの別の LDAP サーバーに対する参照が自動的に実行されます。

## **LDAPv3** リポジトリプラグインクラス名

LDAPv3 リポジトリを実装するクラスファイルの場所を指定します。

## 一般属性名マッピング

フレームワークが認識する共通属性をネイティブデータストアにマップできるようにします。たとえば、フレームワークがユーザー状態の判定に `inetUserStatus` を使用する場合に、ネイティブデータストアが実際には `userStatus` を使用することが可能です。属性定義では大文字と小文字が区別されます。

## **LDAPv3** プラグインでサポートされるタイプおよび操作

この LDAP サーバー上で許可されている、または実行可能な操作を指定します。この LDAPv3 リポジトリプラグインでサポートされている操作はデフォルト操作だけです。LDAPv3 リポジトリプラグインでサポートされている操作は次のとおりです。

- グループ — 読み取り、作成、編集、削除
- ユーザー — 読み取り、作成、編集、削除、サービス
- エージェント — 読み取り、作成、編集、削除

LDAP サーバー設定とタスクに基づいて、これらのリストからアクセス権を削除できますが、アクセス権の追加はできません。

設定された LDAPv3 リポジトリプラグインが Sun Java Systems Directory Server のインスタンスを指している場合は、「ロール」タイプのアクセス権を追加できます。これ以外の場合は、ほかのデータストアではロールがサポートされていない場合があるため、このアクセス権は追加できません。「ロール」タイプのアクセス権は、次のとおりです。

- ロール — 読み取り、作成、編集、削除

サポートする LDAPv3 リポジトリのタイプとして「ユーザー」がある場合は、そのユーザーに対して読み取り、作成、編集、および削除のサービス操作が可能です。言い換えると、「ユーザー」タイプがサポートされている場合は、読み取り、編集、作成、および削除の操作を使用して、アイデンティティーリポジトリからユーザーエントリの読み取り、編集、作成、および削除を実行できます。`user=service` 操作を使用すると、Access Manager サービスがユーザーエントリ内の属性にアクセスできます。さらに、ユーザーが属するレルムまたはロールにサービスが割り当てられている場合、ユーザーは動的サービス属性にアクセスできます。

また、ユーザーは、割り当てられたすべてのサービスのユーザー属性を管理できます。ユーザーに操作として「サービス」 (`user=service`) がある場合は、サービス関連の操作がすべてサポートされるように指定されます。それらの操作には、`assignService`、`unassignService`、`getAssignedServices`、`getServiceAttributes`、`removeServiceAttributes`、および `modifyService` があります。

## LDAPv3 プラグイン検索範囲

LDAPv3 プラグインエントリの検索に使用する範囲を定義します。範囲は次のいずれかにする必要があります。

- SCOPE_BASE
- SCOPE_ONE
- SCOPE_SUB (デフォルト)

## LDAP ユーザー検索属性

このフィールドは、ユーザーに対して検索を行うための属性タイプを定義します。たとえば、ユーザーの DN が「`uid=user1,ou=people,dc=iplanet,dc=com`」である場合、ネーミング属性は `uid` です。

## LDAP ユーザー検索フィルタ

ユーザーエントリの検索に使用する検索フィルタを指定します。

## LDAP ユーザーオブジェクトクラス

ユーザーのオブジェクトクラスを指定します。ユーザーが作成されると、このユーザーオブジェクトクラスのリストがユーザーの属性リストに追加されます。

## LDAP ユーザー属性

ユーザーと関連付けられる属性のリストを定義します。このリストにないユーザー属性の読み取りまたは書き込みは一切行うことができません。属性は大文字と小文字が区別されます。ここでオブジェクトクラスと属性スキーマを定義する前に、Directory Server でオブジェクトクラスと属性スキーマが定義されている必要があります。

## LDAP ユーザー作成属性マッピング

ユーザーを作成するときに必要な属性を指定します。この属性では、次の構文が使用されます。

```
DestinationAttributeName=SourceAttributeName
```

ソース属性名がない場合、デフォルトはユーザー ID (uid) です。次に例を示します。

cn
sn=givenName

ユーザープロファイルを作成するには、cn と sn の両方が必要です。cn は uid という属性の値を取得し、sn は givenName という属性の値を取得します。

## ユーザー状態属性

ユーザーの状態を示す属性名を指定します。

## ユーザー状態のアクティブ値

アクティブなユーザーの状態を示す属性名を指定します。デフォルトは「アクティブ」です。

## ユーザー状態の非アクティブ値

非アクティブなユーザーの状態を示す属性名を指定します。デフォルトは「非アクティブ」です。

## LDAP グループ検索属性

このフィールドは、グループに対して検索を行うための属性タイプを定義します。デフォルトは cn です。

## LDAP グループ検索フィルタ

グループエントリの検索に使用する検索フィルタを指定します。デフォルトは、(objectclass=groupOfUniqueNames) です。

## LDAP グループコンテナネーミング属性

グループがコンテナ内に位置する場合に、グループコンテナのネーミング属性を指定します。コンテナ内に位置しない場合、この属性は空のままとされます。たとえば、「cn=group1,ou=groups,dc=iplanet,dc=com」のグループ DN が ou=groups 内に位置する場合、グループコンテナネーミング属性は ou です。

### LDAP グループコンテナ値

グループコンテナの値を指定します。たとえば、「cn=group1,ou=groups,dc=iplanet,dc=com」のグループ DN がコンテナ名 ou=groups 内に位置する場合、グループコンテナ値は groups になります。

### LDAP グループオブジェクトクラス

グループのオブジェクトクラスを指定します。グループが作成されると、グループオブジェクトクラスのこのリストがグループの属性リストに追加されます。

### LDAP グループ属性

グループと関連付けられる属性のリストを定義します。このリストにないグループ属性の読み取りまたは書き込みは一切行うことができません。属性は大文字と小文字が区別されます。ここでオブジェクトクラスと属性スキーマを定義する前に、Directory Server でオブジェクトクラスと属性スキーマが定義されている必要があります。

### グループメンバーシップ属性

DN が属する全グループの名前がその値である属性の名前を指定します。デフォルトは memberOf です。

### 一意のメンバー属性

このグループに属している DN がその値である属性名を指定します。デフォルトは uniqueMember です。

### グループメンバー URL 属性

このグループに属しているメンバーを決定する LDAP URL がその値である、属性の名前を指定します。デフォルトは memberUrl です。

### LDAP ピープルコンテナネーミング属性

ユーザーがピープルコンテナ内に位置する場合に、ピープルコンテナのネーミング属性を指定します。ユーザーがピープルコンテナ内に位置しない場合、このフィールドは空のままとされます。

### LDAP ピープルコンテナ値

ピープルコンテナの値を指定します。デフォルトは people です。

### LDAP エージェント検索属性

このフィールドは、エージェントの検索を実行するための属性タイプを定義します。デフォルトは uid です。

## LDAP エージェントコンテナネーミング属性

エージェントがエージェントコンテナ内に位置する場合の、エージェントコンテナのネーミング属性。エージェントがエージェントコンテナ内に位置しない場合、このフィールドは空のままとされます。

## LDAP エージェントコンテナ値

エージェントコンテナの値を指定します。デフォルトは `agents` です。

## LDAP エージェント検索フィルタ

エージェントの検索に使用するフィルタを定義します。実際のエージェント検索フィルタは、このフィールドの値の前に「LDAP エージェント検索属性」の値を付加することによって構築されます。

たとえば、「LDAP エージェント検索属性」が `uid` で、「LDAP ユーザー検索フィルタ」が `(objectClass=sunIdentityServerDevice)` の場合、実際のユーザー検索フィルタは次のようになります。`(&(uid=*)(objectClass=sunIdentityServ erDevice))`

## LDAP エージェントオブジェクトクラス

エージェントのオブジェクトクラスを定義します。エージェントが作成されると、エージェントオブジェクトクラスのリストがエージェントの属性リストに追加されます。

## LDAP エージェント属性

エージェントと関連付けられる属性のリストを定義します。このリストにないエージェント属性の読み取りまたは書き込みは一切行うことができません。属性は大文字と小文字が区別されます。ここでオブジェクトクラスと属性スキーマを定義する前に、Directory Server でオブジェクトクラスと属性スキーマが定義されている必要があります。

## 認証可能なアイデンティティータイプ

レルムの認証モジュールモードがデータストアに設定されている場合は、このデータストアがユーザーまたはエージェント、あるいはその両方のアイデンティティータイプを認証できるように指定します。

## 持続検索ベース DN

持続検索に使用するベース DN を定義します。一部の LDAPv3 サーバーは、ルートサフィックスレベルでの持続検索のみをサポートします。

## 持続検索フィルタ

ディレクトリサーバーエントリに特定の変更を返すフィルタを定義します。データストアは、定義されたフィルタに一致する変更のみを受け取ります。

## 再起動前の持続検索の最大アイドル時間

持続検索を再開するまでの最大アイドル時間を定義します。1 よりも大きい値を設定する必要があります。1 以下の値を設定すると、接続のアイドル時間とは無関係に検索を再開します。

Access Manager がロードバランサとともに配備される場合、一部のロードバランサは、指定された時間アイドル状態が続くとタイムアウトします。この場合、ロードバランサに対して指定された時間よりも小さい値を「再起動前の持続検索の最大アイドル時間」に設定することをお勧めします。

## エラーコードのあとの再試行の最大数

「再試行する LDAPException エラーコード」で指定されたエラーコードが返された場合に、持続検索操作を再試行する回数の上限を定義します。

## 再試行の間の遅延時間

各再試行の前に待機する時間を指定します。これは、持続検索接続にのみ適用されます。

## 再試行する **LDAPException** エラーコード

持続検索操作を再試行させるエラーコードを指定します。この属性は持続検索のみに適用され、すべての LDAP 操作に適用されるわけではありません。

## キャッシュ

これを有効にした場合、Access Manager はデータストアから取得されたデータをキャッシュできます。

## キャッシュされた項目の最長有効期間

削除されるまでデータがキャッシュに格納される最長期間を指定します。この値は、秒単位で定義します。

## キャッシュの最大サイズ

キャッシュの最大サイズを指定します。この値を大きくすると、より多くのデータを格納できますが、必要なメモリーも増えます。この値は、バイト単位で定義します。

◆ ◆ ◆ 第 4 章

# 4 認証の管理

認証サービスは、Access Manager の配備先にインストールされたすべてのデフォルト認証タイプに対して Web ベースのユーザーインタフェースを提供します。このインタフェースは、アクセスを要求するユーザーに対して、呼び出される認証モジュールに基づいたログイン条件画面を表示して認証資格を収集する動的かつカスタマイズ可能な手段を提供します。このインタフェースは、開発者が実用的な Web アプリケーションを作成するのに役立つ Java 2 Enterprise Edition (J2EE) プレゼンテーションフレームワークである Sun Java System™ Application Framework (*JATO* と呼ばれることもある) を使用して作成されています。

## 認証の設定

ここでは、配備の認証を設定する方法について説明します。最初の節では、デフォルトの認証モジュールタイプの概要について説明し、必要な事前の設定手順を示します。レルム、ユーザー、ロールなどに対して、同じ認証モジュールタイプの複数の設定インスタンスを設定できます。また、認証に成功するためには複数のインスタンスの条件に合格する必要があるようにするため、認証連鎖を追加することもできます。この節では、次の点を説明します。



### 認証モジュールタイプ

認証モジュールは、ユーザー ID やパスワードなどのユーザー情報を収集し、その情報をデータベース内のエントリで確認するプラグインです。ユーザーが認証条件を満たす情報を入力した場合、そのユーザーは要求するリソースへのアクセスを許可されます。ユーザーが認証条件を満たしていない情報を入力した場合、そのユー

ザーは要求したリソースへのアクセスを拒否されます。Access Manager は、次のタイプの認証モジュールとともにインストールされます。



注 – 一部の認証モジュールタイプでは、認証インスタンスとして使用できるようにするには、事前の設定が必要です。必要に応じて、設定手順がモジュールタイプの説明にリスト表示されています。

## コアシステムサポート

Access Manager では、コア認証モジュールばかりではなく、デフォルトで 15 種類の認証モジュールを提供しています。コア認証モジュールでは、認証モジュールの全体的な設定を行います。Active Directory 認証、匿名認証、証明書に基づく認証、HTTP 基本認証、JDBC 認証、LDAP 認証、任意の認証のモジュールを追加して有効にする前に、コア認証モジュールの追加と有効化を行う必要があります。コア認証モジュールおよび LDAP 認証モジュールは両方とも、デフォルトのレルムに対して自動的に有効になります。

「拡張プロパティー」ボタンをクリックすると、レルムに対して定義できる「コア」認証属性が表示されます。グローバル属性はレルムに適用できないので表示されません。

### Active Directory

Active Directory 認証モジュールでは、LDAP モジュールと同様の認証が実行されますが、LDAP 認証モジュールの場合の Directory Server ではなく、Microsoft の Active Directory™ サーバーが使用されます。LDAP 認証モジュールを Active Directory サー

バー用に設定できますが、このモジュールでは、LDAP と Active Directory 認証の両方を同一レルム下に存在させることができます。

---

注 – このリリースの Active Directory 認証モジュールでは、ユーザー認証のみがサポートされます。パスワードポリシーは LDAP 認証モジュールのみでサポートされます。

---

## 匿名 (anonymous)

デフォルトでは、このモジュールを有効にすると、ユーザーは *anonymous* ユーザーとして Access Manager にログインできるようになります。「有効な匿名ユーザー」のリスト属性を設定して、このモジュールに匿名ユーザーのリストを定義することもできます。匿名アクセスを許可するということは、パスワードなしでアクセスさせるということです。匿名アクセスは、特定の種類のアクセス (読み取りのためのアクセスや検索のためのアクセスなど)、特定のサブツリー、またはディレクトリ内の個別のエントリに制限されます。

## 証明書

証明書に基づく認証では、個人用デジタル証明書 (PDC) を使用してユーザーを特定し、認証します。Directory Server に格納された PDC に一致すること、また証明書失効リスト (CRL) で確認されていることを求めるように、PDC を設定できます。

証明書に基づく認証モジュールをレルムに追加する前に、いくつかの作業を行う必要があります。まず、Access Manager とともにインストールした Web コンテナを保護し、証明書に基づく認証で使用できるように設定する必要があります。

---

注 – SSL 対応の Sun Java System WebsServer 6.1 インスタンスとともに Access Manager Certificate の認証を設定し、証明書ベースの認証要求と証明書ベースでない認証要求の両方を受け入れるように WebServer を定義する場合は、WebServer の `obj.conf` ファイルに次の値を設定する必要があります。

```
PathCheck fn="get-client-cert" dorequest="1" require="0"
```

これは、この動作に対してオプションの属性を設定するときには、WebServer コンソールに制限があるためです。

---

証明書に基づくモジュールを有効にする前に、Web Server に対するこれらの初期設定手順について、『*Sun ONE Web Server 6.1* 管理者ガイド』の第 6 章「証明書と鍵の使用」を参照してください。このマニュアルは、次の場所にあります。

```
http://docs.sun.com/db/prod/s1websrv#hic
```

または、次の場所にある『*Sun ONE Application Sever Administrator's Guide to Security*』を参照してください。

http://docs.sun.com/db/prod/s1appsrv#hic (http://docs.sun.com/db/prod/s1appsrv#hic)

注 – 証明書に基づくモジュールを使用して認証されるユーザーは、ブラウザ用に PDC を要求する必要があります。使用しているブラウザによって、手順が異なります。詳細は、使用しているブラウザのマニュアルを参照してください。

このモジュールを追加するためには、Access Manager にレルム管理者としてログインし、Access Manager と Web コンテナに対して SSL を設定し、クライアント認証を有効化しておく必要があります。詳細については、『*Access Manager Post Installation Guide*』の「Configuring Access Manager in SSL Mode」を参照してください。

## データストア

データストアの認証モジュールを使用すると、レルムのアイデンティティーリポジトリを使用したログインでユーザーを認証できます。同じデータストアリポジトリに対して認証を行う必要がある場合は、データストアのモジュールを使用すると、認証プラグインモジュールを書き、ロードし、認証モジュールを設定する必要がなくなります。さらに、そのレルムの該当リポジトリに対してフラットファイルの認証が必要な場合でも、カスタム認証モジュールを書く必要がありません。

この認証タイプには、Access Manager 認証を設定するときに、ある程度の利便性があります。Access Manager 7.1 より前のリリースでは、LDAPv3 データストアのユーザーが自分のレルムへの認証をできるようにする場合に、次を実行する必要があります。

- LDAPv3 データストアの設定
- 同じレルム対象を参照する LDAP 認証モジュールインスタンスの設定

データストアの認証モジュールを使用すると、レルムのアイデンティティーリポジトリに定義されたユーザーが認証を行うことができます。LDAP 認証の設定は必要ありません。たとえば、レルムのアイデンティティーリポジトリには LDAPv3 データストアが含まれ、同じレルムでデータストア認証が使用されると仮定します。この場合、アイデンティティーリポジトリに定義されたユーザーはすべて、そのレルムへの認証を行うことができます。

## HTTP 基本

このモジュールは、HTTP プロトコルのビルトイン認証サポートである基本認証を使用します。Web サーバーはユーザー名とパスワードを求めるクライアント要求を発行し、その情報を認証済み要求の一部としてサーバーに返します。Access Manager ではユーザー名とパスワードを取得し、LDAP 認証モジュールに対してユーザーを内部的に認証します。HTTP 基本認証が正常に機能するために、LDAP 認証モジュールを追加する必要があります (HTTP 基本モジュールを単独で追加しても機能しない)。いったん認証に成功したユーザーには、以降の認証でユーザー名とパスワードの入力は要求されません。

### JDBC

JDBC (Java Database Connectivity) 認証モジュールでは、JDBC 技術に対応したドライバを提供する SQL データベースを通して Access Manager でユーザーを認証できるようにするメカニズムが提供されます。SQL データベースへの接続は、JDBC ドライバを通して直接行うか、JNDI 接続プールで行います。

---

注 – このモジュールは、MySQL4.0 と Oracle 8i でテストされています。

---

### LDAP *

LDAP 認証モジュールを使用すると、ユーザーがログインするときに、特定のユーザー DN およびパスワードを使用して、LDAP Directory Server にバインドする必要があります。レルムに基づく認証すべてに対して、この認証モジュールがデフォルトとなります。ユーザーが Directory Server に存在するユーザー ID およびパスワードを指定すると、ユーザーは有効な Access Manager セッションへのアクセスが許可され、セットアップされます。コア認証モジュールおよび LDAP 認証モジュールは両方とも、デフォルトのレルムに対して自動的に有効になります。

### メンバーシップ

メンバーシップ認証は、`my.site.com` または `mysun.sun.com` のように、パーソナライズされたサイトのように実装されます。モジュールが有効なときに、ユーザーは管理者の支援なしでアカウントを作成し、パーソナライズします。ユーザーは作成したアカウントを使用し、追加済みユーザーとしてアクセスできます。また、ユーザーはビューアのインタフェースにアクセスできます。ビューアのインタフェースは、認証データおよびユーザー設定として、ユーザープロファイルデータベースに保存されています。

### MSISDN

MSISDN (Mobile Station Integrated Services Digital Network) 認証モジュールでは、携帯電話などのデバイスに関連するモバイル加入者 ISDN を使用して認証できます。これは対話型モジュールではありません。このモジュールは取得した加入者 ISDN について、その番号に一致するユーザーが存在するかどうか Directory Server に対して確認を行います。

### RADIUS

Access Manager は、すでにインストールされている RADIUS サーバーと連携するように設定できます。エンタープライズで認証のために旧バージョンの RADIUS サーバーを使用している場合に便利です。RADIUS 認証モジュールを有効にするには、次の 2 つの手順を行います。

1. RADIUS サーバーを設定します。

詳しい手順については、RADIUS サーバーのマニュアルを参照してください。

2. RADIUS 認証モジュールを登録し、有効にします。

## Sun Java System Application Server で RADIUS を設定する

RADIUS クライアントがそのサーバーに対してソケット接続を作成するとき、デフォルトでは、Application Server の `server.policy` ファイルで SocketPermission の connect アクセス権だけが与えられています。RADIUS 認証を正常に機能させるには、次のアクションを許可する必要があります。

- accept
- connect
- listen
- resolve

ソケット接続のアクセス権を与えるには、Application Server の `server.policy` ファイルにエントリを追加します。SocketPermission は、ホストの指定と、そのホストへの接続方法を指定する一連のアクションとで構成されます。ホストは次のように指定されます。

```
host = hostname | IPaddress:portrange:portrange = portnumber
| -portnumberportnumber-portnumber
```

ホストは、DNS 名または IP アドレスの数値で表されるか、ローカルマシンの場合は localhost と表されます。DNS 名でホストを指定する場合は、ワイルドカード "*" を 1 つだけ使用できます。ワイルドカードを使用する場合は、`*.example.com` のように、左端に置く必要があります。

ポート (またはポート範囲) は省略可能です。`N-` という形式のポート指定は、`N` またはそれ以上の番号を持つすべてのポートを表します。ここで、`N` はポート番号です。`-N` という形式のポート指定は、`N` またはそれ以下の番号を持つすべてのポートを表します。

`listen` アクションは、ローカルホストで使用される場合のみ有効です。`resolve` (ホスト/IP 解決のネームサービスルックアップ) アクションは、ほかのアクションで暗黙的に使用されます。

たとえば、SocketPermission を作成するときに次のアクセス権をコードに与えると、そのコードは `machine1.example.com` のポート `1645` に接続することと、そのポート上で接続を受け入れることができます。

```
permission java.net.SocketPermission machine1.example.com:1645, "connect,accept";
```

同様に、次のアクセス権を与えられたコードは、ローカルホストのポート 1024 ～ 65535 に接続することと、これらのポートで接続受け入れおよび待機を行うことができます。

```
permission java.net.SocketPermission "machine1.example.com:1645", "connect,accept";
permission java.net.SocketPermission "localhost:1024-", "accept,connect,listen";
```

---

注– リモートホストに対する接続受け入れや接続作成のアクセス権をコードに与えると、悪意のあるコードによって、本来アクセス権を持たない第三者に機密データが転送されたり共有されたりしやすくなるので、問題が発生することがあります。適切なアクセス権だけを与えるために、ポート番号を範囲で指定するのではなく、正確なポート番号を指定してください。

---

## SafeWord

Secure Computing の SafeWord™ または SafeWord PremierAccess™ 認証サーバーに対して送られる SafeWord 認証要求を処理するように、Access Manager を設定できます。Access Manager は、SafeWord 認証のクライアント部分を担当します。SafeWord サーバーは、Access Manager のインストールされているシステムにも、別のシステムにも置くことができます。

### Sun Java System Application Server で SafeWord を設定する

SafeWord クライアントがそのサーバーに対してソケット接続を作成するとき、デフォルトでは、Application Server の `server.policy` ファイルで SocketPermission の `connect` アクセス権だけが与えられています。SafeWord 認証を正常に機能させるには、次のアクションを許可する必要があります。

- accept
- connect
- listen
- resolve

ソケット接続のアクセス権を与えるには、Application Server の `server.policy` ファイルにエントリを追加します。SocketPermission は、ホストの指定と、そのホストへの接続方法を指定する一連のアクションとで構成されます。ホストは次のように指定されます。

```
host = (hostname | IPaddress)[:portrange] portrange =
portnumber | -portnumberportnumber-[portnumber]
```

ホストは、DNS 名または IP アドレスの数値で表されるか、ローカルマシンの場合は localhost と表されます。DNS 名でホストを指定する場合は、ワイルドカード "*" を 1 つだけ使用できます。ワイルドカードを使用する場合は、`*.example.com` のように、左端に置く必要があります。

ポート (portrange) は省略可能です。`N-` という形式のポート指定は、`N` またはそれ以上の番号を持つすべてのポートを表します。ここで、`N` はポート番号です。`-N` という形式のポート指定は、`N` またはそれ以下の番号を持つすべてのポートを表します。

listen アクションは、ローカルホストで使用される場合のみ有効です。resolve (ホスト/IP 解決のネームサービスルックアップ) アクションは、ほかのアクションで暗黙的に使用されます。

たとえば、SocketPermission を作成するときに次のアクセス権をコードに与えると、そのコードは machine1.example.com のポート 1645 に接続することと、そのポート上で接続を受け入れることができます。

```
permission java.net.SocketPermission machine1.example.com:5030, "connect,accept";
```

同様に、次のアクセス権を与えられたコードは、ローカルホストのポート 1024 ～ 65535 に接続することと、これらのポートで接続受け入れおよび待機を行うことができます。

```
permission java.net.SocketPermission "machine1.example.com:5030", "connect,accept";
permission java.net.SocketPermission "localhost:1024-", "accept,connect,listen";
```

---

注 – リモートホストに対する接続受け入れや接続作成のアクセス権をコードに与えると、悪意のあるコードによって、本来アクセス権を持たない第三者に機密データが転送されたり共有されたりしやすくなるので、問題が発生することがあります。適切なアクセス権だけを与えるために、ポート番号を範囲で指定するのではなく、正確なポート番号を指定してください。

---

## SAML

SAML (Security Assertion Markup Language) 認証モジュールでは、ターゲットサーバーで SAML 表明の受信と確認が行われます。このモジュールが、Access Manager 2005Q2 から Access Manager 7.1 などへのアップグレード後も含めて、ターゲットマシンで設定されている場合にかぎって、SAML SSO は動作します。

## SecurID

RSA の ACE/Server 認証サーバーに対して送られる SecurID 認証要求を処理するように、Access Manager を設定できます。Access Manager は、SecureID 認証のクライアント部分を担当します。ACE/Server は、Access Manager のインストールされているシステムにも、別のシステムにも置くことができます。ローカルで管理されたユーザー ID を認証する (admintool (1M) を参照) には、root でアクセスする必要があります。

SecurID 認証では、認証ヘルパー amsecuridd を使用します。認証ヘルパーは Access Manager のメインのプロセスとは独立したプロセスです。起動すると、このヘルパーは設定情報を得るためにポートで待機します。Access Manager が nobody として、または root 以外のユーザー ID で実行するようにインストールされている場合でも、*AccessManager-base*/SUNWam/share/bin/amsecuridd プロセスは root ユーザーとして実行する必要があります。amsecuridd ヘルパーの詳細については、『Access Manager Administration Reference』の「amSecurID Helper」を参照してください。

注 – このリリースの Access Manager の場合、Linux プラットフォームと Solaris x86 プラットフォームでは SecurID 認証モジュールを使用できません。この 2 つのプラットフォームでは、SecurID 認証モジュールの登録、設定、有効化を行わないでください。SecurID 認証モジュールは、SPARC のみで使用できます。

## UNIX

Access Manager がインストールされている Solaris または Linux システムで既知の UNIX ユーザー ID およびパスワードに対する認証要求を処理するように、Access Manager を設定できます。UNIX 認証ではレルム属性は 1 つだけしかなく、またグローバル属性は少ししかありませんが、システムの観点から検討すべき点があります。ローカルで管理されたユーザー ID を認証する (`admintool` (1M) 参照) には、root でアクセスする必要があります。

UNIX 認証では、認証ヘルパー `amunixd` を使用します。認証ヘルパーは Access Manager のメインのプロセスとは独立したプロセスです。起動すると、このヘルパーは設定情報を得るためにポートで待機します。UNIX ヘルパーは Access Manager ごとに 1 つだけあり、そのすべてのレルムで共用されます。

Access Manager が `nobody` として、または root 以外のユーザー ID で実行するようにインストールされている場合でも、*AccessManager-base*`/SUNWam/share/bin/amunixd` プロセスは root ユーザーとして実行する必要があります。UNIX 認証モジュールは、UNIX 認証要求を待機するために `localhost:58946` へのソケットを開くことで、`amunixd` デーモンを呼び出します。デフォルトのポートで `amunixd` ヘルパーを実行するには、次のコマンドを入力します。

```
./amunixd
```

デフォルト以外のポートで `amunixd` ヘルパーを実行するには、次のコマンドを入力します。

```
./amunixd [-c portnm] [ipaddress]
```

ipaddress と portnumber は、`AMConfig.properties` 内の `UnixHelper.ipadrs` 属性 (IPV4 形式) と `UnixHelper.port` 属性で指定されています。`amunixd` を `amserver` コマンド行ユーティリティーから実行することもできます。`amserver` は `AMConfig.properties` からポート番号と IP アドレスを取り出し、このプロセスを自動的に実行します。

`/etc/nsswitch.conf` ファイル内の `passwd` エントリでは、`/etc/passwd` および `/etc/shadow` ファイル、または NIS を認証で探すかどうかを指定します。

## Windows デスクトップ SSO

Microsoft Windows デスクトップ SSO 認証モジュールは、Microsoft Windows 2000™ に使用する、Kerberos ベースのプラグインモジュールです。このサービスを使用する

と、Kerberos Distribution Center (KDC) で認証されたユーザーは、ログインの条件を再度提示しなくても Access Manager に認証されます (シングルサインオン)。

ユーザーは、SPNEGO (Simple and Protected GSS-API Negotiation Mechanism) プロトコルで Access Manager に Kerberos トークンを送信します。この認証モジュールで Access Manager への Kerberos ベースのシングルサインオンを実行するには、ユーザーが、クライアントサイドにおいて、SPNEGO プロトコルをサポートして自分自身を認証する必要があります。一般的に、このプロトコルをサポートするすべてのユーザーは、このモジュールを使用して Access Manager に認証できます。クライアントサイドでトークンを使用できるかどうかにより、SPNEGO トークンか Kerberos トークンがこのモジュールによって提供されます。どちらの場合でもプロトコルは同一です。Microsoft Windows 2000 以上で動作している Microsoft Internet Explorer 5.01 以上では、現在このプロトコルがサポートされています。Solaris 9 および 10 上で動作する Mozilla 1.4 では SPNEGO もサポートしていますが、返されるトークンは KERBEROS トークンのみです。これは、Solaris 自体が SPNEGO をサポートしてないためです。

---

注 – Kerberos V5 認証モジュールの新機能を使用するには、JDK 1.4 以上を使用する必要があります。この SPNEGO モジュールで Kerberos ベースの SSO を実行するには、Java GSS API を使用する必要があります。

