Sun Java System Directory Server Enterprise Edition 6.1 管理ガイド

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

Part No: 820–2518
2007 年 6 月

070824@18135

目次

図目次

表目次

例目次

はじめに

『管理ガイド』では、コマンド行から Directory Server および Directory Proxy Server の機能を設定する手順について説明します。Web ベースのインタフェース (Directory Service Control Center) を使用してこれらの機能を設定する方法については、オンラインヘルプで説明しています。

対象読者

この『管理ガイド』の対象読者は、Directory Server ソフトウェアおよび Directory Proxy Server ソフトウェアの管理者です。

このマニュアルをお読みになる前に

このマニュアルでは、ソフトウェアのインストールについては説明していません。インストールについては、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』を参照してください。

古いバージョンの Directory Server または Directory Proxy Server から移行する場合は、サーバーの移行方法について、『Sun Java System Directory Server Enterprise Edition 6.1 Migration Guide』を参照してください。このバージョンの新機能をよく知らない場合は、新機能の概要について、『Sun Java System Directory Server Enterprise Edition 6.1 Evaluation Guide』を参照すると役立ちます。

内容の紹介

パート I「Directory Server による管理」では、Directory Server を管理する手順について説明します。

パート II「Directory Proxy Server による管理」では、Directory Proxy Server を管理する手順について説明します。

このマニュアルで使用する例

一貫性を保つために、このマニュアル全体で同じサンプルデータを使用します。これらの値をお使いのシステムに適した値に置き換えてください。

表 P–1　例で使用するデフォルト値

変数	例で使用する値
サフィックス (SUFFIX_DN)	dc=example,dc=com
インスタンスのパス (INSTANCE_PATH)	Directory Server の場合 : /local/ds/ Directory Proxy Server の場合 : /local/dps/
ホスト名 (HOST)	host1、host2、host3
ポート (PORT)	LDAP: root のデフォルト : 389。root 以外のデフォルト : 1389 SSL のデフォルト : root のデフォルト : 636。root 以外のデフォルト : 1636

Directory Server Enterprise Edition マニュアルセット

この Directory Server Enterprise Edition マニュアルセットでは、Sun Java System Directory Server Enterprise Edition を使用してディレクトリサービスを評価、設計、配備、および管理する方法について説明します。Directory Server Enterprise Edition 用のクライアントアプリケーションを開発する方法も示します。Directory Server Enterprise Edition マニュアルセットは http://docs.sun.com/coll/1224.2 から入手できます。

Directory Server Enterprise Edition について理解を深めるには、次の表に示すドキュメントを順番に参照してください。

表 P–2　Directory Server Enterprise Edition マニュアル

マニュアルタイトル	内容
『Sun Java System Directory Server Enterprise Edition 6.1 リリースノート』	既知の問題を含め、Directory Server Enterprise Edition についての最新情報を提供しています。
『Sun Java System Directory Server Enterprise Edition 6.1 Documentation Center』	マニュアルセットの主な領域へのリンクを提供しています。
『Sun Java System Directory Server Enterprise Edition 6.1 Evaluation Guide』	このリリースの重要な機能を紹介します。これらの機能の仕組みや提供される利点を、単独システムに実装可能な架空の配備のコンテキストに沿って例示します。

表 P–2 Directory Server Enterprise Edition マニュアル (続き)

マニュアルタイトル	内容
『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』	Directory Server Enterprise Edition をベースとする、可用性と拡張性に優れたディレクトリサービスを計画および設計する方法について説明します。配備の計画および設計の基本的な概念および原則を提示します。ソリューションのライフサイクルについて検討し、Directory Server Enterprise Edition ベースのソリューションを計画するために使用する概略レベルのサンプルおよび戦略を提供します。
『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』	Directory Server Enterprise Edition ソフトウェアのインストール方法について説明します。インストールするコンポーネントを選択する方法、インストール後にそれらのコンポーネントを設定する方法、および設定したコンポーネントが正しく機能することを検証する方法を示します。 Directory Editor のインストール手順については、http://docs.sun.com/coll/DirEdit_05q1 を参照してください。 Directory Editor をインストールする前に、『Sun Java System Directory Server Enterprise Edition 6.1 リリースノート』の Directory Editor についての情報を必ずお読みください。
『Sun Java System Directory Server Enterprise Edition 6.1 Migration Guide』	Directory Server、Directory Proxy Server、および Identity Synchronization for Windows の以前のバージョンからコンポーネントをアップグレードする手順を示します。
『Sun Java System Directory Server Enterprise Edition 6.1 管理ガイド』	Directory Server Enterprise Edition をコマンド行から管理するための手順を示します。 Directory Service Control Center (DSCC) を使用して Directory Server Enterprise Edition を管理する際のヒントおよび手順については、DSCC のオンラインヘルプを参照してください。 Directory Editor の管理手順については、http://docs.sun.com/coll/DirEdit_05q1 を参照してください。 Identity Synchronization for Windows のインストールと設定については、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』のパート II「Installing Identity Synchronization for Windows」を参照してください。
『Sun Java System Directory Server Enterprise Edition 6.1 Developer's Guide』	Directory Server Enterprise Edition の一部として提供されているツールと API でディレクトリクライアントアプリケーションを開発する方法を説明しています。
『Sun Java System Directory Server Enterprise Edition 6.1 Reference』	Directory Server Enterprise Edition の技術および概念の基礎を紹介します。コンポーネント、アーキテクチャー、プロセス、および機能について説明しています。開発者 API への参照も示しています。

表 P–2 Directory Server Enterprise Edition マニュアル *(続き)*

マニュアルタイトル	内容
『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』	Directory Server Enterprise Edition を通じて利用可能なコマンド行ツール、スキーマオブジェクト、およびその他の公開インタフェースについて説明しています。このドキュメントの個別の節を、オンラインマニュアルページとしてインストールすることができます。
『Sun Java System Directory Server Enterprise Edition 6.1 Troubleshooting Guide』	さまざまなツールを使用して問題の範囲を特定し、データを収集し、問題部分の障害追跡を行う手順について説明しています。
『Sun Java System Identity Synchronization for Windows 6.0 Deployment Planning Guide 』	Identity Synchronization for Windows の計画と配備に関する一般的なガイドラインやベストプラクティスを示しています。

関連資料

SLAMD 分散負荷生成エンジン (SLAMD) は、ネットワークベースのアプリケーションを対象に負荷テストを実行し、パフォーマンスを分析するために設計された Java™ アプリケーションです。SLAMD は当初 Sun Microsystems, Inc. によって、LDAP ディレクトリサーバーのパフォーマンスをベンチマークおよび分析する目的で開発されました。SLAMD は、OSI が承認したオープンソースライセンスである Sun Public License のもとでオープンソースアプリケーションとして公開されています。SLAMD についての情報を入手するには、`http://www.slamd.com/` を参照してください。SLAMD は java.net プロジェクトとしても公開されています。`https://slamd.dev.java.net/` を参照してください。

Java Naming and Directory Interface (JNDI) 技術は、LDAP および DSML v2 を利用した、Java アプリケーションからのディレクトリサーバーへのアクセスをサポートします。JNDI の詳細については、`http://java.sun.com/products/jndi/` を参照してください。『JNDI チュートリアル』には、JNDI の使用方法についての詳しい説明およびサンプルを収録しています。このチュートリアルは `http://java.sun.com/products/jndi/tutorial/` から入手できます。

Directory Server Enterprise Edition はスタンドアロン製品として、Sun Java Enterprise System のコンポーネントとして、Sun Java Identity Management Suite のような Sun 製品のスイートの一部として、または Sun のその他のソフトウェア製品のアドオンパッケージとしてライセンスを与えることができます。Java Enterprise System は、ネットワークまたはインターネット環境で分散配備されるエンタープライズアプリケーションをサポートするソフトウェアインフラストラクチャーです。Directory Server Enterprise Edition が Java Enterprise System のコンポーネントとしてライセンスされる場合、`http://docs.sun.com/coll/1657.1` から入手可能なシステムマニュアルに目を通してください。

Identity Synchronization for Windows は Message Queue を制限されたライセンスで使用します。Message Queue のマニュアルは `http://docs.sun.com/coll/1661.1` から入手できます。

Identity Synchronization for Windows は、Microsoft Windows のパスワードポリシーを管理するための製品です。

- Windows Server 2003 のパスワードポリシーについての情報は、Microsoft TechNet Web サイトで公開されています。
- Windows Server 2003 でのパスワードの変更やグループポリシーについての情報は、Microsoft TechNet Web サイトで公開されています。
- Microsoft 証明書サービスのエンタープライズルート認証局に関する情報は、Microsoft サポートオンライン Web サイトで公開されています。
- Microsoft システムでの LDAP over SSL の設定に関する情報は、Microsoft サポートオンライン Web サイトで公開されています。

再配布可能なファイル

Directory Server Enterprise Edition では、お客様による再配布が可能なファイルは提供されません。

デフォルトのパスとコマンドの場所

この節では、マニュアルで使用するデフォルトのパスについて説明し、オペレーティングシステムや配備タイプによって異なるコマンドの場所を示します。

デフォルトのパス

次の表では、このドキュメントで使用するデフォルトのパスについて説明します。インストールされるファイルの詳細な説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 14 章「Directory Server File Reference」、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 25 章「Directory Proxy Server File Reference」、または『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の付録 A「Directory Server Resource Kit File Reference」も参照してください。

表 P–3 デフォルトのパス

プレースホルダ	説明	デフォルト値
install-path	Directory Server Enterprise Edition ソフトウェアのベースインストールディレクトリを表します。 ソフトウェアは、このベース *install-path* の下位のディレクトリにインストールされます。たとえば、Directory Server ソフトウェアは *install-path*`/ds6/` にインストールされます。	`dsee_deploy`(1M) を使用して ZIP 形式の配布パッケージからインストールするとき、デフォルトの *install-path* は現在のディレクトリです。*install-path* は、`dsee_deploy` コマンドの `-i` オプションを使用して設定できます。 Java Enterprise System インストーラを使用する場合など、ネイティブパッケージ配布からインストールする場合、デフォルトの *install-path* は次のいずれかの場所になります。 ■ Solaris システム - `/opt/SUNWdsee/` ■ HP-UX システム - `/opt/sun/` ■ Red Hat システム - `/opt/sun/` ■ Windows システム - `C:\Program Files\Sun\JavaES5\DSEE`
instance-path	Directory Proxy Server または Directory Server のインスタンスのフルパスを表します。 このマニュアルでは、Directory Proxy Server には `/local/dps/` を、Directory Server には `/local/ds/` を使用します。	デフォルトのパスはありません。ただしインスタンスのパスは、常にローカルファイルシステム上に存在します。 推奨されるディレクトリは以下のとおりです。 `/var` (Solaris システム) `/global` (Sun Cluster を使用する場合)
serverroot	Identity Synchronization for Windows のインストール先の親ディレクトリを表します	インストールごとに異なります。Directory Server では、*serverroot* の概念が存在しなくなったことに注意してください。
`isw-`*hostname*	Identity Synchronization for Windows インスタンスのディレクトリを表します	インストールごとに異なります
/path/to/`cert8.db`	Identity Synchronization for Windows におけるクライアントの証明書データベースのデフォルトのパスおよびファイル名を表します	*current-working-dir*`/cert8.db`
serverroot`/isw-`*hostname*`/logs/`	システムマネージャー、各コネクタ、および Central Logger のログを Identity Synchronization for Windows がローカルに保存する場所のデフォルトパスを表します	インストールごとに異なります
serverroot`/isw-`*hostname*`/logs/central/`	Identity Synchronization for Windows セントラルログのデフォルトパスを表します	インストールごとに異なります

コマンドの場所

以下の表は、Directory Server Enterprise Edition のマニュアルで使用されるコマンドの場所の一覧です。これらの各コマンドの詳細については、それぞれのマニュアルページを参照してください。

表 P–4 コマンドの場所

コマンド	Java ES ネイティブパッケージ配布	ZIP 形式の配布パッケージ
cacaoadm	Solaris - /usr/sbin/cacaoadm	Solaris - *install-path*/dsee6/cacao_2.0/usr/lib/cacao/bin/cacaoadm
	Red Hat、HP-UX - /opt/sun/cacao/bin/cacaoadm	Red Hat、HP-UX - *install-path*/dsee6/cacao_2.0/cacao/bin/cacaoadm
	Windows - *install-path*\share\cacao_2.0\bin\cacaoadm.bat	Windows - *install-path*\dsee6\cacao_2.0\bin\cacaoadm.bat
certutil	Solaris - /usr/sfw/bin/certutil	*install-path*/dsee6/bin/certutil
	Red Hat、HP-UX - /opt/sun/private/bin/certutil	
dpadm(1M)	*install-path*/dps6/bin/dpadm	*install-path*/dps6/bin/dpadm
dpconf(1M)	*install-path*/dps6/bin/dpconf	*install-path*/dps6/bin/dpconf
dsadm(1M)	*install-path*/ds6/bin/dsadm	*install-path*/ds6/bin/dsadm
dsccmon(1M)	*install-path*/dscc6/bin/dsccmon	*install-path*/dscc6/bin/dsccmon
dsccreg(1M)	*install-path*/dscc6/bin/dsccreg	*install-path*/dscc6/bin/dsccreg
dsccsetup(1M)	*install-path*/dscc6/bin/dsccsetup	*install-path*/dscc6/bin/dsccsetup
dsconf(1M)	*install-path*/ds6/bin/dsconf	*install-path*/ds6/bin/dsconf
dsee_deploy(1M)	提供されていません	*install-path*/dsee6/bin/dsee_deploy
dsmig(1M)	*install-path*/ds6/bin/dsmig	*install-path*/ds6/bin/dsmig
entrycmp(1)	*install-path*/ds6/bin/entrycmp	*install-path*/ds6/bin/entrycmp
fildif(1)	*install-path*/ds6/bin/fildif	*install-path*/ds6/bin/fildif
idsktune(1M)	提供されていません	zip 形式の配布パッケージを解凍したディレクトリにあります

表 P–4 コマンドの場所 *(続き)*

コマンド	Java ES ネイティブパッケージ配布	ZIP 形式の配布パッケージ
insync(1)	*install-path*/ds6/bin/insync	*install-path*/ds6/bin/insync
ns-accountstatus(1M)	*install-path*/ds6/bin/ns-accountstatus	*install-path*/ds6/bin/ns-accountstatus
ns-activate(1M)	*install-path*/ds6/bin/ns-activate	*install-path*/ds6/bin/ns-activate
ns-inactivate(1M)	*install-path*/ds6/bin/ns-inactivate	*install-path*/ds6/bin/ns-inactivate
repldisc(1)	*install-path*/ds6/bin/repldisc	*install-path*/ds6/bin/repldisc
schema_push(1M)	*install-path*/ds6/bin/schema_push	*install-path*/ds6/bin/schema_push
smcwebserver	Solaris、Linux、HP-UX - /usr/sbin/smcwebserver	このコマンドは、ネイティブパッケージ配布を使用してインストールされる Directory Service Control Center のみに関係します。
	Windows - *install-path*\share\webconsole\bin\smcwebserver	
wcadmin	Solaris、Linux、HP-UX - /usr/sbin/wcadmin	このコマンドは、ネイティブパッケージ配布でインストールされる Directory Service Control Center のみに関係します。
	Windows - *install-path*\share\webconsole\bin\wcadmin	