---

## Internet Explorer の既知の制限事項

WindowsDesktopSSO 認証に Microsoft Internet Explorer 6.x を使用しており、WindowsDesktopSSO モジュールで設定されている KDC レルムと一致する、ユーザーの Kerberos/SPNEGO トークンにこのブラウザでアクセスできない場合、WindowsDesktopSSO モジュールへの認証がエラーになったあとで、このブラウザはその他のモジュールに対して不正に動作します。この問題の直接的な原因は、Internet Explorer が WindowsDesktopSSO モジュールでエラーになると、ブラウザを再起動するまで、コールバックを要求されても、別のモジュールのコールバックを Access Manager に渡すことができなくなることです。このため、WindowsDesktopSSO のあとのすべてのモジュールは、ユーザー資格が NULL であるためにエラーとなります。

関連情報については、次の資料を参照してください。

http://support.microsoft.com/default.aspx?scid=kb;en-us;308074
(http://support.microsoft.com/default.aspx?scid=kb;en-us;308074)

http://www.wedgetail.com/jcsi/sso/doc/guide/troubleshooting.html#ieNTLM
(http://support.microsoft.com/default.aspx?scid=kb;en-us;308074)

---

注 – このリリースの Access Manager から、この制限は Microsoft によって解決されています。詳細については、次のマニュアルを参照してください。

http://www.microsoft.com/technet/security/bulletin/ms06-042.mspx

---

## Windows デスクトップ SSO の設定

Windows デスクトップ SSO 認証を有効にするプロセスには、次の 2 つの段階があります。

1. Microsoft Windows 2000 のドメインコントローラにユーザーを作成します。
2. Internet Explorer をセットアップします。

### ▼ Microsoft Windows 2000 のドメインコントローラにユーザーを作成する

1 ドメインコントローラで、**Access Manager** 認証モジュール用のユーザーアカウントを作成します。

**a.** 「スタート」メニューから、「プログラム」>「管理ツール」に進みます。

**b.** 「**Active Directory** ユーザーとコンピュータ」を選択します。

**c.** **Access Manager** ホスト名がユーザー **ID (**ログイン名**)** の新しいユーザーを作成します。**Access Manager** ホスト名には、ドメイン名を含めないでください。

2 ユーザーアカウントをサービスプロバイダの名前と関連付け、**Keytab** ファイルを **Access Manager** がインストールされたシステムにエクスポートします。そのためには、次のコマンドを実行します。

```
ktpass -princ host/hostname.domainname@DCDOMAIN -pass password -mapuser userName-out
hostname.host.keytab
ktpass -princ HTTP/hostname.domainname@DCDOMAIN -pass
password -mapuser userName-out hostname
.HTTP.keytab
```

`ktpass` コマンドには、次のパラメータを使用できます。

**hostname**: Access Manager が稼働する、ドメイン名なしのホスト名です。

**domainname** : Access Manager のドメイン名です。

**DCDOMAIN**: ドメインコントローラのドメイン名です。この名前は、Access Manager のドメイン名とは異なる場合があります。

**password**: ユーザーアカウントのパスワードです。`ktpass` はパスワードを検証しないので、パスワードが正しいことを確認してください。

**userName**: ユーザーアカウント ID です。これはホスト名と同じにする必要があります。

注 – 両方の Keytab ファイルがセキュリティー保護されているようにします。

サービステンプレートの値は、次の例のようになっている必要があります。

「サービス主体」: HTTP/machine1.EXAMPLE.COM@ISQA.EXAMPLE.COM

「**Keytab** ファイル名」: /tmp/machine1.HTTP.keytab

「**Kerberos** レルム」: ISQA.EXAMPLE.COM

「**Kerberos** サーバー名」: machine2.EXAMPLE.com

「ドメイン名を含む主体を返す」: false

「認証レベル」: 22

注 – Windows 2003 または Windows 2003 サービスパックを使用している場合は、次の ktpass コマンド構文を使用します。

```
ktpass /out filename /mapuser username /princ HTTP/hostname.domainname
     /crypto encryptiontype /rndpass /ptype principaltype /target domainname
```

次に例を示します。

```
ktpass /out demo.HTTP.keytab /mapuser http
/princ HTTP/demo.identity.sun.com@IDENTITY.SUN.COM /crypto RC4-HMAC-NT
/rndpass /ptype KRB5_NT_PRINCIPAL /target IDENTITY.SUN.COM
```

構文の定義については、http://technet2.microsoft.com/WindowsServer/en/library/64042138-9a5a-4981-84e9-d576a8db0d051033.mspx?mfr=true の Web サイトを参照してください。

3 サーバーを再起動します。

## ▼ Internet Explorer をセットアップする

この手順は、Microsoft Internet Explorer™ 6 以上に当てはまります。これよりも前のバージョンを使用している場合は、Access Manager がブラウザのインターネットゾーンにあり、ネイティブ Microsoft Windows 認証が有効であることを確認します。

1 「ツール」メニューで、「インターネットオプション」>「詳細設定」>「セキュリティー」に進みます。

2 「統合 **Microsoft Windows** 認証を使用する」オプションを選択します。

3 「セキュリティー」>「イントラネット」に進みます。

a. 「レベルのカスタマイズ」を選択します。「ユーザー認証」の「ログオン」で「イントラネットゾーンでのみ自動的にログオンする」オプションを選択します。

b. 「サイト」に進み、すべてのオプションを選択します。

c. 「詳細設定」をクリックして、**Access Manager** をローカルゾーンに追加します (まだ追加されていない場合)。

## Windows NT

Access Manager は、すでにインストールされている Microsoft Windows NT サーバーまたは Microsoft Windows 2000 サーバーで使用できるように設定できます。Access Manager は、NT 認証のクライアント部分を担当します。

1. NT サーバーを設定します。詳しい手順については、Microsoft Windows NT サーバーのマニュアルを参照してください。
2. Microsoft Windows NT 認証モジュールを追加し、有効にする前に、Samba クライアントを入手してインストールし、Solaris システム上の Access Manager と通信できるようにする必要があります。

## Samba クライアントのインストール

Microsoft Windows NT 認証モジュールをアクティブにするには、Samba Client 2.2.2 をダウンロードして次のディレクトリにインストールする必要があります。

```
AccessManager-base/SUNWam/bin
```

Samba Client は、Microsoft Windows マシンと UNIX マシンを共存させるためのファイルサーバー兼プリントサーバーで、専用の Microsoft Windows NT/2000 Server を必要としません。詳細とダウンロードについては、
http://wwws.sun.com/software/download/products/3e3af224.html を参照してください。

Red Hat Linux とともに出荷される Samba クライアントは、次のディレクトリに置かれています。

```
/usr/bin
```

Linux 用 Microsoft Windows NT 認証モジュールを使って認証を行うためには、クライアントのバイナリを Access Manager の次のディレクトリにコピーします。

**AccessManager-base**/sun/identity/bin

注– 複数のインタフェースがある場合には、追加の設定が必要です。複数のインタフェースは smb.conf ファイルで設定し、それを mbclient へ伝えることにより、設定できます。

# 認証モジュールインスタンス

デフォルトの認証モジュールに基づいて、複数の認証モジュールインスタンスをレルム用に作成できます。同じ認証モジュールを元に、個別に設定した複数のインスタンスを追加できます。

## ▼ 新しい認証モジュールインスタンスを作成する

1 新しい認証モジュールインスタンスを追加するレルムの名前をクリックします。

2 「認証」タブを選択します。

注– 「管理者認証設定」ボタンにより、管理者専用に認証サービスを定義できます。この属性は、管理者とエンドユーザーの認証モジュールを別々のものにする必要がある場合に使用できます。Access Manager コンソールにアクセスするときは、この属性で設定されているモジュールが使用されます。

3 「モジュールインスタンス」リストの「新規」をクリックします。

4 認証モジュールインスタンスの名前を入力します。属性の名前は一意でなければいけません。

5 レルムの認証モジュールタイプの「タイプ」を選択します。

6 「了解」をクリックします。

7 新しく作成したモジュールインスタンスの名前をクリックし、そのモジュールのプロパティーを編集します。各モジュールタイプのプロパティーの定義については、オンラインヘルプの「認証」の節を参照してください。

8 複数のモジュールインスタンスを追加するときは、これらの手順を繰り返します。

# 認証連鎖

1 つ以上の認証モジュールが設定できるので、ユーザーは認証資格をすべての認証モジュールに渡す必要があります。これは、認証連鎖と呼ばれます。Access Manager で

の認証連鎖は、認証サービスに統合された JAAS フレームワークを使用して実現されます。モジュール連鎖は、認証設定サービスの下に設定されます。

## ▼ 新しい認証連鎖を作成する

1 新しい認証連鎖を追加するレルムの名前をクリックします。

2 「認証」タブを選択します。

3 「認証連鎖」リストの「新規」をクリックします。

4 認証連鎖の名前を入力します。

5 「了解」をクリックします。

6 「追加」をクリックし、連鎖に含める認証モジュールインスタンスを定義します。そのためには、「インスタンス」リストからモジュールインスタンス名を選択します。このリストに表示されるモジュールインスタンス名は、「モジュールインスタンス」属性で作成されます。

7 連鎖の条件を選択します。以上のフラグによって、認証モジュールの適用条件が確立されます。適用には階層があります。「必須」がもっとも高く、「オプション」がもっとも低くなります。

   必要　認証にはモジュールインスタンスが必要です。認証に成功すると、「認証連鎖」リストの次のインスタンスへと認証が進行します。認証に失敗すると、制御がただちにアプリケーションに返されます。この場合、「認証連鎖」リストの次のインスタンスには認証が進行しません。

   必須　認証には、このモジュールへの認証が必要です。この連鎖の必須モジュールのいずれかが失敗すると、最終的に認証連鎖全体が失敗します。ただし、必須モジュールが成功しても失敗しても、連鎖の次のモジュールへと制御が進行します。

   十分　認証にモジュールインスタンスは必要ありません。認証に成功するとすぐに、制御がアプリケーションに返されます。この場合、モジュールインスタンスリストの次のインスタンスには認証が進行しません。認証に失敗すると、「認証連鎖」リストの次のインスタンスへと認証が進行します。

   任意　認証にモジュールインスタンスは必要ありません。認証に成功しても失敗しても、「認証連鎖」リストの次のインスタンスへと認証が進行します。

8 連鎖のオプションを入力します。これにより、モジュールの追加オプションがキーと値のペアとして有効になります。複数のオプションを指定するときは、スペースで区切ります。

9 次の属性を定義します。

| | |
|---|---|
| 「ログイン成功時に返す URL」 | 認証が成功した場合にユーザーをリダイレクトする URL を指定します。 |
| 「ログイン失敗時に返す URL」 | 認証が成功しなかった場合にユーザーをリダイレクトする URL を指定します。 |
| 「認証ポストプロセスクラス」 | ログインが成功または失敗したあとに認証ポストプロセスのカスタマイズに使用する Java クラスの名前を定義します。 |

10 「保存」をクリックします。

# 認証タイプ

認証サービスは、さまざまな認証適用方法を提供します。それらの異なる認証方法にアクセスするには、ログイン URL パラメータを指定するか、認証 API を使用します。詳細については、『Sun Java System Access Manager 7.1 Developer's Guide』の第 2 章「Using Authentication APIs and SPIs」を参照してください。認証モジュールを設定する前に、特定の認証モジュール名を含むように、コア認証サービス属性のレルム認証モジュールを修正する必要があります。

認証設定サービスは、次の認証タイプ用の認証モジュールを定義するために使用します。

- 64 ページの「レルムに基づく認証」
- 67 ページの「組織に基づく認証」
- 70 ページの「ロールに基づく認証」
- 73 ページの「サービスに基づく認証」
- 76 ページの「ユーザーに基づく認証」
- 78 ページの「認証レベルに基づく認証」
- 81 ページの「モジュールに基づく認証」

これらの認証タイプのいずれかで認証モジュールを定義すると、認証プロセスの成功または失敗に基づいて、リダイレクト URL、およびポストプロセス Java クラス仕様を提供するように設定できます。

## 認証タイプによってアクセスが決定される方法

各認証方法では、ユーザーは認証に成功するか失敗します。成功または失敗の決定後、各方法では次の手順を実行します。手順 1 ～ 3 は認証が成功した場合に実行され、手順 4 は成功した認証と失敗した認証の両方で実行されます。

1. Access Manager によって、認証されたユーザーが Directory Server データストアに定義されているかどうか、またプロファイルが有効であるかどうかが確認されます。

   コア認証モジュールの「ユーザープロファイル」属性は、「必須」、「動的」、「ユーザーエイリアスを使用して動的に」、または「無視」として定義できます。認証に成功すると、Access Manager によって、認証されたユーザーが Directory Server データストアに定義されているかどうかが確認され、「ユーザープロファイル」の値が「必須」である場合は、プロファイルが有効かどうかが確認されます。これはデフォルトの場合です。「ユーザープロファイル」が動的な設定である場合、認証サービスはユーザープロファイルをデータストアに作成します。「ユーザープロファイル」が「無視」に設定されている場合は、ユーザーの検証は行われません。

2. 認証ポストプロセス SPI が実行されます。

   コア認証モジュールには、値として認証ポストプロセスクラス名を含む「認証ポストプロセスクラス」属性が含まれています。`AMPostAuthProcessInterface` は、ポストプロセスインタフェースです。このインタフェースは、認証の成功または失敗時、またはログアウト時に実行できます。

3. セッショントークンで次のプロパティーが追加または更新され、ユーザーのセッションがアクティブになります。

   **realm**: これは、ユーザーが所属するレルムの DN です。

   **Principal**: ユーザーの DN です。

   **Principals**: ユーザーが認証を受けた名前のリストです。このプロパティーは、パイプで区切られたリストとして定義された複数の値を持つことができます。

   **UserId**: モジュールが返すユーザーの DN であるか、LDAP またはメンバーシップ以外のモジュールの場合はユーザー名です。すべての Principal は、同じユーザーにマッピングされる必要があります。UserId は、すべての Principal がマッピングされるユーザー DN です。

   ---

   注 – このプロパティーは、DN 以外の値になることがあります。

   ---

   **UserToken**: ユーザー名です。すべての Principal は、同じユーザーにマッピングされる必要があります。UserToken は、すべての Principal がマッピングされるユーザー名です。

   **Host**: クライアント用のホスト名または IP アドレスです。

   **authLevel**: ユーザーが認証を受けた最高のレベルです。

   **AuthType**: ユーザーが認証を受けた認証モジュールの、パイプで区切られたリストです (例、`module1|module2|module3`)。

**clientType**: クライアントブラウザのデバイスタイプです。

**Locale**: クライアントのロケールです。

**CharSet**: クライアント用に定めらた文字セットです。

**Role**: ロールに基づく認証にのみ適用可能であり、ユーザーが属すロールです。

**Service**: サービスに基づく認証にのみ適用可能であり、ユーザーが属すサービスです。

4. Access Manager は、成功または失敗した認証のあとにユーザーをリダイレクトする場所についての情報を検索します。

   URL のリダイレクトは、Access Manager のページまたは URL のどちらかにすることができます。リダイレクトは、Access Manager が認証方法に基づいて、また認証が成功したか失敗したかによってリダイレクトを検索する優先順位のもとに行われます。この順序については、次の認証方法についての節の、URL のリダイレクトの部分で詳しく説明します。

### URL のリダイレクト

認証設定サービスでは、成功または失敗した認証に対する URL のリダイレクトを割り当てることができます。その URL 自体は、認証設定サービスの「ログイン成功時に返す URL」および「ログイン失敗時に返す URL」属性で定義します。URL のリダイレクトを有効にするために、ロール、レルム、またはユーザー用に設定するように、認証設定サービスをレルムに追加し、利用可能にする必要があります。認証設定サービスの追加時は、LDAP で必須、というように認証モジュールを追加するようにしてください。

## レルムに基づく認証

この認証方式により、ユーザーはレルムまたはサブレルムの認証を受けることができます。これは、Access Manager のデフォルトの認証方法です。レルムの認証方法は、コア認証モジュールをレルムに登録し、「レルム認証設定」属性を定義することによって設定します。

### レルムに基づく認証ログイン URL

認証のレルムは、ユーザーインタフェースのログイン URL に `realm` パラメータまたは `domain` パラメータを定義して指定できます。認証の要求のレルムは、次の優先順位で判断されます。

1. `domain` パラメータ。
2. `realm` パラメータ。

3. 管理サービスの「DNS エイリアス名」属性の値。

   正しいレルムを呼び出したあと、ユーザーが認証を受ける認証モジュールは、コア認証サービスの「レルム認証設定」属性から取得されます。レルムに基づく認証を指定し、開始するログイン URL を次に示します。

   ```
   http://server_name.domain_name:port/amserver/UI/Login
   http://server_name.domain_name:port/amserver/UI/Login?domain=domain_name
   http://server_name.domain_name:port/amserver/UI/Login?realm=realm_name
   ```

   定義されたパラメータがない場合、レルムはログイン URL に指定されたサーバーホストとドメインから判断されます。

---

注 – 特定のレルムのメンバーであるユーザーが、そのレルムへの認証を行ってから別のレルムへの認証を試みる場合、渡されるパラメータは、realm および module の 2 つのみです。たとえば、User1 が realmA のメンバーで、そのレルムへの認証を行ったあとに、realmB に切り替えるか、またはそのレルムへの認証を行う場合、ユーザーは、realmB 用に定義されたモジュールインスタンスを使用して realmB への新しい認証を開始するか、または既存の realmA との認証済みのセッションに戻るように要求する警告ページを受け取ります。ユーザーが realmB への認証を選択した場合、レルム名とモジュール名 (指定した場合) のみが渡され、新しい認証プロセスの決定に使用されます。

---

## レルムに基づく認証リダイレクト URL

組織に基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### レルムに基づく認証が成功した場合のリダイレクト URL

レルムに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の優先順位で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。

6. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
9. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性に設定された URL。

### レルムに基づく認証に失敗した場合のリダイレクト URL

レルムに基づく認証に失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. gotoOnFail ログイン URL パラメータで設定された URL。
3. ユーザーのエントリ (amUser.xml) の iplanet-am-user-failure-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのエントリ (amUser.xml) の iplanet-am-user-failure-url 属性用に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
9. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

## レルムに基づく認証を設定する

認証モジュールは、最初にコア認証サービスをレルムに追加することで、レルム用に設定できます。

### ▼ レルムの認証属性を設定する

1. 認証連鎖を追加するレルムに移動します。
2. 「認証」タブをクリックします。
3. プルダウンメニューから「デフォルト認証連鎖」を選択します。
4. プルダウンメニューから「管理者認証連鎖」を選択します。この属性は、管理者とエンドユーザーの認証モジュールを別々のものにする必要がある場合に使用できます。デフォルトの認証モジュールは **LDAP** です。
5. 認証連鎖を定義したら、「保存」をクリックします。

## 組織に基づく認証

この認証タイプは、旧バージョンモードでインストールされた Access Manager の配備先にのみ適用されます。

この認証方法では、ユーザーが組織またはサブ組織に対する認証を受けることができます。これは、Access Manager のデフォルトの認証方法です。組織の認証方法は、コア認証モジュールを組織に登録し、「組織認証設定」属性を定義することによって設定します。

### 組織に基づく認証のログイン URL

認証の組織は、ユーザーインタフェースのログイン URL に `org` パラメータまたは `domain` パラメータを定義して指定できます。認証の要求の組織は、次の優先順位で判断されます。

1. `domain` パラメータ。
2. `org` パラメータ。
3. 管理サービスの「`DNS` エイリアス名」 (組織のエイリアス名) 属性の値。

   正しい組織を呼び出したあと、ユーザーが認証を受ける認証モジュールは、コア認証サービスの「組織認証設定」属性から取得されます。組織に基づく認証を指定し、開始するログイン URL を次に示します。

   ```
   http://server_name.domain_name:port/amserver/UI/Login
   http://server_name.domain_name:port/amserver/UI/Login?domain=domain_name
   http://server_name.domain_name:port/amserver/UI/Login?org=org_name
   ```

   定義されたパラメータがない場合、組織はログイン URL に指定されたサーバーホストとドメインから判断されます。

注 – 特定の組織のメンバーであるユーザーが、その組織への認証を行ってから別の組織への認証を試みる場合、渡されるパラメータは、org および module の 2 つのみです。たとえば、User1 が orgA のメンバーで、その組織への認証を行ったあとに、orgB に切り替えるか、またはその組織への認証を行う場合、ユーザーは、orgB 用に定義されたモジュールインスタンスを使用して orgB への新しい認証を開始するか、または既存の orgA との認証済みのセッションに戻るように要求する警告ページを受け取ります。ユーザーが orgB への認証を選択した場合、組織名とモジュール名 (指定した場合) のみが渡され、新しい認証プロセスの決定に使用されます。

## 組織に基づく認証のリダイレクト URL

組織に基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### 組織に基づく認証が成功した場合のリダイレクト URL

組織に基づく認証が成功した場合のリダイレクト URL は、次の場所を次の優先順位で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーの組織エントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
9. ユーザーの組織エントリの iplanet-am-auth-login-success-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性に設定された URL。

### 組織に基づく認証に失敗した場合のリダイレクト URL

組織に基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `gotoOnFail` ログイン URL パラメータで設定された URL。
3. ユーザーのエントリ (`amUser.xml`) の `iplanet-am-user-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーのロールエントリの `iplanet-am-auth-login-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
5. ユーザーの組織エントリの `iplanet-am-auth-login-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
6. グローバルデフォルトとして `iplanet-am-auth-login-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
7. ユーザーのエントリ (`amUser.xml`) の `iplanet-am-user-failure-url` 属性用に設定された URL。
8. ユーザーのロールエントリの `iplanet-am-auth-login-failure-url` 属性に設定された URL。
9. ユーザーの組織エントリの `iplanet-am-auth-login-failure-url` 属性に設定された URL。
10. グローバルデフォルトとして `iplanet-am-auth-login-failure-url` 属性に設定された URL。

## 組織に基づく認証を設定する

認証モジュールは、最初にコア認証サービスを組織に追加することで、組織用に設定できます。

### ▼ 組織の認証属性を設定する

**1** 認証連鎖を追加する組織に移動します。

**2** 「認証」タブをクリックします。

**3** プルダウンメニューから「デフォルト認証連鎖」を選択します。

**4** プルダウンメニューから「管理者認証連鎖」を選択します。この属性は、管理者とエンドユーザーの認証モジュールを別々のものにする必要がある場合に使用できます。デフォルトの認証モジュールは **LDAP** です。

5 認証連鎖を定義したら、「保存」をクリックします。

# ロールに基づく認証

この認証方法では、ユーザーはレルムまたはサブレルム内の (静的またはフィルタリングされた) ロールに対する認証を受けることができます。

---

注 – 認証設定サービスは、レルムに登録してからでなければロールにインスタンスとして登録できません。

---

認証を成功させるには、ユーザーはロールに属し、そのロールに設定された認証設定サービスインスタンスに定義された各モジュールに対する認証を受ける必要があります。ロールに基づく認証の各インスタンスに対して、次の属性を指定できます。

「競合の解決レベル」: 同一のユーザーが含まれることがある 2 つの異なるロールに定義された認証設定サービスインスタンスに対する優先レベルを設定します。たとえば、`User 1` が `Role 1` および `Role 2` に割り当てられている場合を想定します。ユーザーが認証を試みたときに、認証の成功または失敗時のリダイレクトや認証後プロセスに対して `Role 1` の優先順位がより高くなるように、`Role1` により高い競合解決レベルを設定することができます。

「認証設定」: ロールの認証プロセスに設定された認証モジュールを定義します。

「ログイン成功時に返す **URL**」: 認証が成功した場合にユーザーがリダイレクトされる URL を定義します。

「ログイン失敗時に返す **URL**」: 認証が失敗した場合にユーザーがリダイレクトされる URL を定義します。

「認証ポストプロセスクラス」: 認証後インタフェースを定義します。

## ロールに基づく認証のログイン URL

ロールに基づく認証は、ユーザーインタフェースのログイン URL にロールパラメータを定義して指定できます。正しいロールを呼び出したあと、ユーザーが認証を受ける認証モジュールは、そのロールに定義された認証設定サービスインスタンスから取得されます。

このロールに基づく認証を指定し開始するログイン URL を次に示します。

```
http://server_name.domain_name:port/amserver/UI/Login?role=role_name
http://server_name.domain_name:port/amserver/UI/Login?realm=realm_name&role=role_name
```

realm パラメータが設定されていない場合、ロールが属すレルムはログイン URL そのものに指定されたサーバーホストおよびドメインから判断されます。

## ロールに基づく認証のリダイレクト URL

ロールに基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### ロールに基づく認証が成功した場合のリダイレクト URL

ロールに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `goto` ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーが認証を受けたロールの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
5. 認証されたユーザーの別のロールエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。このオプションは、前のリダイレクト URL が失敗した場合の代替リダイレクト URL です。
6. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
7. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
8. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性に設定された URL。
9. ユーザーが認証を受けたロールの `iplanet-am-auth-login-success-url` 属性に設定された URL。
10. 認証されたユーザーの別のロールエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。このオプションは、前のリダイレクト URL が失敗した場合の代替リダイレクト URL です。
11. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。
12. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性に設定された URL。

### ロールに基づく認証が失敗した場合のリダイレクト URL

ロールに基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-failure-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーが認証を受けたロールの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
5. 認証されたユーザーの別のロールエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。このオプションは、前のリダイレクト URL が失敗した場合の代替リダイレクト URL です。
6. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
8. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-failure-url 属性に設定された URL。
9. ユーザーが認証を受けたロールの iplanet-am-auth-login-failure-url 属性に設定された URL。
10. 認証されたユーザーの別のロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。このオプションは、前のリダイレクト URL が失敗した場合の代替リダイレクト URL です。
11. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
12. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

## ▼ ロールに基づく認証を設定する

**1** 認証設定サービスを追加するレルム (または組織) に移動します。

**2** 「対象」タブをクリックします。

**3** 「フィルタロール」または「ロール」を選択します。

4. 認証設定を設定するロールを選択します。
   認証設定サービスがロールに追加されていない場合は、「追加」をクリックし、「認証サービス」を選択して「次へ」をクリックします。
5. プルダウンメニューから、有効にする「デフォルト認証連鎖」を選択します。
6. 「保存」をクリックします。

---

注 – 新しいロールを作成している場合、そのロールに認証設定サービスは自動的に割り当てられません。ロールを作成する前に、ロールプロファイルページの上部で認証設定サービスを選択していることを確認してください。

ロールに基づく認証が有効になっているときは、LDAP 認証モジュールはデフォルトのままにでき、メンバーシップを設定する必要はありません。

---

# サービスに基づく認証

この認証方法では、ユーザーはレルムまたはサブレルムに登録された特定のサービスまたはアプリケーションに対する認証を受けることができます。このサービスは、認証設定サービス内でサービスインスタンスとして設定され、インスタンス名が関連付けられます。認証を成功させるには、ユーザーはそのサービスに対して設定されている、認証設定サービスインスタンスで定義済みの各モジュールから認証を受けなければなりません。サービスに基づく認証の各インスタンスに対して、次の属性を指定できます。

「認証設定」: サービスの認証プロセスに設定された認証モジュールを定義します。

「ログイン成功時に返す **URL**」: 認証が成功した場合にユーザーがリダイレクトされる URL を定義します。

「ログイン失敗時に返す **URL**」: 認証が失敗した場合にユーザーがリダイレクトされる URL を定義します。

「認証ポストプロセスクラス」: 認証後インタフェースを定義します。

## サービスに基づく認証のログイン URL

サービスに基づく認証は、ユーザーインタフェースのログイン URL にサービスパラメータを定義して指定できます。サービスを呼び出したあと、ユーザーが認証を受ける認証モジュールは、そのサービスに定義された認証設定サービスインスタンスから取得されます。

このサービスに基づく認証を指定し開始するログイン URL を次に示します。

```
http://server_name.domain_name:port/amserver/UI/
Login?service=auth-chain-name
```

および

```
http://server_name.domain_name:port/amserver
     /UI/Login?realm=realm_name&service=auth-chain-name
```

org パラメータが設定されていない場合、レルムはログイン URL そのものに指定されたサーバーホストとドメインから判断されます。

## サービスに基づく認証のリダイレクト URL

サービスに基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### サービスに基づく認証が成功した場合のリダイレクト URL

サービスに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーが認証を受けたサービスの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
6. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
7. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
8. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性に設定された URL。
9. ユーザーが認証を受けたサービスの iplanet-am-auth-login-success-url 属性に設定された URL。
10. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
11. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性に設定された URL。

12. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性に設定された URL。

### サービスに基づく認証が失敗した場合のリダイレクト URL

サービスに基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-failure-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーが認証を受けたサービスの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
6. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
8. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-failure-url 属性に設定された URL。
9. ユーザーが認証を受けたサービスの iplanet-am-auth-login-failure-url 属性に設定された URL。
10. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
11. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
12. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

## ▼ サービスに基づく認証を設定する

認証モジュールは、認証設定サービスを追加すると、サービス用に設定されます。そのためには、次の手順を実行します。

**1** サービスに基づく認証を設定するレルムを選択します。

**2** 「認証」タブをクリックします。

3 認証モジュールインスタンスを作成します。

4 認証連鎖を作成します。

5 「保存」をクリックします。

6 レルムのサービスに基づく認証にアクセスするには、次のアドレスを入力します。

`http://`*server_name.domain_name:port*`/amserver/UI/Login?realm=`*realm_name*`&service=`*auth-chain-name*

# ユーザーに基づく認証

この認証方法では、ユーザーはユーザー専用に設定された認証プロセスに対する認証を受けることができます。このプロセスは、ユーザーのプロファイルの「ユーザー認証設定」属性の値として設定されます。認証を成功させるには、ユーザーは定義された各モジュールに対して認証する必要があります。

## ユーザーに基づく認証のログイン URL

ユーザーに基づく認証は、ユーザーインタフェースのログイン URL にユーザーパラメータを定義して指定できます。正しいユーザーを呼び出したあと、ユーザーが認証を受ける認証モジュールは、そのユーザーに定義されたユーザー認証インスタンスから取得されます。

このロールに基づく認証を指定し開始するログイン URL を次に示します。

```
http://server_name.domain_name:port/amserver/UI/Login?user=user_name
http://server_name.domain_name:port/amserver/UI/Login?org=org_name&user=user_name
```

realm パラメータが設定されていない場合、ロールが属すレルムはログイン URL そのものに指定されたサーバーホストおよびドメインから判断されます。

## ユーザーエイリアスリスト属性

ユーザーに基づく認証の要求を受け取ると、認証サービスはまずユーザーが有効なユーザーであることを確認してから、ユーザーの認証設定データを取得します。ユーザーログイン URL パラメータの値に複数の有効なユーザープロファイルが関連付けられている場合は、すべてのプロファイルを指定されたユーザーにマップする必要があります。ユーザープロファイルのユーザーエイリアス属性 (`iplanet-am-user-alias-list`) には、ユーザーに属すその他のプロファイルを定義できます。マッピングが失敗すると、ユーザーは有効なセッションを拒否されます。ユーザーの 1 人が最上位管理者の場合は、例外として、ユーザーのマッピングの検証が行われていなくても、そのユーザーには最上位管理者の権限が与えられます。