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–5 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 machine_name% you have mail.
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	machine_name% **su** Password:

表 P–5 表記上の規則 (続き)

字体または記号	意味	例
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。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	`sun% grep '^#define \` `XV_VERSION_STRING'`

コード例は次のように表示されます。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bourne シェルおよび Korn シェル

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

- Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

コマンド例のシェルプロンプト

次の表は、デフォルトのシステムプロンプトとスーパーユーザープロンプトを示しています。

表 P–6 シェルプロンプト

シェル	プロンプト
UNIX および Linux システムの C シェル	machine_name%
UNIX および Linux システムの C シェルのスーパーユーザー	machine_name#
UNIX および Linux システムの Bourne シェルおよび Korn シェル	$
UNIX および Linux システムの Bourne シェルおよび Korn シェルのスーパーユーザー	#
Microsoft Windows のコマンド行	C:\

記号の規則

次の表に、このマニュアルで使用する記号の表記規則を示します。

表 P–7 記号の規則

記号	説明	例	意味
[ ]	省略可能な引数やコマンドオプションが含まれます。	ls [-l]	-l オプションは必須ではありません。
{ | }	必須のコマンドオプションの選択肢を囲みます。	-d {y|n}	-d オプションには y 引数か n 引数のいずれかを使用する必要があります。
${ }	変数参照を示します。	${com.sun.javaRoot}	com.sun.javaRoot 変数の値を参照します。
-	同時に押すキーを示します。	Control-A	Control キーを押しながら A キーを押します。
+	順番に押すキーを示します。	Ctrl+A+N	Control キーを押してから放し、それに続くキーを押します。
→	グラフィカルユーザーインタフェースでのメニュー項目の選択順序を示します。	「ファイル」→「新規」→「テンプレート」	「ファイル」メニューから「新規」を選択します。「新規」サブメニューから「テンプレート」を選択します。

マニュアル、サポート、およびトレーニング

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

パート I

Directory Server による管理

◆ ◆ ◆ 第 1 章

Directory Server のツール

Sun Java™ System Directory Server Enterprise Edition は、複数のサーバー、インスタンス、サフィックスをレプリケートされた環境で管理するためのブラウザインタフェースとコマンド行ツールを備えています。この章では、Directory Server の管理ツールの概要を説明します。

この章の内容は次のとおりです。

Directory Server の管理の概要

Directory Server の管理フレームワークについては、このマニュアルセットの別のマニュアルで説明しています。

- Directory Server の管理フレームワークの概要については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「Directory Server Enterprise Edition の管理モデル」を参照してください。
- Directory Server の管理フレームワークの詳細な情報については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 1 章「Directory Server Overview」を参照してください。

DSCC を使用する場合とコマンド行を使用する場合の判断

Directory Server Enterprise Edition には、Directory Server と Directory Proxy Server を管理するためのユーザーインタフェースが 2 つあります。ブラウザインタフェースである Directory Service Control Center (DSCC) とコマンド行インタフェースです。

DSCC を使用して手順を実行できるかどうかの判断

このマニュアルの手順のほとんどは、コマンド行または DSCC を使用して実行できます。このマニュアルの手順は、コマンド行を使用して手順を実行する方法を説明しています。ほとんどの場合、DSCC を使用しても同じ作業を実行できます。DSCC が特定の手順に使用できる場合、手順の初めにその旨が表記されています。

DSCC のオンラインヘルプには、DSCC を使用してこのマニュアルの手順を実行する詳細な指示が記載されています。

DSCC を使用した方が良い場合

DSCC は、次の節で説明するように、一部の操作と作業をコマンド行から実行するより簡単に実行できます。一般に、いくつかのサーバーに適用するコマンドは、DSCC を使用した方がうまくいきます。

サーバーとサフィックスのレプリケーション状態の表示

DSCC は、DSCC に登録されているすべてのサーバーインスタンスと設定されているすべてのサフィックス、およびそれぞれの状態を表に表示します。

サーバーの表は、「ディレクトリサーバー」タブにあり、サーバーの操作状態を示します。可能なサーバー状態の完全な一覧については、Directory Server のオンラインヘルプを参照してください。

サフィックスの表は「サフィックス」タブにあり、エントリの数やレプリケートされてない変更の数や経過時間など、レプリケーション状態の情報を示します。この表に表示される情報の詳細については、Directory Server のオンラインヘルプを参照してください。

サーバーのグループの管理

サーバーグループによって、サーバーの監視や設定を円滑に行えます。グループを作成し、そのグループにサーバを割り当てられます。たとえば、地理的な位置や機能によってサーバーをグループ化できます。多数のサーバーがある場合、グループ内のサーバーのみを表示するよう「ディレクトリサーバー」タブに表示されるサー

バーにフィルタを適用できます。また、あるサーバーのサーバー設定 (たとえば、インデックスやキャッシュ設定など) をグループ内のほかのすべてのサーバーにコピーすることもできます。サーバーグループの設定と使用方法については、Directory Server のオンラインヘルプを参照してください。

設定のコピー

DSCC では、既存のサーバー、サフィックス、またはレプリケーションアグリーメントの設定を 1 つまたは複数のほかのサーバー、サフィックス、レプリケーションアグリーメントにコピーできます。これらの作業の実行方法については、Directory Server のオンラインヘルプを参照してください。

レプリケーションの設定

DSCC を使用すると、レプリケーショントポロジをすばやく簡単に設定できます。単純にサーバーインスタンスを作成し、DSCC によって提供される手順を使用して、各サーバーのロールを割り当てます。DSCC によってレプリケーションアグリーメントが自動的に作成されます。DSCC を使用してレプリケーションを設定する方法については、Directory Server のオンラインヘルプを参照してください。

Directory Service Control Center のインタフェース

Directory Service Control Center (DSCC) は、ブラウザを使用して Directory Server と Directory Proxy Server を管理できるユーザーインタフェースです。

DSCC を設定するには、72 ページの「DSCC の設定」を参照してください。DSCC の使用については、次の節を参照してください。

DSCC の管理ユーザー

DSCC では次の管理者権限でのアクセスが必要となる場合があります。

- **OS** ユーザー。サーバーインスタンスを作成し、`dsadm` コマンドを使用してサーバーインスタンス上でオペレーティングシステムコマンドを実行する権限を持った唯一のユーザーです。DSCC は、場合によっては、OS ユーザーパスワードを要求することがあります。このユーザーはパスワードを持ち、ディレクトリサーバーインスタンスを作成できます。
- ディレクトリマネージャー。サーバーの LDAP スーパーユーザーです。デフォルト DN は `cn=Directory Manager` です。
- ディレクトリ管理者。Directory Server を管理します。このユーザーはアクセス制御、パスワードポリシー、認証の要件に関する条件付きで、ディレクトリマネージャーと同じ権限を持ちます。ディレクトリ管理者は必要な数だけ作成できます。

- **Directory Service Manager**。DSCC を使用して複数のマシン上のデータとサーバー設定を管理します。このユーザーには、DSCC に登録されている各サーバーに対してディレクトリマネージャーと同じ権限があり、ディレクトリ管理者グループのメンバーです。

▼ DSCC にアクセスする

DSCC へのアクセスに問題がある場合は、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「To Troubleshoot Directory Service Control Center Access」を参照してください。

1. **DSCC が『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「Software Installation」で説明されているとおりに正しくインストールされていることを確認します。**

2. ネイティブパッケージインストールによって **DSCC** をインストールした場合は、次の手順に従います。

   a. ブラウザを開き、**DSCC** ホスト **URL** を次の形式で入力します。

      ```
      https://hostname:6789
      ```

      次に例を示します。

      ```
      https://host1:6789
      ```

      ここで、ホスト名は、DSCC ソフトウェアをインストールしたシステムです。

      Sun Java Web Console のデフォルトポートは 6789 です。

      次の図は、Sun Java Web Console のログインウィンドウを示しています。

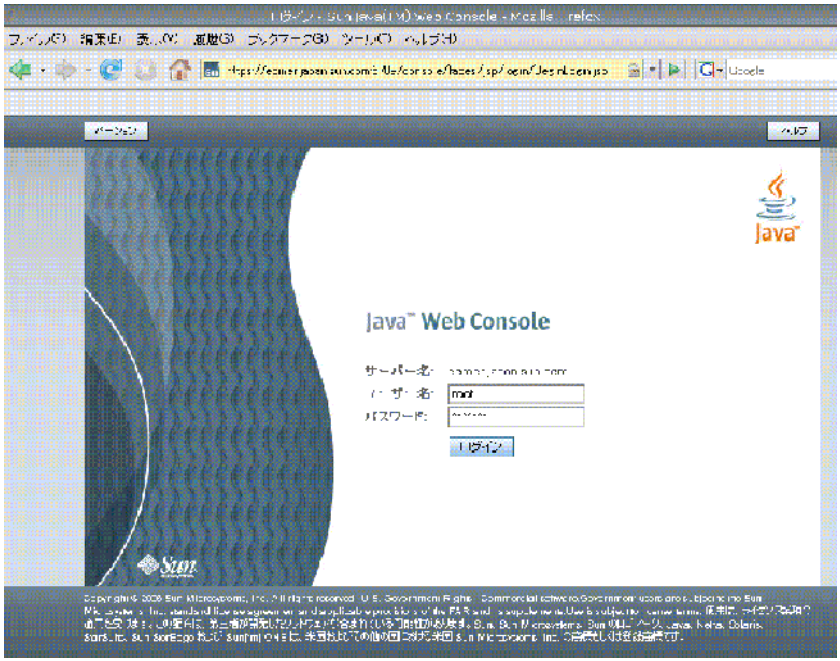


図 1–1　Sun Java Web Console のログインウィンドウ

b. **Sun Java Web Console** にログインします。

- Sun Java Web Console に初めてログインする場合は、DSCC ソフトウェアをインストールしたシステムに `root` として ログインします。
- これが 2 回目以降のログインである場合は、オペレーティングシステムのユーザー名とパスワードを入力します。このユーザーは、Directory Server インスタンスを起動、停止、および管理する特権が必要です。

ログインすると、アプリケーションの一覧が表示されます。

c. **Directory Service Control Center (DSCC)** を選択します。
DSCC ログインウィンドウが表示されます。

3 **zip** 形式の配布パッケージによって **DSCC** をインストールした場合は、次の手順に従います。

a. 好みのアプリケーションサーバーで、**DSCC** のホスト **URL** を入力して、**DSCC** に直接アクセスします。**DSCC** のホスト **URL** は、アプリケーションサーバーの設定に応じて次のいずれかにできます。

`https://`*hostname*`:6789`

または

```
http://hostname:6789
```

b. 次のコマンドを使用して **DSCC** を初期化します。

```
$ install path/dscc6/bin/dsccsetup ads-create
```

4 **DSCC** にログインします。

DSCC に初めてログインする場合は、Directory Service Manager パスワードを設定する必要があります。2 回目以降のログインの場合は、初回のログイン時に設定したパスワードを使用します。

DSCC にログインすると「共通操作」タブが表示されます。

図 1–2 DSCC の「共通操作」タブ

5 タブを使用して移動します。

- 「共通操作」タブには、一般的に使用するウィンドウやウィザードへのショートカットが表示されます。
- 「ディレクトリサーバー」タブには、DSCC で管理される Directory Server がすべて表示されます。特定のサーバーを管理および設定するためのオプションを表示するには、サーバー名をクリックします。

- 「プロキシサーバー」タブには、DSCC で管理される Directory Proxy Server がすべて表示されます。特定のサーバーを管理および設定するためのオプションを表示するには、サーバー名をクリックします。

注 – DSCC を使用して作業を実行する方法については、DSCC のオンラインヘルプを参照してください。

DSCC のタブの説明

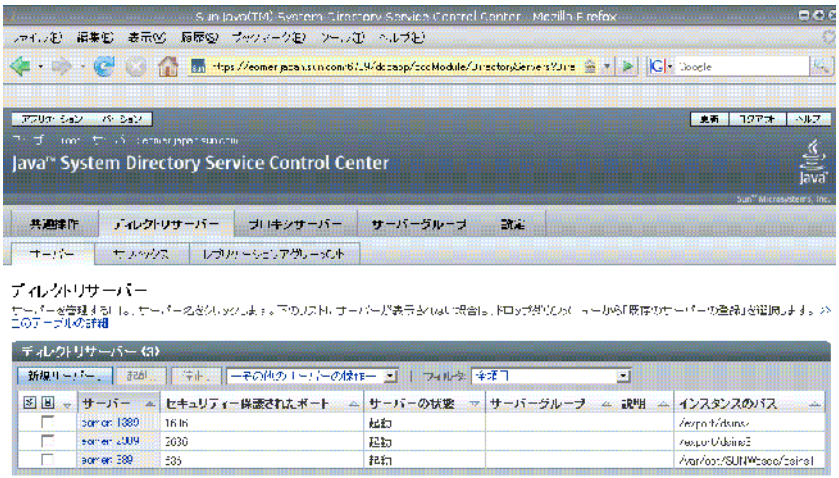


図 1–3 「サーバー」サブタブ上の Directory Server の一覧

DSCC のタブを使用してインタフェースを移動します。

「共通操作」タブ

「共通操作」タブ (図 1–2を参照) は、DSCC を開いたときに最初に表示されるインタフェースです。ここには、ディレクトリデータの検索、ログの確認、サーバーの管理など、一般的に使用する管理作業へのリンクが表示されます。

「ディレクトリサーバー」タブ

「ディレクトリサーバー」タブ (図 1–3を参照) には、DSCC に登録されているディレクトリサーバーがすべて一覧表示されます。各サーバーのサーバーの状態とインスタンスがどこにあるかを示すインスタンスパスを確認できます。

サーバー名をクリックすると、そのサーバーにのみ関連するタブのセットが表示されます。

「プロキシサーバー」タブ

「プロキシサーバー」タブは、DSCC に登録されているディレクトリプロキシサーバーをすべて一覧表示します。各サーバーのサーバーの状態とインスタンスがどこにあるかを示すサーバーインスタンスパスを確認できます。

サーバー名をクリックすると、そのサーバーにのみ関連するタブのセットが表示されます。

「サーバーグループ」タブ

「サーバーグループ」タブでは、サーバーをグループに割り当て、サーバー管理を簡素化できます。多数のサーバーがある場合、フィルタを使用して特定のグループのサーバーのみを表示できます。また、あるサーバーのサーバー設定 (たとえば、インデックスやキャッシュ設定など) をグループ内のほかのすべてのサーバーにコピーすることもできます。

「設定」タブ

このタブには、DSCC ポート番号が表示され、Directory Service Manager を作成および削除できます。

DSCC のオンラインヘルプ

オンラインヘルプには、次の情報が表示されます。

- 現在使用しているページのコンテキスト依存ヘルプ。
- DSCC を使用して管理および設定手順を実行するための一般的なヘルプ。

ほとんどのページで画面の右上にある「ヘルプ」ボタンをクリックすればヘルプにアクセスできます。ウィザードから、「ヘルプ」タブをクリックするとヘルプにアクセスできます。また、「共通操作」タブからオンラインヘルプにアクセスすることもできます。