## ユーザーに基づく認証のリダイレクト URL

ユーザーに基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### ユーザーに基づく認証が成功した場合のリダイレクト URL

ユーザーに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の優先順位で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `goto` ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーのロールエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
6. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
7. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性に設定された URL。
8. ユーザーのロールエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。
9. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。
10. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性に設定された URL。

### ユーザーに基づく認証に失敗した場合のリダイレクト URL

ユーザーに基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `gotoOnFail` ログイン URL パラメータで設定された URL。
3. ユーザーのエントリ (`amUser.xml`) の `iplanet-am-user-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーのロールエントリの `iplanet-am-auth-login-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。

5. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのエントリ (amUser.xml) の iplanet-am-user-failure-url 属性用に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
9. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

### ▼ ユーザーに基づく認証を設定する

1 ユーザーの認証を設定するレルムに移動します。

2 「対象」タブをクリックし、「ユーザー」をクリックします。

3 変更するユーザーの名前をクリックします。
ユーザープロファイルが表示されます。

---

注 – 新しいユーザーを作成している場合、そのユーザーに認証設定サービスは自動的に割り当てられません。ユーザーを作成する前に、サービスプロファイルで認証設定サービスを選択していることを確認してください。このオプションを選択しないと、ユーザーはロールに定義された認証設定を継承しません。

---

4 「ユーザー認証設定」属性で、適用する認証連鎖を選択します。

5 「保存」をクリックします。

## 認証レベルに基づく認証

それぞれの認証モジュールは、その認証レベルに整数値が関連付けられています。認証レベルを割り当てるには、「設定」で認証モジュールをクリックし、モジュールの「認証レベル」属性で対応する値を変更します。認証レベルが高いということは、1 つ以上の認証モジュールで認証を受けたそのユーザーの信頼性のレベルが高いということです。

ユーザーがそのモジュールに対する認証に成功すると、認証レベルがユーザーの SSO トークンに設定されます。複数の認証モジュールに対して認証を受ける必要があり、認証に成功した場合は、最高の認証レベルの値がユーザーの SSO トークンに設定されます。

ユーザーがサービスへのアクセスを試みる場合、サービスでは、そのユーザーの SSO トークンの認証レベルを確認することで、そのユーザーがアクセスを許可されているかどうかを判別できます。次に、設定された認証レベルで認証モジュールにパスするように、ユーザーをリダイレクトします。

ユーザーは特定の認証レベルで認証モジュールにアクセスすることもできます。たとえばユーザーが次の構文でログインします。

```
http://hostname:port/deploy_URI/UI/Login?authlevel=
auth_level_value
```

認証レベルが *auth_level_value* 以上であるすべてのモジュールが、ユーザーが選択するための認証メニューとして表示されます。一致するモジュールが 1 つしかなかった場合は、その認証モジュールのログインページが直接表示されます。

この認証の方法では、ID が認証を受けられるモジュールのセキュリティーレベルを、管理者が指定できます。各認証モジュールには、それぞれの「認証レベル」属性があり、この属性の値は任意の有効な整数として定義できます。認証レベルに基づく認証では、認証サービスは、ログイン URL パラメータに指定された値以上の認証レベルを持つ認証モジュールを含むメニューを持つモジュールログインページを表示します。ユーザーは、提示されたリストからモジュールを選択します。ユーザーがモジュールを選択すると、以降のプロセスはモジュールに基づく認証に基づきます。

## 認証レベルに基づく認証のログイン URL

認証レベルに基づく認証は、ユーザーインタフェースのログイン URL に authlevel パラメータを定義して指定できます。関連するモジュールのリストを示すログイン画面を呼び出したあと、ユーザーは認証を受けるモジュールを選択する必要があります。認証レベルに基づく認証を指定し開始するログイン URL を次に示します。

```
http://server_name.domain_name:port/amserver/UI/Login?authlevel=authentication_level
```

および

```
http://server_name.domain_name:port/amserver/UI/
Login?realm=realm_name&authlevel=authentication_level
```

`realm` パラメータが設定されていない場合、ユーザーが属すレルムはログイン URL そのものに指定されたサーバーホストおよびドメインから判断されます。

## 認証レベルに基づく認証のリダイレクト URL

認証レベルに基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### 認証レベルに基づく認証が成功した場合のリダイレクト URL

認証レベルに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の優先順位で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `goto` ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーのロールエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
6. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性用に `clientType` カスタムファイルに設定された URL。
7. ユーザーのプロファイル (`amUser.xml`) の `iplanet-am-user-success-url` 属性に設定された URL。
8. ユーザーのロールエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。
9. ユーザーのレルムエントリの `iplanet-am-auth-login-success-url` 属性に設定された URL。
10. グローバルデフォルトとして `iplanet-am-auth-login-success-url` 属性に設定された URL。

### 認証レベルに基づく認証が失敗した場合のリダイレクト URL

認証レベルに基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. `gotoOnFail` ログイン URL パラメータで設定された URL。
3. ユーザーのエントリ (`amUser.xml`) の `iplanet-am-user-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。
4. ユーザーのロールエントリの `iplanet-am-auth-login-failure-url` 属性用に `clientType` カスタムファイルに設定された URL。

5. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのエントリ (amUser.xml) の iplanet-am-user-failure-url 属性に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
9. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

# モジュールに基づく認証

ユーザーは次の構文を使用して、特定の認証モジュールにアクセスできます。

```
http://hostname:port/deploy_URI/UI/Login?module=
module_name
```

認証モジュールにアクセスする前に、その認証モジュール名を含むように、コア認証サービス属性のレルム認証モジュールを修正する必要があります。認証モジュール名がこの属性に含まれていない場合、ユーザーが認証を試みると「認証モジュールが拒否されました」ページが表示されます。

この認証の方法では、ユーザーは認証を受けるモジュールを指定できます。指定するモジュールは、ユーザーがアクセスするレルムまたはサブレルムに登録する必要があります。これは、レルムのコア認証サービスの「レルム認証モジュール」属性に設定されます。モジュールに基づく認証の要求を受け取ると、認証サービスは、モジュールが指定されたように正しく設定されていることを確認し、モジュールが定義されていない場合は、ユーザーはアクセスを拒否されます。

## モジュールに基づく認証のログイン URL

モジュールパラメータを定義して、ユーザーインタフェースのログイン URL にモジュールに基づく認証を指定できます。モジュールに基づく認証を指定し開始するログイン URL を次に示します。

```
http://server_name.domain_name:port/amserver/UI/Login?module=authentication_module_name
http://server_name.domain_name:port/amserver/UI/
Login?org=org_name&module=authentication_module_name
```

org パラメータが設定されていない場合、ユーザーが属すレルムはログイン URL そのものに指定されたサーバーホストおよびドメインから判断されます。

## モジュールに基づく認証のリダイレクト URL

モジュールに基づく認証が成功または失敗したときに、Access Manager はユーザーのリダイレクト先の情報を検索します。この情報をアプリケーションが検索する優先順位を次に示します。

### モジュールに基づく認証が成功した場合のリダイレクト URL

モジュールに基づく認証が成功した場合のリダイレクト URL は、次の場所を次の優先順位で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. goto ログイン URL パラメータで設定された URL。
3. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのプロファイル (amUser.xml) の iplanet-am-user-success-url 属性に設定された URL。
8. ユーザーのロールエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
9. ユーザーのレルムエントリの iplanet-am-auth-login-success-url 属性に設定された URL。
10. グローバルデフォルトとして iplanet-am-auth-login-success-url 属性に設定された URL。

### モジュールに基づく認証に失敗した場合のリダイレクト URL

モジュールに基づく認証が失敗した場合のリダイレクト URL は、次の場所を次の順序で確認することによって判断されます。

1. 認証モジュールで設定された URL。
2. gotoOnFail ログイン URL パラメータで設定された URL。

3. ユーザーのエントリ (amUser.xml) の iplanet-am-user-failure-url 属性用に clientType カスタムファイルに設定された URL。
4. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
5. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
6. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性用に clientType カスタムファイルに設定された URL。
7. ユーザーのロールエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
8. ユーザーのレルムエントリの iplanet-am-auth-login-failure-url 属性に設定された URL。
9. グローバルデフォルトとして iplanet-am-auth-login-failure-url 属性に設定された URL。

# ユーザーインタフェースのログイン URL

認証サービスユーザーインタフェースには、Web ブラウザの場所ツールバーにログイン URL を入力してアクセスします。この URL は次のとおりです。

```
http://AccessManager-root/.domain_name:port /service_deploy_uri /UI/Login
```

---

注 – インストールの間に、*service_deploy_uri* は amserver として設定されます。このマニュアル全体にわたり、このデフォルトのサービス配備 URI が使用されています。

---

特定の認証方法や、成功、失敗した認証の URL のリダイレクトを定義するために、ユーザーインタフェースのログイン URL にログイン URL パラメータを付加することもできます。

## ログイン URL パラメータ

URL パラメータは、URL の終わりに付加される名前と値のペアです。このパラメータは疑問符 (?) で始まり、名前=値の形式をとります。名前=値の形式をとります。たとえば、次のようにいくつかのパラメータを 1 つのログイン URL に結合できます。

```
http://server_name.domain_name:port/amserver/UI/
Login?module=LDAP&locale=ja&goto=http://www.sun.com
```

複数のパラメータを指定する場合は、アンパサンド (&) で区切ります。複数のパラメータを指定する場合は、次のガイドラインに従う必要があります。

- 各パラメータは、1 つの URL に 1 回のみ指定できます。たとえば、`module=LDAP&module=NT` は受け入れられません。
- `org` パラメータと `domain` パラメータは両方ともログインレルムを決定します。この場合、ログイン URL にはこの 2 つのパラメータどちらかを使用する必要があります。両方を使用する場合に優先順位を指定しないと、1 つのみが有効になります。
- `user`、`role`、`service`、`module`、および `authlevel` は、それぞれの基準に基づいて認証モジュールを定義するためのパラメータです。このため、ログイン URL にはこれらのパラメータのいずれか 1 つのみを使用する必要があります。複数のパラメータを使用する場合に優先順位を指定しないと、1 つのみが有効になります。

次の節では、ユーザーインタフェースのログイン URL に付加され、Web ブラウザの場所ツールバーに入力されたときに、さまざまな認証機能を実現するパラメータについて説明します。

---

注 – レルム全体に配布する認証 URL およびパラメータを単純なものにするには、管理者は単純な URL を持つ HTML ページを作成し、そのページにすべての設定された認証方法に対するより複雑なログイン URL へのリンクを含めることができます。

---

## goto パラメータ

`goto=`*successful_authentication_URL* パラメータは、認証設定サービスの「ログイン成功時に返す URL」に定義された値を置き換えます。このパラメータは、認証が成功すると指定された URL にリンクします。`goto=logout_URL` パラメータも、ユーザーのログアウト時に指定された URL にリンクするのに使用できます。次に認証成功時に返す URL の例を示します。

```
http://server_name.domain_name:port/amserver/
UI/Login?goto=http://www.sun.com/homepage.html
```

次に `goto` ログアウト URL の例を示します。

```
http://server_name.domain_name:port/amserver/
UI/Logout?goto=http://www.sun.com/logout.html.
```

---

注 – Access Manager が認証成功時にどの URL にリダイレクトさせるかについては優先順位があります。リダイレクト URL とそれらの順番は認証方法に基づいているので、この順番および関連情報については、「認証タイプ」の節で詳しく説明します。

---

## gotoOnFail パラメータ

gotoOnFail=failed_authentication_URL パラメータは、認証設定サービスの「ログイン失敗時に返す URL」に定義された値を置き換えます。ユーザーが認証に失敗すると、指定された URL にリンクします。次に gotoOnFail URL の例を示します。

http:// *server_name.domain_name*:*port* /amserver/UI/Login?
gotoOnFail=http://www.sun.com/auth_fail.html

---

注 – Access Manager が認証失敗時にどの URL にリダイレクトさせるかについては優先順位があります。リダイレクト URL とそれらの順番は認証方法に基づいているので、この順番および関連情報については、「認証タイプ」の節で詳しく説明します。

---

## realm パラメータ

org=*realmName* パラメータを使用すると、指定されたレルムのユーザーとしてユーザーを認証することができます。

---

注 – 指定されたレルムのメンバーになっていないユーザーが、realm パラメータで認証を試みると、エラーメッセージを受け取ります。ただし、次の条件をすべて満たせば、Directory Server に動的にユーザープロファイルを作成できます。

- コア認証サービスの「ユーザープロファイル」属性に、「動的」または「ユーザーエイリアスを使用して動的に」が設定されている。
- ユーザーが、必要なモジュールに対する認証に成功している。
- ユーザーのプロファイルは、まだ Directory Server にない。

このパラメータから、レルムおよびそのロケールの設定に基づいて、正しいログインページが表示されます。このパラメータが設定されない場合は、デフォルトは最上位のレルムになります。たとえば、org URL は次のようになります。

http://server_name.domain_name:port/amserver/UI/Login?realm=sun

---

## org パラメータ

org=*orgName* パラメータを使用すると、指定された組織のユーザーとしてユーザーを認証することができます。

注 – 指定された組織のメンバーになっていないユーザーが、org パラメータで認証を試みると、エラーメッセージを受け取ります。ただし、次の条件をすべて満たせば、Directory Server に動的にユーザープロファイルを作成できます。

- コア認証サービスの「ユーザープロファイル」属性に、「動的」または「ユーザーエイリアスを使用して動的に」が設定されている。
- ユーザーが、必要なモジュールに対する認証に成功している。
- ユーザーのプロファイルは、まだ Directory Server にない。

このパラメータから、組織およびそのロケールの設定に基づいて、正しいログインページが表示されます。このパラメータが設定されない場合は、デフォルトは最上位の組織になります。たとえば、org URL は次のようになります。

```
http://server_name.domain_name:port/amserver/UI/Login?org=sun
```

## user パラメータ

user=*userName* パラメータは、ユーザーのプロファイルの「ユーザー認証設定」属性に設定されたモジュールに基づいて、認証を強制します。たとえば、あるユーザーのプロファイルを、証明書モジュールを使用して認証し、別のユーザーを LDAP モジュールを使用して認証するように設定できます。このパラメータを追加すると、ユーザーはユーザーの組織に設定された認証方法ではなく、ユーザー用に設定された認証プロセスに送られます。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?user=jsmith
```

## role パラメータ

role=roleName パラメータは、ユーザーを指定されたロール用に設定された認証プロセスに送ります。指定されたロールのメンバーになっていないユーザーが、このパラメータで認証を試みると、エラーメッセージを受け取ります。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?role=manager
```

## locale パラメータ

Access Manager は、認証プロセスとコンソール自身について、ローカライズされた (英語以外の言語に翻訳された) 画面を表示することができます。locale=*localeName* パラメータは、指定されたロケールをその他の定義されたロケールよりも優先させることができます。ログインのロケールは、次の順序で次の場所から設定を検索したあとに、クライアントに表示されます。

1. ログイン URL のロケールパラメータの値

   `locale=`*localeName* パラメータの値は、定義されたその他のすべてのロケールよりも優先されます。

2. ユーザーのプロファイルに定義されたロケール

   URL パラメータがない場合は、ロケールはユーザープロファイルの「ユーザー設定言語」属性に設定された値に基づいて表示されます。

3. HTTP ヘッダーに定義されたロケール

   このロケールは、Web ブラウザによって設定されます。

4. コア認証サービスに定義されたロケール

   これは、コア認証モジュールの「デフォルト認証ロケール」属性の値です。

5. プラットフォームサービスに定義されたロケール

   これは、プラットフォームサービスの「プラットフォームロケール」属性の値です。

オペレーティングシステムのロケール

この優先順位から導き出されるロケールは、ユーザーのセッショントークンに格納され、Access Manager が、ローカライズされた認証モジュールのロードだけに使用します。認証に成功すると、ユーザーのプロファイルの「ユーザー設定言語」属性に定義されたロケールが使用されます。ロケールが設定されていない場合は、認証に使用されたロケールが引き続き使用されます。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?locale=ja
```

---

注 – 画面のテキストおよびエラーメッセージのローカライズの方法については、Access Manager を参照してください。

---

## module パラメータ

`module=`*moduleName* パラメータを使用すると、指定した認証モジュールによって認証を行うことができます。どのモジュールでも指定できますが、まずユーザーが所属するレルムに登録し、コア認証モジュールでそのレルムの認証モジュールの 1 つとして選択する必要があります。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?module=Unix
```

---

注 – 認証モジュール名は、URL パラメータで使用する場合には大文字と小文字が区別されます。

---

## service パラメータ

service=*serviceName* パラメータを使用すると、サービスの設定された認証スキームによってユーザーを認証できます。認証設定サービスを使用して、異なるサービスに異なる認証スキームを設定できます。たとえば、オンラインの給料支払いアプリケーションにはより安全な証明書認証モジュールを使用した認証が必要になり、レルムの社員のディレクトリアプリケーションには LDAP 認証モジュールのみが必要になるなどです。認証スキームを、それらの各サービスに設定および指定できます。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?service=sv1
```

注 – 認証設定サービスを使用して、サービスに基づく認証のスキームを定義します。

## arg パラメータ

arg=*newsession* パラメータを使用して、ユーザーの現在のセッションを終了し、新しいセッションを開始します。認証サービスは、1 回の要求でユーザーの既存のセッショントークンを破棄し、新しいログインを実行します。このオプションは、通常匿名の認証モジュールで使用されます。ユーザーは、まず匿名セッションで認証を受けてから、登録リンクまたはログインリンクをヒットします。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?arg=newsession
```

## authlevel パラメータ

authlevel=*value* パラメータは、指定された認証レベル値以上の認証レベルのモジュールを呼び出すように認証サービスに指示します。各認証モジュールは、固定整数の認証レベルで定義されます。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?authlevel=1
```

注 – 認証レベルは、各モジュールの特定のプロファイルに設定されます。

## domain パラメータ

このパラメータを使用すると、指定されたドメインとして識別されるレルムにユーザーがログインできます。指定するドメインは、レルムのプロファイルの「ドメイン名」属性に定義された値に一致する必要があります。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?domain=sun.com
```

注 – 指定されたドメイン、つまりレルムのメンバーになっていないユーザーが、org パラメータで認証を試みると、エラーメッセージを受け取ります。ただし、次の条件をすべて満たせば、Directory Server に動的にユーザープロファイルを作成できます。

- コア認証サービスの「ユーザープロファイル」属性に、「動的」または「ユーザーエイリアスを使用して動的に」が設定されている。
- ユーザーが、必要なモジュールに対する認証に成功している。
- ユーザーのプロファイルは、まだ Directory Server にない。

## iPSPCookie パラメータ

iPSPCookie=yes パラメータを使用すると、ユーザーは持続 Cookie でログインできます。持続 Cookie とは、ブラウザウィンドウが閉じられたあとも存在し続ける Cookie のことです。このパラメータを使用するには、ユーザーがログインするレルムのコア認証モジュールで「持続 Cookie」が有効になっている必要があります。ユーザーが認証されブラウザを閉じると、ユーザーは新しいブラウザセッションでログインすることが可能であり、再認証する必要なくコンソールにダイレクトされます。これは、コアサービスに指定された「Cookie の最大持続時間」属性の値まで有効です。次に例を示します。

```
http://server_name.domain_name:port/amserver/UI/Login?org=example&iPSPCookie=yes
```

## IDTokenN パラメータ

このパラメータオプションを使用すると、ユーザーは URL または HTML 形式で認証資格を渡すことができます。IDTokenN=*value* パラメータを使用すると、ユーザーは認証サービスユーザーインタフェースにアクセスせずに認証を受けることができます。この処理は、ゼロページログインと呼ばれます。ゼロページログインは、1 つのログインページを使用する認証モジュールの場合にのみ機能します。IDToken0、IDToken1、...、IDTokenN の値は、認証モジュールのログインページのフィールドにマッピングされます。たとえば、LDAP 認証モジュールは、userID 情報に IDToken1 を、パスワード情報に IDToken2 を使用できます。この場合、LDAP モジュールの IDTokenN URL は次のようになります。

```
http://server_name.domain_name:port/amserver/UI/
Login?module=LDAP&IDToken1=userID&IDToken2=password
```

LDAP がデフォルトの認証モジュールである場合は、module=LDAP を省略できます。

匿名認証の場合は、ログイン URL パラメータは次のようになります。

```
http://server_name.domain_name:port/amserver/UI/Login?module=Anonymous&IDToken1=anonymousUserID
```

注 – 以前のリリースのトークン名 Login.Token0、Login.Token1、...、Login.TokenN は、現在はサポートされていますが今後のリリースではサポートされません。新しい IDTokenN パラメータを使用することをお勧めします。

# アカウントのロック

認証サービスには、定義された回数失敗すると、ユーザーが認証からロックアウトされる機能があります。この機能はデフォルトではオフになっていますが、Access Manager コンソールを使用して有効にできます。

注 – パスワードの間違いを検出できるモジュールのみがアカウントロック機能を利用できます。

コア認証サービスには、この機能を有効化およびカスタマイズするための次の属性 (ただし、これらに限定されない) が含まれています。

- 「ログイン失敗時のロックアウトモードを有効」は、アカウントロックを有効にします。
- 「ログイン失敗が右の回数に達したらロックアウト」は、ユーザーがロックアウトされるまでに認証を試みることができる回数を定義します。この回数は、ユーザー ID ごとにのみ有効であり、同一ユーザー ID が指定された回数だけ認証に失敗するとロックアウトされます。
- 「ログイン失敗回数を加算する間隔」は、分単位で定義された時間内にユーザーのログイン失敗の回数が「ログイン失敗が右の回数に達したらロックアウト」で定義された値に達した場合に、そのユーザーをロックアウトすることを意味します。
- 「ロックアウト通知の送信先電子メールアドレス」は、ユーザーロックアウト通知が送信される電子メールアドレスを指定します。
- 「ユーザーに警告を出すまでの失敗回数」は、認証の失敗回数がここで定義した値に達したら、警告メッセージをユーザーに表示することを意味します。ロックアウトの発生が迫っていることを知らせる警告をユーザーが受けたあとに、さらに実行できるログインの試みを、管理者が設定できます。
- 「ログイン失敗時のロックアウト持続時間」は、ロックアウト後、次に認証を再試行できるようになるまで、ユーザーが待たなければならない時間を分単位で定義します。
- 「ロックアウト属性名」は、物理ロックのためにユーザーのプロファイルのどの LDAP 属性を「非アクティブ」に設定するかを定義します。

- 「ロックアウト属性値」は、「ロックアウト属性名」に指定された LDAP 属性を「非アクティブ」または「アクティブ」のどちらに設定するかを定義します。

アカウントのロックアウトに関する電子メールの通知が、管理者に送信されます。アカウントロックのアクティビティーはログにも記録されます。

---

注 – Microsoft® Windows 2000 オペレーティングシステムでこの機能を使用する場合の特別な指示については、付録 A、「AMConfig.properties ファイル」の「Simple Mail Transfer Protocol (SMTP)」を参照してください。

---

Access Manager では、物理ロックとメモリーロックの 2 つのタイプのアカウントロックがサポートされます。次の節では、この 2 つについて説明します。

# 物理ロック

物理ロックは、Access Manager のデフォルトのロック動作です。このロックは、ユーザーのプロファイルの LDAP 属性の状態を非アクティブに変更することによって開始されます。「ロックアウト属性名」属性は、ロックの目的で使用する LDAP 属性を定義します。

---

注 – エイリアス化されたユーザーとは、LDAP プロファイルで「ユーザーエイリアスリスト」属性 (amUser.xml の `iplanet-am-user-alias-list`) を設定して既存の LDAP ユーザープロファイルにマッピングされるユーザーのことです。エイリアス化されたユーザーは、コア認証サービスの「エイリアス検索属性名」フィールドに `iplanet-am-user-alias-list` を追加することによって確認できます。つまり、エイリアス化されたユーザーがロックアウトされると、ユーザーがエイリアス化された実際の LDAP プロファイルがロックされます。これは、LDAP およびメンバーシップ以外の認証モジュールの物理ロックアウトにのみ関係します。

---

## メモリーのロック

メモリーロックは、「ログイン失敗時のロックアウト持続時間」属性の値を 0 よりも大きな値に変更すると有効になります。有効にすると、ユーザーのアカウントは指定された時間メモリー上でロックされます。指定された期間が過ぎると、このアカウントはロック解除されます。メモリーロック機能を使用する場合の考慮事項を次に示します。

- Access Manager が再起動されると、メモリー上でロックされたすべてのアカウントはロック解除されます。

- ユーザーのアカウントがメモリー上でロックされ、管理者がロックアウト持続時間を 0 に設定してアカウントロックメカニズムを物理ロックに変更すると、ユーザーのアカウントはメモリーでロック解除され、ロックカウントがリセットされます。
- メモリーのロックアウト後、LDAP およびメンバーシップ以外の認証モジュールを使用するときに、ユーザーが正しいパスワードでログインを試みると、「このレルムにユーザーのプロファイルがありません。」というエラーが返され、「ユーザーがアクティブではありません。」というエラーは返されません。

---

注 – ユーザーのプロファイルに「ログイン失敗時に返す URL」属性を設定すると、ロックアウト警告メッセージもアカウントがロックされたことを示すメッセージも表示されず、ユーザーは定義された URL にリダイレクトされます。

---

# 認証サービスのフェイルオーバー

プライマリサーバーにハードウェアまたはソフトウェア上の問題で障害が発生した場合、または一時的にシャットダウンした場合には、認証サービスのフェイルオーバーにより、認証要求はセカンダリサーバーへ自動的にリダイレクトされます。

認証コンテキストは認証サービスが使用可能な Access Manager のインスタンス上でまず作成されなければなりません。Access Manager のこのインスタンスが使用できない場合は、認証フェイルオーバーメカニズムにより Access Manager の別のインスタンス上に認証コンテキストが作成されます。認証コンテキストは次のような順序でサーバーが使用可能かどうか確認します。

1. 認証サービス URL を AutoContext API に送ります。次に例を示します。

   ```
   AuthContext(orgName, url)
   ```

   この API を使う場合は、URL で参照されたサーバーのみを使用します。この場合、そのサーバー上で認証サービスが使用可能であっても、フェイルオーバーは起きません。
2. 認証コンテキストが `AMConfig.properties` ファイルの `com.iplanet.am.server*` 属性に定義されたサーバーをチェックします。
3. 手順 2 で失敗すると、認証コンテキストはネーミングサービスが利用可能なサーバーからのプラットフォームリストを照会します。ディレクトリサーバーの 1 つのインスタンスを共有する複数のインスタンスが、(主にフェイルオーバーを目的として) Access Manager 上にインストールされたときに、このプラットフォームリストが自動的に作成されます。

たとえば、プラットフォームリストに Server1、Server2、および Server3 の URL が含まれていると、認証コンテキストは Server1、Server2、および Server3 のいずれかで認証が成功するまでループします。

プラットフォームリストは、ネーミングサービスの有無に依存しているので、常に同一のサーバーから得られるわけではありません。さらに、ネーミングサービスのフェイルオーバーが最初に起こります。複数ネーミングサービス URL は AMConfing.properties の com.iplanet.am.naming.url プロパティーに定義されます。利用可能な最初のネーミングサービス URL は、認証フェイルオーバーが発生する (プラットフォームサーバーリスト中の) サーバーのリストを持つサーバーを特定するのに使われます。

# 完全修飾ドメイン名のマッピング

完全修飾ドメイン名 (FQDN) のマッピングは、ユーザーが誤った URL を入力した (保護されたリソースにアクセスするために部分的なホスト名または IP アドレスを指定したなど) 場合に認証サービスが訂正を行うことができるようにします。FQDN のマッピングは、AMConfig.properties ファイルで com.sun.identity.server.fqdnMap 属性を変更することによって可能になります。このプロパティーは次の形式で指定します。

```
com.sun.identity.server.fqdnMap[invalid-name ]=valid-name
```

*invalid-name* の値はユーザーが入力する可能性がある無効な FQDN ホスト名であり、*valid-name* はフィルタがユーザーをリダイレクトする実際のホスト名です。定められた要件に準拠するかぎり、コード例 1-1 に示すようにいくつでもマッピングを指定できます。このプロパティーを設定しない場合は、ユーザーは com.iplanet.am.server.host=*server_name* プロパティーに設定されたデフォルトのサーバー名に送信されます。このプロパティーも AMConfig.properties ファイルにあります。

例 4–1 AMConfig.properties の FQDN マッピング属性

```
com.sun.identity.server.fqdnMap[isserver]=isserver.mydomain.com
com.sun.identity.server.fqdnMap[isserver.mydomain]=isserver.mydomain.com
com.sun.identity.server.fqdnMap[
            IP address]=isserver.mydomain.com
```

## FQDN のマッピングの使用例

このプロパティーは、サーバーにホストされたアプリケーションが複数のホスト名でアクセス可能な場合に、複数のホスト名のマッピングを作成するために使用できます。このプロパティーを使用して、特定の URL について訂正を行わないように Access Manager を設定することもできます。たとえば、IP アドレスを使用してアプリケーションにアクセスするユーザーにリダイレクトが必要ない場合は、この機能は次のようなマップエントリを指定して実現できます。

`com.sun.identity.server.fqdnMap[`*IP address* `]=`*IP address*

---

注 – 複数のマッピングを定義する場合は、無効な FQDN 名で値が重複しないようにします。そのようにしないと、アプリケーションにアクセスできなくなる場合があります。

---

# 持続 Cookie

持続 Cookie とは、Web ブラウザを閉じたあとも存在する Cookie のことであり、ユーザーが再認証なしに新しいブラウザセッションにログインすることを可能にします。Cookie の名前は、`AMConfig.properties` の `com.iplanet.am.pcookie.name` プロパティーに定義されます。デフォルト値は、`DProPCookie` です。Cookie の値は、3DES で暗号化された文字列であり、この文字列には、ユーザー DN、レルム名、認証モジュール名、最大セッション時間、アイドル時間、およびキャッシュ時間が含まれます。