Directory Server のコマンド行ツール

DSCC で実行する作業のほとんどは、コマンド行ツールを使用して実行できます。これらのツールによって、コマンド行から直接 Directory Server を管理し、スクリプトを使用してサーバーを管理できます。

主なディレクトリサーバーのコマンドは、`dsadm` と `dsconf` です。これらのコマンドを使用して、バックアップ、LDIF へのエクスポート、証明書の管理などを行えます。これらのコマンドについては、`dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

この節では、Directory Server コマンド行ツールの次の情報について説明します。

Directory Server コマンドの場所

Directory Server コマンド行ツールは、デフォルトインストールディレクトリにあります。

install-path/ds6/bin

インストールディレクトリは、オペレーティングシステムによって異なります。すべてのオペレーティングシステムのインストールパスは、33 ページの「デフォルトのパスとコマンドの場所」に一覧表示されています。

dsconf の環境変数の設定

dsconf コマンドでは、いくつかのオプションが必要となりますが、それらは環境変数によってあらかじめ設定することができます。コマンドを使用する際にオプションが指定されていない場合や、環境変数が設定されていない場合は、デフォルト設定が使用されます。環境変数は次のオプションに対して設定できます。

-D *user DN*	ユーザーバインド DN。環境変数: LDAP_ADMIN_USER。デフォルト: cn=Directory Manager。
-w *password-file*	ユーザーバインド DN のパスワードファイル。環境変数: LDAP_ADMIN_PWF。デフォルト: パスワードのプロンプト。
-h *host*	ホスト名。環境変数: DIRSERV_HOST。デフォルト: localhost。
-p *LDAP-port*	LDAP ポート番号。環境変数: DIRSERV_PORT。デフォルト: 389。
-e, --unsecured	dsconf がデフォルトで開くクリア接続を指定します。環境変数: DIRSERV_UNSECURED。この変数が設定されていない場合、dsconf はデフォルトでセキュリティー保護された接続を開きます。

詳細は、dsconf(1M) のマニュアルページを参照してください。

dsadm と dsconf の比較

次の表に、dsadm コマンドと dsconf コマンドの比較を示します。

表 1–1　dsadm コマンドと dsconf コマンドの比較

	dsadm コマンド	dsconf コマンド
説明	ローカルホストで直接実行する必要がある管理コマンド。次に例を示します。 ■ サーバーの起動または停止 ■ サーバーインスタンスの作成	リモートホストから実行できる管理コマンド。次に例を示します。 ■ レプリケーションの有効化 ■ キャッシュサイズの設定
注	サーバーは停止している必要があります (dsadm stop コマンドと dsadm info コマンドを除く)。 サーバーはサーバーインスタンスパス (*instance-path*) によって特定されます。 サーバーインスタンスパスへの OS アクセス権を持っている必要があります。	サーバーが実行中である必要があります。 サーバーはホスト名 (-h)、ポート (-p) または LDAPS セキュアポート (-P) によって特定されます。 ポート番号を特定しないと、dsconf はデフォルトポート (LDAP の場合は、389) を使用します。 たとえば、ユーザー cn=admin,cn=Administrators,cn=config など設定データへの LDAP アクセス権を持っている必要があります。

dsadm と dsconf を使用するためのヘルプの表示

dsadm コマンドと dsconf コマンドの使用方法についての詳細は、dsadm(1M) および dsconf(1M) のマニュアルページを参照してください。

- サブコマンドの一覧を表示するには、次の該当するコマンドを入力します。

```
$ dsadm --help

$ dsconf --help
```

- サブコマンドの使用方法についての説明を表示するには、次の該当するコマンドを入力します。

```
$ dsadm subcommand --help

$ dsconf subcommand --help
```

dsconf を使用した設定プロパティーの変更

dsconf のさまざまなサブコマンドを使って、ユーザーは設定プロパティーを表示したり、変更したりできます。

- Directory Server で使用する設定プロパティーを一覧表示するには、次のように入力します。

```
$ dsconf help-properties
```

- 特定のプロパティーを見つけるには、ヘルププロパティーの出力を検索します。

  たとえば、UNIX® プラットフォームを使用している場合は、リフェラルに関連するプロパティーをすべて検索するには、次のコマンドを使用します。

```
$ dsconf help-properties | grep -i referral
SER  referral-url                      rw  M  LDAP_URL | undefined
     Referrals returned to clients requesting a DN not stored in this
     Directory Server (Default: undefined)
SUF  referral-mode                     rw     disabled|enabled|only-on-write
     Specifies how referrals are used for requests involving the suffix
     (Default: disabled)
SUF  referral-url                      rw  M  LDAP_URL | undefined
     Server(s) to which updates are referred (Default: undefined)
SUF  repl-rewrite-referrals-enabled    rw     on|off
     Specifies whether automatic referrals are overwritten (Default: off)
```

  プロパティーは、サフィックス (SUF) やサーバー (SER) などターゲットオブジェクトによってグループ化されることに注意してください。rw キーワードは、そのプロパティーが読み書き可能であることを示します。M キーワードは、そのプロパティーが複数の値を持つことを示します。

- サーバー属性を表示するには、冗長モードを使用します。たとえば、UNIX システムでは次のように入力します。

```
$ dsconf help-properties -v | grep -i referral-mode
SUF  referral-mode    rw  disabled|enabled|only-on-write  nsslapd-state
  Specifies how referrals are used for requests involving the suffix
  (Default: disabled)
```

個々のプロパティーの詳細は、各プロパティーのマニュアルページを参照してください。マニュアルページは、『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』にあります。

dsconf を使用した複数値プロパティーの設定

特定の Directory Server プロパティーは複数の値をとることができます。これらの値を指定する構文は、次のとおりです。

```
$ dsconf set-container-prop -h host -p port container-name \
 property:value1 property:value2
```

たとえば、サーバーに対して複数の暗号化方式を設定するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host1 -p 1389 ssl-cipher-family:SSL_RSA_WITH_RC4_128_MD5 \
 ssl-cipher-family:SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
```

すでに値が含まれている複数値プロパティーに値を追加するには、次の構文を使用します。

```
$ dsconf set-container-prop -h host -p port container-name \
 property+:value
```

すでに値が含まれている複数値プロパティーから値を削除するには、次の構文を使用します。

```
$ dsconf set-container-prop -h host -p port container-name \
 property-:value
```

たとえば、前述の例で、暗号化方式のリストに SHA 暗号化方式を追加するには、次のコマンドを実行します。

```
$ dsconf set-server-prop -h host1 -p 1389 \
 ssl-cipher-family+:TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
```

このリストから MD5 暗号化方式を削除するには、次のコマンドを実行します。

```
$ dsconf set-server-prop -h host1 -p 1389 ssl-cipher-family-:SSL_RSA_WITH_RC4_128_MD5
```

マニュアルページ

マニュアルページには、Directory Server で使用するコマンドと属性すべての説明が記載されています。さらに、マニュアルページには配備でコマンドを使用する方法についての有効な例もいくつか記載されています。

旧バージョンのツール

旧バージョンのツールは、下位互換性のために通常の Directory Server ツールに含まれています。これらのツールは、含まれてはいますが、推奨されていません。

◆ ◆ ◆ 第 2 章

Directory Server のインスタンスとサフィックス

この章では、Directory Server のインスタンスとサフィックスを作成、管理する方法について説明します。その他多くのディレクトリ管理業務はサフィックスレベルで設定されますが、このマニュアルでは別の章で説明しています。

この章の内容は次のとおりです。

サーバーインスタンスとサフィックスを手短に作成する手順

この章では、サーバーインスタンスとサフィックスを作成する方法について詳しく説明します。Directory Server のインスタンスとサフィックスを手短に作成し、サンプルデータをインポートする必要がある場合は、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「Server Instance Creation」を参照してください。

Directory Server インスタンスの作成と削除

この節では、Directory Server インスタンスの作成と削除の方法について説明します。

▼ Directory Server インスタンスを作成する

データを管理する前に、コマンド行ツールまたはブラウザインタフェース Directory Service Control Center (DSCC) を使用して Directory Server インスタンスを作成する必要があります。DSCC で、Directory Server インスタンスは単に「Directory Server」と呼ばれることがあります。

Directory Server インスタンスを作成する場合、Directory Server に必要なファイルとディレクトリは、指定した *instance-path* で作成されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用して新しいサーバーインスタンスを作成する場合は、既存のサーバーからサーバー設定の一部またはすべてをコピーするよう選択できます。

1 新しい **Directory Server** インスタンスを作成して、インスタンスパスを設定します。

```
$ dsadm create instance-path
```

このサーバーのディレクトリマネージャーのパスワードを設定するよう要求されます。

サーバーインスタンスにデフォルト以外のポート番号またはその他のパラメータを指定するには、`dsadm`(1M) のマニュアルページを参照してください。

たとえば、ディレクトリ `/local/ds` で新しいインスタンスを作成するには、次のコマンドを使用します。

```
$ dsadm create /local/ds
Choose the Directory Manager password:
Confirm the Directory Manager password:
Use 'dsadm start /local/ds' to start the instance
```

2 サーバーインスタンスが正しく作成されていることを確認します。

```
$ dsadm info instance-path
```

次に例を示します。

```
$ dsadm info /local/ds1
インスタンスのパス:         /local/ds1
所有者:                 user1(group1)
```

```
セキュリティーが確保されていないポート:       1389
セキュアポート:           1636
ビットフォーマット:            64-bit
状態:                稼働中
サーバーの PID:            22555
DSCC URL:              -
SMF アプリケーション名:  -
ブート時に起動する:          無効
インスタンスのバージョン:      D-A00
```

3 **(省略可能) Java Enterprise System インストーラまたはネイティブのパッケージインストールを使用して Directory Server をインストールし、お使いの OS にサービス管理ソリューションがある場合は、次の表で示すように、サーバーがサービスとして管理されるようにできます。**

オペレーティングシステム	コマンド
Solaris 10	Sun クラスタ環境で作業している場合、次のコマンドを使用します。 `dsadm enable-service --type CLUSTER` *instance-path resource-group* そうでない場合は、次のようになります。 `dsadm enable-service --type SMF` *instance-path*
Solaris 9	Sun クラスタ環境で作業している場合、次のコマンドを使用します。 `dsadm enable-service --type CLUSTER` *instance-path resource_group* そうでない場合は、次のようになります。 `dsadm autostart` *instance-path*
Linux、HP-UX	`dsadm autostart` *instance-path*
Windows	`dsadm enable-service --type WIN_SERVICE` *instance-path*

4 **Directory Server を起動します。**

```
$ dsadm start instance-path
```

注 – サーバーは実行されますが、データやサフィックスは含まれていません。サフィックスを作成するには、 `dsconf` を使用します。

5 **(省略可能) 次のいずれかの方法で、サーバーインスタンスを登録します。**

- URL `https://`*host*`:6789` にアクセスし、DSCC によってサーバーを登録します。
- `dsccreg add-server` コマンドを使用します。
  詳細については、`dsccreg`(1M) のマニュアルページを参照してください。

6 **Directory Server** インスタンスがスタンドアロンでパスワードポリシーを使用する場合、または DS6-only パスワードポリシーモードにすでに移行したレプリケーショントポロジに属している場合は、インスタンスをこのモードに移行します。

```
$ dsconf pwd-compat -h host -p port to-DS6-migration-mode

## Beginning password policy compatibility changes .
## Password policy compatibility changes finished.

Task completed (slapd exit code: 0).
$ dsconf pwd-compat -h host -p port to-DS6-mode

## Beginning password policy compatibility changes .
## Password policy compatibility changes finished.

Task completed (slapd exit code: 0).
```

▼ Directory Server インスタンスを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **Directory Server** を停止します。

```
$ dsadm stop instance-path
```

2 以前に **DSCC** を使用してサーバーを管理していた場合は、コマンド行を使用してサーバーを登録解除します。

```
$ dsccreg remove-server /local/ds
Enter DSCC administrator's password:
/local/ds is an instance of DS
Enter password of "cn=Directory Manager" for /local/ds:
This operation will restart /local/ds.
Do you want to continue ? (y/n) y
Unregistering /local/ds from DSCC on localhost.
Connecting to /local/ds
Disabling DSCC access to /local/ds
Restarting /local/ds
```

詳細については、dsccreg(1M) のマニュアルページを参照してください。

3 **(省略可能)** 以前にサーバー管理ソリューションでサーバーインスタンスを有効にした場合は、サービスとしてのサーバーの管理を無効にします。

オペレーティングシステム	コマンド
Solaris 10	Sun クラスタ環境で作業している場合、次のコマンドを使用します。 `dsadm disable-service --type CLUSTER` *instance-path* そうでない場合は、次のようになります。 `dsadm disable-service --type SMF` *instance-path*
Solaris 9	Sun クラスタ環境で作業している場合、次のコマンドを使用します。 `dsadm disable-service --type CLUSTER` *instance-path* そうでない場合は、次のようになります。 `dsadm autostart --off` *instance-path*
Linux、HP-UX	`dsadm autostart --off` *instance-path*
Windows	`dsadm disable-service --type WIN_SERVICE` *instance-path*

4 サーバーインスタンスを削除します。

```
$ dsadm delete instance-path
```

注意 – このコマンドによって、データベースやデータを含むすべてが削除されます。

インスタンスがサービスとして有効にされている場合、またはインスタンスがシステムの起動時に自動的に起動されている場合には、ルートアクセス権を持つ管理ユーザーで、`dsadm delete` を実行する必要があります。

Directory Server インスタンスの起動、停止、および再起動

コマンド行からサーバーを起動、停止、または再起動するには、それぞれコマンド `dsadm start`、`dsadm stop`、または `dsadm restart` を使用します。

注 – エントリを維持するよう設定されたメモリーに大容量のキャッシュがある状態で Directory Server インスタンスを停止して再起動する場合、キャッシュを復元するにはしばらく時間がかかります。キャッシュを復元している間、インスタンスの応答時間は遅くなります。

これらのコマンドは、Directory Server を作成したものと同じ UID と GID で実行するか、root で実行する必要があります。たとえば、Directory Server を `user1` として実行している場合、`start`、`stop`、および `restart` ユーティリティーも `user1` として実行する必要があります。

注 – Solaris 上では、役割に基づくアクセス制御によって、root 以外のユーザーとして Directory Server を実行できます。

▼ Directory Server を起動、停止、および再起動する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。ただし、これは、サービス管理を有効および無効にする手順には適用されません。サービス管理の有効化と無効化は、Directory Server の起動および停止時にコマンド行で実行する必要があります。

- **Directory Server** を起動、停止、および再起動するには、次のいずれかを実行します。
  - サーバーを起動するには、次のように入力します。

    ```
    $ dsadm start instance-path
    ```

    たとえば、インスタンスパスが `/local/ds` のサーバーを起動するには、次のコマンドを使用します。

    ```
    $ dsadm start /local/ds
    ```
  - サーバーを停止するには、次のように入力します。

    ```
    $ dsadm stop instance-path
    ```

    次に例を示します。

    ```
    $ dsadm stop /local/ds
    ```
  - サーバーを再起動するには、次のように入力します。

    ```
    $ dsadm restart instance-path
    ```

    次に例を示します。

    ```
    $ dsadm restart /local/ds
    ```

サフィックスの作成

Directory Server インスタンスを作成した後、サーバーのディレクトリ情報ツリー (DIT) に対して 1 つまたは複数のサフィックスを作成します。DIT はサーバー内のすべてのエントリから構成され、エントリはそれぞれ DN (識別名) によって識別されます。DN は階層構造を持つため、ツリー内のデータ構成を決定する分岐のエントリおよびリーフエントリが作成されます。DIT は、サフィックスとサブサフィックスの点で管理上、定義され、管理されます。DSCC は、これらの要素すべての作成と管理を制御します。または、コマンド行ツールを使用することもできます。

ディレクトリデータの構造化と一般的なサフィックスの概念的な情報については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

次の手順で説明しているように、`dsconf create-suffix` コマンドを使用して、ディレクトリでサフィックス設定を作成できます。ルートサフィックスとサブサフィックスは、サーバーによって内部的に同じ方法で管理されるため、それらをコマンド行から作成する手順はほとんど同じです。手順では、必要なオプションのみと `dsconf create-suffix` コマンドを使用しています。このコマンドのその他のオプションについては、`dsconf`(1M) のマニュアルページを参照するか、次のコマンドを実行してください。