## ▼ 持続 Cookie を有効にする

1. コア認証モジュールで「持続 `Cookie` モード」をオンにします。
2. コア認証モジュールで「`Cookie` の最大持続時間」属性の時間値を設定します。
3. ユーザーインタフェースのログイン **URL** に `yes` の値で **iPSPCookie** パラメータを付加します。

   ユーザーがこの URL を使用して認証を受けると、ブラウザを閉じた場合、新しいブラウザウィンドウを開くことができ、再認証なしにコンソールにリダイレクトされます。手順 2 で定義された時間が経過するまでこのようになります。

   持続 Cookie モードは、次の認証 SPI メソッドを使用してオンにできます。

   ```
   AMLoginModule.setPersistentCookieOn()
   ```

# 旧バージョンモードにおけるマルチ LDAP 認証モジュール設定

フェイルオーバーの形式として、あるいは、Access Manager コンソールで値フィールドが 1 つだけ提供されている場合に、属性に複数の値を設定するために、管理者は 1 つのレルムに複数の LDAP 認証モジュール設定を定義できます。これら追加の設定はコンソールに表示されませんが、要求を行っているユーザーの承認が初期検索で見つからない場合に、主設定とともに機能します。たとえば、1 つのレルムで 2 つの異なるドメインでの認証に LDAP サーバーを介した検索を定義したり、あるいは 1 つのドメインに複数のユーザーネーミング属性を設定できます。後者の場合、コンソールにはテキストフィールドが 1 つのみあり、第 1 の検索基準でユーザーが見つからない場合は、LDAP モジュールは第 2 の検索範囲で検索します。追加の LDAP 構成を設定する手順を次に示します。

## ▼ 追加の LDAP 構成を設定する

1 **2 番目 (または 3 番目) の LDAP 認証設定に必要な属性および新しい値の完全なセットを含めた XML ファイルを作成します。**

利用可能な属性は、`/etc/opt/SUNWam/config/xml` にある `amAuthLDAP.xml` で参照できます。この手順で作成された XML ファイルは、`amAuthLDAP.xml` とは異なり、`amadmin.dtd` の構造に基づいています。このファイルには、任意のまたはすべての属性を定義できます。コード例 1-2 は、LDAP 認証設定に利用できるすべての属性の値が含まれる副設定ファイルの例です。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--
  Copyright (c) 2002 Sun Microsystems, Inc. All rights reserved.
  Use is subject to license terms.
-->
<!DOCTYPE Requests
    PUBLIC "-//iPlanet//Sun ONE Access Manager 6.0 Admin CLI DTD//EN"
    "jar://com/iplanet/am/admin/cli/amAdmin.dtd"
>
<!--
  Before adding subConfiguration load the schema with
GlobalConfiguration defined and replace corresponding
 serviceName and subConfigID in this sample file OR load
 serviceConfigurationRequests.xml before loading this sample
-->
<Requests>
<realmRequests DN="dc=iplanet,dc=com">
    <AddSubConfiguration subConfigName = "ssc"
        subConfigId = "serverconfig"
```

```
priority = "0" serviceName="iPlanetAMAuthLDAPService">

      <AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-server"/>
    <Value>vbrao.red.iplanet.com:389</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-base-dn"/>
    <Value>dc=iplanet,dc=com</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="planet-am-auth-ldap-bind-dn"/>
    <Value>cn=amldapuser,ou=DSAME Users,dc=iplanet,dc=com</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-bind-passwd"/>
    <Value>
          plain text password</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-user-naming-attribute"/>
    <Value>uid</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-user-search-attributes"/>
    <Value>uid</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-search-scope"/>
    <Value>SUBTREE</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-ssl-enabled"/>
    <Value>false</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-return-user-dn"/>
    <Value>true</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-auth-level"/>
    <Value>0</Value>
</AttributeValuePair>
<AttributeValuePair>
    <Attribute name="iplanet-am-auth-ldap-server-check"/>
    <Value>15</Value>
</AttributeValuePair>
```

```
    </AddSubConfiguration>

</realmRequests>
</Requests>
```

2 手順 1 で作成した **XML** ファイルで **iplanet-am-auth-ldap-bind-passwd** の値としてプレーンテキストのパスワードをコピーします。

この属性の値は、コード例に太字で示されています。

3 `amadmin` コマンド行ツールを使用して、**XML** ファイルをロードします。

```
./amadmin -u amadmin -w administrator_password -v -t name_of_XML_file.
```

この 2 番目の LDAP 設定は、コンソールを使用して表示も変更もできません。

---

ヒント – 複数 LDAP 設定に利用できるサンプルが用意されています。`/AccessManager-base /SUNWam/samples/admin/cli/bulk-ops/` にある `serviceAddMultipleLDAPConfigurationRequests.xml` コマンド行テンプレートを参照してください。手順については、`/AccesManager-base/SUNWam/samples/admin/cli/` にある `Readme.html` を参照してください。

---

# セッションのアップグレード

認証サービスでは、1 つのレルムに対して同一ユーザーが実行した 2 回目の成功した認証に基づいて有効なセッショントークンのアップグレードを行うことができます。有効なセッショントークンを持つユーザーが、現在のレルムによってセキュリティー保護されているリソースに対して認証を試み、この 2 回目の認証が成功すると、セッションは新しい認証に基づく新しいプロパティーで更新されます。認証が失敗すると、ユーザーの現在のセッションがアップグレードなしに戻されます。有効なセッショントークンを持つユーザーが、別のレルムによってセキュリティー保護されているリソースに対して認証を試みると、そのユーザーは新しいレルムの認証を受けるどうかを尋ねるメッセージを受け取ります。この時点では、ユーザーは、現在のセッションを維持するか、または新しいレルムの認証を受けることができます。認証が成功すると、古いセッションは破棄され、新しいセッションが作成されます。

セッションのアップグレード時に、タイムアウトになると、これまで設定されていた「成功時に返す URL」へのリダイレクトが発生します。タイムアウト値は、次の条件に基づいて判断されます。

- 各モジュールに設定されたページタイムアウト値 (デフォルトは 1 分)
- AMConfig.properties の com.iplanet.am.invalidMaxSessionTime プロパティー (デフォルトは 10 分)
- iplanet-am-max-session-time (デフォルトは 120 分)

com.iplanet.am.invalidMaxSessionTimeout と iplanet-am-max-session-time の値は、ページタイムアウト値よりも大きい必要があり、そうでない場合はセッションアップグレード時に有効なセッション情報が失われ、これまで設定されていた「成功時に返す URL」 へのリダイレクトは失敗します。

# 検証プラグインインタフェース

管理者は、レルムに適したユーザー名またはパスワード検証ロジックを作成し、そのロジックを認証サービスにプラグインできます。この機能は、LDAP およびメンバーシップの認証モジュールのみでサポートされます。ユーザーを認証したりパスワードを変更したりする前に、Access Manager はこのプラグインを呼び出します。検証が成功すると認証が継続され、失敗すると認証失敗時に返すページがスローされます。プラグインは、サービス管理 SDK の一部である com.iplanet.am.sdk.AMUserPasswordValidation クラスを拡張します。SDK についての情報は、Access Manager Javadocs の com.iplanet.am.sdk パッケージを参照してください。

## ▼ 検証プラグインを作成して設定する

1. 新しいプラグインクラスは、com.iplanet.am.sdk.AMUserPasswordValidation クラスを拡張し、validateUserID() および validatePassword() メソッドを実装します。検証が失敗した場合は、AMException がスローされます。

2. プラグインクラスをコンパイルし、.class ファイルを必要な場所に配置します。実行時に **Access Manager** がプラグインにアクセスできるように、クラスパスを更新します。

3. 最上位の管理者として **Access Manager** コンソールにログインします。「設定」タブをクリックし、管理サービスの属性にアクセスします。「ユーザー ID とパスワードの検証プラグインクラス」フィールドにパッケージ名を含むプラグインクラスの名前を入力します。

4. ログアウトし、ログインし直します。

# JAAS 共有状態

JAAS 共有状態を有効にすることで、ユーザー ID とパスワードの両方を認証モジュール間で共有できます。各認証モジュールに対して次のオプションを定義します。

- レルム (または、組織)
- ユーザー
- サービス
- 役割

失敗した場合、モジュールは必要な資格を要求します。認証の失敗後、モジュールが停止するか、ログアウト共有状態がクリアされます。

## JAAS 共有状態の有効化

JAAS 共有状態を設定するには、次のようにします。

- `iplanet-am-auth-shared-state-enabled` オプションを使用します。
- 共有状態オプションの使用法を次に示します。
  `iplanet-am-auth-shared-state-enabled=true`
- このオプションのデフォルトは、true です。
- この変数は、認証連鎖設定の「オプション」列で指定されます。

失敗すると、認証モジュールは、JAAS の仕様で提案される `tryFirstPass` オプションの動作に従って必要な資格を要求します。

### JAAS 共有状態ストアオプション

「JAAS 共有状態ストア」オプションを設定するには、次のようにします。

- `iplanet-amauth-store-shared-state-enabled` オプションを使用します。
- 「共有状態」オプションの使用法を次に示します。
  `iplanet-am-auth-store-shared-state-enabled=true`
- このオプションのデフォルトは、false です。
- この変数は、認証連鎖設定の「オプション」列で指定されます。

コミット、中断、またはログアウト後に、共有状態はクリアされます。

◆ ◆ ◆ 第 5 章

# ポリシーの管理

この章では、Sun Java™ System Access Manager のポリシー管理機能について説明します。Access Manager のポリシー管理機能では、最上位レベル管理者または最上位レベルポリシー管理者が、すべてのレルムで使用できる特定サービスのポリシーの表示、作成、削除、修正を行うことができるようになります。レルムまたはサブレルムの管理者あるいはポリシー管理者が、レルムレベルでポリシーを表示、作成、削除、修正することもできます。

この章の内容は次のとおりです。



## 概要

ポリシーは、組織の保護されたリソースに対するアクセス権限を指定するルールを定義します。ビジネスには、保護、管理、監視しなければならない、リソース、アプリケーション、およびサービスがあります。ポリシーはこれらのリソースに対するアクセス権と使用を管理し、ユーザーがあるリソースに対して、いつ、どのようにアクションを実行できるかを定義します。ポリシーによって、特定の主体のリソースが定義されます。

注 – 主体には、個人、企業、ロール、グループなど、アイデンティティーを持つことができるすべてのものが該当します。詳細については、『Java™ 2 Platform Standard Edition Javadoc (http://java.sun.com/j2se/1.4.2/docs/api/java/security/Principal.html)』を参照してください。

1 個のポリシーでは、二者択一または任意設定の決定を定義できます。二者択一の決定とは、「はい」/「いいえ」、「真」/「偽」または「許可する」/「許可しない」の形式で表されるものを指します。二者択一でない決定とは、属性の値で表されるものを指します。たとえば、メールサービスは、ユーザーごとの最大保存容量の値セットを持つ `mailboxQuota` 属性を含んでいます。一般的に、ポリシーは主体が、どのような条件でどのリソースに何をできるかを定義するように設定されています。

# ポリシー管理機能

ポリシー管理機能には、ポリシーの作成および管理を行うポリシーサービスが備わっています。ポリシーサービスにより、管理者は Access Manager の配備の中でリソースを保護するために、アクセス権を定義、変更、付与、無効化、および削除することができます。通常、ポリシーサービスには、データストアと、ポリシーの作成、管理、評価ができるインタフェースライブラリ、およびポリシーエンフォーサ (ポリシーエージェント) が含まれています。デフォルトでは、Access Manager は Sun Java Enterprise System Directory Server をデータ保存に使い、ポリシーの評価とポリシーサービスのカスタマイズのために Java と C の API を提供します。詳細は、『Sun Java System Access Manager 7.1 Developer's Guide』を参照してください。また、管理者は Access Manager コンソールを使用してポリシー管理を行うこともできます。Access Manager は、ダウンロード可能なポリシーエージェントを使用してポリシーを適用する、URL ポリシーエージェントサービスを提供します。

## URL ポリシーエージェントサービス

Access Manager をインストールするときに、HTTP URL を保護するポリシーを定義するための URL ポリシーエージェントサービスが実装されます。このサービスにより、管理者はポリシーエンフォーサ、つまりポリシーエージェントにより、ポリシーの作成および管理を行えます。

### ポリシーエージェント

ポリシーエージェントは企業のリソースが保存されているサーバーへのポリシー適用ポイント (Policy Enforcement Point、PEP) です。ポリシーエージェントは Access Manager とは別に Web サーバー上にインストールされており、ユーザーが保護され

た Web サーバー 上のリソースを要求すると、追加の認証ステップとして働きます。この認証は、リソースが実行するあらゆるユーザー認証要求に追加されます。ポリシーエージェントは Web サーバーを保護し、一方、認証プラグインはリソースを保護します。

たとえば、リモートインストールされた Access Manager で保護されている人事部の Web サーバーにエージェントがインストールされているとします。このエージェントにより、適切なポリシーを持っていない担当者には機密の給与情報またはその他の秘密情報は表示されません。このポリシーは Access Manager の管理者が定義し、Access Manager の配備に保存され、リモート Web サーバーのコンテンツにユーザーがアクセスするのをポリシーエージェントが許可または拒否するのに使われます。

最新の Access Manager 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のインストールおよび管理の詳細については、『Sun Java System Access Manager Policy Agent 2.2 User's Guide』を参照してください。

---

注 – ポリシーの評価に特定の順序はありませんが、評価の途中であるアクションの値が「許可しない」となったときには、ポリシー設定サービスにより「拒否決定で評価を続行」属性が有効になっていないかぎり、以降のポリシーの評価は中止されます。

---

Access Manager ポリシーエージェントが決定を適用するのは Web URL (`http://...` または `https//...`) だけですが、Java と C の Policy Evaluation API を使ってエージェントをプログラミングすれば、ほかのリソースにもポリシーを適用可能です。

この場合、追加作業として、ポリシー設定サービスの「リソースコンパレータ」属性をデフォルトから次のように変更する必要があります。

```
serviceType=Name_of_LDAPService
|class=com.sun.identity.policy.plugins.SuffixResourceName|wildcard=*

|delimiter=,|caseSensitive=false
```

または、`LDAPResourceName` などの実装により、`com.sun.identity.policy.interfaces.ResourceName` を実装して、その上で「リソースコンパレータ」を適切に設定するという方法もあります。

## ポリシーエージェントプロセス

Web ブラウザがポリシーエージェントによって保護されたサーバー上の URL を要求すると、保護された Web リソースに対するプロセスが始まります。このサーバーにインストールされたポリシーエージェントはこの要求を傍受し、既存の認証資格をチェックします ( セッショントークン)。

エージェントが要求を傍受し、既存のセッショントークンを検証したら、プロセスは次のように続きます。

1. セッショントークンが有効であれば、ユーザーのアクセスは許可または拒否されます。ユーザーのトークンが無効であれば、次の手順にあるようにユーザーを認証サービスにリダイレクトします。

   既存のセッショントークンが存在しない要求をエージェントが傍受した場合は、そのリソースが異なる認証方法で保護されているときでも、エージェントはユーザーをログインページにリダイレクトします。
2. ユーザーの資格が適切に認証されると、エージェントは Access Manager の内部サービスの接続に使う URL を定義するネーミングサービスに要求を出します。
3. ポリシーを適用しないリソースリストがエージェントに設定されている場合、リソースがそのリストに一致する場合には、アクセスが許可されます。
4. ネーミングサービスはポリシーサービス、セッションサービス、およびログサービスのロケータを返します。
5. エージェントはユーザーに適用されるポリシー決定を取得するためにポリシーサービスに要求を送信します。
6. アクセスされるリソースに関してのポリシー決定に基づいて、ユーザーのアクセスが許可または拒否されます。ポリシー決定へのアドバイスが異なる認証レベルまたは認証メカニズムを示している場合、エージェントはすべての基準が検証されるまで要求を認証サービスにリダイレクトします。

# ポリシータイプ

Access Manager を使用して設定できるポリシーは、次の 2 種類です。

- 104 ページの「標準ポリシー」
- 110 ページの「参照ポリシー」

## 標準ポリシー

Access Manager では、アクセス権を定義するポリシーを標準ポリシーと呼びます。標準ポリシーは、ルール、対象、条件、および応答プロバイダから構成されます。

### ルール

ルールは 1 つのサービスタイプ、1 つ以上のアクション、および 1 つの値からなります。基本的に、ルールがポリシーを定義します。

- サービスタイプは、保護されるリソースのタイプを定義します。

- アクションはリソースに対して実行される操作の名前です。Web サーバーアクションの例としては POST または GET があります。たとえば、人事サービスに対して自宅電話番号を変更するアクションを許可できます。
- 値は各アクションの可否を示します。たとえば、allow (許可する) または deny (許可しない) です。

---

注 – 一部のサービスに対しては、リソースなしでアクションを定義することも可能です。

---

## 対象

対象はポリシーが影響を与えるユーザーまたはユーザーの集合 (たとえば、グループ、または特定のロールを持つ複数のユーザー) を定義します。対象の一般則は、ユーザーがポリシー中の少なくとも 1 つの対象のメンバーである場合にのみポリシーが適用される、というものです。デフォルトの対象は次のとおりです。

| | |
|---|---|
| Access Manager アイデンティティー対象 | この対象は、「レルム対象」タブで作成して管理しているアイデンティティーを対象のメンバーとして追加できることを示します。 |
| 認証済みユーザー | この対象タイプは、有効な SSOToken を持つ任意のユーザーがこの対象のメンバーであることを示します。<br><br>すべての認証済みユーザーは、ポリシーが定義されている組織とは別のレルムに認証しても、この対象のメンバーになります。リソース所有者が、別の組織のユーザー用に管理されているリソースにアクセスできるようにする場合は、これが便利です。特定組織のメンバーに保護されているリソースに対するアクセスを制限する場合は、組織対象を使用してください。 |
| Web サービスクライアント | この対象タイプは、SSOToken に含まれる主体の DN がこの対象の選択された値のいずれかに一致する場合、SSOToken が特定する Web サービスクライアント (WSC) がこの対象のメンバーであることを示します。有効な値は、信頼できる WSC の証明書に対応する、ローカル JKS キーストア内の信頼できる証明書の DN です。この対象は、Liberty Web サービス |

フレームワークに依存し、Liberty サービスプロバイダが WSC を承認するためにのみ使用する必要があります。

この対象をポリシーに追加する前に、キーストアが作成されていることを確認します。キーストアの設定に関する説明は、次の場所にあります。

*AccessManager-base*
`/SUNWam/samples/saml/xmlsig/keytool.html`

さらに次の対象は、レルムのポリシー設定サービスで選択することによって使用可能になります。

| | |
|---|---|
| Access Manager ロール | この対象タイプは、ある Access Manager ロールの任意のメンバーがこの対象のメンバーであることを示します。Access Manager ロールは、旧バージョンモードで動作している Access Manager と 6.3 ベースのコンソールを使用して作成されます。これらのロールは Access Manager が規定するオブジェクトクラスを持ちます。Access Manager のロールには、ホスティングする Access Manager のポリシーサービス経由でのみアクセスできます。 |
| LDAP グループ | この対象タイプは、ある LDAP グループの任意のメンバーがこの対象のメンバーであることを示します。 |
| LDAP ロール | この対象タイプは、ある LDAP ロールの任意のメンバーがこの対象のメンバーであることを示します。LDAP ロールは、Directory Server ロール機能を使用するロール定義です。LDAP ロールは、Directory Server ロール定義が規定するオブジェクトクラスを持ちます。ポリシー設定サービスで LDAP ロール検索フィルタを変更して、検索範囲を絞り込みパフォーマンスを向上させることができます。 |
| LDAP ユーザー | この対象タイプは、任意の LDAP ユーザーがこの対象のメンバーであることを示します。 |
| 組織 | この対象タイプは、あるレルムの任意のメンバーがこの対象のメンバーであることを示します。 |

## Access Manager ロールと LDAP ロールの比較

Access Manager ロールは Access Manager を使用して作成します。これらのロールは Access Manager が規定するオブジェクトクラスを持ちます。LDAP ロールは、Directory Server ロール機能を使用するロール定義です。LDAP ロールは、Directory Server ロール定義が規定するオブジェクトクラスを持ちます。すべての Access

Manager ロールは Directory Server のロールとして使用できます。しかし、すべての Directory Server ロールが必ずしも Access Manager ロールというわけではありません。LDAP ロールは、131 ページの「ポリシー設定サービス」を設定することにより、既存のディレクトリから利用できます。Access Manager のロールには、ホスティングする Access Manager のポリシーサービス経由でのみアクセスできます。ポリシー設定サービスで LDAP ロール検索フィルタを変更して、検索範囲を絞り込みパフォーマンスを向上させることができます。

## 入れ子ロール

入れ子ロールはポリシー定義の対象の LDAP ロールとして正しく評価できます。

## 条件

条件によって、ポリシーに制約を定義できます。たとえば、給与アプリケーション用のポリシーを定義する場合、アプリケーションへのアクセスを特定の時間帯だけに制限するようにアクションに対して条件を定義することができます。また、所定の IP アドレスまたは企業のイントラネットからの要求に対してのみアクションを許可するように条件を定義することもできます。

条件は、同じドメインの別の URI で別のポリシーを設定するために、補助的に使用されることもあります。たとえば、`http://org.example.com/hr/*jsp` は `org.example.net` ドメインより午前 9 時 ～ 午後 5 時の間だけアクセスできるようにします。これは IP 条件と時間条件を使用して実現します。またルールのリソースを `http://org.example.com/hr/*.jsp` に指定することで、ポリシーは `http://org.example.com/hr` 以下、サブディレクトリ内を含むすべての JSP に適用されるようになります。

---

注 – 参照、ルール、リソース、対象、条件、アクション、値の各用語は、`policy.dtd` 内の *Referral*、*Rule*、*ResourceName*、*Subject*、*Condition*、*Attribute*、*Value* の各要素に対応しています。

---

追加できるデフォルトの条件は、次のとおりです。

## アクティブなセッション時間

ユーザーセッションのデータに基づいて条件を設定します。変更できるフィールドは、次のとおりです。

最大セッション時間　セッションを開始してから、ポリシーがユーザーセッションに適用され続ける時間の最大値を指定します。

セッションを終了　選択すると、「最大セッション時間」フィールドで定義した許可される最大値をセッション時間が超えた場合に、ユー

ザーセッションを終了します。

### 認証連鎖

このポリシーは、指定したレルムの認証連鎖にユーザーが正しく認証された場合に適用されます。レルムを指定しなかった場合は、認証連鎖でどのレルムに対する認証も条件を満たします。

### 認証レベル(大きいか等しい)

ユーザーの認証レベルが条件に設定された認証レベル以上である場合にポリシーが適用されます。この属性は、指定したレルム内での認証の信用レベルを示しています。

### 認証レベル(小さいか等しい)

ユーザーの認証レベルが条件に設定された認証レベル以下である場合にポリシーが適用されます。この属性は、指定したレルム内での認証の信用レベルを示しています。

### 認証モジュールインスタンス

このポリシーは、指定したレルムの認証モジュールにユーザーが正しく認証された場合に適用されます。レルムを指定しなかった場合、認証モジュールでどのレルムに対する認証も条件を満たします。

### 現在のセッションプロパティー

ユーザーの Access Manager セッションに設定されたプロパティーの値に基づいて要求にポリシーを適用できるかどうかを決定します。ポリシーの評価中に条件が「真」を返すのは、ユーザーのセッションに条件で定義されたすべてのプロパティー値が存在する場合だけです。条件の中で複数の値を使用して定義されたプロパティーについては、トークンのプロパティーの値が少なくとも 1 つあれば十分です。

### IP アドレス/DNS 名

IP アドレスの範囲に基づいて条件を設定します。定義できるフィールドは、次のとおりです。

| | |
|---|---|
| 「IP アドレス 開始 / 終了」 | IP アドレスの範囲を指定します。 |
| DNS 名 | DNS 名を指定します。このフィールドには、完全修飾ホスト名または次の形式の文字列を指定できます。<br><br>*domainname* |

**.domainname*

### LDAP フィルタ条件

このポリシーは、定義された LDAP フィルタがポリシー設定サービス内で指定された LDAP ディレクトリでユーザーエントリを検索するときに適用されます。このポリシーが適用されるのは、ポリシーが定義されたレルム内のみです。

### レルム認証

このポリシーは、指定したレルムにユーザーが認証された場合に適用されます。

### 時間 (曜日、日付、時刻、およびタイムゾーン)

時間の制約に基づいて条件を設定します。フィールドは次のとおりです。

| | |
|---|---|
| 開始日付 / 終了日付 | 日付の範囲を指定します。 |
| 時刻 | 1 日での時間の範囲を指定します。 |
| 曜日 | 曜日を指定します。 |
| タイムゾーン | タイムゾーンを標準またはカスタムで指定します。カスタムのタイムゾーンとして指定できるのは、Java で認識されるタイムゾーン ID だけです (PST など)。値を指定しない場合は、デフォルト値は Access Manager JVM に設定されたタイムゾーンになります。 |

## 応答プロバイダ

応答プロバイダは、ポリシーベースの応答属性を提供するプラグインです。応答プロバイダ属性は、ポリシー決定とともに PEP に送信されます。Access Manager には、`IDResponseProvider` が 1 つだけ実装されています。このバージョンの Access Manager では、カスタム応答プロバイダはサポートされていません。エージェントの PEP は通常、これらの応答属性をヘッダーとしてアプリケーションに渡します。アプリケーションは通常、これらの属性を使用して、ポータルページなどのアプリケーションページをポリシーに基づいて設定します。

## ポリシーアドバイス

条件で決定したようにポリシーを適用できない場合は、ポリシーを要求に適用できなかった理由を示すアドバイスメッセージを条件によって作成できます。このアドバイスメッセージは、ポリシー決定でポリシー適用ポイントに伝わります。ポリシー適用ポイントでは、このアドバイスを取得し、認証メカニズムにユーザーを戻してより高いレベルに認証するなど、適切なアクションを実行しようとします。ア

ドバイスの適切なアクションを実行したあとでポリシーが適用可能になると、ユーザーはより高いレベルの認証を要求され、リソースにアクセスできるようになることがあります。

詳細については、次のクラスを参照してください。

```
com.sun.identity.policy.ConditionDecision.getAdvices()
```

条件が満たされない場合、アドバイスを提供するのは `AuthLevelCondiiton` と `AuthSchemeCondition` のみです。

`AuthLevelCondition` アドバイスは、次のキーに関連します。

```
com.sun.identity.policy.plugin.AuthLevelCondition.AUTH_LEVEL_CONDITION_ADVICE
```

`AuthSchemeCondition` アドバイスは、次のキーに関連します。

```
com.sun.identity.policy.plugin.AuthLevelCondition.AUTH_SCHEME_CONDITION_ADVICE
```

カスタム条件でもアドバイスを作成できます。ただし、Access Manager ポリシーエージェントは、認証レベルアドバイスと認証方式アドバイスのみに応答します。カスタムエージェントを作成してより多くのアドバイスを理解させて応答させたり、既存の Access Manager エージェントを拡張してより多くのアドバイスを理解させて応答させたりすることができます。詳細は、『Sun Java System Access Manager Policy Agent 2.2 User's Guide』を参照してください。

# 参照ポリシー

管理者は、あるレルムのポリシーの定義と決定を別のレルムに委任することが必要になる場合があります。または、あるリソースに対するポリシー決定を別のポリシー製品に委任することもできます。参照ポリシーは、ポリシーの作成と評価の両方に対するポリシーの委任を管理します。1 つ以上のルールと、1 つ以上の参照で構成されます。

ポリシー設定サービスには、組織エイリアス参照というグローバル属性が含まれます。この属性を使用すると、最上位のレルムまたは親レルムから参照ポリシーを作成しなくても、サブレルムにポリシーを作成できます。作成できるポリシーは、完全修飾ホスト名が、レルムまたはレルムの DNS エイリアスと一致する HTTP または HTTPS リソースを保護するポリシーのみです。デフォルトでは、この属性は「No」に設定されています。

## ルール

ルールは、ポリシーの定義と評価が参照されるリソースを定義します。

### 参照

参照は、ポリシーの評価をどの組織に対して参照するかを定義します。デフォルトでは、2 種類の参照があります。ピアレルムとサブレルムです。それぞれ、同じレベルのレルム、下位レベルのレルムを表します。詳細は、122 ページの「ピアレルムおよびサブレルムのポリシーの作成」を参照してください。

---

注 – 参照先のレルムでは、そのレルムをすでに参照済みのリソース (またはサブリソース) のポリシーのみを定義または評価できます。ただし、この制約は最上位のレルムには適用されません。

---

## ポリシー DTD

作成して設定したポリシーは、Directory Server に XML 形式で保存されます。Directory Server では、XML でエンコードされたデータは 1 か所に保管されます。ポリシーは、`amAdmin.dtd` (またはコンソール) を使って定義され設定されますが、実際には `policy.dtd` に基づく XML として、Directory Server に保存されます。`policy.dtd` は、`amAdmin.dtd` から抽出されたポリシー要素タグ (ポリシー作成タグを除く) を含んでいます。したがって、ポリシーサービスは Directory Server からポリシーをロードすると、`policy.dtd` に基づいて XML をパースします。ポリシーをコマンド行から作成するときには、`amAdmin.dtd` のみが使われます。この節では、`policy.dtd` の構造について説明します。`policy.dtd` は次の場所にあります。