```
$ dsconf create-suffix --help
```

管理ユーザーが設定エントリを作成できます。ただし、サフィックスの最上位エントリは、ディレクトリマネージャーが作成するか、`cn=admin,cn=Administrators,cn=config` のように、ディレクトリの管理者として作成される必要があります。

▼ サフィックスを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用して新しいサフィックスを作成するには、既存のサフィックスからサフィックス設定の一部またはすべてをコピーするよう選択できます。

1 ルートサフィックスを作成します。

サーバーが起動中であることを確認して、次のコマンドを入力します。

```
$ dsconf create-suffix -h host -p port suffix-DN
```

ここで、*suffix-DN* は新しいサフィックスの完全な DN です。ルートサフィックスでは、ドメインコンポーネント (`dc`) のネーミング属性が使用されます。

たとえば、DN dc=example,dc=com のサフィックスを作成する場合は、次のコマンドを使用します。

```
$ dsconf create-suffix -h host1 -p 1389 dc=example,dc=com
```

このコマンドによって、次のように新しいサフィックスが作成されます。

- ルートサフィックスの最上位レベル (またはベース) エントリが作成されます。
- cn=config 内にサフィックスとデータベースの両方に対する設定エントリが作成されます。
- デフォルトデータベース名は、サフィックス DN に基づきます。

作成された新しいサフィックスを含む、すべてのサフィックスについては、次のコマンドを使用します。

```
$ dsconf list-suffixes -h host -p port -v
```

-v オプションによって冗長モードが表示されます。これによって、サフィックス上のエントリの数とレプリケーション情報が表示されます。

注 – 複数の Directory Server インスタンスがある場合、 -h *host name* と -p *port number* オプションを使用して、サフィックスの属するサーバーインスタンスを指定します。

データベースファイル用にデフォルト以外のパスを指定する場合は、-L オプションを使用します。サフィックスデータベースパスはあとで変更できます。これを実行するには、コマンド dsconf set-suffix-prop *suffix-DN* db-path:*new-db-path* を使用してから、サーバーを停止し、データベースファイルを手動で移動して、サーバーを再起動します。

サフィックスの作成時に使用できるオプションをすべて確認するには、dsconf(1M) のマニュアルページを参照してください。

---

2 必要に応じてサブサフィックスを作成します。

```
$ dsconf create-suffix -h host -p port subSuffix-DN
```

その後、サブサフィックスをルートサフィックスに追加します。

```
$ dsconf set-suffix-prop -h host -p port subSuffix-DN parent-suffix-dn:parentSuffix-DN
```

ここで、*parentSuffix-DN* は、前の手順の *suffix-DN* と同じ値にします。サブサフィックスの *suffix-DN* には、サブサフィックスの相対識別名 (RDN) と親サフィックスの DN が含まれます。

たとえば、サブサフィックス ou=Contractors,dc=example,dc=com を作成して、サブサフィックスをルートサフィックスに追加するには、次のように入力します。

```
$ dsconf create-suffix -h host1 -p 1389 ou=Contractors,dc=example,dc=com
$ dsconf set-suffix-prop -h host1 -p 1389 ou=Contractors,dc=example,dc=com \
 parent-suffix-dn:dc=example,dc=com
```

このエントリがディレクトリに追加されると、サーバーのデータベースモジュールは、次のディレクトリにデータベースファイルを自動的に作成します。

instance-path/db/*database-name*

ここで、*database-name* は、サフィックスの一部から自動的に構築された名前です。たとえば、前の例で、*database-name* は Contractors となります。

3 (省略可能) サフィックスをデータで初期化します。212 ページの「サフィックスの初期化」を参照してください。

サフィックスの無効化と有効化

場合によっては、保守のためにサフィックスを使用不可にしたり、セキュリティー上の理由からその内容を使用不可にする必要のあることがあります。サフィックスを無効にすることによって、クライアント操作に応えてサーバーがサフィックスの内容を読み書きするのを防げます。サフィックスを無効にすると、そのサフィックスにアクセスすることはできなくなり、リフェラルモードは自動的に無効になります。

▼ サフィックスを無効にしてから有効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サフィックスを無効にします。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN enabled:off
```

注 – レプリケートされたサフィックスのほとんどのプロパティーがレプリケーションメカニズムによって決定されるため、レプリケーションが有効になっているサフィックスを無効にすることはできません。

---

2 サフィックスを有効にします。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN enabled:on
```

リフェラルを設定し、サフィックスを読み取り専用にする

サフィックスを完全に無効にすることなくサフィックスへのアクセスを制限するには、アクセス権を変更して、読み取り専用アクセスを許可することもできます。この場合、書き込み操作に対しては、別のサーバーへのリフェラルを定義する必要があります。また、読み取りアクセスと書き込みアクセスの両方を拒否し、サフィックスへのすべての操作に対するリフェラルを定義できます。

さらに、リフェラルを使用して、クライアントアプリケーションが一時的に別のサーバーを使用するように設定することもできます。たとえば、サフィックスの内容をバックアップしている間、別のサフィックスへリフェラルを追加できます。

サフィックスがレプリケートされた環境のコンシューマである場合、レプリケーションメカニズムによって、リフェラル設定の値が決まります。リフェラルの設定は手動で変更できますが、リフェラルは次のレプリケーションの更新時に上書きされます。レプリケーションのリフェラルの設定については、250 ページの「コンシューマの詳細設定を行う」を参照してください。

リフェラルはラベル化された URL なので、LDAP URL には空白文字とラベルが続く場合があります。次に例を示します。

```
ldap://phonebook.example.com:389/
```

または

```
ldap://phonebook.example.com:389/ou=All%20People,dc=example,dc=com
```

リフェラルの URL 部分にある空白文字は、`%20` を使用してエスケープする必要があります。

▼ リフェラルを設定して、サフィックスを読み取り専用にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 リフェラルの **URL** を設定します。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN referral-url:LDAP-URL
```

ここで、*LDAP-URL* はターゲットのホスト名、ポート名、DN を含む有効な URL です。

次に例を示します。

```
$ dsconf set-suffix-prop -h host1 -p 1389 dc=example,dc=com \
 referral-url:ldap://phonebook.example.com:389/
```

LDAP URL は任意の個数だけ指定できます。

2 サフィックスを読み取り専用にするためにリフェラルモードを設定します。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN referral-mode:only-on-write
```

サフィックスを読み取りも書き込みもできないようにし、すべての要求にリフェラルを返すには`referral-mode` を `enabled` に設定します。

3 コマンドが正常に実行されるとすぐに、サフィックスは読み取り専用またはアクセス不可になり、リフェラルを返す準備ができます。

4 (省略可能) サフィックスが使用できるようになったら、ふたたびサフィックスの読み書きができるようにリフェラルを無効にします。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN referral-mode:disabled
```

リフェラルが無効になると、サフィックスの `enabled` プロパティーを `off` に設定してサフィックス自体を無効にしていない限り、サフィックスは自動的に読み書き可能になります。

サフィックスの削除

サフィックスを削除すると、DIT からそのエントリ全体が削除されます。

注 – サフィックスを削除すると、ディレクトリからそのデータエントリすべてが完全に削除されます。レプリケーション設定を含むサフィックス設定情報もすべて削除されます。

親サフィックスを削除し、そのサブサフィックスを、DIT で新しいルートサフィックスとして保持することはできません。サブサフィックスを含むエントリ全体を削除する場合は、削除された親のサブサフィックスと考えられるサブサフィックスも削除します。

▼ サフィックスを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- サフィックスの設定エントリを削除します。

  ```
  $ dsconf delete-suffix -h host -p port [subSuffix-DN] suffix-DN
  ```

  このコマンドによって、*suffix-DN* のベースエントリで始まるサフィックスがサーバーから削除されます。これで、サフィックスはディレクトリに表示されなくなり、アクセスできなくなります。

サフィックスの圧縮

Directory Server 6.1 では、オフラインでのサフィックスの圧縮がサポートされます。このリリースでは、オンラインでの圧縮はサポートされません。記憶領域を使用できる場合、サフィックスを圧縮すると、データベースキーが再構成されることによりデータベースのサイズが縮小されます。

▼ サフィックスをオフラインで圧縮する

この手順を実行する前に、サーバーを停止してください。

- 必要なサフィックスを圧縮します。

  ```
  $ dsadm repack instance-path suffix-dn
  ```

  指定したサフィックスに関連するすべての `.db3` ファイルが圧縮されます。

  このコマンドに `-b` オプションを付けて実行すると、サフィックス DN の代わりにバックエンドデータベース名を指定できます。少なくとも 1 つのサフィックスまたはバックエンドを指定してください。

◆ ◆ ◆ 第 3 章

3 Directory Server の設定

この章では、Directory Server の設定方法について説明します。これには dsconf コマンドを使用できます (dsconf(1M) のマニュアルページを参照)。

また、Directory Service Control Center (DSCC) も使用でき、こちらが推奨される方法です。DSCC では、設定プロセスの間に追加のチェックが行われ、これによりエラーを最小限に抑えることができます。さらに、DSCC では、設定をあるサーバーインスタンスから別のサーバーインスタンスにコピーできます。DSCC の使い方の詳細は、DSCC のオンラインヘルプを参照してください。

Directory Server インスタンスの設定の表示

Directory Server インスタンスの設定を表示するには、dsconf info を実行します。

```
$ dsconf info -h host -p port
インスタンスのパス    :  instance path
グローバルな状態      :  read-write
ホスト名          :  host
ポート             :  port
セキュリティー保護されたポート      :  secure port
エントリ合計数    :  20844

サフィックス          :  suffix-DN

ターゲットサーバー    :  host:port

実行中のタスク   :  import
完了したタスク   :  backup
```

この出力は、サフィックス、およびターゲットサーバーとのレプリケーションアグリーメントが作成されていることを前提としています。実行中のインポート処理と完了したバックアップ処理も表示されます。

DSCC による設定の変更

設定を変更する方法としては、DSCC の使用を推奨します。このブラウザインタフェースには、タスクベースの制御が用意されており、迅速かつ効率的な設定に役立ちます。DSCC を使えば、1 つのサーバーの設定を変更してから、その設定をほかのサーバーにコピーできます。また、DSCC のインタフェースは、設定の複雑さや相互依存の解決に役立ちます。DSCC による設定変更の詳しい手順については、DSCC のオンラインヘルプを参照してください。

コマンド行からの設定の変更

コマンド行ツールを使用するスクリプトを作成することで、設定タスクを自動化することができます。

`dsconf` コマンドを使用して、コマンド行から設定を変更します。このコマンドは、LDAP を使用して `cn=config` サブツリーを変更します。`dsconf` の詳細は、50 ページの「Directory Server のコマンド行ツール」を参照してください。

`dsconf` では実行できないタスクの場合は、`ldapmodify` コマンドを使用します。

注 – `dsconf set-server-prop` コマンドを使用してサーバー設定プロパティーを変更する場合は、変更できるプロパティーとそれらのデフォルト値がわかっている必要があります。すべてのプロパティーのヘルプを表示するには、次のコマンドを使用します。

```
$ dsconf help-properties -v
```

必要な項目についてプロパティーのヘルプを検索します。たとえば、UNIX プラットフォームの場合は、次のように入力してメモリーキャッシュのプロパティーを検索します。

```
$ dsconf help-properties -v | grep cache
```

`cn=config` 内の設定エントリの詳細と、許容値の範囲を含むすべての設定エントリおよび属性の詳しい説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

dse.ldif ファイルの変更

Directory Server では、その設定情報が次のファイル内に格納されます。

instance-path/config/dse.ldif

注意 – dse.ldif ファイルの内容を直接編集して設定を変更することは、エラーが生じる可能性が高くなるため、お勧めできません。ただし、このファイルを手動で編集する場合は、ファイルを編集する前にサーバーを停止し、編集が終わったらサーバーを再起動します。

dse.ldif ファイルの形式は、LDIF (LDAP Data Interchange Format) です。LDIF は、エントリ、属性、およびその値をテキスト表現したもので、RFC2849 (2849 (http://www.ietf.org/rfc/rfc2849)) に定義されている標準形式です。

dse.ldif ファイルにある Directory Server の設定は、次のもので構成されます。

- cn=config エントリの属性と値。
- cn=config の下のサブツリーに含まれるすべてのエントリと、その属性および属性値。
- ルートエントリ ("") と cn=monitor エントリのオブジェクトクラス、およびアクセス制御命令。これらのエントリのその他の属性は、サーバーによって生成されます。

  このファイルの読み書き権限を持っているのは、Directory Server インスタンスを所有するシステムユーザーのみです。

  Directory Server では、LDAP を通じてすべての設定を読み取り、書き込むことができます。デフォルトでは、ディレクトリの cn=config ブランチは、許可されているすべてのユーザーが読み取りでき、ディレクトリマネージャー (cn=Directory Manager) だけが cn=Administrators,cn=config の下の管理ユーザーに書き込むことができます。管理ユーザーは、他のディレクトリエントリと同様に、設定エントリを表示、変更できます。

  cn=config エントリの下には設定エントリ以外のものは作成しないでください。通常のエントリとは異なり、cn=config 配下では、作成されたエントリは、スケーラブルなデータベースとは異なる dse.ldif ファイルに格納されるためです。多くのエントリ、特に頻繁に更新されるエントリが cn=config の下に格納されている場合、パフォーマンスが低下する可能性が高くなります。ただし、レプリケーションマネージャー (サプライヤバインド DN) などの特別なユーザーエントリを cn=config の下に格納しておくと、設定情報を集中管理できて便利です。

管理ユーザーの設定

Directory Server には、デフォルトの管理ユーザーであるディレクトリマネージャーと cn=admin,cn=Administrators,cn=config ユーザーが含まれています。これらのユーザーは両方とも同じアクセス権限を持ちますが、ACI の対象は cn=admin,cn=Administrators,cn=config です。

この節では、ルートアクセス権を持つ管理ユーザーを作成する方法と、ディレクトリマネージャーを設定する方法について説明します。

▼ ルートアクセス権を持つ管理ユーザーを作成する

cn=admin,cn=Administrators,cn=config と同じ権限を持つ新しい管理ユーザーを作成する場合は、グループ cn=Administrators,cn=config 内に新しいユーザーを作成します。このグループ内のすべてのユーザーが、ディレクトリマネージャーと同じ権限を許可するグローバル ACI の対象となります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- 新しい管理ユーザーを作成します。

  たとえば、cn=Admin24,cn=Administrators,cn=config という新しいユーザーを作成するには、次のように入力します。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=admin24,cn=Administrators,cn=config
changetype: add
objectclass: top
objectclass: person
userPassword: password
description: Administration user with the same access rights as Directory Manager.
```

  --D オプションと --w オプションでは、それぞれ、このエントリの作成に必要な権限を持つユーザーのバインド DN とパスワードを指定します。

▼ ディレクトリマネージャーを設定する

ディレクトリマネージャーとは、特権を持つサーバー管理者のことで、UNIX システムの root ユーザーにあたります。アクセス制御はディレクトリマネージャーには適用されません。

ほとんどの管理タスクでは、ディレクトリマネージャーを使用する必要はありません。代わりに、ユーザー cn=admin,cn=Administrators,cn=config を使用するか、cn=Administrators,cn=config の下に作成するその他のユーザーを使用できます。

ディレクトリマネージャーを必要とするタスクは、ルート ACI の変更と、レプリケーションの修復や削除記録 (tombstone) の検索などのレプリケーションのトラブルシューティングタスクだけです。

ディレクトリマネージャー DN およびパスワードを変更でき、パスワードを自動的に読み取ることができるファイルを作成することもできます。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 既存のディレクトリマネージャー **DN** を見つけます。

```
$ dsconf get-server-prop -h host -p port root-dn
root-dn:cn=Directory Manager
```

2 必要に応じてディレクトリマネージャーの設定を変更します。

- ディレクトリマネージャー DN を変更するには、次のように入力します。

  ```
  $ dsconf set-server-prop -h host -p port root-pwd-file:new-root-dn-password-file
  ```

  ディレクトリマネージャー DN のなかに空白文字がある場合は、引用符を使います。次に例を示します。

  ```
  $ dsconf set-server-prop -h host1 -p 1389 root-dn:"cn=New Directory Manager"
  ```

- ディレクトリマネージャーパスワードを変更するには、次のように入力します。

  ```
  $ dsconf set-server-prop -h host -p port root-pwd:new-root-dn-password
  ```

  セキュリティー上の理由でコマンド行引数としてクリアテキストのパスワードを渡したくない場合は、パスワード設定用の一時ファイルを作成します。

  ```
  $ echo password > /tmp/pwd.txt
  ```

  このファイルが一度読み取られ、パスワードは将来使用するために格納されます。サーバールートのパスワードファイルプロパティーを設定します。

  ```
  $ dsconf set-server-prop -h host -p port root-pwd-file:/tmp/pwd.txt
  ```

  このコマンドは、サーバーにパスワードファイルの読み取りを要求します。パスワードファイルプロパティーの設定が完了したら、一時パスワードファイルを削除します。

  ```
  $ rm /tmp/pwd.txt
  ```

設定情報の保護

Directory Server のルートエントリ (長さゼロの DN "" によるベースオブジェクト検索で返されるエントリ) と、 cn=config、cn=monitor、cn=schema の下のサブツリーには、Directory Server によって自動的に生成されるアクセス制御命令 (ACI) が含まれます。これらの ACI は、ディレクトリエントリに対するユーザーアクセス権を確認するために使われます。評価目的としては、これらの ACI は十分に使えます。しかし、本稼働環境への配備の場合には、アクセス制御要件を評価し、独自のアクセス制御を設計する必要があります。

セキュリティー上の理由で 1 つまたは複数の追加のサブツリーの存在を非表示にし、設定情報を保護する場合は、追加の ACI を DIT 上に配置する必要があります。

- ACI 属性を、非表示にするサブツリーのベースにあるエントリに配置する。
- ACI をルート DSE エントリの namingContexts 属性に配置する。namingContexts というルート DSE エントリに、Directory Server の各データベースのベース DN のリストがあります。
- ACI を cn=config サブツリーと cn=monitor サブツリーに配置する。サブツリー DN も、cn=config と cn=monitor の下のマッピングツリーエントリ内に格納されます。

ACI の作成の詳細は、第 6 章を参照してください。

DSCC の設定

この節では、DSCC の設定に関する次のような点について記載します。

▼ 共通エージェントコンテナのポート番号を変更する

デフォルトの共通エージェントコンテナのポート番号は 11162 です。共通エージェントコンテナは、DSCC エージェントポートを jmxmp-connector-port として定義します。管理上の理由で、DSCC エージェントと共通エージェントコンテナに別のポート番号を使用する必要がある場合は、次の手順に従います。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **root** として、`jmxmp-connector-port` の既存のポート番号を確認します。

```
$ su
Password:
# cacaoadm list-params
...
jmxmp-connector-port=11162
...
```

2 **DSCC** エージェントのポート番号を変更します。

DSCC エージェントのポート番号を変更するときは、共通エージェントコンテナを停止する必要があります。

```
# cacaoadm stop
# cacaoadm set-param jmxmp-connector-port=new-port
# cacaoadm start
```

このコマンドの場所については、35 ページの「コマンドの場所」を参照してください。

3 **DSCC** で、サーバーの登録を解除してから、新しい**DSCC** エージェントのポート番号でそれらを再登録します。

また、新しいサーバーを作成する場合は、デフォルト以外の DSCC エージェントのポート番号を指定する必要があります。

▼ Directory Service Manager パスワードをリセットする

Directory Service Manager パスワードをリセットするには、次の手順で示すように DSCC を使用します。

1 46 ページの「**DSCC** にアクセスする」で説明するように **DSCC** にアクセスします。

2 「設定」タブをクリックし、次に「**Directory Service Manager**」を選びます。

3 パスワードを変更する **Directory Service Manager** の名前をクリックします。

4 プロパティー画面で新しいパスワードを入力します。

新しいパスワードをもう一度「パスワードの確認」フィールドに入力して確認します。［OK］をクリックして、変更を保存します

▼ DSCC セッションの自動タイムアウト遅延を延長する

DSCC セッションは、一定の時間が経過するとタイムアウトになり、ユーザーは DSCC からログアウトさせられます。タイムアウト遅延を延長するには、次の手順に従います。この手順では、DSCC および Sun Java Web Console 内のほかのすべてのアプリケーションのタイムアウトを延長します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **root** として、タイムアウト遅延を延長します。

```
# wcadmin add -p -a ROOT session.timeout.value=mm
```

mm は、タイムアウトまでの時間 (分) です。

たとえば、タイムアウトを 2 時間に設定するには、次のように入力します。

```
$ su
Password:
# wcadmin add -p -a ROOT session.timeout.value=120
Set 1 properties for the ROOT application.
# wcadmin list -p
Shared service properties (name, value):
    session.timeout.value  120
    ...
```

2 **Sun Java Web Console** を再起動します。

```
# smcwebserver restart
Shutting down Sun Java(TM) Web Console Version 3.0.2 ...
Starting Sun Java(TM) Web Console Version 3.0.2 ...
The console is running.
```

これらのコマンドの場所については、35 ページの「コマンドの場所」を参照してください。

DSCC に対するフェイルオーバーの設定

DSCC には、DSCC に登録されているサーバーが表示されます。

DSCC がインストールされているマシンで問題が発見された場合は、DSCC を別のマシンにインストールし、次にサーバーを再登録します。ただし、これにはかなり時間がかかる可能性があります。DSCC を使ってサーバーにすぐにアクセスする場合は、DSCC フェイルオーバーを設定できます。

DSCC フェイルオーバーを設定する場合は、次のような点を考慮に入れてください。

- 登録済みサーバーのすべての情報は、DSCC レジストリに格納される。このレジストリは Directory Server インスタンスである。管理コマンド `dsadm` および `dsconf` で、レジストリを管理できます。
- DSCC レジストリには、次のようなデフォルトの特性があります。

サーバーインスタンス	Solaris — `/var/opt/SUNWdsee/dscc6/dcc/ads` Linux および HP-UX — `/var/opt/sun/dscc6/dcc/ads` Windows — `C:\Program Files\Sun\DSEE\var\dscc6\dcc\ads`
サフィックス	`cn=dscc`
ポート	LDAP `3998`、LDAPS `3999`

- DSCC を複数のマシンにインストールしたあとは、DSCC レジストリサフィックス間にレプリケーションを設定できます。第 10 章で説明するレプリケーションコマンド行の手順に従います。あるいは、単純なレプリケーション設定の例については、`dsconf`(1M) のマニュアルページを参照してください。

  レプリケーションを設定したら、別のマシンから DSCC に登録されているサーバーと同じサーバーにアクセスできます。たとえば、`host1` および `host2` 上の DSCC レジストリサフィックス間にレプリケーションを設定する場合は、`https://host1:6789` または `https://host2:6789` のいずれかにある DSCC で同じサーバーを管理できます。ホストで障害が発生した場合は、もう一方のホストから DSCC にアクセスします。

DSCC のトラブルシューティング

DSCC のトラブルシューティングの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「To Troubleshoot Directory Service Control Center Access」を参照してください。

Directory Server のポート番号の変更

DSCC を使うか、`dsconf set-server-prop` コマンドを使って、ディレクトリサーバーの LDAP ポートまたは LDAPS セキュアポート番号を変更できます。

ポート番号を変更する場合は、次の点に注意してください。

- ほかのユーザーがアクセスするマシンに Directory Server がインストールされている場合、Directory Server のポートを root 権限を必要としないポート番号に設定すると、このポートはほかのアプリケーションによってハイジャックされる危険にさらされることになります。ほかのアプリケーションが同じアドレスとポートのペアにバインドできることになり、そのアプリケーションが、Directory Server になりすまして要求を処理できるようになってしまいます。つまり、そのアプリケーションを使って認証プロセスで使われるパスワードを取得したり、クライアント要求やサーバー応答を変更したり、サービス拒否攻撃を生成したりすることが可能となってしまいます。こうしたセキュリティー上のリスクを回避するには、`listen-address` または `secure-listen-address` プロパティーを使用して、Directory Server が待機するインタフェース (アドレス) を指定します。

コマンド行でポート番号を変更する場合は、次の点に注意してください。

- ほかのサーバー上に定義されているレプリケーションアグリーメントで Directory Server が参照される場合、そのレプリケーションアグリーメントは新しい ポート番号を使用するよう更新する必要があります。
- 以前に DSCC を使ってサーバーを管理していた場合は、ポート番号の変更後、一時的にサーバーを表示できなくなります。サーバーを再度表示するには、サーバーの登録を解除してから、新しいポート番号を使って DSCC で再度登録する必要があります。

▼ ポート番号を変更する、ポートを使用可能にする、ポートを使用不可にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

注 – 設定を変更したあとは、変更を有効にするためにサーバーを再起動してください。

---

1 ポートの既存の設定を確認します。

```
$ dsconf get-server-prop -h host -p port port-type
```

ここで、*port-type* は次のいずれかです。

`ldap-port`	LDAP デフォルトポート
`ldap-secure-port`	LDAPS セキュアポート
`dsml-port`	DSML デフォルトポート

`dsml-secure-port`　DSMLセキュアポート

たとえば、LDAPSセキュアポートを表示するには、次のように入力します。

```
$ dsconf get-server-prop -h host1 -p 2501 ldap-secure-port
Enter "cn=Directory Manager" password:
ldap-secure-port : 2511
```

返される結果が整数の場合、ポートは使用可能です。返される結果が`disabled`の場合、ポートは使用不可です。

注 – また、`dsadm`を使ってLDAPデフォルトポートとLDAPSセキュアポートのリストを表示することもできます。

---

2 必要に応じて、ポート番号を変更するか、ポートを使用可能にします。

```
$ dsconf set-server-prop -h host -p port port-type:new-port
```

たとえば、LDAPポート番号を1389から1390に変更するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host1 -p 1389 ldap-port:1390
```

ポート番号2250のDSMLセキュアポートを使用可能にするには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host1 -p 1389 dsml-secure-port:2250
```

3 必要に応じて、ポートを使用不可にします。

```
$ dsconf set-server-prop -h host -p port port-type:disabled
```

たとえば、DSMLセキュアポートを使用不可にするには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host1 -p 1389 dsml-secure-port:disabled
```

DSMLの設定

Directory Serverは、LDAP (Lightweight Directory Access Protocol)で要求を処理するほか、DSMLv2 (Directory Service Markup Language version 2)で送信された要求も処理します。DSMLは、クライアントがディレクトリ操作をエンコードするためのもう1つの方法です。サーバーはDSMLを、同じアクセス制御とセキュリティー機能のすべてを持つほかの要求として処理します。このDSML処理により、さまざまな種類のクライアントがディレクトリの内容にアクセスできるようになります。

Directory Server では、ハイパーテキスト転送プロトコル (HTTP/1.1) で DSMLv2 を使用できます。また、DSML の内容を転送するためのプログラミングプロトコルとして SOAP (Simple Object Access Protocol) version 1.1 が使われます。これらのプロトコルの詳細と DSML 要求の例については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 10 章「Directory Server DSMLv2」を参照してください。

この節の内容は、次のとおりです。

- 78 ページの「DSML-over-HTTP サービスを有効にする」
- 79 ページの「DSML-over-HTTP サービスを無効にする」
- 80 ページの「DSML のアイデンティティーマッピング」

▼ DSML-over-HTTP サービスを有効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **DSML** モードを on に設定します。

   ```
   $ dsconf set-server-prop -h host -p port dsml-enabled:on
   ```

2. セキュア **DSML** ポートを設定します。

   ```
   $ dsconf set-server-prop -h host -p port dsml-secure-port:port
   ```

3. セキュアではない **DSML** ポートを設定します。

   ```
   $ dsconf set-server-prop -h host -p port dsml-port:port
   ```

   デフォルトでは、このポートは `disabled` に設定されます。

4. サーバーを再起動します。

   ```
   $ dsadm restart instance-path
   ```

次の手順　ユーザーによって定義されたパラメータと属性値に従い、DSML クライアントは次の URL を使用して、このサーバーに要求を送信できます。

`http://`*host*`:`*DSML-port*`/` *relative-URL*

`https://`*host*`:`*secure-DSML-port* `/`*relative-URL*

注 – *relative-URL* の読み取りおよび設定は、`dsml-relative-root-url` プロパティーを使用して行うことができます。

▼ DSML-over-HTTP サービスを無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **DSML** モードを `off` に設定します。