```
AccessManager-base/SUNWam/dtd (Solairs)
AccessManager-base/identity/dtd (Linux)
AccessManager-base/identity/dtd (HP-UX)
AccessManager-base\identity\dtd (Windows)
```

---

注 – 本章ではこれ以降、ディレクトリ情報は Solaris についてのみ記します。Linux、HP-UX、および Windows のディレクトリ構造は異なっていることに注意してください。

---

### Policy 要素

*Policy* はアクセス権、つまりポリシーのルールと、ルールの適用先、つまり対象を定義するルート要素です。また、ポリシーが参照 (委任) ポリシーかどうか、およびポリシーになんらかの制限 (または条件) があるかどうかも定義します。この要素には、*Rule*、*Conditions*、*Subjects*、*Referrals*、*response providers* というサブ要素のうち 1 つ以上が含まれることがあります。必須の XML 属性は `name` で、これはポリシーの名

前を指定します。`referralPolicy` 属性は、ポリシーが参照ポリシーであるかどうかを識別します。指定がなければ標準ポリシーとして扱われます。オプションの XML 属性には *name* と *description* があります。

---

注 – ポリシーに *referral* タグを付けると、対象と条件はポリシー評価の際、無視されます。逆に、ポリシーに *normal* タグを付けると、Referrals は無視されます。

---

# Rule 要素

*Rule* 要素では、ポリシーの詳細を定義します。*ServiceName*、*ResourceName* 、*AttributeValuePair* という 3 つのサブ要素を持つことができます。この要素は、リソース名やそこで実行されるアクションだけではなく、ポリシーが作成されたサービスやアプリケーションのタイプを定義します。アクションを持たないルールも定義できます。たとえば、参照ポリシールールにはアクションはありません。

---

注 – *ResourceName* 要素を含まないポリシーの定義も可能です。

---

## ServiceName 要素

*ServiceName* 要素は、ポリシーを適用するサービスの名前を定義します。この要素は、サービスのタイプを表しています。ほかの要素は含みません。値は、そのサービスの XML ファイルで (`sms.dtd` に基づいて) 定義されているとおりです。
*ServiceName* 要素の XML サービス属性はサービスの名前 (値は文字列) です。

## ResourceName 要素

*ResourceName* 要素は対象となるオブジェクトを定義します。ポリシーは、このオブジェクトを保護するように設定されています。ほかの要素は含みません。
*ResourceName* 要素の XML サービス属性は、オブジェクトの名前です。*ResourceName* の例としては、Web サーバー上の `http://www.sunone.com:8080/images`、またはディレクトリサーバー上の `ldap://sunone.com:389/dc=example,dc=com` があります。より具体的なリソースとしては、`salary://uid=jsmith,ou=people,dc=example,dc=com` などが考えられます。この場合、処理対象のオブジェクトは John Smith の給与情報です。

## AttributeValuePair 要素

*AttributeValuePair* 要素はアクションとその値を定義します。この要素は 114 ページの「Subject 要素」、114 ページの「Referral 要素」、および 115 ページの「Condition 要素」 のサブ要素として扱われます。これは *Attribute* 要素と *Value* 要素を含み、XML サービス属性は含んでいません。

### Attribute 要素

*Attribute* 要素はアクションの名前を定義します。アクションとは処理、つまりリソースに対して行われるイベントのことです。POST や GET は Web サーバーリソースに対して行われるアクションであり、READ や SEARCH はディレクトリサーバーリソースに対して行われるアクションです。*Attribute* 要素は *Value* 要素とペアにする必要があります。*Attribute* 要素自体は、ほかの要素を含みません。*Attribute* 要素に対する XML サービス属性は、アクションの名前です。

### Value 要素

*Value* 要素はアクションの値を定義します。Allow (許可する)/Deny (許可しない)、Yes (はい) /No (いいえ) はアクションの値の例です。ほかのアクションの値は、ブール型、数値型、または文字列型が可能です。値は、そのサービスの XML ファイルの中で (`sms.dtd` に基づいて) 定義されています。*Value* 要素はほかの要素を含まず、また XML サービス属性も含みません。

---

注 – 許可しないルールは許可するルールより優先度が高くなります。たとえば、あるポリシーはアクセスを拒否し、別のポリシーが許可すると、(両方のポリシーのほかの条件がすべて一致する場合) 結果は拒否となります。許可しないポリシーを使用すると、ポリシー間で潜在的に衝突が生じるおそれがあるため、十分に注意して拒否ポリシーを使用することをお勧めします。明示的な許可しないルールを使用すると、さまざまな対象 (ロールやグループメンバーシップなど) を通じてユーザーに割り当てられたポリシーが、アクセスを拒否する可能性があります。通常は、ポリシー定義プロセスでは、許可するルールのみを使うべきです。デフォルトで「許可しない」というのは、ほかに適用するポリシーがない場合に使用します。

---

## Subjects 要素

*Subjects* サブ要素はポリシーが適用される主体の集合を特定します。この集合はグループのメンバーシップ、ロールの所有権、または個別ユーザーに基づいて選択されます。この要素は、*Subject* というサブ要素を持ちます。定義できる XML 属性は次のとおりです。

**name**: 集合の名前を定義します。

**description**: 対象の説明を定義します。

**includeType**: 現在は使われていません。

## Subject 要素

*Subject* サブ要素はポリシーを適用する主体の集合を特定します。この集合は、Subjects 要素が定義する集合をさらに絞り込んだものです。メンバーシップは、ロール、グループメンバーシップ、または単なる個別ユーザーのリストに基づきます。この要素は、112 ページの「AttributeValuePair 要素」というサブ要素を含みます。必須の XML 属性は `type` で、特定の定義済み対象を持つ、オブジェクトの一般的な集合を特定します。ほかの XLM 属性には、集合の名前を定義する `name`、対象のメンバーでないユーザーに対してポリシーを適用するかどうかに関して、集合が定義されたとおりになっているかどうかを定義する `includeType` があります。

---

注 – 複数の対象を定義した場合、ポリシーを適用するためには少なくとも 1 つの対象をユーザーに適用しなければなりません。`includeType` を false にして対象を定義するときは、ユーザーがその対象のメンバーではないことが必要です。

---

## Referrals 要素

*Referrals* サブ要素はポリシー参照の集合を特定します。この要素は、*Referral* というサブ要素を持ちます。ともに定義できる XML 属性には、集合の名前を定義する `name`、説明を含む `description` があります。

## Referral 要素

*Referral* サブ要素は個別のポリシー参照を特定します。この要素は、サブ要素として112 ページの「AttributeValuePair 要素」 を取ります。必須の XML 属性は `type` で、特定の定義済み参照を持つ、割り当ての一般的な集合を特定します。また、集合の名前を定義する `name` 属性を持つこともできます。

## Conditions 要素

*Conditions* サブ要素はポリシーの制限 (時間範囲、認証レベル、その他) の集合を特定します。この要素は複数の *Condition* サブ要素を含んでいなければなりません。ともに定義できる XML 属性には、集合の名前を定義する `name`、説明を含む `description` があります。

---

注 – conditions 要素は、ポリシーの中ではオプションの要素です。

---

### Condition 要素

*Condition* サブ要素は特定のポリシーの制限 (時間範囲、認証レベル、その他) を特定します。この要素は、サブ要素として 112 ページの「AttributeValuePair 要素」 を取ります。必須の XML 属性は `type` で、特定の定義済み対象を持つ、オブジェクトの一般的な集合を特定します。また、集合の名前を定義する `name` 属性を持つこともできます。

## ポリシーを有効にしたサービスの追加

サービススキーマの `<Policy>` 要素が `sms.dtd` に設定されている場合にのみ、そのサービスのリソースにポリシーを定義することができます。

デフォルトで、Access Manager は URL ポリシーエージェントサービス (`iPlanetAMWebAgentService`) を提供します。このサービスは、XML ファイル形式で定義され、次のディレクトリにあります。

```
/etc/opt/SUNWam/config/xml/
```

Access Manager にさらにポリシーサービスを追加することもできます。ポリシーサービスを作成したら、`amadmin` コマンド行ユーティリティーを使って、これを Access Manager に追加します。

### ▼ 新しいポリシーを有効にしたサービスを追加する

1. 新しいポリシーサービスを `sms.dtd` に基づいて **XML** ファイルで作成します。新しいポリシーサービスファイルの雛型にできるポリシーサービス **XML** ファイルが **2** つ、**Access Manager** から提供されます。

   `amWebAgent.xml` はデフォルトの URL ポリシーエージェントサービスのための XML ファイルです。これは `/etc/opt/SUNWam/config/xml/` にあります。

   `SampleWebService.xml` は、ポリシーサービスファイルのサンプルであり、`SampleWebService.xml` にあります。

2. 新しいポリシーサービスをロードするディレクトリに **XML** ファイルを保存します。次に例を示します。

   ```
   /config/xml/newPolicyService.xml
   ```

3 新しいポリシーサービスを amadmin コマンド行ユーティリティーを使ってロードします。次に例を示します。

```
AccessManager-base/SUNWam/bin/amadmin
    --runasdn “uid=amAdmin,ou=People,default_org,
root_suffix
    --password password
    --schema /config/xml/newPolicyService.xml
```

4 新しいポリシーサービスをロードしたあと、**Access Manager** コンソールから作業を行うか、または、amadmin で新しいポリシーをロードすることにより、ポリシー定義のルールを定義できます。

# ポリシーの作成

Policy API と Access Manager コンソールを使用してポリシーを作成、変更、および削除でき、amadmin コマンド行ツールを使用してポリシーを作成および削除できます。また、amadmin ユーティリティーを使用して、ポリシーの一覧を XML 形式で取得して表示することもできます。この節では、amadmin コマンド行ユーティリティーと Access Manager コンソールを使用してポリシーを作成することを中心に説明します。Policy API の詳細は、『Sun Java System Access Manager 7.1 Developer’s Guide』を参照してください。

通常ポリシーは XML ファイルで作成し、amadmin コマンド行ユーティリティーを使って Access Manager に追加し、Access Manager のコンソールを使って管理します (ただし、ポリシーをコンソールで作成することもできる)。これは、ポリシーが amadmin を使って直接変更できないからです。ポリシーを修正するには、そのポリシーを Access Manager から削除してから、修正したポリシーを amadmin を使用して追加します。

一般に、ポリシーはレルムツリー全体で使用するために、レルムまたはサブレルムレベルで作成します。

## ▼ amadmin でポリシーを作成する

1 ポリシーの **XML** ファイルを amadmin.dtd に基づいて作成します。このファイルは次のディレクトリにあります。

*AccessManager-base* /SUNWam/dtd.

ポリシーの XML ファイルの例を次に示します。この例には、デフォルトの対象および条件の値がすべて含まれます。これらの値の定義については、104 ページの「ポリシータイプ」を参照してください。

```
<Policy name="bigpolicy" referralPolicy="false" active="true" >
<Rule name="rule1">
<ServiceName name="iPlanetAMWebAgentService" />
<ResourceName name="http://thehost.thedomain.com:80/*.html" />
<AttributeValuePair>
<Attribute name="POST" />
<Value>allow</Value>
</AttributeValuePair>
<AttributeValuePair>
<Attribute name="GET" />
<Value>allow</Value>
</AttributeValuePair>
</Rule>
<Subjects name="subjects" description="desccription">
<Subject name="webservicescleint" type="WebServicesClients" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/><Value>CN=sun-unix,
OU=SUN Java System Access Manager, O=Sun, C=US</Value>
</AttributeValuePair>
</Subject>
<Subject name="amrole" type="IdentityServerRoles" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/><Value>
cn=organization admin role,o=realm1,dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
<Subject name="au" type="AuthenticatedUsers" includeType="inclusive">
</Subject>
<Subject name="ldaporganization" type="Organization" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/>
<Value>dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
<Subject name="ldapuser" type="LDAPUsers" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/>
<Value>uid=amAdmin,ou=People,dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
<Subject name="ldaprole" type="LDAPRoles" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/>
<Value>cn=Organization Admin Role,o=realm1,dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
<Subject name="ldapgroup" type="LDAPGroups" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/>
```

```
<Value>cn=g1,ou=Groups,dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
<Subject name="amidentitysubject" type="AMIdentitySubject" includeType="inclusive">
<AttributeValuePair><Attribute name="Values"/>
<Value>id=amAdmin,ou=user,dc=red,dc=iplanet,dc=com</Value>
</AttributeValuePair>
</Subject>
</Subjects>
<Conditions name="conditions" description="description">
<Condition name="ldapfilter" type="LDAPFilterCondition">
<AttributeValuePair><Attribute name="ldapFilter"/>
<Value>dept=finance</Value>
</AttributeValuePair>
</Condition>
<Condition name="authlevelge-nonrealmqualified" type="AuthLevelCondition">
<AttributeValuePair><Attribute name="AuthLevel"/>
<Value>1</Value>
</AttributeValuePair>
</Condition>
<Condition name="authlevelle-realmqaulfied" type="LEAuthLevelCondition">
<AttributeValuePair><Attribute name="AuthLevel"/>
<Value>/:2</Value>
</AttributeValuePair>
</Condition>
<Condition name="sessionproperties" type="SessionPropertyCondition">
<AttributeValuePair><Attribute name="valueCaseInsensitive"/>
<Value>true</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="a"/><Value>10</Value>
<Value>20</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="b"/><Value>15</Value>
<Value>25</Value>
</AttributeValuePair>
</Condition>
<Condition name="activesessiontime" type="SessionCondition">
<AttributeValuePair><Attribute name="TerminateSession"/>
<Value>session_condition_false_value</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="MaxSessionTime"/>
<Value>30</Value>
</AttributeValuePair>
```

```
</Condition>
<Condition name="authelevelle-nonrealmqualfied"
        type="LEAuthLevelCondition">
<AttributeValuePair><Attribute name="AuthLevel"/>
<Value>2</Value>
</AttributeValuePair>
</Condition>
<Condition name="ipcondition" type="IPCondition">
<AttributeValuePair><Attribute name="DnsName"/>
<Value>*.iplanet.com</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="EndIp"/>
<Value>145.15.15.15</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="StartIp"/>
<Value>120.10.10.10</Value>
</AttributeValuePair>
</Condition>
<Condition name="authchain-realmqualfied"
        type="AuthenticateToServiceCondition">
<AttributeValuePair><Attribute name="AuthenticateToService"/>
<Value>/:ldapService</Value>
</AttributeValuePair>
</Condition>
<Condition name="auth to realm"
     type="AuthenticateToRealmCondition">
<AttributeValuePair><Attribute name="AuthenticateToRealm"/>
<Value>/</Value>
</AttributeValuePair>
</Condition>
<Condition name="authlevelge-realmqualified"
     type="AuthLevelCondition">
<AttributeValuePair><Attribute name="AuthLevel"/>
<Value>/:2</Value>
</AttributeValuePair>
</Condition>
<Condition name="authchain-nonrealmqualfied"
    type="AuthenticateToServiceCondition">
<AttributeValuePair><Attribute name="AuthenticateToService"/>
<Value>ldapService</Value>
</AttributeValuePair>
</Condition>
<Condition name="timecondition" type="SimpleTimeCondition">
```

```
<AttributeValuePair><Attribute name="EndTime"/>
<Value>17:00</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="StartTime"/>
<Value>08:00</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="EndDate"/>
<Value>2006:07:28</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="EnforcementTimeZone"/>
<Value>America/Los_Angeles</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="StartDay"/>
<Value>mon</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="StartDate"/>
<Value>2006:01:02</Value>
</AttributeValuePair>
<AttributeValuePair><Attribute name="EndDay"/>
<Value>fri</Value>
</AttributeValuePair>
</Condition>
</Conditions>
<ResponseProviders name="responseproviders"
     description="description">
<ResponseProvider name="idresponseprovidere"
   type="IDRepoResponseProvider">
<AttributeValuePair>
<Attribute name="DynamicAttribute"/>
</AttributeValuePair>
<AttributeValuePair>
<Attribute name="StaticAttribute"/>
<Value>m=10</Value>
<Value>n=30</Value>
</AttributeValuePair>
</ResponseProvider>
</ResponseProviders>
</Policy>
```

**2** ポリシーの **XML** ファイルを作成したら、次のコマンドを使用してロードできます。

```
AccessManager-base/SUNWam/bin/amadmin
--runasdn "uid=amAdmin,ou=People,default_org,
root_suffix"
```

`--password` *password*
`--data` *policy.xml*

複数のポリシーを同時に追加するには、各 XML ファイルにポリシーを 1 つずつ置くのではなく、1 つの XML ファイルにすべてのポリシーを置きます。複数の XML ファイルでポリシーを次々とロードすると、内部ポリシーインデックスが破損したり、ポリシーの評価に参加できないポリシーが生じたりするおそれがあります。

ポリシーを `amadmin` で作成するときは、認証スキーム条件を作成中にレルムに認証モジュールを登録することの確認、レルム、LDAP グループ、LDAP ロール、および LDAP ユーザーの対象を作成中に、対応する LDAP オブジェクト (レルム、グループ、ロール、およびユーザー) が存在することの確認、`IdentityServerRoles` 対象を作成中に、Access Manager ロールが存在することの確認、そしてサブレルムまたはピアレルムの参照を作成中に、関連があるレルムが存在することの確認を行ってください。

`SubrealmReferral`、`PeerRealmReferral`、`Realm` 対象、`IdentityServerRoles` 対象、`LDAPGroups` 対象、`LDAPRoles` 対象、および `LDAPUsers` 対象の Value 要素のテキストには、完全な DN を指定する必要があります。

## ▼ Access Manager コンソールを使って標準ポリシーを作成する

1. ポリシーを作成するレルムを選択します。
2. 「ポリシー」タブをクリックします。
3. 「ポリシー」リストから「新規ポリシー」をクリックします。
4. ポリシーの名前と説明を入力します。
5. ポリシーをアクティブにする場合は、「アクティブ」属性で「はい」を選択します。
6. この時点では、標準ポリシーのフィールドすべてを定義する必要はありません。ポリシーの作成後、ルール、対象、条件、および応答プロバイダを追加できます。詳細は、**124** ページの「ポリシーの管理」を参照してください。
7. 「了解」をクリックします。

### ▼ Access Manager コンソールを使って参照ポリシーを作成する

1 ポリシーを作成するレルムを選択します。

2 「ポリシー」タブから「新規参照」をクリックします。

3 ポリシーの名前と説明を入力します。

4 ポリシーをアクティブにする場合は、「アクティブ」属性で「はい」を選択します。

5 この時点では、参照ポリシーのフィールドすべてを定義する必要はありません。ポリシーの作成後、ルールおよび参照を追加できます。詳細は、124 ページの「ポリシーの管理」を参照してください。

6 「了解」をクリックします。

## ピアレルムおよびサブレルムのポリシーの作成

ピアレルムまたはサブレルムのポリシーを作成するには、まず親レルムまたは別のピアレルムで参照ポリシーを作成する必要があります。参照ポリシーのルールの定義には、サブレルムが管理するリソースプレフィックスを含める必要があります。親レルムまたは別のピアレルムで参照ポリシーを作成すれば、サブレルムまたはピアレルムで標準ポリシーを作成できます。

次の例では、`o=isp` は親レルム、`o=example.com` は、`http://www.example.com` のリソースとサブリソースを管理するサブレルムです。

### ▼ サブレルムのポリシーを作成する

1 `o=isp` で参照ポリシーを作成します。参照ポリシーについては、129 ページの「参照ポリシーの修正」の手順を参照してください。

参照ポリシーは、`http://www.example.com` をリソースとしてルールに定義し、参照内に `example.com` を値として持つ `SubRealmReferral` を含んでいる必要があります。

2 `example.com` というサブレルムに移動します。

3 これで isp によってリソースが example.com の管理に委ねられたので、http://www.example.com というリソース、または http://www.example.com から始まる任意のリソースに対して標準ポリシーを作成できます。

example.com で管理する別のリソースのポリシーを定義するには、追加の参照ポリシーを o=isp に作成する必要があります。

# ほかの Access Manager インスタンスへのポリシーのエクスポート

Access Manager では、amadmin コマンド行ツールを使用してポリシーのエクスポートが可能です。この機能は、多数の既存のポリシーを別の Access Manager インスタンスに移動する場合や、既存のポリシーに加えた変更をバッチモードで検査する場合に便利です。ポリシーをエクスポートするには、amadmin コマンド行ユーティリティーを使用して指定したポリシーをファイルにエクスポートします。構文は次のとおりです。

amamdin - u *username* –w *password* –ofilename *output_file.xml* –t *policy_data_file.xml*

ポリシー名にはワイルドカード (*) を使用して、任意の文字列を表すことができます。

*policy_data_file.xml* の例を次に示します。

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--
    Copyright (c) 2005 Sun Microsystems, Inc. All rights reserved
    Use is subject to license terms.
-->

<!DOCTYPE Requests
    PUBLIC "-//iPlanet//Sun Java System Access Manager 6.2 Admin CLI DTD//EN"
    "/opt/SUNWam/dtd/amAdmin.dtd"
>>

<!--  CREATE REQUESTS -->

<!-- to export to file use option -ofilename fileName -->

<Requests>

<RealmRequests >
<RealmGetPolicies realm="/" >
<AttributeValuePair>
```

```
<Attribute name="policyName"/>
<Value>p*</Value>
</AttributeValuePair>
</RealmGetPolicies>
</RealmRequests>

<RealmRequests >
<RealmGetPolicies realm="/" >
<AttributeValuePair>
<Attribute name="policyName"/>
<Value>g10</Value>
<Value>g11</Value>
</AttributeValuePair>
</RealmGetPolicies>

</RealmRequests>
<RealmRequests >
<RealmGetPolicies realm="/realm1" >
<AttributeValuePair>
<Attribute name="policyName"/>
<Value>*</Value>
</AttributeValuePair>
</RealmGetPolicies>
</RealmRequests>

</Requests>
```

ポリシーは *Output_file.xml* ファイルにエクスポートされます。このファイルに含まれるポリシー定義に任意の変更を加えることができます。ポリシーを別の Access Manager インスタンスにインポートする前に、`amadmin` コマンドユーティリティーと互換性を持つように出力ファイルを変更しておく必要があります。`amadmin` と互換性のあるポリシーデータファイルの例を含む、ポリシーのインポート手順については、「amadmin でポリシーを作成する」を参照してください。

# ポリシーの管理

標準または参照ポリシーを作成し、Access Manager に追加したあとは、ポリシーの管理は、Access Manager のコンソールを使用して、ルール、対象、条件と参照を変更することにより行えます。

# 標準ポリシーの修正

「ポリシー」タブを使用して、アクセス権を定義する標準ポリシーを変更することができます。複数のルール、対象、条件、およびリソースコンパレータを定義および設定できます。ここでは、標準ポリシーを変更する手順のいくつかについて説明します。

## ▼ 標準ポリシーのルールを追加または変更する

1 すでにポリシーを作成済みである場合は、ルールを追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、**121 ページの「Access Manager コンソールを使って標準ポリシーを作成する」**を参照してください。

2 「ルール」メニューから「新規」をクリックします。

3 次のデフォルトのサービスタイプから、ルール用に 1 つ選択します。ポリシーに複数のサービスが使用できる場合は、さらに多くのサービスが一覧表示されます。

| | |
|---|---|
| ディスカバリサービス | ディスカバリサービスクエリー用の認証アクションを定義し、指定されたリソースについて、Web サービスクライアントによるプロトコル呼び出しを変更します。 |
| Liberty 個人プロファイルサービス | Liberty 個人プロファイルサービスクエリー用の認証アクションを定義し、指定されたリソースについて、Web サービスクライアントによるプロトコル呼び出しを変更します。 |
| URL ポリシーエージェント | URL ポリシーエージェントサービス用の認証アクションを定義します。このサービスは、HTTP および HTTPS の URL を保護するポリシーの定義に使用します。このサービスは、Access Manager ポリシーのもっとも一般的な使用例です。 |

4 「次へ」をクリックします。

5 ルールの名前とリソース名を入力します。

現在、Access Manager ポリシーエージェントでサポートされているリソースは `http://` と `https://` だけです。また、ホスト名の代わりに IP アドレスを使用することはできません。

プロトコル、ホスト、ポート、およびリソース名にはワイルドカードを使用できます。次に例を示します。

```
http*://*:*/*.html
```

URL ポリシーエージェントサービスでは、ポート番号が入力されていない場合のデフォルトのポート番号は、`http://` では 80、`https://` では 443 となります。

**6** ルールのアクションを選択します。サービスタイプに応じて、次のアクションを選択できます。

- LOOKUP (ディスカバリサービス)
- UPDATE (ディスカバリサービス)
- MODIFY (Liberty 個人プロファイルサービス)
- QUERY (Liberty 個人プロファイルサービス)
- GET (URL ポリシーエージェント)
- POST (URL ポリシーエージェント)

**7** アクションの値を選択します。

- 同意を求める対話プロトコルの呼び出し — リソースに関する同意を求める Liberty 対話プロトコルを呼び出します。これは、Liberty 個人プロファイルサービスタイプにのみ使用できます。
- 値を求める対話プロトコルの呼び出し — リソースに関する値を求める Liberty 対話プロトコルを呼び出します。これは、Liberty 個人プロファイルサービスタイプにのみ使用できます。
- 許可 — ルールに定義されたリソースに一致するリソースへのアクセスを許可
- 拒否 — ルールに定義されたリソースに一致するリソースへのアクセスを拒否

  ポリシーでは、拒否ルールが許可ルールよりも優先されます。たとえばあるリソースに 2 つのポリシーがあり、1 つはアクセス拒否でもう 1 つはアクセス許可の場合、その結果はアクセスの拒否になります (両方のポリシーの条件が一致する場合)。拒否ポリシーを使用すると、ポリシー間で潜在的に衝突が生じるおそれがあるため、十分に注意して拒否ポリシーを使用することをお勧めします。通常は、ポリシー定義プロセスでは許可ルールだけを使用し、拒否の場合を実現するのに適用するポリシーがない場合にデフォルトの拒否を使用してください。

  拒否ルールを明示的に使用すると、1 つ以上のポリシーでアクセスが許可される場合でも、異なる対象 (ロールやグループのメンバーシップ) を通じてユーザーに割り当てられたポリシーによって、リソースへのアクセスを拒否されるおそれがあります。たとえば、1 つのリソースについて、Employee ロールに適用される拒否ポリシーと、Manager ロールに適用される許可ポリシーがあるとします。この場合、Employee ロールと Manager ロールの両方を割り当てられているユーザーへのポリシーの決定は拒否されます。

  このような問題を解決する 1 つの方法は、条件プラグインを使ってポリシーを設計することです。上記の例では、Employee ロールに認証されたユーザーには拒否ポリシーを適用し、Manager ロールに認証されたユーザーには許可ポリシーを適

用するという "ロール条件" を利用することで、2 つのポリシーを区別できます。Manager ロールにはより高い認証レベルが与えられることから、認証レベル条件を使用する方法もあります。

8 「終了」をクリックします。

## ▼ 標準ポリシーの対象を追加および変更する

1 すでにポリシーを作成済みである場合は、対象を追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、121 ページの「Access Manager コンソールを使って標準ポリシーを作成する」を参照してください。

2 「対象」リストから、「新規」をクリックします。

3 デフォルトの対象タイプを次の中から選択します。対象タイプの説明は、105 ページの「対象」を参照してください。

4 「次へ」をクリックします。

5 対象の名前を入力します。

6 「排他的」フィールドを選択または選択解除します。

このフィールドが選択されていないと (デフォルト)、ポリシーは、対象のメンバーであるアイデンティティーに適用されます。このフィールドが選択されていると、ポリシーは、対象のメンバーではないアイデンティティーに適用されます。

ポリシーに複数の対象が存在する場合は、指定されたアイデンティティーにポリシーが適用されることが少なくとも 1 つの対象で示されている場合に、そのポリシーがアイデンティティーに適用されます。

7 検索を実行して、対象に追加するアイデンティティーを表示します。この手順は、「認証済みユーザー」対象や「Web サービスクライアント」対象には適用できません。

デフォルト (*) の検索パターンでは、該当するすべてのエントリが表示されます。

8 対象に追加する個々のアイデンティティーを選択するか、または「すべて追加」を選択して一度にすべてのアイデンティティーを追加します。「追加」をクリックしてアイデンティティーを選択リストに移動します。この手順は、「認証済みユーザー」対象には適用されません。

9 「終了」をクリックします。

10 ポリシーから対象を削除するには、対象を選択して「削除」をクリックします。対象の名前をクリックすると、対象の定義を編集できます。

## ▼ 標準ポリシーに条件を追加する

1 すでにポリシーを作成済みである場合は、条件を追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、121 ページの「Access Manager コンソールを使って標準ポリシーを作成する」を参照してください。

2 「条件」リストから、「新規」をクリックします。

3 条件タイプを選択して、「次へ」をクリックします。

4 条件タイプのフィールドを定義します。

5 「終了」をクリックします。

## ▼ 標準ポリシーに応答プロバイダを追加する

1 すでにポリシーを作成済みである場合は、応答プロバイダを追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、121 ページの「Access Manager コンソールを使って標準ポリシーを作成する」を参照してください。

2 「応答プロバイダ」リストから、「新規」をクリックします。

3 応答プロバイダの名前を入力します。

4 次の値を定義します。

| | |
|---|---|
| StaticAttribute | 属性値形式の静的属性であり、ポリシーに保存されている `IDResponseProvider` のインスタンスに定義されています。 |
| DynamicAttribute | ここで選択する応答属性は、対応するレルムのポリシー設定サービス内で事前に定義されている必要があります。定義する属性名は、設定済みデータストア (IDRepository) 内に存在する属性名のサブセットになるようにしてください。属性の定義方法の詳細については、ポリシー設定属性定義を参照してください。特定または複数の属性を選択するには、Ctrl キーを押しながらマウスの左ボタンをクリックします。 |

5 「終了」をクリックします。

6 ポリシーから応答プロバイダを削除する場合は、そのプロバイダを選択し、「削除」をクリックします。名前をクリックすると、応答プロバイダの定義を編集できます。

# 参照ポリシーの修正

参照ポリシーを使用すると、あるレルムのポリシーの定義や判断を別のレルムに委任できます。カスタム参照を使用すると、任意のポリシー適用先からポリシー決定を取得できます。参照ポリシーを作成したら、関連付けられているルール、参照、および応答プロバイダを追加または変更できます。

## ▼ 参照ポリシーのルールを追加および変更する

1 すでにポリシーを作成済みである場合は、ルールを追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、122 ページの「Access Manager コンソールを使って参照ポリシーを作成する」を参照してください。

2 「ルール」メニューから「新規」をクリックします。

3 次のデフォルトのサービスタイプから、ルール用に 1 つ選択します。ポリシーに複数のサービスが使用できる場合は、さらに多くのサービスが一覧表示されます。

| | |
|---|---|
| ディスカバリサービス | ディスカバリサービスクエリー用の認証アクションを定義し、指定されたリソースについて、Web サービスクライアントによるプロトコル呼び出しを変更します。 |
| Liberty 個人プロファイルサービス | Liberty 個人プロファイルサービスクエリー用の認証アクションを定義し、指定されたリソースについて、Web サービスクライアントによるプロトコル呼び出しを変更します。 |
| URL ポリシーエージェント | URL ポリシーエージェントサービス用の認証アクションを定義します。このサービスは、HTTP および HTTPS の URL を保護するポリシーの定義に使用します。このサービスは、Access Manager ポリシーのもっとも一般的な使用例です。 |

4 「次へ」をクリックします。

5 ルールの名前とリソース名を入力します。

現在、Access Manager ポリシーエージェントでサポートされているリソースは http:// と https:// だけです。また、ホスト名の代わりに IP アドレスを使用することはできません。