   ```
   $ dsconf set-server-prop -h host -p port dsml-enabled:off
   ```

2. セキュア **DSML** ポートを `disabled` に設定します。

   ```
   $ dsconf set-server-prop -h host -p port dsml-secure-port:disabled
   ```

3. サーバーを再起動します。

   ```
   $ dsasm restart instance-path
   ```

▼ DSML セキュリティーを設定する

DSML 要求を受け入れるために必要なセキュリティーレベルを設定できます。このためには、DSML クライアント認証を設定する必要があります。

- **DSML** クライアント認証モードを設定します。

  ```
  $ dsconf set-server-prop -h host -p port dsml-client-auth-mode:dsml-mode
  ```

  デフォルトでは、`dsml-client-auth-mode` プロパティーは `client-cert-first` に設定されます。

  dsml-mode は、次のいずれかです。

  - `http-basic-only` - これはデフォルト値です。サーバーは HTTP Authorization ヘッダーの内容を使用して、ディレクトリ内のエントリに対応付けるユーザー名を見つけます。このプロセスとその設定は SSL によって暗号化されますが、クライアント証明書は使用しません。これについては、80 ページの「DSML のアイデンティティーマッピング」で説明しています。
  - `client-cert-only`: サーバーはクライアント証明書の資格情報を使用してクライアントを識別します。この設定では、DSML クライアントはすべて、セキュリティー保護された HTTPS ポートを使用して DSML 要求を送信し、証明書を提示する必要があります。サーバーは、このクライアント証明書がディレクトリ内のエントリと一致するかどうかを確認します。詳細は、第 5 章を参照してください。
  - `client-cert-first`: クライアント証明書が提示された場合、サーバーはまずその証明書を使用してクライアントの認証を試みます。それ以外の場合は、Authorization ヘッダーの内容を使用してクライアントを認証します。

HTTP 要求に証明書も Authorization ヘッダーもない場合は、匿名バインドを使用して DSML 要求を実行します。匿名バインドは、次の場合にも使われます。

- `client-cert-only` が指定されている場合で、クライアントから有効な Authorization ヘッダーが提示されたが、証明書は提示されていないとき。
- `http-basic-only` が指定されている場合で、クライアントから有効な証明書が提示されたが、Authorization ヘッダーは提示されていないとき。

証明書が提示されていてもエントリと一致しない場合や、HTTP Authorization ヘッダーが指定されていてもユーザーのエントリに対応付けることができない場合、クライアント認証方式にかかわらず DSML 要求は拒否され、エラーメッセージ 403「Forbidden」が返されます。

DSML のアイデンティティーマッピング

証明書を使わない基本認証を実行するときは、Directory Server はアイデンティティーマッピングというメカニズムを使用して、DSML 要求を受け入れるときに使うバインド DN を決定します。このメカニズムでは、HTTP 要求の Authorization ヘッダーから情報が抽出され、バインドに使うアイデンティティーを決定します。

DSML/HTTP のデフォルトのアイデンティティーマッピングは、サーバー設定の次のエントリで指定されます。

```
dn: cn=default,cn=HTTP-BASIC,cn=identity mapping,cn=config
objectClass: top
objectClass: nsContainer
objectClass: dsIdentityMapping
cn: default
dsSearchBaseDN: ou=people
dsSearchFilter: (uid=${Authorization})
```

この設定は、サーバーでは、Directory Server サフィックスに格納された DN の `uid` 値として、HTTP ユーザー ID を使用するべきであることを示します。たとえば、HTTP ユーザーが `bjensen` の場合、サーバーは、DN `uid=bjensen,ou=people` を使ってバインドを実行しようとします。

したがって、マッピングを適切に処理するためには、`dsSearchBaseDN` の値を完全にする必要があります。たとえば、`dsSearchBaseDN` の値を `ou=people,dc=example,dc=com` に変更することができます。そのあと、HTTP ユーザーが `bjensen` の場合、サーバーは、DN `uid=bjensen,ou=people,dc=example,dc=com` を使ってバインドを実行しようとします。

```
dn: cn=default,cn=HTTP-BASIC,cn=identity mapping,cn=config
objectClass: top
objectClass: nsContainer
```

```
objectClass: dsIdentityMapping
cn: default
dsSearchBaseDN: ou=people,dc=example,dc=com
dsSearchFilter: (uid=${Authorization})
```

マッピングエントリ属性 `dsSearchFilter` 内では、`${`*header*`}` という形式のプレースホルダを使用できます。ここで、*header* は HTTP ヘッダーの名前です。

DSML マッピングでもっともよく使われるヘッダーは、次のとおりです。

`${Authorization}`	この文字列は、HTTP Authorization ヘッダーに格納されているユーザー名で置き換えられます。Authorization ヘッダーにはユーザー名とパスワードの両方が格納されていますが、このプレースホルダにはユーザー名だけが入ります。
`${From}`	この文字列は、HTTP From ヘッダーに格納されている電子メールアドレスで置き換えられます。
`${host}`	この文字列は、DSML 要求の URL に含まれるホスト名とポート番号 (サーバーのホスト名とポート番号) に置き換えられます。

DSML 要求で別の種類のアイデンティティーマッピングを実行するには、HTTP ヘッダーのアイデンティティーマッピングを新しく定義します。

▼ HTTP ヘッダーの新しいアイデンティティーマッピングを定義する

1. デフォルトの **DSML-over-HTTP** アイデンティティーマッピングを編集するか、このプロトコル用のカスタムマッピングを作成します。

   マッピングエントリは、エントリ `cn=HTTP-BASIC,cn=identity mapping,cn=config` の下になければなりません。

   91 ページの「ldapmodify によるエントリの追加」で説明するように、`ldapmodify` コマンドを使ってこのエントリをコマンド行から追加します。

2. **Directory Server** を再起動して、新しいマッピングを有効にします。

   最初にカスタムマッピングが評価されます。カスタムマッピングが正常に行われていない場合は、デフォルトマッピングが評価されます。どのマッピングを使用しても、DSML 要求の認証に使うバインド DN を特定できない場合、DSML 要求は禁止され拒否されます (エラー 403)。

読み取り専用としてのサーバーの設定

ディレクトリ内のそれぞれのサフィックスは、単独で読み取り専用モードにすることができ、特定のリフェラルが 1 つ定義されていれば、それを返すことができます。また、Directory Server には、すべてのサフィックスに適用されるサーバー用の読み取り専用モードがあり、グローバルリフェラルが 1 つ定義されている場合は、それを返すことができます。

サーバーの読み取り専用モードは、管理者がサフィックスのインデックスを作成し直すときなどに、途中でディレクトリの内容が変更されないようにするために使います。このため、サーバーの読み取り専用モードは、次のエントリには適用されません。

- `cn=config`
- `cn=monitor`
- `cn=schema`

これらのエントリは、読み取り専用の設定とは無関係に、管理者以外のユーザーによる変更に対して常にアクセス制御命令 (ACI) によって保護してください (第 6 章を参照)。グローバルな読み取り専用モードでは、ディレクトリマネージャーで開始された更新操作を含む、ディレクトリ内のほかのすべてのサフィックスに対する更新操作が行われません。

読み取り専用モードでは、このモードが適用されているサフィックスについてはレプリケーションも中断されます。レプリケーションの対象となる変更がマスターレプリカに加えられることはなくなります。ただし、読み取り専用モードが適用される前に加えられた変更は、引き続きレプリケートされます。読み取り専用モードが無効になるまでは、コンシューマレプリカは更新を受け取りません。マルチマスターレプリケーション環境のマスターは、レプリケーションの対象となる変更が加えられることはなく、他のマスターから更新を受け取りません。

▼ サーバー読み取り専用モードを有効または無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 グローバルな読み取り専用モードを有効にします。

```
$ dsconf set-server-prop -h host -p port read-write-mode:read-only
```

2 準備ができたら、読み取り専用モードを無効にします。

```
$ dsconf set-server-prop -h host -p port read-write-mode:read-write
```

メモリーの設定

この節では、さまざまなタイプのメモリーについて説明します。さまざまなタイプのキャッシュの説明と、キャッシュチューニングの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 5 章「Directory Server Data Caching」を参照してください。

キャッシュのプライミング (priming)

キャッシュのプライミングとは、キャッシュをデータで満たすことを意味するため、それ以降の Directory Server の動作は、立ち上げ時のパフォーマンスではなく、通常動作時のパフォーマンスを反映します。キャッシュプライミングを使うと、ベンチマーク時に再現性のある結果に到達させるのに便利です。また、可能性のある最適化を測定し分析するためにも便利です。

可能なかぎり、キャッシュのプライミングは積極的には行わないでください。キャッシュのプライミングは、パフォーマンスを測定する前に、Directory Server を使って通常または一般的なクライアント対話によって行なってください。

データベースキャッシュのプライミングツールについては、`http://www.slamd.com` を参照してください。

▼ データベースキャッシュを変更する

注意 – キャッシュを変更すると、サーバーのパフォーマンスに重大な影響を与える可能性があります。キャッシュを変更する場合は十分に注意してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 現在のデータベースのキャッシュレベルを取得します。

```
$ dsconf get-server-prop -h host -p port db-cache-size
```

2. データベースキャッシュレベルを変更します。

```
$ dsconf set-server-prop -h host -p port db-cache-size:size
```

size は、G バイト (G)、M バイト (M)、K バイト (k)、バイト (b) のいずれかの単位で表せます。マシンでサポートされるサイズを指定してください。

▼ データベースキャッシュを監視する

インストール時のキャッシュのデフォルトレベルはテスト環境に適したものであり、本稼働環境に適したものではありません。チューニング目的の場合は、サーバーのデータベースキャッシュを監視することもできます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- データベースキャッシュを監視します。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -b "cn=monitor,cn=ldbm database,cn=plugins,cn=config" "(objectclass=*)"
```

データベースキャッシュのサイズが十分に大きく、キャッシュのプライミングが終わっている場合は、ヒット率 (`dbcachehitratio`) を高くしてください。また、(`dbcachepagein`) で読み取られるページ数と (`dbcacheroevict`) で書き出されるクリーンページは、低くしてください。この場合、「高い」と「低い」というのは、配備の制約に対して相対的に高いか低いかを意味します。

▼ エントリキャッシュを監視する

チューニング目的では、1 つまたは複数のサフィックスのエントリキャッシュをチェックすることもできます。エントリキャッシュレベルを表示するには、この手順を使用します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- エントリキャッシュを監視します。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -b "cn=monitor,cn=db-name,cn=ldbm database,cn=plugins,cn=config" "(objectclass=*)"
```

サフィックスのエントリキャッシュの大きさがサフィックス内の大半のエントリを保持するには十分な場合で、キャッシュが事前準備されている場合、ヒット率 (`entrycachehitratio`) は高くしてください。

キャッシュを事前準備した場合は、以前に空だったエントリキャッシュが満たされると、エントリキャッシュのサイズ (`currententrycachesize`) がエントリキャッシュの最大サイズ (`maxentrycachesize`) に近づいていることがわかります。理想としては、エントリ内のサイズ (`currententrycachecount`) は、サフィックス内のエントリの総数 (`ldapentrycachecount`) と等しいか非常に近いものにしてください。

▼ エントリキャッシュを変更する

注意 – キャッシュを変更すると、サーバーのパフォーマンスに重大な影響を与える可能性があります。キャッシュを変更する場合は十分に注意してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 現在のエントリキャッシュレベルを取得します。

   ```
   $ dsconf get-suffix-prop -h host -p port suffix-DN entry-cache-count entry-cache-size
   ```

2. キャッシュのエントリ数を変更します。

   ```
   $ dsconf set-suffix-prop -h host -p port suffix-DN entry-cache-count:integer
   ```

   integer は、キャッシュに格納されるエントリの数です。

3. エントリキャッシュのサイズを変更します。

   ```
   $ dsconf set-suffix-prop -h host -p port suffix-DN entry-cache-size:size
   ```

   size は、G バイト (G)、M バイト (M)、K バイト (k)、バイト (b) のいずれかの単位で表されるキャッシュサイズです。マシンでサポートされるサイズを指定してください。

▼ ヒープメモリーのしきい値を設定する

`nsslapd` プロセスで使用するヒープメモリーの量を制限する場合は、動的メモリーのフットプリントのしきい値を設定できます。このしきい値は、リソースが共有されているか sparse 状態であるマシン上で Directory Server が実行中の場合に設定することもできます。

注 – このしきい値は、Solaris™ および Linux プラットフォームのみに設定できます。

メモリーサイズの設定の詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「Directory Server とメモリー」を参照してください。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

注 – デフォルトでは、heap-high-threshold-size および heap-low-threshold-size プロパティーは undefined に設定されます。

1 最大ヒープの高メモリーしきい値を設定します。

```
$ dsconf set-server-prop -h host -p port heap-high-threshold-size:value
```

value は、undefinedか、またはGバイト (G)、Mバイト (M)、Kバイト (k)、バイト (b) のいずれかの単位で表されるメモリーサイズです。マシンでサポートされるサイズを指定してください。

heap-high-threshold-size に使用する値に関する推奨事項については、server(5dsconf) のマニュアルページを参照してください。

2 オプションで、最大ヒープ低メモリーしきい値を設定します。

```
$ dsconf set-server-prop -h host -p port heap-low-threshold-size:value
```

value は、undefinedか、またはGバイト (G)、Mバイト (M)、Kバイト (k)、バイト (b) のいずれかの単位で表されるメモリーサイズです。マシンでサポートされるサイズを指定してください。

heap-low-threshold-size に使用する値に関する推奨事項については、server(5dsconf) のマニュアルページを参照してください。

各クライアントアカウントのリソース制限の設定

各クライアントアカウントに対する、サーバー上の検索操作のリソース制限を制御できます。このような制限をアカウントのオペレーショナル属性に設定すると、それらの制限は、クライアントがディレクトリへのバインドに使用するアカウントに基づいて、Directory Server で実施されます。

設定できる制限は次のとおりです。

- 検索制限は、検索処理で参照されるエントリの最大数を指定する。
- サイズ制限は、検索操作に応答して返されるエントリの最大数を指定する。
- 時間制限は、検索処理に費やせる最大時間を指定する。
- アイドルタイムアウトは、接続が切断されるまでに、クライアント接続がアイドル状態でいられる最大時間を指定する。

注 – デフォルトでは、ディレクトリマネージャーは無制限にリソースを利用できます。

特定のユーザーアカウントに対して設定したリソース制限は、サーバー規模の設定で設定したリソース制限より優先されます。この節では、各アカウントに対するリソース制限の設定について説明します。

この節で示す例では、エントリの属性に直接リソース制限を設定します。また、サービスクラス (CoS) メカニズムを使ってアカウントにリソース制限を設定することもできます。クライアントアプリケーション用にエントリが取得されるとき、CoS メカニズムによって計算された属性が生成されます。CoS の定義の詳細は、231 ページの「サービスクラス」を参照してください。

▼ ヒープメモリーのしきい値を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 `dsconf get-server-prop` コマンドを使用して、リソース制限サーバープロパティーを読み取ります。

```
$ dsconf get-server-prop -h host -p port look-through-limit search-size-limit \
 search-time-limit idle-timeout
look-through-limit  :  5000
search-size-limit   :  2000
search-time-limit   :  3600
idle-timeout        :  none
```