プロトコル、ホスト、ポート、およびリソース名にはワイルドカードを使用できます。次に例を示します。

```
http*://*:*/*.html
```

URL ポリシーエージェントサービスでは、ポート番号が入力されていない場合のデフォルトのポート番号は、http:// では 80、https:// では 443 となります。

6 「終了」をクリックします。

## ▼ ポリシーの参照を追加または変更する

1 すでにポリシーを作成済みである場合は、応答プロバイダを追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、122 ページの「Access Manager コンソールを使って参照ポリシーを作成する」を参照してください。

2 「参照」リストから、「新規」をクリックします。

3 「ルール」フィールドにリソースを定義します。フィールドには、次のものがあります。

参照— 現在の参照タイプが表示されます。

名前— 参照の名前を入力します。

リソース名— リソースの名前を入力します。

フィルター— 「値」フィールドに表示するレルム名を絞り込むためのフィルタを指定します。デフォルトでは、すべてのレルム名が表示されます。

値 — 参照のレルム名を選択します。

4 「終了」をクリックします。

ポリシーから参照を削除するには、参照を選択して「削除」をクリックします。

参照名の横にある「編集」リンクをクリックすれば、参照の定義を編集できます。

## ▼ 参照ポリシーに応答プロバイダを追加する

1 すでにポリシーを作成済みである場合は、応答プロバイダを追加するポリシーの名前をクリックします。ポリシーを作成済みでない場合は、122 ページの「Access Manager コンソールを使って参照ポリシーを作成する」を参照してください。

2 「応答プロバイダ」リストから、「新規」をクリックします。

3 応答プロバイダの名前を入力します。

4 次の値を定義します。

StaticAttribute　属性値形式の静的属性であり、ポリシーに保存されている `IDResponseProvider` のインスタンスに定義されています。

DynamicAttribute　ここで選択する応答属性は、対応するレルムのポリシー設定サービス内で事前に定義されている必要があります。定義する属性名は、設定済みデータストア (IDRepository) 内に存在する属性名のサブセットになるようにしてください。属性の定義方法の詳細については、ポリシー設定属性定義を参照してください。特定または複数の属性を選択するには、Ctrl キーを押しながらマウスの左ボタンをクリックします。

5 「終了」をクリックします。

6 ポリシーから応答プロバイダを削除する場合は、そのプロバイダを選択し、「削除」をクリックします。名前をクリックすると、応答プロバイダの定義を編集できます。

# ポリシー設定サービス

各組織のポリシーに関連する属性の設定を Access Manager コンソールから行うにはポリシー設定サービスを使用します。Access Manager ポリシーフレームワークで使用するリソース名の実装および Directory Server データストアを定義することもできます。ポリシー設定サービスに指定した Directory Server は、LDAP ユーザー、LDAP グループ、LDAP ロール、および組織ポリシー対象のメンバーシップの評価に使用されます。

## 対象結果の有効時間

ポリシー評価のパフォーマンスを上げるため、ポリシー設定サービスの「対象結果の有効時間」属性で定義されるとおり、メンバーシップ評価を一定の時間キャッシュします。これらのキャッシュされたメンバーシップ決定は「対象結果の有効時間」属性で定義された時間が経つまで使用されます。この時間が経過したあとは、ディレクトリ内のユーザーの現在の状態を反映するために、メンバーシップ評価が使用されます。

## 動的属性

これらは使用可能な動的属性の名前であり、ポリシーの応答プロバイダの動的属性を定義する際に一覧表示され、選択されます。定義される名前は、データリポジトリ内に定義された属性名と同じである必要があります。

## amldapuserの定義

amldapuserはインストール時に作成されたユーザーで、ポリシー設定サービスに指定したDirectory Serverでデフォルトで使用されます。amldapuserは、レルムの管理者またはポリシー管理者が必要に応じて変更できます。

## ポリシー設定サービスの追加

レルムを作成すると、そのレルムのためにポリシー設定サービスの属性が自動的に設定されます。ただし、これらの属性は必要に応じて変更できます。

# リソースベースの認証

組織によっては高度な認証シナリオが必要です。この場合、ユーザー認証は、アクセスしようとするリソースごとに特定のモジュールで行われます。リソースベースの認証は、Access Managerの機能で、ユーザーはデフォルトの認証モジュールではなく、そのリソースを保護している認証モジュールから認証される必要があります。この機能は、ユーザーがはじめて認証されるときにのみ適用可能です。

---

注 – これは、97ページの「セッションのアップグレード」で説明しているリソースベースの認証とは別の機能です。その機能には何の制限もありません。

---

## 制限

リソースベースの認証には、次のような制限があります。

- リソースに適用可能なポリシーに認証モジュールが複数個含まれていた場合、そのどちらかの認証モジュールがシステムによって自動的に選択されます。
- このポリシーについて定義できる条件はレベルと方式だけです。
- この機能は、異なるDNSドメイン間では働きません。

## ▼ リソースベースの認証を設定する

Access Manager とポリシーエージェントをインストールしたら、リソースベースの認証を設定できます。そのためには、Access Manager が Gateway サーブレットを指す必要があります。

1 `AMAgent.properties` を開きます。

`AMAgent.properties` は Solaris 環境では `/etc/opt/SUNWam/agents/config/` にあります。

2 次の行をコメントアウトします。

```
#com.sun.am.policy.am.loginURL = http://Access
Manager_server_host.domain_name:port/amserver/UI/Login.
```

3 ファイルに次の行を追加します。

```
com.sun.am.policy.am.loginURL =
http://AccessManager_host.domain_name:port/amserver/gateway
```

---

注 – ゲートウェイサーブレットは Policy Evaluation API を使って開発します。このサーブレットを使えば、リソースベースの認証を実現するためのカスタムメカニズムを記述できます。『Access Manager Developer's Guide』の『Sun Java System Access Manager 7.1 Developer's Guide』の第 3 章「Using the Policy APIs」を参照してください。

---

4 エージェントを再起動します。

◆ ◆ ◆ 第 6 章

# 対象の管理

「対象」インタフェースにより、レルム内部で基本的なアイデンティティー管理を行うことができます。「対象」インタフェースで作成するすべてのアイデンティティーは、Access Manager アイデンティティー対象タイプを使って作成されるポリシーに対する対象定義で使用できます。

作成および変更できるアイデンティティーには、次のものがあります。



## ユーザー

ユーザーは、個人のアイデンティティーを表現します。グループ内でユーザーを作成および削除できます。また、ロールやグループにユーザーを追加したり、ロールやグループからユーザーを削除することができます。サービスをユーザーに割り当てることもできます。

### ▼ ユーザーを作成または変更する

1 「ユーザー」タブをクリックします。

2 「新規」をクリックします。

3 次のフィールドのデータを入力します。

「ユーザー **ID**」**:** このフィールドには、ユーザーが Access Manager へログインするために使用する名前を指定します。このプロパティーは、DN 以外の値になることがあります。

「名」**:** このフィールドには、ユーザーの名 (ファーストネーム) を指定します。

「姓」**:** このフィールドには、ユーザーの姓 (ラストネーム) を指定します。

「フルネーム」**:** このフィールドには、ユーザーのフルネームを指定します。

「パスワード」**:** — このフィールドには、「ユーザー ID」フィールドで指定した名前のパスワードを指定します。

「パスワード **(**確認**)**」**:** 確認のためにパスワードを再入力します。

「ユーザー状態」**:** このオプションは、ユーザーが Access Manager 経由の認証を許可されるかどうかを示します。

4 「了解」をクリックします。

5 ユーザーの作成後は、ユーザーの名前をクリックすることによってユーザー情報を編集できます。ユーザー属性の詳細については、「ユーザー属性」を参照してください。実行できるその他の変更には、次のものがあります。

- 135 ページの「ユーザーを作成または変更する」
- 136 ページの「ロールおよびグループにユーザーを追加する」
- 137 ページの「サービスをアイデンティティーに追加する」

## ▼ ロールおよびグループにユーザーを追加する

1 変更するユーザーの名前をクリックします。

2 「ロール」または「グループ」を選択します。ユーザーにすでに割り当てられているロールおよびグループだけが表示されます。

3 「利用可能」リストからロールまたはグループを選択して「追加」をクリックします。

4 ロールまたはグループが「選択」リストに表示されたら、「保存」をクリックします。

### ▼ サービスをアイデンティティーに追加する

1. サービスを追加するアイデンティティーを選択します。
2. 「サービス」タブをクリックします。
3. 「追加」をクリックします。
4. 選択したアイデンティティータイプに応じて、次のサービスのリストが表示されます。
   - 認証設定
   - ディスカバリサービス
   - Liberty 個人プロファイルサービス
   - セッション
   - ユーザー
5. 追加するサービスを選択して「次へ」をクリックします。
6. サービスの属性を編集します。サービスの説明を見るには、ステップ **4** でサービス名をクリックします。
7. 「終了」をクリックします。

## エージェントプロファイル

Access Manager ポリシーエージェントは、Web サーバーおよび Web プロキシサーバー上のコンテンツを無許可の侵入から保護します。管理者が設定したポリシーに基づいてサービスおよび Web リソースへのアクセスを制御します。

エージェントオブジェクトは、ポリシーエージェントプロファイルを定義します。また、Access Manager リソースを保護している特定のエージェントの認証情報やその他のプロファイル情報を Access Manager で保存できるようにします。Access Manager コンソールを使用して、管理者はエージェントプロファイルを表示、作成、変更、および削除できます。

エージェントオブジェクト作成ページは、エージェントが Access Manager の認証を受ける UID およびパスワードを定義できる場所です。同一の Access Manager を使用して複数の Web コンテナを設定した場合は、これにより、異なるエージェントに対して複数の ID を有効にすることができ、Access Manager から独立してそれらの ID を有効にしたり無効にしたりできます。また、マシンごとに `AMAgent.properties` を編集するのではなく、エージェントの設定値によっては集中的に管理することもできます。

## ▼ エージェントを作成または変更する

1 「エージェント」タブをクリックします。

2 「新規」をクリックします。

3 次のフィールドの値を入力します。

「名前」: エージェントの名前またはアイデンティティーを入力します。この名前を使用してエージェントは Access Manager にログインします。名前に複数バイト文字を含めることはできません。

「パスワード」: エージェントのパスワードを入力します。このパスワードは、LDAP 認証時にエージェントが使用するパスワードとは異なっている必要があります。

「パスワードを確認」: パスワードを確認します。

「デバイスの状態」: エージェントのデバイスの状態を入力します。「アクティブ」に設定されている場合、エージェントは Access Manager に対して認証および通信を行うことができます。「非アクティブ」に設定されている場合、エージェントは Access Manager に対して認証を行うことができません。

4 「了解」をクリックします。

5 エージェントを作成したあとで、さらに次のフィールドを編集できます。

「説明」: エージェントの簡単な説明を入力します。たとえば、エージェントインスタンス名またはそれが保護しているアプリケーションの名前を入力できます。

「エージェントキー値」: キーと値のペアでエージェントのプロパティーを設定します。このプロパティーは、ユーザーに関する資格表明の要求をエージェントから受け付けるために Access Manager によって使用されます。現時点では 1 つのプロパティーだけが有効であり、その他のプロパティーはすべて無視されます。次の書式で入力します。

*agentRootURL*=`protocol://` *hostname:port/*

正確に入力する必要があり、`agentRootURL` では大文字と小文字を区別します。

protocol　使用するプロトコルを表します。HTTP と HTTPS のいずれかです。

hostname　エージェントがインストールされているマシンのホスト名を表します。このマシンには、エージェントが保護するリソースも格納されます。

port　エージェントがインストールされているポート番号を表します。エージェントは、このポートで受信トラフィックを待機し、ホスト上のリソースにアクセスするための要求をすべて遮断します。

## Cookie ハイジャックから保護するための Access Manager の設定

Cookie ハイジャックは、詐称者 (おそらく信頼できないアプリケーションを使用するハッカー) が Cookie に未承認でアクセスしている状況を表します。ハイジャックされた Cookie がセッション Cookie である場合は、システムの設定方法によっては、Cookie ハイジャックにより、保護された Web リソースに未承認でアクセスするおそれが増す可能性があります。

Sun のドキュメントには、「Access Management Deployment でのセッション Cookie ハイジャックに対する予防措置」というタイトルのテクニカルノートがあり、セッション Cookie ハイジャック関連のセキュリティー上の脅威に対する予防措置に関する情報が記述されています。次のドキュメントを参照してください。

『Technical Note: Precautions Against Cookie Hijacking in an Access Manager Deployment』

# フィルタを適用したロール

フィルタロールは、LDAP フィルタを使用して作成される動的なロールです。ユーザーはすべてフィルタを通してまとめられ、ロールの作成時にそのロールに割り当てられます。フィルタはエントリの属性と値のペア (`ca=user*` など) を検索して、その属性を含むユーザーをロールに自動的に割り当てます。

## ▼ フィルタロールを作成する

1. ナビゲーション区画で、ロールを作成する組織に移動します。
2. 「新規」をクリックします。
3. フィルタロールの名前を入力します。
4. 検索条件を入力します。
   例を示します。

   ```
   (&(uid=user1)(|(inetuserstatus=active)(!(inetuserstatus=*))))
   ```

   フィルタを空白のままにすると、次のロールが作成されます。

   ```
   (objectclass = inetorgperson)
   ```

5. 「了解」をクリックして、フィルタ条件に基づく検索を開始します。フィルタ条件によって定義されたアイデンティティーは、ロールに自動的に割り当てられます。

6 フィルタロールが作成されたら、ロールの名前をクリックして、そのロールに属するユーザーを表示します。「サービス」タブをクリックして、ロールにサービスを追加することもできます。

# ロール

ロールのメンバーは、ロールを所有する LDAP エントリです。ロール自体の基準は、属性を持つ LDAP エントリとして定義されます。このエントリは、エントリの識別名 (DN) 属性で特定されます。ロールが作成されたら、サービスとユーザーを手動で追加できます。

## ▼ ロールを作成または変更する

1 「ロール」タブをクリックします。

2 「ロール」リストで「新規」をクリックします。

3 ロールの名前を入力します。

4 「了解」をクリックします。

## ▼ ユーザーをロールまたはグループに追加する

1 ユーザーを追加するロールまたはグループの名前をクリックします。

2 「ユーザー」タブをクリックします。

3 追加するユーザーを「利用可能」リストから選択して「追加」をクリックします。

4 ユーザーが「選択」リストに表示されたら、「保存」をクリックします。

# グループ

グループは、共通の職務、特徴、または興味を持つユーザーの集合を表現します。通常、このグループには関連付けられた権限はありません。グループは、組織内および管理されているほかのグループ内という、2 つのレベルに存在できます。

## ▼ グループを作成または変更する

1 「グループ」タブをクリックします。

2 「グループ」リストから「新規」をクリックします。

3 グループの名前を入力します。

4 「了解」をクリックします。
グループを作成したら、グループ名、「ユーザー」タブの順にクリックして、ユーザーをグループに追加できます。

パート II

# ディレクトリ管理とデフォルトサービス

これは『Sun Java System Access Manager 7.1 管理ガイド』の第 2 部です。「ディレクトリ管理」の章では、Access Manager を旧バージョンモードで配備するときにディレクトリオブジェクトを管理する方法について説明します。その他の章では、Access Manager の一部のデフォルトサービスを設定および使用する方法について説明します。次の章で構成されています。

◆ ◆ ◆ 第 7 章

# 7 ディレクトリ管理

「ディレクトリ管理」タブは、Access Manager を旧バージョンモードでインストールした場合にのみ表示されます。このディレクトリ管理機能は、Sun Java System の Directory Server に対応した Access Manager の配備に必要なアイデンティティー管理ソリューションを提供します。

インストール時の旧バージョンモードオプションについては、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』を参照してください。

## ディレクトリオブジェクトの管理

「ディレクトリ管理」タブには、Directory Server オブジェクトの表示および管理に必要なすべてのコンポーネントが含まれています。この節では、オブジェクトタイプと、それらを設定する方法の詳細について説明します。Access Manager コンソールまたはコマンド行インタフェースを使用して、ユーザー、ロール、グループ、組織、サブ組織、およびコンテナの各オブジェクトを定義、修正、または削除できます。コンソールには、ディレクトリオブジェクトを作成および管理する権限が異なる、複数のデフォルト管理者が存在しています。ロールに基づいて、管理者を追加作成できます。管理者は、Access Manager でインストールしたときに、Directory Server 内に定義されます。次の Directory Server オブジェクトを管理できます。

# 組織

組織は、企業が部門とリソースの管理に使用する最上位レベルの階層構造を表します。インストール時に、Access Manager は最上位レベルの組織 (インストール時に定義) を動的に作成して、Access Manager の企業構成を管理します。インストール後に組織を追加作成して、企業を個別に管理できます。作成した組織はすべて、最上位レベルの組織の下に入ります。

## ▼ 組織を作成する

1 「ディレクトリ管理」タブをクリックします。

2 「組織」リストで、「新規」をクリックします。

3 フィールドの値を入力します。「名前」だけが必須です。フィールドは次のとおりです。

| | |
|---|---|
| 名前 | 組織の名前の値を入力します。 |
| 「ドメイン名」 | ドメインネームシステム (DNS) を使用している場合は、DNS の完全な名前を入力します。 |
| 「組織の状態」 | 「アクティブ」または「非アクティブ」の状態を選択します。デフォルトは「アクティブ」です。これは、その組織の存続期間中であればいつでも、「プロパティー」アイコンを選択して変更できます。「非アクティブ」を選択すると、その組織にログインした場合、ユーザーアクセスが無効になります。 |
| 「組織のエイリアス」 | このフィールドでは、組織のエイリアス名を指定します。URL ログインで、認証にエイリアスを使用できるようになります。たとえば `exampleorg` という組織があり、エイリアスとして `123` および `abc` を指定すると、次の URL を使用して組織にログインできます。<br><br>`http://machine.example.com/amserver/UI/Login?org=exampleorg`<br><br>`http://machine.example.com/amserver/UI/Login?org=abc`<br><br>`http://machine.example.com/amserver/UI/Login?org=123`<br><br>組織のエイリアス名は、組織全体で一意でなければなりません。「一意の属性リスト」を使用して一意性を実現できます。 |

「DNS エイリアス名」

組織の DNS 名にエイリアス名を追加できます。この属性では、実際のドメインエイリアスだけを使用できます。ランダムな文字列は使用できません。たとえば example.com という DNS があり、exampleorg という組織のエイリアスとして example1.com および example2.com を指定すると、次の URL を使用して組織にログインできます。

```
http://machine.example.com/amserver/UI/

Login?org=exampleorg

http://machine.example1.com/amserver/

UI/Login?org=exampleorg

http://machine.example2.com/amserver/

UI/Login?org=exampleorg
```

「一意の属性リスト」

組織内のユーザー用に一意の属性名リストを追加できます。たとえば、電子メールアドレスを示す一意の属性名を追加した場合、同一の電子メールアドレスを持つ 2 人のユーザーを作成できなくなります。このフィールドには、コンマ区切りのリストも指定できます。リスト内の属性名は、どれも一意性を定義します。たとえば、このフィールドに次の属性名リストが指定されたとします。

```
PreferredDomain, AssociatedDomain
```

また、特定のユーザーに対して、PreferredDomain は http://www.example.com と定義されています。この場合、コンマ区切りのリスト全体が、その URL に関して一意であると定義されます。「一意の属性リスト」にネーミング属性「ou」を追加しても、デフォルトの groups、people コンテナでは一意性は要求されません。(ou=Groups、ou=People)。

すべてのサブ組織で一意性が要求されます。

---

注 – 一意の属性はレルムモードには設定できません。また、一意の属性は、7.0 または 7.1 ベースのコンソールで旧バージョンモードに設定することもできません。一意の属性リストを作成するには、6.3 ベースのコンソールにログインする必要があります。詳細は、21 ページの「旧バージョンモード 6.3 コンソール」を参照してください。

---

4 「了解」をクリックします。
新しい組織が「組織」リストに表示されます。組織の作成時に定義したプロパティーを編集するには、編集対象の組織の名前をクリックし、プロパティーを変更して「保存」をクリックします。

### ▼ 組織を削除する

1 削除する組織名の横にあるチェックボックスを選択します。

2 「削除」をクリックします。

---

注 – 削除を実行するときに警告メッセージは表示されません。組織内のエントリがすべて削除されます。この操作を元に戻すことはできません。

---

## ポリシーに組織を追加する

Access Manager オブジェクトは、ポリシーの対象定義を通じてポリシーに追加されます。ポリシーを作成または修正するときに、組織、ロール、グループ、ユーザーを対象として定義できます。対象を定義すると、ポリシーがオブジェクトに適用されます。詳細は、124 ページの「ポリシーの管理」を参照してください。

# コンテナ

コンテナエントリは、オブジェクトクラスおよび属性が異なるために組織エントリが使用できない場合に使用します。Access Manager コンテナエントリと Access Manager 組織エントリは、必ずしも LDAP オブジェクトクラス `organizationalUnit` および `organization` と同等とはかぎらないことに留意してください。これらは抽象アイデンティティーエントリです。可能であれば、コンテナエントリではなく組織エントリを使用します。

---

注 – コンテナの表示は必要に応じて行います。コンテナを表示するには、「設定」>「コンソールプロパティー」の「管理」サービスでコンテナを表示するようにオプションを選択します。

---

### ▼ コンテナを作成する

1 新しいコンテナを作成する組織またはコンテナのリンクを選択します。

2 「コンテナ」タブをクリックします。

3 「コンテナ」リストで「新規」をクリックします。

4 作成するコンテナの名前を入力します。

5 「了解」をクリックします。

### ▼ コンテナを削除する

1 「コンテナ」タブをクリックします。

2 削除するコンテナ名の横にあるチェックボックスを選択します。

3 「削除」をクリックします。

---

注– コンテナを削除すると、そのコンテナに含まれるオブジェクトがすべて削除されます。すべてのオブジェクトとサブコンテナが対象になります。

---

## グループコンテナ

グループコンテナを使用してグループを管理します。グループコンテナにはグループとほかのグループコンテナだけを含めることができます。グループコンテナの「グループ」は、すべての管理されているグループの親エントリとしてダイナミックに割り当てられます。必要に応じて、グループコンテナを追加することができます。

---

注– グループコンテナの表示は必要に応じて行います。グループコンテナを表示するには、「設定」>「コンソールプロパティー」の「管理」サービスでグループコンテナを有効にするようにオプションを選択します。

---

### ▼ グループコンテナを作成する

1 新しいグループコンテナを追加する組織またはグループコンテナのリンクを選択します。

2 「グループコンテナ」タブを選択します。

3 「グループコンテナ」リストで「新規」をクリックします。

4 「名前」フィールドに値を入力して、「了解」をクリックします。「グループコンテナ」リストに新しいグループコンテナが表示されます。

### ▼ グループコンテナを削除する

1 削除対象のグループコンテナを含む組織に移動します。

2 「グループコンテナ」タブを選択します。

3 削除するグループコンテナの横にあるチェックボックスを選択します。

4 「削除」をクリックします。

## グループ

グループは、共通の機能、特徴、または関心事を持つユーザーの集まりを表します。通常、このグループには関連付けられた権限はありません。グループは、組織内および管理されているほかのグループ内という、2 つのレベルに存在できます。ほかのグループ内に存在するグループは、サブグループと呼ばれます。サブグループは、親グループ内に「物理的に」存在する子ノードです。

Access Manager は、入れ子グループもサポートします。入れ子グループは、1 つのグループに含まれる既存のグループを表します。サブグループとは対照的に、入れ子グループは DIT 内のどこにでも存在できます。入れ子グループは、多数のユーザーに対するアクセス権の設定を簡単にします。

作成できるグループには、静的グループと動的グループの 2 種類があります。ユーザーを手動で追加できるのは静的グループのみです。動的グループでは、フィルタによってユーザーの追加が制御されます。入れ子グループまたはサブグループは、両方に追加できます。

静的グループ

静的グループは、指定した管理されているグループタイプに基づいて作成されます。グループメンバーは、`groupOfNames` または `groupOfUniqueNames` オブジェクトクラスを使用してグループエントリに追加されます。

---

注 – 管理されているグループタイプのデフォルトは dynamic です。このデフォルトは、管理サービス設定で変更できます。

---

動的グループ

動的グループは、LDAP フィルタを使用して作成されます。エントリはすべてフィルタを通してまとめられ、グループに動的に割り当てられます。フィルタはエントリ

の属性を検索して、その属性を含むエントリを返します。たとえば、建物番号に基づいてグループを作成する場合、フィルタを使用すると建物番号属性を含むすべてのユーザーのリストを返します。

---

注 – Access Manager は、Directory Server とともに参照整合性プラグインを使用するように設定されている必要があります。参照整合性プラグインが有効になっているときは、削除操作や名前の変更操作の直後に、指定された属性について整合性更新が実行されます。これにより、関連するエントリ間の関係がデータベース全体で維持されます。Directory Server では、データベースインデックスによって検索パフォーマンスが向上します。このプラグインを有効にする方法の詳細は、『Sun Java System Access Manager 6 Migration Guide』を参照してください。

---

## ▼ 静的グループを作成する

1. 新しいグループを作成する組織、グループ、またはグループコンテナに移動します。
2. 「グループ」リストから「新規静的」をクリックします。
3. 「グループ名」フィールドにグループの名前を入力します。「次へ」をクリックします。
4. 「ユーザーのグループ加入を有効」属性を選択すると、ユーザーが自分でそのグループに加入できるようになります。
5. 「了解」をクリックします。
   グループの作成後、グループの名前を選択して「一般」をクリックすることにより、「ユーザーのグループ加入を有効」属性を編集できます。

## ▼ 静的グループのメンバーを追加または削除する

1. 「グループ」リストから、メンバーを追加するグループを選択します。
2. 「アクションの選択」メニューで実行するアクションを選択します。実行できるアクションは次のとおりです。

   「新規ユーザー」　このアクションでは、新規ユーザーが作成され、ユーザー情報の保存時にユーザーがグループに追加されます。

   「ユーザーの追加」　このアクションでは、既存のユーザーがグループに追加されます。このアクションを選択する場合は、追加するユーザーを指定する検索条件を作成します。条件の作成に使用する

フィールドでは、「いずれか」または「すべて」演算子を使用します。「すべて」は、指定したすべてのフィールドに一致するユーザーを返します。「いずれか」は、指定したいずれか 1 つのフィールドに一致するユーザーを返します。フィールドを空白のままにすると、その属性に関してはすべてのエントリが一致したとみなされます。

検索条件を作成したら、「次へ」をクリックします。返されたユーザーのリストから、追加するユーザーを選択し、「終了」をクリックします。

「グループを追加」　このアクションでは、入れ子グループが現在のグループに追加されます。このアクションを選択すると、検索範囲、グループの名前 ("*" ワイルドカードを使用可能) を含む検索条件を作成し、ユーザーが自分でグループに加入できるかどうかを指定できます。情報を入力したら、「次へ」をクリックします。返されたグループのリストから、追加するグループを選択し、「終了」をクリックします。

「メンバーを消去」　このアクションでは、グループからメンバー (ユーザーとグループを含む) が消去されますが、メンバー自身の削除はされません。消去するメンバーを選択し、「アクションの選択」メニューから「メンバーを消去」を選択します。

「メンバーを削除」　このアクションでは、選択されたメンバー自身のエントリーが Directory Server から物理的に削除されます。削除するメンバーを選択し、「メンバーを削除」を選択します。

## ▼ 動的グループを作成する

1. 新しいグループを作成する組織またはグループに移動します。
2. 「グループ」タブをクリックします。
3. 「新規動的」をクリックします。
4. 「グループ名」フィールドにグループの名前を入力します。
5. **LDAP** 検索フィルタを作成します。

   デフォルトでは、Access Manager は基本検索フィルタインタフェースを表示します。フィルタの作成に使用する基本フィールドでは、「いずれか」または「すべて」演算子を使用します。「すべて」は、指定したすべてのフィールドに一致するユーザーを返します。「いずれか」は、指定したいずれか 1 つのフィールドに一致する

ユーザーを返します。フィールドを空白のままにすると、その属性に関してはすべてのエントリが一致したとみなされます。

6 「了解」をクリックすると、検索条件に一致するすべてのユーザーが自動的にグループに追加されます。

### ▼ 動的グループのメンバーを追加または削除する

1 「グループ」リストで、メンバーを追加するグループの名前をクリックします。

2 「アクションの選択」メニューで実行するアクションを選択します。実行できるアクションは次のとおりです。

| | |
|---|---|
| 「グループを追加」 | このアクションでは、入れ子グループが現在のグループに追加されます。このアクションを選択すると、検索範囲、グループの名前 ("*" ワイルドカードを使用可能) を含む検索条件を作成し、ユーザーが自分でグループに加入できるかどうかを指定できます。情報を入力したら、「次へ」をクリックします。返されたグループのリストから、追加するグループを選択し、「終了」をクリックします。 |
| 「メンバーを消去」 | このアクションでは、グループからメンバー (グループを含む) が消去されますが、メンバー自身の削除はされません。消去するメンバーを選択し、「メンバーを消去」を選択します。 |
| 「メンバーを削除」 | このアクションでは、選択されたメンバー自身のエントリーが Directory Server から物理的に削除されます。削除するメンバーを選択し、「メンバーを削除」を選択します。 |

### ポリシーにグループを追加する

Access Manager オブジェクトは、ポリシーの対象定義を通じてポリシーに追加されます。ポリシーを作成または修正するときに、ポリシーの「対象」ページで、組織、ロール、グループ、ユーザーを対象として定義できます。対象を定義すると、ポリシーがオブジェクトに適用されます。詳細は、124 ページの「ポリシーの管理」を参照してください。

## ピープルコンテナ

ピープルコンテナはデフォルトの LDAP 組織単位です。ユーザーはすべて、組織内で作成されるときにその組織単位に割り当てられます。ピープルコンテナは組織レベルで、あるいはサブピープルコンテナとしてピープルコンテナレベルで表示され

ます。ピープルコンテナにはほかのピープルコンテナとユーザーだけを含めることができます。必要に応じて、ピープルコンテナを組織に追加することができます。

---

注 – ピープルコンテナの表示は必要に応じて行います。ピープルコンテナを表示するには、管理サービスで「ピープルコンテナを有効」を選択します

---

### ▼ ピープルコンテナを作成する

1. ピープルコンテナを作成する組織またはピープルコンテナに移動します。
2. 「ピープルコンテナ」リストで「新規」をクリックします。
3. 作成するピープルコンテナの名前を入力します。
4. 「了解」をクリックします。