この出力は、検索により最大 5000 エントリが調べられ、最大 2000 エントリが返され、検索の処理には最大 1 時間 (3600 秒) が費やされることを示しています。

2 検索制限を変更します。

```
$ dsconf set-server-prop -h host -p port look-through-limit:integer
```

integer は、検索処理で参照されるエントリの最大数です。

3 検索サイズ制限を変更します。

```
$ dsconf set-server-prop -h host -p port search-size-limit:integer
```

integer は、検索処理で返されるエントリの最大数です。

4 検索時間制限を変更します。

```
$ dsconf set-server-prop -h host -p port serach-time-limit:integer
```

integer は、検索処理に費やせる最大時間です。

5 アイドルタイムアウトを変更します。

```
$ dsconf set-server-prop -h host -p port idle-timeout:integer
```

integer は、接続が切断されるまでに、クライアント接続がアイドル状態でいられる最大時間を指定します。

◆ ◆ ◆ 第 4 章

Directory Server のエントリ

この章では、ディレクトリ内のデータエントリを管理する方法について説明します。また、リフェラルを設定する方法と属性値を暗号化する方法についても説明します。

ディレクトリの配備を計画する場合には、ディレクトリに格納するデータの種類を把握する必要があります。エントリの作成とデフォルトスキーマの変更に入る前に、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の関連する章に目を通すようにしてください。

適切な ACI (アクセス制御命令) が定義されていない場合、ディレクトリは変更できません。詳細は、第 6 章を参照してください。

この章の内容は次のとおりです。

エントリの管理

エントリを管理するための最良な方法は、状況によって異なります。

- DSCC を主に管理用に使用していて、検索や変更を行いたいエントリは少ししかない場合は、DSCC を使用します。DSCC の詳細については、45 ページの「Directory Service Control Center のインタフェース」を参照してください。
- Directory Server で管理タスクを実行せず、検索や変更を行いたいエントリが少ししかない場合は、Directory Editor を使用します。Directory Editor の詳細は、『Sun Java System Directory Editor 1 2005Q1 Installation and Configuration Guide』を参照してください。

- 多数のエントリを検索したり変更したりする場合は、コマンド行ユーティリティー `ldapmodify` および `ldapdelete` を使用します。

DSCC によるエントリの管理

DSCCでは、エントリの読み取り可能なすべての属性を表示し、書き込み可能な属性を編集できます。また、属性の追加と削除、複数値属性の設定、エントリのオブジェクトクラスの管理も行えます。DSCC によるエントリの管理方法の詳細は、DSCC のオンラインヘルプを参照してください。DSCC の概要については、45 ページの「Directory Service Control Center のインタフェース」を参照してください。

Directory Editor によるエントリの管理

Directory Editor は、管理者とエンドユーザーがデータを検索、作成、編集するための、使いやすいディレクトリ編集ツールです。扱えるデータの種類は、ユーザー、グループ、およびコンテナです。

`ldapmodify` および `ldapdelete` によるエントリの管理

コマンド行ユーティリティー `ldapmodify` および `ldapdelete` には、ディレクトリの内容を追加、編集、削除するための完全な機能が用意されています。これらのユーティリティーを使用して、サーバーの設定エントリと、ユーザーエントリに含まれるデータの両方を管理できます。これらのユーティリティーは、1 つまたは複数のディレクトリの一括管理を実行するためのスクリプトの作成にも利用できます。

`ldapmodify` コマンドと `ldapdelete` コマンドは、このマニュアル全体の手順で使用されます。次の節で、これらの手順の実行に必要な基本操作について説明します。`ldapmodify` コマンドと `ldapdelete` コマンドの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

コマンド行ユーティリティーへの入力は、常に LDIF 形式で行います。この形式の入力は、コマンド行から直接指定できるだけでなく、入力ファイルからも行うことができます。次の節では、LDIF 入力について説明し、それ以降の節では各種変更処理で使われる LDIF 入力について説明します。

LDIF 入力の正しい形式については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Guidelines for Providing LDIF Input」を参照してください。

これらの基本操作については、次の節で説明します。

ldapmodify によるエントリの追加

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

ldapmodify の -a オプションを使用して、ディレクトリに 1 つまたは複数のエントリを追加できます。次の例では、まずユーザーが属するエントリを作成し、次にユーザーエントリを作成しています。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: ou=People,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: People
description: Container for user entries

dn: uid=bjensen,ou=People,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetorgPerson
uid: bjensen
givenName: Barbara
sn: Jensen
cn: Babs Jensen
telephoneNumber: (408) 555-3922
facsimileTelephoneNumber: (408) 555-4000
mail: bjensen@example.com
userPassword: secret
```

-D オプションと -w オプションは、これらのエントリの作成に必要な権限を持つユーザーのバインド DN とパスワードを指定します。-a オプションは、LDIF に指定されているすべてのエントリが追加されることを示します。各エントリは DN と属性値でリストされ、エントリとエントリの間には空白行が挿入されます。ldapmodify ユーティリティーは、入力されるすべてのエントリを順番に作成し、エラーが発生した場合は、それをレポートします。

慣例により、エントリの LDIF には次の属性が指定されます。

1. エントリの DN。
2. オブジェクトクラスのリスト。
3. 1 つまたは複数のネーミング属性。これは DN で使用される属性で、必須属性である必要はありません。
4. すべてのオブジェクトクラスの必須属性。
5. エントリに指定する、許可されているその他の属性。

userpassword 属性の値を入力するときは、パスワードをクリアテキストで指定します。サーバーはこの値を暗号化し、暗号化された値だけが格納されます。LDIF ファイルに表示されるクリアテキストのパスワードを保護するために、読み取りアクセス権を制限してください。

-a オプションを必要としない、別の形式の LDIF をコマンド行に指定することもできます。この形式の利点は、エントリを追加する文と、次の例で説明するエントリを変更する文を組み合わせて指定できることです。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: ou=People,dc=example,dc=com
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: People
description: Container for user entries

dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: add
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetorgPerson
uid: bjensen
givenName: Barbara
sn: Jensen
cn: Barbara Jensen
telephoneNumber: (408) 555-3922
facsimileTelephoneNumber: (408) 555-4000
mail: bjensen@example.com
userPassword: secret
```

changetype: add キーワードは、指定の DN を持つエントリが、それ以後のすべての属性を持った状態で作成されることを示します。それ以外のすべてのオプションと LDIF の表記は、この節の前のほうで説明した表記と同じです。

どちらの例でも、-f *filename* オプションを使うことで、端末からの入力の代わりにファイルから LDIF を読み取ることができます。LDIF ファイルには、-a オプションの使用の有無に応じて、端末から入力する場合と同じ形式で情報を指定する必要があります。

ldapmodify によるエントリの変更

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

既存のエントリの属性と属性値を追加、置換、または削除するときは、changetype: modify キーワードを使います。changetype: modify を指定する場合は、エントリの変更方法を示す、1 つまたは複数の変更操作も指定する必要があります。次の例には、3 種類の LDIF 変更操作が指定されています。

```
dn: entryDN
changetype: modify
add: attribute
attribute: value...
-
replace: attribute
attribute: newValue...
-
delete: attribute
[attribute: value]
...
```

同じエントリに対する操作を区切るときはハイフン (-) を使い、異なるエントリに対する操作セットを区切るときは空白行を使います。各操作の対象となる *attribute*: *value* ペアを複数指定することもできます。

属性値の追加

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

次の例は、同じ add LDIF 文を使用して、既存の複数値属性と、まだ存在しない属性に値を追加する方法を示しています。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: cn
cn: Babs Jensen
-
```

```
add: mobile
mobile: (408) 555-7844
```

次のいずれかの場合は、処理が失敗し、エラーが返されることがあります。

- 指定した値がその属性にすでに存在する。
- 値が、属性に定義されている構文に準拠していない。
- エントリのオブジェクトクラスが、その属性タイプを必要としないか、許可しない。
- 属性タイプが複数値ではなく、その属性にすでに値が存在する。

バイナリ属性サブタイプの使用

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

attribute;binary サブタイプは、実際の構文にかかわらず、値がバイナリデータとして LDAP 上を転送されることを示しています。このサブタイプは、userCertificate など、LDAP 文字列表現を持たない複雑な構文用に設計されたものです。この目的以外でバイナリサブタイプを使用しないでください。

ldapmodify コマンドで使用する場合は、どの LDIF 文でも、属性名に適切なサブタイプを追加できます。

バイナリ値を入力するには、LDIF テキストに直接入力するか、別のファイルから読み取ります。次の例は、ファイルから読み取る LDIF の構文を示しています。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
version: 1
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: userCertificate;binary
userCertificate;binary:< file:///local/cert-file
```

ファイル名の指定に :< 構文を使用するには、LDIF 文を version: 1 という行から開始する必要があります。ldapmodify がこの文を処理するときに、このツールは、指定ファイルの内容全体から読み取った値を属性に設定します。

言語サブタイプを持つ属性の追加

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

属性の言語とふりがなのサブタイプは、ローカライズされた値を特定します。属性に対して言語サブタイプを指定すると、そのサブタイプが属性名に次のように追加されます。

attribute;lang-*CC*

ここで、*attribute* は既存の属性タイプを示し、*cc* は言語を特定する 2 文字の国コードを示します。オプションとして、言語サブタイプにふりがなのサブタイプを追加し、ローカライズされた値の発音表記を指定することもできます。この場合、属性名は次のようになります。

attribute;lang-*CC*;phonetic

サブタイプを持つ属性に対して処理を行うには、そのタイプを明示的に一致させる必要があります。たとえば、lang-fr の言語サブタイプを持つ属性値を変更する場合は、次の例に示すように、変更操作に lang-fr を含める必要があります。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: homePostalAddress;lang-fr
homePostalAddress;lang-fr: 34, rue de la Paix
```

注 – 属性値に ASCII 以外の文字が含まれている場合は、UTF-8 で符号化する必要があります。

属性値の変更

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

次の例に、LDIF で replace 構文を使用して属性の値を変更する方法を示します。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
replace: sn
sn: Morris
-
replace: cn
cn: Barbara Morris
cn: Babs Morris
```

指定された属性の現在の値が削除され、指定された値が追加されます。

属性値を変更した後、ldapsearch コマンドで変更を確認します。

属性値内の後続の空白文字

属性値を変更する場合、値末尾の後ろに空白文字を残さないでください。値の後ろに空白文字を残すと、その値が base-64 方式で表示される場合があります (`34xy57eg` など)。

属性値の後ろに空白文字がある場合、その空白文字は属性値の一部としてエンコードされます。DSCC コンソールまたは `ldapsearch` コマンドを使用して変更を確認する場合、値はプレーンテキストで表示されますが、base-64 方式のテキストで表示される場合もあります。これは、使用している Directory Server クライアントにより異なります。

属性値の削除

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

次の例は、属性全体、または複数値属性の 1 つの値だけを削除する方法を示しています。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
delete: facsimileTelephoneNumber
-
delete: cn
cn: Babs Morris
```

attribute: value のペアを指定せずに `delete` 構文を使用すると、属性のすべての値が削除されます。*attribute: value* のペアを指定した場合は、その値だけが削除されます。

複数値属性の 1 つの値の変更

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

`ldapmodify` コマンドを使用して、複数値属性の 1 つの値を変更するには、次の例に示すように、2 段階の処理が必要です。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
delete: mobile
```

```
mobile: (408) 555-7845
-
add: mobile
mobile: (408) 555-5487
```

ldapdelete によるエントリの削除

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

ディレクトリからエントリを削除するときは、`ldapdelete` コマンド行ユーティリティーを使います。このユーティリティーは、ディレクトリサーバーにバインドし、DN を基にした 1 つまたは複数のエントリを削除します。指定のエントリを削除する権限を持つバインド DN を指定する必要があります。

子エントリのあるエントリは削除できません。LDAP プロトコルでは、親を持たない子エントリが存在する状況を禁止しています。たとえば、組織単位に属するすべてのエントリを先に削除しない限り、組織単位エントリは削除できません。

次の例では、組織単位には 1 つのエントリしか存在しません。このエントリを削除すれば、その親エントリを削除できます。

```
$ ldapdelete -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
uid=bjensen,ou=People,dc=example,dc=com
ou=People,dc=example,dc=com
```

ldapmodify によるエントリの削除

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

`ldapmodify` ユーティリティーを使用する場合は、`changetype: delete` キーワードを利用してエントリを削除することもできます。前の節で説明したように、`ldapdelete` の利用時と同じ制限事項がすべて適用されます。LDIF 構文を使用してエントリを削除する利点は、1 つの LDIF ファイルで複数の処理を組み合わせて実行できることです。

次の例は、前述の例と同じ削除処理を行います。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: delete

dn: ou=People,dc=example,dc=com
changetype: delete
```

ldapsearch によるエントリの検索

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

ldapsearch コマンド行ユーティリティーは、ディレクトリエントリの検索と取得に使用できます。ldapsearch ユーティリティーは Solais プラットフォームに付属しているユーティリティーではありませんが、Directory Server Resource Kit の一部です。

ldapsearch の使い方、共通の ldapsearch オプション、受け入れられる形式、および例については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference 』を参照してください。

▼ ldapmodify を使用してエントリを移動または名前変更する

この手順では、DN 変更操作を使用します。この操作を始める前に、100 ページの「DN の変更操作に関するガイドラインと制限」の節に記載されている内容を十分に理解してください。

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

注 – グループの uniquemember であるエントリの DN を変更するときは、参照の完全性プラグインを有効にしておいてください。参照の完全性により、エントリが移動されたときにグループメンバーが調整されます。参照の完全性を有効にし、設定する方法の詳細は、244 ページの「参照の完全性プラグインを設定する」を参照してください。

---

1 エントリをある親から別の親に移動する場合は、親エントリの **ACI** 権限を拡張します。

- 移動するエントリの現在の親エントリでは、ACI で export 操作が許可されていることを、allow (export ...) 構文を使用して確認します。
- エントリの移動先の親エントリでは、ACI で import 操作が許可されていることを、allow (import ...) 構文を使用して確認します。

ACIU の使い方については、第 6 章を参照してください。

2 **DN 変更操作がグローバルに有効か、少なくとも移動操作で影響を受けるサフィックスに対して有効であることを確認します。**

Directory Server の以前のリリースとの互換性を保つため、デフォルトでは DN 変更操作は有効ではありません。

すでに DN 変更操作があらかじめ有効になっている場合は、次の手順に進みます。

DN 変更操作をサーバーに対してグローバルに有効にするには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port moddn-enabled:on
```

3 **`ldapmodify` コマンドを実行します。**

この手順では、DN 変更操作を使用します。次のいずれかの操作を行います。

- エントリを移動します。

  たとえば、次のコマンドで、エントリ `uid=bjensen` がパート社員のサブツリーである `ou=Contractors,dc=example,dc=com` から、従業員のサブツリーである `ou=People,dc=example,dc=com` に移動します。

  ```
  $ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
  Enter bind password:
  dn: uid=bjensen,ou=Contractors,dc=example,dc=com
  changetype: modrdn
  newrdn: uid=bjensen
  deleteoldrdn: 0
  newsuperior: ou=People,dc=example,dc=com
  ```