### ▼ ピープルコンテナを削除する

1. 削除対象のピープルコンテナを含む組織またはピープルコンテナに移動します。
2. 削除するピープルコンテナ名の横にあるチェックボックスを選択します。
3. 「削除」をクリックします。

---

注 – ピープルコンテナを削除すると、そのピープルコンテナに含まれるオブジェクトがすべて削除されます。すべてのユーザーとサブピープルコンテナが対象になります。

---

## ユーザー

ユーザーは、個人のアイデンティティーを表します。Access Manager のアイデンティティー管理モジュールを使用して、組織、コンテナ、およびグループに対するユーザーの作成と削除、ロールやグループに対するユーザーの追加と削除が可能です。サービスをユーザーに割り当てることもできます。

注 – amadmin と同じユーザー ID でサブ組織のユーザーを作成すると、amadmin のログインが失敗します。このような問題が起こったら、管理者はディレクトリサーバーコンソールを使って、そのユーザーの ID を変更する必要があります。これで、管理者がデフォルトの組織にログインできるようになります。さらに、認証サービスの「ユーザー検索の開始 DN」をピープルコンテナ DN に設定し、ログイン処理で一意のユーザーが特定できます。

## ▼ ユーザーを作成する

1 ユーザーを作成する組織、コンテナ、またはピープルコンテナに移動します。

2 「ユーザー」タブをクリックします。

3 「ユーザー」リストで「新規」をクリックします。

4 次の値のデータを入力します。

| | |
|---|---|
| 「ユーザー ID」 | このフィールドには、ユーザーが Access Manager へログインするために使用する名前を指定します。このプロパティーは、DN 以外の値になることがあります。 |
| 「名」 | このフィールドには、ユーザーの名 (ファーストネーム) を指定します。名 (ファーストネーム) の値と姓 (ラストネーム) の値によって、「現在ログイン中」フィールドのユーザーが識別されます。これは必須の値ではありません。 |
| 「姓」 | このフィールドには、ユーザーの姓 (ラストネーム) を指定します。名 ( ファーストネーム) の値と姓 ( ラストネーム) の値によってユーザーが識別されます。 |
| 「フルネーム」 | このフィールドには、ユーザーのフルネームを指定します。 |
| パスワード | このフィールドには、「ユーザー ID」フィールドで指定した名前のパスワードを指定します。 |
| 「パスワード (確認)」 | パスワードを確認します。 |
| 「ユーザー状態」 | このオプションは、Access Manager による認証をユーザーに許可するかどうかを指定します。アクティブなユーザーだけが認証を受けることができます。デフォルト値は「アクティブ」です。 |

5 「了解」をクリックします。

## ▼ ユーザープロファイルを編集する

管理者ロールを割り当てられていないユーザーが Access Manager に認証されると、そのユーザーのユーザープロファイルがデフォルトの表示になります。また、適切な権限を持つ管理者はユーザープロファイルを編集できます。この表示では、ユーザーが各自の個人プロファイルに特有の属性値を編集できます。ユーザープロファイルビューに表示された属性は、展開することができます。オブジェクトおよびアイデンティティーのためにカスタマイズされた属性の追加については、『Access Manager Developer's Guide』を参照してください。

1 プロファイルが編集されるユーザーを選択します。デフォルトでは、「一般」表示となっています。

2 次のフィールドを編集します。

「名」
このフィールドには、ユーザーの名 (ファーストネーム) を指定します。

「姓」
このフィールドには、ユーザーの姓 (ラストネーム) を指定します。

「フルネーム」
このフィールドには、ユーザーのフルネームを指定します。

パスワード
「編集」リンクをクリックして、ユーザーのパスワードを追加し、確認します。

「電子メールアドレス」
このフィールドには、ユーザーの電子メールアドレスを指定します。

「社員番号」
このフィールドには、ユーザーの社員番号を指定します。

「電話番号」
このフィールドには、ユーザーの電話番号を指定します。

「ホームアドレス」
このフィールドには、ユーザーのホームアドレスを指定します。

「ユーザー状態」
このオプションは、Access Manager による認証をユーザーに許可するかどうかを指定します。アクティブなユーザーだけが Access Manager を使用して認証を受けることができます。デフォルト値は「アクティブ」です。プルダウンメニューから次のどちらかを選択することができます。

- 「アクティブ」— ユーザーは Access Manager を使用して認証を受けることができます。
- 「非アクティブ」— ユーザーは Access Manager を使用して認証を受けることはできませんが、ユーザープロファイルはそのままディレクトリに格納されます。

注 – ユーザー状態を「非アクティブ」に変えても、Access Manager による認証に影響するだけです。Directory Server は、*nsAccountLock* 属性を使用してユーザーアカウント状態を判別しますが、Access Manager での「ユーザー状態」の設定は、nsAccountLock 属性には影響しません。Access Manager にて、ユーザー状態を「非アクティブ」にしたユーザーアカウントでも、Access Manager を必要としないタスクは実行できます。Access Manager 認証だけではなく、ディレクトリのユーザーアカウントも無効にするには、*nsAccountLock* の値を「false」に設定します。サイトの委託管理者がユーザーを定期的に無効にしている場合は、*nsAccountLock* 属性を Access Manager のユーザープロファイルページに追加することを検討してください。詳細は、『Sun Java System Access Manager 7.1 Developer's Guide』を参照してください。

「アカウント有効期限」
この属性が存在し、その値が現在の日時以前であれば、認証サービスはログインを無効にします。この属性の形式は次のとおりです。*mm/dd/yyyy hh:mm*

「ユーザー認証設定」
この属性は、ユーザーの認証連鎖を設定します。

「ユーザーエイリアスリスト」
このフィールドは、ユーザーに適用される可能性のあるエイリアスを定義します。この属性に設定されたエイリアスを使用するために、`iplanet-am-user-alias-list` 属性を LDAP サービスのユーザーエントリ検索属性フィールドに追加して、LDAP サービスを修正する必要があります。

「設定ロケール」
このフィールドは、ユーザーのロケールを指定します。

「成功 URL」
この属性は、認証が成功した場合にユーザーをリダイレクトする URL を指定します。

「失敗 URL」
この属性は、認証が失敗した場合にユーザーをリダイレクトする URL を指定します。

「パスワードリセットオプション」
ここでは、パスワードを忘れた場合に使用する質問を選択します。パスワードを忘れたときは、選択した質問に答えることで、パスワードを回復できます。

「ユーザーディスカバリのリソースオファリング」
ユーザーの、ユーザーディスカバリサービスのリソースオファリングを設定します。

「MSISDN 番号」
MSISDN 認証を使用している場合に、ユーザーの MSISDN 番号を定義します。

### ▼ ロールおよびグループにユーザーを追加する

1 「ユーザー」タブをクリックします。

2 変更するユーザーの名前をクリックします。

3 「ロール」タブまたは「グループ」タブを選択します。

4 ユーザーを追加するロールまたはグループを選択し、「追加」をクリックします。

5 「保存」をクリックします。

---

注 – ロールまたはグループからユーザーを削除するには、ロールまたはグループを選択し、「削除」をクリックして「保存」をクリックします。

---

## ポリシーにユーザーを追加する

Access Manager オブジェクトは、ポリシーの対象定義を通じてポリシーに追加されます。ポリシーを作成または修正するときに、ポリシーの「対象」ページで、組織、ロール、グループ、ユーザーを対象として定義できます。対象を定義すると、ポリシーがオブジェクトに適用されます。詳細は、124 ページの「ポリシーの管理」を参照してください。

# ロール

ロールとは、グループの概念に似た、Directory Server の 1 つのエントリメカニズムです。グループにはメンバーがあるように、ロールにもメンバーがあります。ロールのメンバーは、ロールを持つ LDAP エントリです。ロール自体の基準は、属性を持つ LDAP エントリとして定義されます。このエントリは、エントリの識別名 (DN) 属性で特定されます。Directory Server にはさまざまなタイプのロールがありますが、Access Manager で管理できるのは、管理ロールだけです。

---

注 – そのほかの Directory Server ロールタイプもディレクトリの配備で使用できますが、Access Manager コンソールで管理することはできません。ポリシーの対象定義にほかの Directory Server タイプを使用することもできます。ポリシー対象については、116 ページの「ポリシーの作成」を参照してください。

---

ユーザーには 1 つ以上のロールを持たせることができます。たとえば、セッションサービスとパスワードリセットサービスの属性を持つ契約社員ロールを作成できます。新しい契約社員が入社したときに、管理者は契約社員エントリの別の属性を設定しなくても、契約社員にこのロールを割り当てることができます。契約社員がエ

ンジニアリング部門に属し、エンジニアリング社員に適用可能なサービスとアクセス権が必要な場合は、管理者は契約社員にエンジニアリングロールおよび契約社員ロールを割り当てることができます。

Access Manager では、ロールを使用して、アクセス制御の命令を適用します。Access Manager をはじめてインストールしたときに、管理者アクセス権を定義するアクセス制御命令 (ACI) が定義されます。次にこれらの ACI をロール (組織管理者ロール、組織ヘルプデスク管理者ロールなど) に割り当てます。このロールをユーザーに割り当てると、ユーザーのアクセス権が定義されます。

ユーザーは、管理サービスで「ユーザープロファイルページにロールを表示」属性が有効である場合だけ、割り当てられたロールを確認できます。

---

注 – Access Manager は、Directory Server とともに参照整合性プラグインを使用するように設定されている必要があります。参照整合性プラグインが有効になっているときは、削除操作や名前の変更操作の直後に、指定された属性について整合性更新が実行されます。これにより、関連するエントリどうしの関係がデータベース全体で維持されます。Directory Server では、データベースインデックスによって検索パフォーマンスが向上します。

---

ロールには、次の 2 種類があります。

- 静的ロール — 静的ロールは、ロールの作成時にユーザーを追加せずに作成されます。ロールを作成したあとで、特定のユーザーをそのロールに追加できます。これにより、ユーザーを指定されたロールに追加するときにより細かく制御できます。
- 動的ロール – 動的ロールは、LDAP フィルタを使用して作成されます。ユーザーはすべてフィルタを通してまとめられ、ロールの作成時にそのロールに割り当てられます。フィルタはエントリの属性と値のペア (`ca=user*` など) を検索して、その属性を含むユーザーをロールに自動的に割り当てます。

## ▼ 静的ロールを作成する

1. ロールを作成する組織に移動します。
2. 「ロール」タブをクリックします。

   組織の設定時にデフォルトのロールが作成され、「ロール」リストに表示されます。デフォルトのロールは次のとおりです。

   コンテナヘルプデスク管理者: コンテナのヘルプデスク管理者ロールは、組織単位のすべてのエントリに対する読み取りアクセス権、およびそのコンテナ単位だけにあるユーザーエントリの `userPassword` 属性に対する書き込みアクセス権を持っています。

組織ヘルプデスク管理者: 組織ヘルプデスク管理者は、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 userPassword 属性に対する書き込みアクセス権を持っています。

---

注 – サブ組織を作成するときは、管理者ロールは親組織ではなくサブ組織に作成してください。

---

コンテナ管理者: コンテナ管理者ロールは、LDAP 組織単位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。Access Manager では、LDAP 組織単位をコンテナと呼ぶことがあります。

組織ポリシー管理者: 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おり、組織内のすべてのポリシーについて作成、割り当て、修正、および削除ができます。

ピープル管理者: デフォルトで、新規に作成した組織のユーザーエントリはその組織のメンバーです。ピープル管理者は、組織のすべてのユーザー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。なお、このロールは、ロールおよびグループ DN を含む属性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ないため、ロールまたはグループの属性を変更したり、ロールまたはグループからユーザーを消去したりすることができません。

---

注 – ほかのコンテナは、Access Manager とともに設定して、ユーザーエントリ、グループエントリ、またはほかのコンテナを保持することができます。組織を構成したあとで、作成されたコンテナに管理者ロールを適用するには、デフォルトのコンテナ管理者ロールまたはコンテナヘルプデスク管理者を使用します。

---

グループ管理者: グループ作成時に作成されたグループ管理者は、特定グループのすべてのメンバーに対する読み取りアクセス権および書き込みアクセス権を持っており、新しいユーザーの作成、管理しているグループへのユーザーの割り当て、および作成したユーザーの削除を行うことができます。

グループを作成すると、そのグループを管理するのに必要な権限を持つグループ管理者ロールが自動的に作成されます。このロールはグループのメンバーに自動的には割り当てられません。グループの作成者、またはグループ管理者ロールへのアクセス権を持つ人が割り当てる必要があります。

最上位レベル管理者: 最上位レベル管理者は、最上位レベル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。言い換えれば、最上位レベル管理者ロールには、Access Manager アプリケーション内のすべての設定主体に対する権限があります。

組織管理者: 組織管理者は、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。組織を作成すると、その組織を管理するのに必要な権限を持つ組織管理者ロールが自動的に作成されます。

3 「新規静的」ボタンをクリックします。

4 ロールの名前を入力します。

5 ロールの詳細を入力します。

6 「タイプ」メニューからロールのタイプを選択します。

ロールは、管理者ロールまたはサービスロールにすることができます。ロールのタイプは、Access Manager コンソールが、コンソール内でどのユーザーをどの位置から始動させるかを決定するために使用します。管理者ロールは、ロールの所有者が管理者権限を持っていることをコンソールに通知します。サービスロールは、その所有者がエンドユーザーであることをコンソールに通知します。

7 「アクセス権」メニューから、ロールに適用する権限のデフォルトセットを選択します。これは、組織内のエントリにアクセスする権限です。ここで示すデフォルトの権限は順不同です。権限は次のとおりです。

| | |
|---|---|
| アクセス権なし | ロールにアクセス権が設定されません。 |
| 組織管理者 | 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。 |
| 組織ヘルプデスク管理者 | 組織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 `userPassword` 属性に対する書き込みアクセス権を持っています。 |
| 組織ポリシー管理者 | 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。組織のポリシー管理者は、ピア組織に対する参照ポリシーを作成できません。<br><br>一般に、「アクセス権なし」ACI をサービスロールに割り当て、管理者ロールにはデフォルト ACI のいずれかを割り当てます。 |

## ▼ 静的ロールにユーザーを追加する

1 ユーザーを追加するロールの名前をクリックします。

2 「メンバー」リストで、「アクションの選択」メニューから「ユーザーの追加」を選択します。

3 検索条件を入力します。表示される 1 つ以上のフィールドを基に、ユーザーの検索方法を選択できます。フィールドは次のとおりです。

| | |
|---|---|
| 「一致」 | フィルタ用として含めるフィールドを選択できます。「すべて」は、指定したすべてのフィールドに一致するユーザーを返します。「いずれか」は、指定したいずれか 1 つのフィールドに一致するユーザーを返します。 |
| 「名」 | 名 (ファーストネーム) でユーザーを検索します。 |
| 「ユーザー ID」 | ユーザー ID でユーザーを検索します。 |
| 「姓」 | 姓 (ラストネーム) でユーザーを検索します。 |
| 「フルネーム」 | フルネームでユーザーを検索します。 |
| 「ユーザー状態」 | ユーザーの状態 (アクティブまたは非アクティブ) でユーザーを検索します。 |

4 「次へ」をクリックすると、検索が始まります。検索結果が表示されます。

5 ユーザー名の横にあるチェックボックスを選択して、返された名前の中からユーザーを選択します。

6 「終了」をクリックします。
これで、ユーザーがロールに割り当てられます。

## ▼ 動的ロールを作成する

1 ロールを作成する組織に移動します。

2 「ロール」タブをクリックします。
組織の設定時にデフォルトのロールが作成され、「ロール」リストに表示されます。デフォルトのロールは次のとおりです。

コンテナヘルプデスク管理者: コンテナのヘルプデスク管理者ロールは、組織単位のすべてのエントリに対する読み取りアクセス権、およびそのコンテナ単位だけにあるユーザーエントリの `userPassword` 属性に対する書き込みアクセス権を持っています。

組織ヘルプデスク管理者: 組織ヘルプデスク管理者は、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 `userPassword` 属性に対する書き込みアクセス権を持っています。

---

注 – サブ組織を作成するときは、管理者ロールは親組織ではなくサブ組織に作成してください。

---

コンテナ管理者: コンテナ管理者ロールは、LDAP 組織単位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。Access Manager では、LDAP 組織単位をコンテナと呼ぶことがあります。

組織ポリシー管理者: 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おり、組織内のすべてのポリシーについて作成、割り当て、修正、および削除ができます。

ピープル管理者: デフォルトで、新規に作成した組織のユーザーエントリはその組織のメンバーです。ピープル管理者は、組織のすべてのユーザー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。なお、このロールは、ロールおよびグループ DN を含む属性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ないため、ロールまたはグループの属性を変更したり、ロールまたはグループからユーザーを消去したりすることができません。

---

注 – ほかのコンテナは、Access Manager とともに設定して、ユーザーエントリ、グループエントリ、またはほかのコンテナを保持することができます。組織を構成したあとで、作成されたコンテナに管理者ロールを適用するには、デフォルトのコンテナ管理者ロールまたはコンテナヘルプデスク管理者を使用します。

---

グループ管理者: グループ作成時に作成されたグループ管理者は、特定グループのすべてのメンバーに対する読み取りアクセス権および書き込みアクセス権を持っており、新しいユーザーの作成、管理しているグループへのユーザーの割り当て、および作成したユーザーの削除を行うことができます。

グループを作成すると、そのグループを管理するのに必要な権限を持つグループ管理者ロールが自動的に作成されます。このロールはグループのメンバーに自動的には割り当てられません。グループの作成者、またはグループ管理者ロールへのアクセス権を持つ人が割り当てる必要があります。

最上位レベル管理者: 最上位レベル管理者は、最上位レベル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。言い換えれば、最上位レベル管理者ロールには、Access Manager アプリケーション内のすべての設定主体に対する権限があります。

組織管理者: 組織管理者は、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。組織を作成すると、その組織を管理するのに必要な権限を持つ組織管理者ロールが自動的に作成されます。

3 「新規動的」ボタンをクリックします。

4 ロールの名前を入力します。

5 ロールの詳細を入力します。

6 「タイプ」メニューからロールのタイプを選択します。

ロールは、管理者ロールまたはサービスロールにすることができます。ロールのタイプは、Access Manager コンソール内でどのユーザーをどの位置から始動させるかを決定するために使用します。管理者ロールは、ロールの所有者が管理者権限を持っていることをコンソールに通知します。サービスロールは、その所有者がエンドユーザーであることをコンソールに通知します。

7 「アクセス権」メニューから、ロールに適用する権限のデフォルトセットを選択します。これは、組織内のエントリにアクセスする権限です。ここで示すデフォルトの権限は順不同です。権限は次のとおりです。

| | |
|---|---|
| アクセス権なし | ロールにアクセス権が設定されません。 |
| 組織管理者 | 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。 |
| 組織ヘルプデスク管理者 | 組織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 `userPassword` 属性に対する書き込みアクセス権を持っています。 |
| 組織ポリシー管理者 | 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。組織のポリシー管理者は、ピア組織に対する参照ポリシーを作成できません。<br><br>一般に、「アクセス権なし」ACI をサービスロールに割り当て、管理者ロールにはデフォルト ACI のいずれかを割り当てます。 |

8 検索条件を入力します。フィールドは次のとおりです。

| | |
|---|---|
| 「一致」 | 演算子を含めるフィルタのフィールドに、演算子を含めることができます。「すべて」は、指定したすべてのフィールドに一致するユーザーを返します。「いずれか」は、指定したいずれか 1 つのフィールドに一致するユーザーを返します。 |
| 「名」 | 名 (ファーストネーム) でユーザーを検索します。 |
| 「ユーザー ID」 | ユーザー ID でユーザーを検索します。 |
| 「姓」 | 姓 (ラストネーム) でユーザーを検索します。 |
| 「フルネーム」 | フルネームでユーザーを検索します。 |
| 「ユーザー状態」 | ユーザーの状態 (アクティブまたは非アクティブ) でユーザーを検索します。 |

9 「了解」をクリックして、フィルタ条件を基に、検索を開始します。そのフィルタ条件で定義されたユーザーがロールに自動的に割り当てられます。

## ▼ ロールからユーザーを消去する

1 変更するロールを含む組織に移動します。
アイデンティティー管理モジュールで「表示」メニューから「組織」を選択し、「ロール」タブを選択します。

2 変更するロールを選択します。

3 「表示」メニューから「ユーザー」を選択します。

4 消去する各ユーザーの横にあるチェックボックスを選択します。

5 「アクションの選択」メニューから「ユーザーの消去」をクリックします。
これで、ロールからユーザーが消去されます。

## ポリシーにロールを追加する

Access Manager オブジェクトは、ポリシーの対象定義を通じてポリシーに追加されます。ポリシーを作成または修正するときに、ポリシーの「対象」ページで、組織、ロール、グループ、ユーザーを対象として定義できます。対象を定義すると、ポリシーがオブジェクトに適用されます。詳細は、124 ページの「ポリシーの管理」を参照してください。

◆ ◆ ◆ 第 8 章

# 8 現在のセッション

この章では、Access Manager のセッション管理機能について説明します。セッション管理モジュールでは、ユーザーセッションの情報を確認したり、ユーザーセッションを管理したりする手段を用意しています。さまざまなセッションの時間を追跡するほかに、管理者がセッションを終了することができます。システム管理者は、プラットフォームサーバーリストに表示されたロードバランササーバーを無視してください。

## 現在のセッションのインタフェース

「現在のセッション」モジュールインタフェースを使用すると、適切な権限を持った管理者は、Access Manager にログインしている任意のユーザーのセッション情報を参照できます。

### セッション管理

セッション管理フレームには、現在管理されている Access Manager の名前が表示されます。

### セッション情報

セッション情報ウィンドウには、Access Manager に現在ログイン中のすべてのユーザーと、各ユーザーのセッション時間が表示されます。表示フィールドは次のとおりです。

「ユーザー **ID**」: 現在ログイン中のユーザーのユーザー ID が表示されます。

「残り時間」: ユーザーの再認証までの、セッションの残り時間 (分単位) が表示されます。

「最大セッション時間」: ユーザーがログインした状態でいられる最大時間 (分単位) が表示されます。この時間が経過すると、セッションが期限切れになり、ユーザーはアクセスするために再度認証を受ける必要があります。

「アイドル時間」: ユーザーがアイドル状態になっている時間 (分単位) が表示されます。

「最大アイドル時間」: ユーザーの再認証が必要になるまでの残りの最大アイドル時間 (分単位) が表示されます。

時間の制限値は、管理者がセッション管理サービスに定義します。

「ユーザー ID」フィールドに入力して「フィルタ」をクリックすれば、特定のユーザーのセッションや特定の範囲のセッションを表示できます。ワイルドカードも使用できます。

「更新」ボタンをクリックすれば、セッションの表示が更新されます。

# セッションの終了

適切な権限を持った管理者は、ユーザーのセッションをいつでも終了させることができます。

## ▼ セッションを終了させる

1. 終了させるユーザーのセッションを選択します。
2. 「セッションを終了」をクリックします。

◆ ◆ ◆ 第 9 章

# パスワードリセットサービス

Access Manager では、Access Manager によって保護されている特定のサービスやアプリケーションにアクセスするためのパスワードをユーザー自身がリセットできるように、パスワードリセットサービスが用意されています。パスワードリセットサービス属性は、最上位レベル管理者が定義します。この属性を使用して、ユーザーを検証するための証明情報 (秘密の質問形式) を制御し、新規または既存のパスワード通知の機構を制御し、不正なユーザーに適用するロックアウト間隔を設定します。

この章の内容は次のとおりです。



## パスワードリセットサービスの登録

パスワードリセットサービスは、ユーザーが属しているレルムに対して登録する必要はありません。ユーザーが割り当てられている組織にパスワードリセットサービスが存在しない場合は、「設定」タブの「パスワードリセット」に定義されている値が継承されます。

### ▼ 別のレルムに存在するユーザーのパスワードリセットを登録する

1 そのユーザーのパスワードを登録するレルムに移動します。

2 レルム名をクリックし、「サービス」タブをクリックします。
選択したレルムにレルム名が追加されていない場合は、「追加」ボタンをクリックします。

3 「パスワードリセット」を選択し、「次へ」をクリックします。
パスワードリセットサービス属性が表示されます。属性の定義については、オンラインヘルプを参照してください。

4 「終了」をクリックします。

# パスワードリセットサービスの設定

パスワードリセットサービスの登録が完了したら、管理者権限を持っているユーザーがこのサービスを設定する必要があります。

## ▼ サービスを設定する

1 パスワードリセットサービスが登録されているレルムを選択します。

2 「サービス」タブをクリックします。

3 サービスリストから「パスワードリセット」をクリックします。

4 「パスワードリセット」属性が表示され、ここでパスワードリセットサービスの要件を定義できます。パスワードリセットサービスが有効になっていることを確認します (デフォルトでは有効)。少なくとも次の属性を定義する必要があります。

- ユーザー検証
  - 秘密の質問
  - バインド DN
  - バインドパスワード

    「バインド DN」属性には、パスワードをリセットする権限を持っているユーザー (ヘルプデスク管理者など) を指定する必要があります。Directory Server の制限により、バインド DN が `cn=Directory Manager` の場合はパスワードリセットは機能しません。

    残りの属性は省略可能です。これらのサービス属性の説明については、オンラインヘルプを参照してください。

注 – Access Manager では、ランダムなパスワードを生成するパスワードリセット Web アプリケーションが自動的にインストールされます。ただし、パスワードの生成や通知を行う独自のプラグインクラスを記述することもできます。このようなプラグインクラスのサンプルについては、次の場所にある Readme.html ファイルを参照してください。

PasswordGenerator:

```
AccessManager-base/SUNWam/samples/console/PasswordGenerator
```

NotifyPassword:

```
AccessManager-base/SUNWam/samples/console/NotifyPassword
```

5 独自の質問を定義するユーザーには、「個人的な質問を有効」属性を選択します。属性を定義し終えたら、「保存」をクリックします。

## ▼ 秘密の質問をローカライズする

ローカライズ版の Access Manager を実行していて、秘密の質問をロケールに固有の文字セットで表示する場合は、次の手順を実行します。

1 パスワードリセットサービスの「秘密の質問」属性の下の「現在の値」リストに秘密の質問キーを追加します。たとえば、`favorite-color` を追加するとします。

2 このキーとキーの値を表示するために必要な質問を `amPasswordReset.properties` ファイルに追加します。次に例を示します。

`favorite-color`=好きな色はなんですか。

3 同じキーとローカライズされた質問を `/opt/SUNWam/locale` にある `AMPasswordReset_`*locale*`.properties` に追加します。ユーザーが自分のパスワードの変更を試みると、ローカライズされた質問が表示されます。

# パスワードリセットのロックアウト

パスワードリセットサービスには、ユーザーが秘密の質問に正しく回答するまでの回数を制限するために、ロックアウト機能が用意されています。ロックアウト機能は、パスワードリセットサービス属性を使用して設定します。これらのサービス属性の説明については、オンラインヘルプを参照してください。パスワードリセットでは、メモリーロックアウトと物理的なロックアウトの 2 種類がサポートされています。

### メモリーロックアウト

これは一時的なロックアウトです。「パスワードリセット失敗のロックアウト持続時間」属性の値が 0 より大きく、「パスワードリセット失敗のロックアウトを有効」属性が有効になっている場合にのみ機能します。ロックアウトされたユーザーは、パスワードリセット Web アプリケーションを使用してもパスワードをリセットできなくなります。「パスワードリセット失敗のロックアウト持続時間」に指定した時間が経過するまで、またはサーバーが再起動されるまで、ロックアウトは持続します。これらのサービス属性の説明については、オンラインヘルプを参照してください。

### 物理的なロックアウト

メモリーロックアウトより永続的なロックアウトです。「パスワードリセット失敗のロックアウトカウント」属性の値が 0 に設定され、「パスワードリセット失敗のロックアウトを有効」属性が有効になっている場合には、秘密の質問に正しく回答できなかったユーザーのアカウント状態が非アクティブに変更されます。これらのサービス属性の説明については、オンラインヘルプを参照してください。

# エンドユーザーから見たパスワードリセット

以降の節では、ユーザーの観点からパスワードリセットサービスについて説明します。

## パスワードリセットのカスタマイズ

管理者がパスワードリセットサービスを有効にし、属性を定義したら、ユーザーは Access Manager コンソールにログインして秘密の質問をカスタマイズできます。

### ▼ パスワードリセットをカスタマイズする

1. ユーザー名とパスワードを入力して認証に成功したら、**Access Manager** コンソールにログインします。
2. 「ユーザープロファイル」ページで、「パスワードリセットのオプション」を選択します。「質問と回答」画面が表示されます。
3. 管理者が定義した、このサービスで選択できる質問が表示されます。たとえば、次のような質問が表示されます。
   - ペットの名前は何でしょうか？
   - 好きなテレビ番組は何でしょうか？

- 母親の旧姓は何でしょうか?
- よく行くレストランの名前は何でしょうか?

4 秘密の質問を選択します。管理者がこのレルムに定義した最大質問数(パスワードリセットサービスに定義)まで選択できます。選択した質問への回答を指定します。これらの質問と回答を元に、自分でパスワードをリセットすることができます(次の節を参照)。管理者が「個人的な質問を有効」属性を選択している場合は、自分だけの秘密の質問と回答を入力できるように、テキストフィールドが表示されます。

5 「保存」をクリックします。

# パスワードを忘れた場合のリセット

パスワードを忘れた場合には、パスワードリセット Web アプリケーションによって新しいパスワードがランダムに生成され、それがユーザーに通知されます。パスワードを忘れた場合の標準的な手順を次に示します。

## ▼ パスワードを忘れた場合にリセットする

1 管理者から渡された **URL** を使って、パスワードリセット **Web** アプリケーションにログインします。次に例を示します。

`http://hostname:port /ampassword` (デフォルトのレルムの場合)

または、

`http://hostname: port/`*`deploy_uri`*` /UI/PWResetUserValidation?realm=realmname`。realmname はレルムの名前です。

---

注 – パスワードリセットサービスがサブレルムで有効になっていても、親レルムで無効になっている場合は、次の構文を使ってサービスにアクセスする必要があります。

`http://hostname: port/`*`deploy_uri`*`/UI/PWResetUserValidation?realm=realmname`

---

2 ユーザー **ID** を入力します。

3 パスワードリセットサービスに定義されている質問のうち、パスワードリセット設定をカスタマイズした際にユーザーが選択した質問が表示されます。事前に「ユーザープロファイル」ページにログインしておらず、パスワードリセット設定をカスタマイズしていない場合には、パスワードは生成されません。

質問に正しく回答すると、新しいパスワードが生成され、電子メールで通知されます。また、回答が正しいかどうかにかかわらず、パスワードをリセットしようとし

たことが通知されます。新しいパスワードおよびパスワードをリセットしようとしたことの通知を受け取るには、「ユーザープロファイル」ページに電子メールアドレスを入力しておく必要があります。

# パスワードポリシー

パスワードポリシーは、特定のディレクトリ内でパスワードがどのように使用されるかを規定した規則の集合です。パスワードポリシーは Directory Server 内に、通常は Directory Server コンソールを使用して定義されます。安全なパスワードポリシーとして次のような条件をパスワードに適用すると、推測されやすいパスワードによるリスクを最小限に抑えることができます。

- ユーザーは定期的にパスワードを変更しなければならない。
- ユーザーは、容易に推測できないパスワードを指定しなければならない。
- 不正なパスワードを何度か使用すると、アカウントがロックされることがある。

Directory Server では、いくつかの方法により、ツリー内の任意のノードにパスワードポリシーを設定できます。詳細は、

『Directory Server Enterprise Edition 6.0 管理ガイド』の「Directory Server のパスワードポリシー」を参照してください。

---

注 – Directory Server では、パスワードポリシーに `passwordExp` 属性が含まれます。この属性は、一定の秒数が経過するとユーザーパスワードが期限切れになるかどうかを定義します。管理者が `passwordExp` 属性を `on` に設定すると、エンドユーザーパスワードの有効期限と、`amldap`、 `dsame`、`puser` などの Access Manager の管理アカウントの有効期限が設定されます。Access Manager 管理者アカウントのパスワードが期限切れになったときにエンドユーザーがログインしている場合は、そのユーザーにパスワード変更画面が表示されます。ただし、Access Manager では、そのパスワード変更画面に関係するユーザーを特定しません。この場合、このパスワード変更画面は管理者向けに表示されており、エンドユーザーはパスワードを変更できません。

この問題を解決するには、管理者が Directory Server にログインし、`amldap`、`dsame`、および `puser` の各パスワードを変更するか、または将来のために `passwordExpirationTime` 属性を変更します。

---

◆ ◆ ◆ 第 10 章

# 10 ログサービス

Sun Java™ System Access Manager には、ユーザーアクティビティー、トラフィックパターン、認証違反などの情報を記録するために、ログサービスが用意されています。また、管理者はデバッグファイルを利用して、インストールの障害追跡を行うことができます。

## ログファイル

ログファイルには、ログサービスが監視するイベントが記録されます。管理者は、ログファイルを定期的に確認することをお勧めします。ログファイルのデフォルトのディレクトリは、SPARC システムの場合は `/var/opt/SUNWam/logs` 、Linux システムの場合は `/var/opt/sun/identity`、HP-UX の場合は `/var/opt/sun/identity`、そして Windows の場合は *jes-install-dir*`\identity` です。ログサービスのログファイルディレクトリを設定するときには、Access Manager コンソールを使用します。

ログファイルのデフォルトタイプ、記録される情報、ログファイルの形式の詳細については、『Sun Java System Access Manager 7.1 Technical Overview』の「Logging Overview」を参照してください。

ログサービスの属性定義については、Access Manager コンソールの「ヘルプ」ボタンをクリックしてオンラインヘルプを参照してください。

## Access Manager サービスのログ

サービスログファイルには、アクセスログファイルとエラーログファイルの 2 種類があります。アクセスログファイルには、アクションが実行されたことと正常に実行されたアクションの結果が記録されます。エラーログファイルには、Access Manager サービスで発生したエラーが記録されます。フラットログファイルには、`.error` または `.access` という拡張子が付きます。データベース列名の最後には、Oracle データベースの場合は `_ERROR` または `_ACCESS`、MySQL データベースの場合は

_error または _access が付きます。たとえば、コンソールイベントを記録するフラットファイルログには amConsole.access という名前が付き、コンソールイベントを記録するデータベース列には AMCONSOLE_ACCESS という名前が付きます。以下の節では、ログサービスによって記録されるログファイルについて説明します。

## セッションログ

ログサービスでは、次のセッションサービスイベントが記録されます。

- ログイン
- ログアウト
- セッションのアイドルタイムアウト
- 最大セッション数によるタイムアウト
- ログインの失敗
- セッションの再起動
- セッションの破棄

セッションログのファイル名は、amSSO で始まります。

## コンソールログ

Access Manager コンソールログには、アイデンティティー関連のオブジェクト、ポリシー、およびサービスが作成、削除、および変更されたことが記録されます。たとえば、組織、組織単位、ユーザー、ロール、ポリシー、グループが記録されます。コンソールログには、パスワードなどのユーザー属性が変更されたことや、ロールとグループに対してユーザーが追加または削除されたことも記録されます。また、委託やデータストアのアクティビティーも記録されます。コンソールログのファイル名は、amConsole で始まります。

## 認証ログ

認証コンポーネントでは、ユーザーのログインおよびログアウトがログとして記録されます。認証ログのファイル名は、amAuthentication で始まります。

## 連携ログ

連携コンポーネントでは、連携関連のイベントなどがログとして記録されます。たとえば、認証ドメインが作成されたことや、ホストプロバイダが作成されたことが記録されます。連携ログのファイル名は、amFederation で始まります。

## ポリシーログ

ポリシーコンポーネントでは、ポリシー関連のイベントなどがログとして記録されます。たとえば、ポリシー管理 (ポリシーの作成、削除、および変更) やポリシー評価が記録されます。ポリシーログのファイル名は、amPolicy で始まります。

## エージェントログ

ポリシーエージェントログには、ユーザーがアクセスを許可または拒否されているログリソースに関する例外が記録されます。エージェントログのファイル名は、amAgent で始まります。amAgent ログは、エージェントサーバーだけに存在します。エージェントイベントのログは、Access Manager サーバー上の認証ログに記録されます。この機能の詳細については、ポリシーエージェントのマニュアルを参照してください。

## SAML ログ

SAML コンポーネントでは、SAML 関連のイベントなどが記録されます。たとえば、表明およびアーティファクトが作成または削除されたこと、応答および要求の詳細、SOAP エラーなどが記録されます。SAML ログのファイル名は、amSAML で始まります。

## amAdmin ログ

コマンド行ログには、コマンド行ツールを使用する操作中に発生したイベントエラーが記録されます。たとえば、サービススキーマがロードされたこと、ポリシーが作成されたこと、ユーザーが削除されたことなどが記録されます。コマンド行ログのファイル名は amAdmin. で始まります。 amadmin.access および amadmin.error の各ログファイルは、主要なログディレクトリのサブディレクトリに置かれます。デフォルトでは、amadmin コマンド行ツールのログファイルは /var/opt/SUNWam/logs に置かれます。

# ログ機能

ログサービスにはいくつかの特殊な機能が用意されていて、追加機能として有効にできます。このような機能として、「セキュリティー保護されたログを有効」、「コマンド行ログ」、および「リモートログ」があります。

# セキュリティー保護されたログ

このオプションの機能を追加すると、ログ機能のセキュリティーが強化されます。セキュリティー保護されたログを有効にすると、セキュリティーログに対する未承認の変更や改ざんを検出できます。この機能を使用するために、特にコーディングする必要はありません。セキュリティー保護されたログには、システム管理者が事前に登録した証明書が必要です。この MAC (Manifest Analysis and Certification) は、すべてのログレコードについて生成および格納されます。また、特別な「署名」ログレコードが定期的に挿入されます。このログレコードは、その時点までに書き込まれたログの内容に対する署名となります。これらの 2 つのレコードによって、ログが改ざんされていないことが保証されます。

## ▼ セキュリティー保護されたログを有効にする

1 `Logger` という名前の証明書を作成し、**Access Manager** を実行する配備コンテナにインストールします。配備コンテナの詳細については、マニュアルを参照してください。

2 **Access Manager** コンソールを使用して「ログサービス」設定の「セキュリティー保護されたログ」を有効にし、変更を保存します。管理者は、「ログサービス」のその他の属性のデフォルト値を変更することもできます。

ログディレクトリがデフォルトのディレクトリ (`/var/opt/SUMWam/logs`) から変更されている場合は、アクセス権が 0700 に設定されていることを確認してください。このディレクトリが存在しない場合には、ログサービスによって作成されますが、そのアクセス権は 0755 に設定されます。

また、デフォルトではないログディレクトリを指定した場合は、Web コンテナの `server.policy` ファイルで次のパラメータをその 新しいディレクトリに変更する必要があります。

```
permission java.io.FilePermission “/var/opt/SUNWam/logs/*”,”delete,write”
```

3 *AccessManager-base*`/SUNWam/config` ディレクトリに証明書データベースパスワードを含むファイルを作成し、`.wtpass` という名前を付けます。

---

注 – ファイル名とそのパスは、`AMConfig.properties` ファイルに設定できます。詳細については、『*Access Manager Administration Reference*』の AMConfig.properties ファイルリファレンスの「Certificate Database」を参照してください

セキュリティー上の理由から、このファイルへの読み取りアクセス権を持つ管理者だけが配備コンテナを使用できるようにしてください。

---

4 サーバーを再起動します。

このとき、セキュリティー保護されたログディレクトリをクリアすることをお勧めします。セキュリティー保護されたログを開始したときに、誤解を招きやすい検証エラーが /var/opt/SUNWam/debug/amLog ファイルに書き込まれることがあります。

セキュリティーログに対する未承認の変更や改ざんを検出するには、検証プロセスによって /var/opt/SUNWam/debug/amLog に書き込まれたエラーメッセージを探してください。改ざんを手動で確認する場合は、VerifyArchive ユーティリティーを実行します。詳細については、『*Access Manager Administration Reference*』を参照してください。

# コマンド行ログ

amadmin コマンド行ツールを使用して、Directory Server のアイデンティティーオブジェクト (組織、ユーザー、ロールなど) を作成、変更、および削除することができます。このツールを使用して、サービステンプレートをロード、作成、および登録することもできます。-t オプションを指定すれば、ログサービスでこれらのアクションを記録できます。AMConfig.properties の com.iplanet.am.logstatus プロパティーが有効 (ACTIVE) になっている場合は、ログレコードが作成されます。このプロパティーはデフォルトでは有効になっています。コマンド行ログのファイル名は、amAdmin で始まります。詳細については、『*Access Manager Administration Reference*』の「The amadmin Command Line Tool」を参照してください。

# ログプロパティー

AMConfig.properties ファイルには、ログ出力を制御するプロパティーが入っています。

| | |
|---|---|
| com.iplanet.am.logstatus=ACTIVE | このプロパティーを使用して、ログの有効または無効を切り替えます。デフォルトでは ACTIVE になっています。 |
| iplanet-am-logging.*service*.level= *level* | *service* にはログを記録するサービス名を指定します。この名前はそのままログファイルの名前になります。たとえば、amSAML.access にログレベルを指定するには、iplanet-am-logging.amSAML.access.level プロパティーを使用します。*level* は java.util.logging.Level の値のいずれかであり、記録されるログファイルの詳細レベルを表します。指定できるレベルは、OFF、SEVERE、WARNING、INFO、CONFIG、FINE、FINER、および ALL で |

す。ほとんどのサービスでは、INFO レベルより詳細なログは記録されません。

# リモートログ

Access Manager では、リモートログがサポートされます。これにより、Access Manager SDK がインストールされたホストを使用するクライアントアプリケーションが、リモートマシン上に配備された Access Manager インスタンス上にログレコードを作成できるようになります。リモートログは、次のいずれかの場合に開始されます。

1. ある Access Manager インスタンスのネームサービスのログ URL にリモートの Access Manager インスタンスの URL が指定されていて、2 つのインスタンスの間に信頼関係が設定されている場合に、リモート Access Manager インスタンスにログが記録されます。
2. Access Manager SDK がリモート Access Manager インスタンスにインストールされていて、SDK サーバーで実行されているクライアント (または Java クラス) がログ API を使用している場合に、リモート Access Manager マシンにログが記録されます。
3. ログ API が Access Manager エージェントで使用されているとき。

## ▼ リモートログを有効にする

**1** **Sun Java System Web Server** を使用する場合は、`server.xml` 設定ファイルに次の環境変数を設定する必要があります。

- `java.util.logging.manager=com.sun.identity.log.LogManager`
- `java.util.logging.config.file=/` *AccessManager-base* `/SUNwam/lib/LogConfig.properties`
- Java™ 2 Platform, Standard Edition の 1.4 以上を使用している場合には、コマンド行で次のコマンドを実行すると、リモートログが有効になります。

  `java -cp /`*AccessManager-base* `/SUNWam/lib/am_logging.jar:/` *AccessManager-base* `/SUNWam/lib/xercesImpl.jar:/` *AccessManager-base* `/SUNWam/lib/xmlParserAPIs.jar:/` *AccessManager-base* `/SUNWam/lib/jaas.jar:/` *AccessManager-base* `/SUNWam/lib/xmlParserAPIs.jar:/` *AccessManager-base* `/SUNWam/lib/servlet.jar:/` *AccessManager-base* `/SUNWam/locale:/` *AccessManager-base*`/SUNWam/lib/am_services.jar:/` *AccessManager-base*`/SUNWam/lib/am_sdk.jar:/` *AccessManager-base*`/SUNWam/lib/jss311.jar:/` *AccessManager-base*`/SUNWam/lib:.`

  `-Djava.util.logging.manager=com.sun.identity.log.LogManager`

-Djava.util.logging.config.file=/ *AccessManager-base* /SUNwam/lib/LogConfig.properties <logTestClass>

- 1.4 より前の Java 2 Platform, Standard Edition を使用している場合には、コマンド行で次のコマンドを実行すると、リモートログが有効になります。

  java -Xbootclasspath/a:/ *AccessManager-base* /SUNWam/lib/jdk_logging.jar -cp /*AccessManager-base* /SUNWam/lib/am_logging.jar:/ *AccessManager-base* /SUNWam/lib/xercesImpl.jar:/ *AccessManager-base* /SUNWam/lib/xmlParserAPIs.jar:/ *AccessManager-base* /SUNWam/lib/jaas.jar:/ *AccessManager-base* /SUNWam/lib/xmlParserAPIs.jar:/ *AccessManager-base* /SUNWam/lib/servlet.jar:/ *AccessManager-base* /SUNWam/locale:/ *AccessManager-base*/SUNWam/lib/am_services.jar:/ *AccessManager-base*/SUNWam/lib/am_sdk.jar:/ *AccessManager-base*/SUNWam/lib/jss311.jar:/ *AccessManager-base*/SUNWam/lib:.

  -Djava.util.logging.manager=com.sun.identity.log.LogManager

  -Djava.util.logging.config.file=/ *AccessManager-base* /SUNwam/lib/LogConfig.properties <logTestClass>

**2** *AccessManager-base*/SUNWam/lib の LogConfig.properties に次のパラメータが設定されていることを確認します。

- iplanet-am-logging-remote-handler=com.sun.identity.

  log.handlers.RemoteHandler

- iplanet-am-logging-remote-formatter=com.sun.

  identity.log.handlers.RemoteFormatter

- iplanet-am-logging-remote-buffer-size=1

  リモートログでは、ログレコード数に基づいてバッファー機能がサポートされます。この値には、レコード数でログバッファーサイズを定義します。バッファーがいっぱいになったら、バッファーに保管されたすべてのレコードがサーバーにフラッシュされます。

- iplanet-am-logging-buffer-time-in-seconds=3600

  この値には、ログバッファーとクリーナのスレッドを呼び出すときの、タイムアウト期間を定義します。

- iplanet-am-logging-time-buffering-status=OFF

  この値には、ログバッファー (およびバッファークリーナのスレッド) を有効にするかどうかを定義します。デフォルトでは、この機能は無効になっています。

注 – ログファイルが空の場合、セキュリティー保護されたログに「verification failure」と表示される可能性があります。これは、作成されたファイルの数がアーカイブサイズに等しい場合、セキュリティー保護されたログが、このセットからアーカイブし、再起動するからです。ほとんどの場合、このエラーは無視してもかまいません。レコード数がアーカイブサイズに等しくなると、このエラーは表示されなくなります。

3 プログラムをクライアント **SDK** とともに使用している場合は、**AMConfig.properties** ファイル内の次のプロパティーをそれに応じて設定する必要があります。

- com.iplanet.am.naming.url
- com.sun.identityagents.app.username
- com.iplanet.am.service.password
- com.iplanet.am.server.protocol
- com.iplanet.am.server.host
- com.iplanet.am.server.port

`/opt/SUNWam/war` ディレクトリの `README.clientsdk` で、クライアント SDK のサンプルを参照してください。ここには、`AMConfig.properties` ファイルおよび make ファイルが `/opt/SUNWam/war/clientsdk-samples` ディレクトリに生成される経緯が詳しく説明されています。これらのファイルは、サンプルの make ファイルのコンパイルおよび実行エントリによって順番に使用されます。

# エラーログとアクセスログ

Access Manager には 2 種類のログファイルが存在します。アクセスログファイルとエラーログファイルです。

アクセスログファイルには、Access Manager 配備に関する一般的な監査情報が記録されます。ログには、認証の成功など、特定のイベントに対する単一のレコードが含まれる場合があります。またその同じイベントに対する複数のレコードが含まれる場合もあります。たとえば、管理者がコンソールを使って特定の属性の値を変更した場合、ログサービスは、その変更の試みを 1 つのレコードとしてログに記録します。ログサービスはさらに、その変更の実行結果も、2 番目のレコードとしてログに記録します。

エラーログファイルには、アプリケーション内で発生したエラーが記録されます。ある処理のエラーはエラーログに記録されますが、その処理が試みられたことはアクセスログファイルに記録されます。

フラットログファイルには、`.error`、`.access` のいずれかの拡張子が付けられます。データベーステーブル名は、`_ERROR`、`_ACCESS` のいずれかで終わります。たとえば、コンソールイベントのログを記録するフラットファイルの名前が `amConsole.access`

である場合、その同じイベントのログを記録するデータベーステーブルの名前は、AMCONSOLE_ACCESS または amConsole_access になります。

次の表では、各 Access Manager コンポーネントが生成するログファイルについて簡単に説明します。

表 10–1 Access Manager コンポーネントのログ

| コンポーネント | ログファイル名のプレフィックス | ログに記録される情報 |
|---|---|---|
| セッション | amSSO | ログイン時刻、ログアウト時刻、タイムアウト制限などのセッション管理属性値。 |
| 管理コンソール | amConsole | アイデンティティー関連のオブジェクト、レルム、ポリシーの作成、削除、変更など、管理コンソール経由で実行されるユーザーアクション。 |
| 認証 | amAuthentication | ユーザーのログインとログアウト。 |
| アイデンティティー連携 | amFederation | 認証ドメインの作成やホストプロバイダの作成など、連携関連のイベント。認証ログのファイル名は、amFederation で始まります。 |
| 承認 (ポリシー) | amPolicy | ポリシーの作成、削除、変更やポリシー評価など、ポリシー関連のイベント。 |
| ポリシーエージェント | amAgent | 特定のユーザーからのアクセスが許可または拒否されたリソースに関する例外。amAgent ログは、ポリシーエージェントがインストールされたサーバー上に存在します。エージェントイベントのログは、Access Manager マシン上の認証ログに記録されます。 |
| SAML | amSAML | 表明、アーティファクトの作成または削除、応答や要求の詳細、SOAP エラーなど、SAML 関連のイベント。 |
| コマンド行 | amAdmin | amadmin コマンド行ツールによる操作中に発生したイベントエラー。フラットファイルログを指定した場合、amAdmin ログファイルは、主要なログディレクトリ (デフォルトでは /var/opt/SUNWam/logs) の下の amadmincli サブディレクトリに置かれます。例: サービススキーマの読み込み、ポリシーの作成、ユーザーの削除。 |

Access Manager のログファイルの一覧と説明については、『*Access Manager Administration Reference*』の「Access Manager Log File Reference」を参照してください。

# デバッグファイル

デバッグファイルは、ログサービスの機能ではありません。デバッグファイルは、ログ API とは別の API を使用して記録されます。デバッグファイルは、`/var/opt/SUNWam/debug` に格納されます。この場所は、デバッグ情報のレベルと一緒に、*AccessManager-base*`/SUNWam/lib/` ディレクトリの `AMConfig.properties` ファイルで設定できます。デバッグプロパティーの詳細については、『*Access Manager Administration Reference*』の「AMConfig.properties file reference」を参照してください。

## デバッグレベル

デバッグファイルに記録できる情報には、いくつかのレベルがあります。デバッグレベルは、`AMConfig.properties` の `com.iplanet.services.debug.level` プロパティーを使用して設定します。

1. off—デバッグ情報は記録されません。
2. error—このレベルは本稼働環境で使用されます。運用者は、本稼働環境ではデバッグファイルにエラーが記録されることのないように努めてください。
3. warning—現在、このレベルを使用することは推奨されていません。
4. message—コードトレースを使用するときに発生する可能性のある問題を警告します。ほとんどの Access Manager モジュールでは、このレベルを使用してデバッグメッセージが送信されます。

---

注 – warning および message レベルは、本稼働環境では使用してはいけません。パフォーマンスが大きく低下し、大量のデバッグメッセージが送信されます。

---

## デバッグ出力ファイル

デバッグファイルは、モジュールがデバッグファイルに書き込むときに作成されます。したがって、デフォルトの「`error`」モードでは、デバッグファイルは生成されません。デバッグレベルを「`message`」に設定した状態で基本的なログインを実行すると、次のデバッグファイルが作成されます。

- amAuth
- amAuthConfig
- amAuthContextLocal
- amAuthLDAP
- amCallback
- amClientDetection
- amConsole

- amFileLookup
- amJSS
- amLog
- amLoginModule
- amLoginViewBean
- amNaming
- amProfile
- amSDK
- amSSOProvider
- amSessionEncodeURL
- amThreadManager

もっとも頻繁に使用されるファイルは、`amSDK`、`amProfile`、および認証に関連するすべてのファイルです。記録された情報には、日付、時刻、およびメッセージタイプ (エラー、警告、message) が含まれます。

## デバッグファイルの使用

デバッグレベルは、デフォルトでは「`error`」に設定されています。デバッグファイルは、管理者が次の作業を行っているときに役立ちます。

- カスタム認証モジュールを作成するとき。
- Access Manager SDK を使用してカスタムアプリケーションを作成するとき。`amProfile` および `amSDK` デバッグファイルがこの情報を記録します。
- コンソールまたは SDK を使用しているときにアクセス権の障害を追跡するとき。`amProfile` および `amSDK` デバッグファイルは、この情報も記録します。
- SSL の障害を追跡するとき。
- LDAP 認証モジュールの障害を追跡するとき。`amAuthLDAP` デバッグファイルがこの情報を記録します。

デバッグファイルは、今後提供する予定の障害追跡ガイドと一緒に使用することをお勧めします。たとえば、SSL に障害が発生したときには、「message」レベルのデバッグを有効にして、`amJSS` デバッグファイルで証明書固有のエラーを探してみるとよいでしょう。

## 複数の **Access Manager** インスタンスとデバッグファイル

Access Manager には、多数のサーバーインスタンスを設定するときに使用できるように、`ammultiserverinstall` スクリプトが用意されています。複数のサーバーインスタンスがそれぞれ異なるデバッグディレクトリを使用するように設定されている場

合、各インスタンスに対してそれぞれのデバッグディレクトリへの読み取りアクセス権と書き込みアクセス権を割り当てる必要があります。

◆ ◆ ◆ 第 11 章

# 11 通知サービス

Sun Java System Access Manager 7.1 Notification Service を使用すると、セッション通知をリモートの web コンテナに送信できます。このサービスを、Access Manager サーバー本体から離れた場所で実行されている SDK アプリケーションが使用できるようにする必要があります。この章では、リモート web コンテナで通知を受信できるようにする方法について説明します。次の節があります。



## 概要

Notification Service を使用すると、Access Manager SDK をリモートで実行している web コンテナにセッション通知を送信できます。通知は、セッションサービス、ポリシーサービス、およびネームサービスにだけ適用されますまた、リモートアプリケーションは web コンテナ内で実行されなければなりません。通知には次のような目的があります。

- 個々のサービスのクライアント側のキャッシュを同期化する。
- クライアントでよりリアルタイムの更新を可能にする。(通知がない場合はポーリングが使用される。)
- 通知をサポートするために、クライアントアプリケーションの変更は必要ありません。

リモート SDK が web コンテナにインストールされている場合にのみ、通知を受信できます。

# 通知サービスの有効化

次に、セッション通知を受信するためのリモート SSO SDK の設定手順を示します。

## ▼ セッション通知を受信する

1 **Machine 1** に **Access Manager** をインストールします。

2 **Machine 2** に **Sun Java System Web Server** をインストールします。

3 **Web Server** と同じマシンに `SUNWamsdk` をインストールします。

Access Manager SDK をリモートでインストールする手順については、『Sun Java Enterprise System 5 インストールガイド』を参照してください。

4 **SDK** がインストールされているマシンに関して、次のことが当てはまることを確認します。

**a.** **SDK** がインストールされているサーバーの */remote_SDK_server/* `SUNWam/lib` ディレクトリおよび */remote_SDK_server* / `SUNWam/locale` ディレクトリに対して正しいアクセス権が設定されている。

これらのディレクトリには、リモートサーバーのファイルおよび jar ファイルが含まれます。

**b.** **Web Server** の `server.policy` ファイルの「**Grant**」セクションに次のアクセス権が設定されていることを確認します。

`server.policy` が、Web Server インストールの `config` ディレクトリに含まれる。これらのアクセス権は必要に応じてコピーおよびペーストできます。

```
permission java.security.SecurityPermission
"putProviderProperty.Mozilla-JSS"

permission java.security.SecurityPermission "insertProvider.Mozilla-JSS";
```

**c.** `server.xml` に正しいクラスパスが設定されていることを確認します。

`server.xml` も、Web Server インストールの `config` ディレクトリに含まれる。標準的なクラスパスは次のとおりです。

```
<JAVA javahome="/export/home/ws61/bin/https/jdk"
serverclasspath="/export/home/ws61/bin/https/jar/webserv-rt.jar:${java.home}/lib/tools.
    bin/https/jar/nova.jar"
classpathsuffix="::/IS_CLASSPATH_BEGIN_DELIM:                //usr/share/lib/xalan.jar:
                //lib:/export/SUNWam/locale:                //usr/share/lib/mps/jss3.ja
envclasspathignored="true" debug="false"
```

```
debugoptions="-Xdebug -Xrunjdwp:transport=dt_socket,
server=y,suspend=n"
javacoptions="-g"
dynamicreloadinterval="2">
```

5 リモート **SDK** サーバーにインストールされた **SSO** サンプルを確認の目的で使用します。

**a.** / *remote_SDK_server* /SUNWam/samples/sso ディレクトリに変更します。

**b.** gmake を実行します。

**c.** 生成されたクラスファイルを / *remote_SDK_server* /SUNWam/samples/sso から / *remote_SDK_server* /SUNWam/lib/ へコピーします。

6 am.encryption.pwd プロパティーの暗号化値を、**Access Manager** とともにインストールされた AMConfig.properties ファイルから、**SDK** がインストールされたリモートサーバー上の AMConfig.properties ファイルにコピーします。

am.encryption.pwd の値は、暗号化および復号化のためのパスワードとして使用されます。

7 **Access Manager** に amadmin としてログインします。

http://*AcceessManager-HostName* :3000/amconsole

8 ブラウザの場所フィールドに次のアドレスを入力してサーブレットを実行し、SSOToken を検証します。http://*remote_SDK_host*:58080/servlet/SSOTokenSampleServlet

SSOTokenSampleServlet は、セッショントークンの検証とリスナーの追加に使用されます。サーブレットを実行すると、次のメッセージが出力されます。

```
SSOToken host name: 192.18.149.33 SSOToken Principal name:
uid=amAdmin,ou=People,dc=red,dc=iplanet,dc=com Authentication type used: LDAP
IPAddress of the host: 192.18.149.33 The token id is
AQIC5wM2LY4SfcyURnObg7vEgdkb+32T43+RZN30Req/BGE= Property: Company is - Sun
Microsystems Property: Country is - USA SSO Token Validation test Succeeded
```

9 クライアント **SDK** がインストールされているマシンの AMConfig.properties に com.iplanet.am.notification.url= プロパティーを設定します。

```
com.iplanet.am.notification.url=http://clientSDK_host.domain:port
/servlet
    com.iplanet.services.comm.client.PLLNotificationServlet
```

10 **Web Server** を再起動します。

11 **Access Manager** に amadmin としてログインします。

```
http://AcceessManager-HostName :3000/amconsole
```

12 ブラウザの場所フィールドに次のアドレスを入力してサーブレットを実行し、再度 SSOToken を検証します。http://*remote_SDK_host*:58080/servlet/SSOTokenSampleServlet

リモート SDK が動作しているマシンが通知を受信すると、セッション状態が変わったときに個々のリスナーを呼び出します。リモート SDK が web コンテナにインストールされている場合にのみ、通知を受信できます。

## ▼ ポータルのみのインストール環境で通知サービスを有効化する

この節では、ポータルのみのインストール環境の WebLogic 8.1 で通知サービスを有効にする手順を説明します。デフォルトでは、WebLogic 8.1 は polling モードで動作します。amserver コンポーネントも含むポータルインスタンスの場合、これらの手順は不要です。amserver コンポーネントは、通知を実行するように自動的に設定されます。

1 PLLNotificationServlet を **WebLogic** に登録します。

WebLogic 8.1 では、web アプリケーションが配備されている必要があります。また、サーブレット URL は有効でなければなりません。有効な場合、ブラウザからアクセスすると次のメッセージが返されます。

```
Webtop 2.5 Platform Low Level notification servlet
```

2 登録された **URL** を AMConfig.properties に次のように入力します。

```
com.iplanet.am.notifaction.url=http://
weblogic_instance-host.domain:port/notification/PLLNotificationServlet
```

3 **AMConfig.properties** でポーリングを無効にします。これにより、通知が自動的に有効になります。

```
com.iplanet.am.session.client.polling.enable=false
```

4 **WebLogic** を再起動して、設定をテストします。

デバッグモードを message に設定している場合は、トリガーされたときにポータルにセッション通知が到達します。たとえば、Access Manager コンソールからあるユーザーを終了するなどのアクションにより、通知イベントが発生します。

# 索引

## ね



## ひ



## ほ



## ゆ



## り

## る



## れ



## ろ