- エントリの名前を変更します。

  たとえば、次のコマンドで、エントリ `uid=bbjensen` の名前が `uid=bjensen` に変更されます。

  ```
  $ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
  Enter bind password:
  dn: uid=bbjensen,ou=People,dc=example,dc=com
  changetype: modrdn
  newrdn: uid=bjensen
  deleteoldrdn: 1
  ```

LDIF 文を作成する際は、次の属性に注意してください。

- `dn` - 名前変更または移動するエントリを指定します。
- `changetype: modrdn` - DN の変更操作の使用を指定します。
- `newrdn` - 新しいネーミング属性を指定します。
- `deleteoldrdn`: 以前のネーミング属性をエントリから削除するかどうかを指定します (`1` であれば削除、`0` であれば削除しない)。

ネーミング属性がエントリ定義で必須である場合、この属性はエントリから削除できないことに注意してください。

- `newsuperior`: エントリの新しい上位属性を指定します。

`ldapmodify` コマンドとそのオプションの詳細は、`ldapmodify`(1) のマニュアルページを参照してください。

4 多数のエントリが含まれているサブツリーを移動したり名前変更したりするときに、リソース制限エラーが発生した場合は、データベースで使用できるロック数を増やします。

```
$ dsconf set-server-prop -h host -p port db-lock-count:value
```

このプロパティーを変更する場合は、変更を有効にするためにサーバーを再起動する必要があります。

DN の変更操作に関するガイドラインと制限

DN の変更操作を、前の節で説明したように使用する場合は、次の節で説明するガイドラインにしたがってください。

DN の変更操作に関する一般的なガイドライン

- DN の変更操作は、エントリのサフィックス間の移動や、ルートサフィックスの名前変更または移動には使用しないでください。
- 実行中の Directory Server が 5.2 2005Q1 以降であることを確認してください。Directory Server 5.2 2005Q1 以前のバージョンの Directory Server では、DN の変更操作を使用できません。
- `entryid` オペレーショナル属性は、内部使用専用の属性のため、DN 変更操作では使用しないでください。エントリの `entryid` 属性は、エントリが移動すると変更されます。
- DN の変更操作はサーバー上のすべてのサフィックスに対してグローバルに、または操作を実行する各サフィックスで個別に有効にすることができます。デフォルトでは、DN の変更操作は無効になっています。
- サフィックス上で、DN の変更操作を実行するには、そのサフィックスの ACI 権限を拡張します。`Import` アクセス権を許可すると、指定された DN にエントリをインポートすることができます。`Export` アクセス権を許可すると、指定された DN からエントリをエクスポートすることができます。
- DN の変更操作を実行する前に、この操作によってクライアント認証が無効にならないことを確認してください。クライアント証明書を参照しているエントリを移動すると、クライアント認証が無効になってしまいます。エントリの移動によって、証明書が無効になっていないかどうかを確認してください。

- DN の変更操作を実行する前に、この操作によってアプリケーションが中断されることがないようにしてください。エントリの名前変更または移動によりいくつかのサフィックスに影響が及ぶ場合があります。あるいはエントリの次の特性が変更される可能性があります。
  - エントリのフィルタが適用されたロールの範囲。
  - エントリの入れ子のロール。入れ子のロールにはフィルタ化されたロールが格納されます。
  - エントリのダイナミックグループのメンバーシップ。

レプリケーション環境での DN の変更操作に関する一般的なガイドライン

注意 – 次の要件を満たさずに DN の変更操作を実行すると、レプリケーションが中断され、ディレクトリサービスが停止する可能性があります。

- レプリケーショントポロジのすべてのサーバーが Directory Server 5.2 以降を実行していることを確認します。Directory Server 5.2 より前のバージョンの Directory Server では、DN の変更操作を使用できません。
- レプリケーショントポロジのすべてのサーバーで、DN の変更操作を有効にします。DN の変更操作がマスターサーバーでサポートされていてもコンシューマサーバーでサポートされていない場合、レプリケーションは失敗します。サプライヤサーバーのエラーログに、次のようなメッセージが書き込まれます。

  `Unable to start a replication session with MODDN enabled` レプリケーションを再開するには、レプリケーショントポロジを再設定してすべてのサーバーで `DN` の変更操作を有効にします。そのあと、次のいずれかの方法で、レプリケーションセッションを開始します。

  - 283 ページの「レプリケーションの更新を強制的に実行する」の手順に従います。
  - サプライヤサーバーでエントリを変更します。変更内容はコンシューマサーバーにレプリケートされます。
- トポロジのすべてのマスターレプリカで、参照の完全性プラグインを有効にし設定します。このアクションにより、サーバーがグループとロールに対して参照の完全性を維持できます。参照の完全性を有効にし、設定する方法の詳細は、244 ページの「参照の完全性プラグインを設定する」を参照してください。

  DN の変更操作を実行した後で、参照の完全性プラグインにより変更内容がレプリケートされるまで待機します。

リフェラルの設定

情報をローカルに取得できない場合に、どのサーバーに接続すべきかをクライアントアプリケーションに通知するには、リフェラルを使います。リフェラルとは、リモートサフィックスへのポインタ、つまり Directory Server が結果の代わりにクライアントへ返すエントリへのポインタです。クライアントは、リフェラルで指定されたリモートサーバー上で、再度、操作を実行する必要があります。

リダイレクションは、次の 3 つの場合に行われます。

- クライアントアプリケーションがローカルサーバーに存在しないエントリを要求し、サーバーがデフォルトのリフェラルを返すよう構成されている場合。
- サフィックス全体がメンテナンスまたはセキュリティー上の理由で無効になっている場合。

  サーバーは、該当のサフィックスで定義されているリフェラルを返します。サフィックスレベルのリフェラルについては、64 ページの「リフェラルを設定し、サフィックスを読み取り専用にする」を参照してください。クライアントが書き込み処理を要求する場合、サフィックスの読み取り専用レプリカも、マスターサーバーへのリフェラルを返します。
- クライアントが特にスマートリフェラルにアクセスする場合。

  スマートリフェラルは、ユーザーが作成するエントリです。サーバーは、スマートリフェラルが定義するリフェラルを返します。

いずれの場合も、リフェラルは LDAP URL であり、ホスト名、ポート番号、およびオプションとして別のサーバー上の DN を含みます。たとえば、`ldap://east.example.com:389` です。

ディレクトリの配備でリフェラルを使用する方法の概念情報は、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

次に、ディレクトリのデフォルトリフェラルを設定する手順と、スマートリフェラルを作成および定義する手順について説明します。

デフォルトリフェラルの設定

デフォルトリフェラルは、Directory Server で管理されているサフィックスに含まれない DN に対して、操作を送信するクライアントアプリケーションに返されます。サーバーは定義されているすべてのリフェラルを返しますが、返す順序は定義されていません。

▼ デフォルトリフェラルを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 1つ以上のデフォルトリフェラルを設定するには、dsconf コマンド行ユーティリティーを使用します。

  ```
  $ dsconf set-server-prop -h host -p port suffix-DN referral-url:referral-URL
  ```

  次に例を示します。

  ```
  $ dsconf set-server-prop -h host1 -p 1389 dc=example,dc=com \
   referral-url:ldap://east.example.com:1389
  ```

スマートリフェラルの設定

スマートリフェラルを使用して、ディレクトリエントリやディレクトリツリーを、特定の LDAP URL に割り当てることができます。スマートリフェラルを使用すると、クライアントアプリケーションに、特定のサーバーや特定のサーバーにある特定のエントリを参照させることができます。

多くの場合、スマートリフェラルは別のサーバー上の同じ DN を持つ実際のエントリを指しています。ただし、同じサーバーまたは別のサーバーのあらゆるエントリに対するスマートリフェラルを定義できます。たとえば、次の DN を持つエントリをスマートリフェラルとして定義することができます。

```
uid=bjensen,ou=People,dc=example,dc=com
```

このスマートリフェラルは、east.example.com というサーバー上の次のエントリを指しています。

```
cn=Babs Jensen,ou=Sales,o=east,dc=example,dc=com
```

ディレクトリがスマートリフェラルを使用する方法は、RFC 4511 (http://www.ietf.org/rfc/rfc4511.txt (http://www.ietf.org/rfc/rfc4511.txt)) のセクション 4.1.10 に指定されている標準に準拠する必要があります。

▼ スマートリフェラルを作成および変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. スマートリフェラルを作成するには、referral オブジェクトクラスと extensibleObject オブジェクトクラスを持つエントリを作成します。

   referral オブジェクトクラスでは、ref 属性で LDAP URL を指定します。extensibleObject オブジェクトクラスは、ターゲットエントリと一致させるために、任意のスキーマ属性をネーミング属性として使用することを許可します。

たとえば、次のエントリを uid=bjensen エントリの代わりにスマートリフェラルを返すよう定義するには、次のコマンドを使用します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: referral
uid: bjensen
ref: ldap://east.example.com/cn=Babs%20Jensen,ou=Sales,o=east,dc=example,dc=com
```

注 – サーバーでは、LDAP URL で空白のあとに続く情報はすべて無視されます。このため、リフェラルとして使用する予定のある LDAP URL では、空白の代わりに %20 を使用する必要があります。その他の特殊文字はエスケープする必要があります。

スマートリフェラルを定義すると、別のサーバー上の cn=Babs Jensen エントリで、uid=bjensen エントリの修正が実際に行われます。ldapmodify コマンドは、たとえば次のように、自動的にリフェラルをたどります。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: replace
replace: telephoneNumber
telephoneNumber: (408) 555-1234
```

2 (省略可能) スマートリフェラルエントリを変更するには、ldapmodify の -M オプションを使用します。

```
$ ldapmodify -M -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: replace
replace: ref
ref: ldap://east.example.com/cn=Babs%20Jensen,ou=Marketing,o=east,dc=example,dc=com
```

有効な属性構文のチェック

Directory Server では、次の操作を行うときは常に、属性の完全性をチェックできます。

- dsadm import または dsconf import によるデータのインポート。
- LDAP または DSML によるエントリの追加、エントリの変更、またはエントリの DN の変更。

このチェックにより、属性値を IETF 勧告に準拠させることができます。準拠していないすべての属性は拒否され、エラーログに記録されます。ログメッセージには、当てはまる場合は接続および操作 ID が含まれます。

デフォルトでは、サーバーによって前述の操作の構文が自動的にチェックされます。構文チェックをオフにする場合は、次の手順に従います。

注 – 構文チェックはスキーマチェックとは異なります。スキーマチェックの詳細は、295 ページの「スキーマ検査の管理」を参照してください。

▼ 自動構文チェックをオフにする

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- 自動構文チェックをオフにするには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port check-syntax-enabled:off
```

ディレクトリエントリへの変更の記録

デフォルトでは、サーバーは、新たに作成または変更されたエントリの特別な属性を LDAP v3 仕様の指定どおりに保持します。これらの特別な属性は、サフィックス内のエントリに格納され、次のものが組み込まれます。

- `creatorsName` — 最初にエントリを作成したユーザーの DN。
- `createTimestamp` — エントリが作成されたときのタイムスタンプ (GMT 形式)。
- `modifiersName` — エントリを最後に変更したユーザーの DN。
- `modifyTimestamp` — エントリが変更されたときのタイムスタンプ (GMT 形式)。

▼ エントリ変更記録をオフにする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

注意 – エントリの変更記録をオフにすると、データが準拠しないものになります。多くのアプリケーションはこれらの属性に依存しているため、また、これらの機能を無効にすると最低限のパフォーマンスしか得られなくなるため、エントリ変更記録はオフにしないことをお勧めします。

- サーバーのエントリ変更記録をオフにします。

```
$ dsconf set-server-prop -h host -p port suffix-DN mod-tracking-enabled:off
```

属性値の暗号化

属性の暗号化はディレクトリに格納されている機密データを保護します。この機能を使用すると、エントリの特定の属性を暗号化された形式で格納するように指定できます。これにより、データベースファイル、バックアップファイル、およびエクスポートされた LDIF ファイルに格納されているデータが読み取られることを防ぎます。

この機能では、属性値は Directory Server データベースに格納される前に暗号化され、クライアントに返される前に元の値に復号化されます。クライアントと Directory Server との間でやり取りを行うときに、アクセス制御を使用してクライアントがこれらの属性に許可なくアクセスすることを防ぎ、SSL を使用して属性値を暗号化する必要があります。データセキュリティー全般のアーキテクチャー上の概要と、属性の暗号化の詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

属性の暗号化は、サーバー上で SSL が設定され有効になっている場合だけアクティブになります。ただし、デフォルトでは、どの属性も暗号化されません。属性の暗号化はサフィックスレベルで設定されます。つまり、サフィックス内でその属性が現れるすべてのエントリについて、属性が暗号化されます。ディレクトリ全体で属性を暗号化するには、すべてのサフィックスでその属性の暗号化を有効にする必要があります。

注意 – 属性を暗号化すると、サフィックスに関連付けられたすべてのデータとインデックスファイルが影響を受けます。既存のサフィックスについて暗号化設定を変更するときは、まずサフィックスの内容をエクスポートし、設定を変更してからその内容をふたたびインポートする必要があります。これらの手順は、DSCCを使用して実行できます。DSCCの使用については、45 ページの「Directory Service Control Center のインタフェース」を参照してください。

さらにセキュリティーについて考慮するならば、属性の暗号化をオンにする際に、暗号化されていない値がまだ含まれている可能性があるデータベースキャッシュファイルとデータベースログファイルを、手動で削除してください。これらのファイルを削除する手順については、109 ページの「属性の暗号化を設定する」を参照してください。

新しいサフィックスにデータを読み込むときや作成するときは、暗号化されているすべての属性を有効にしてください。

一部のエントリがネーミング属性として使用している属性を暗号化すると、DN に表示される値は暗号化されないままですが、エントリ内に格納されている値は暗号化されています。

暗号化のために `userPassword` 属性を選択することはできますが、パスワードがクリアテキストとして格納される必要がある場合を除き、実質的なセキュリティー上の利点はありません。これは、DIGEST-MD5 SASL 認証などの場合です。パスワードポリシーにパスワードの暗号化メカニズムが定義されている場合、それをさらに暗号化してもセキュリティーの強化にはならず、バインド操作のたびにパフォーマンスが低下するという結果になるだけです。

暗号化された上で格納されている属性には、適用された暗号化アルゴリズムを示す暗号化方式タグが最初に付けられます。DES 暗号化アルゴリズムを使用して暗号化された属性は、次のように表示されます。

```
{CKM_DES_CBC}3hakc&jla+=snda%
```

データを暗号化するためにオンラインで (dsconf コマンドを使って) インポートする場合、サーバーへの認証に使用される鍵データベースのパスワードはすでに指定されているため、2 回目には要求されなくなります。データをオフラインで (dsadm コマンドを使って) インポートする場合、インポートするデータを暗号化するたびに Directory Server はパスワードを要求します。より機密性の高い操作であるデータの復号化では、エクスポートをオンライン、オフラインのどちらで行うかに関係なく、Directory Server は常に鍵データベースのパスワードを要求します。これにより、セキュリティーはさらに高まります。

注 – 証明書や非公開鍵を変更しない限り、サーバーにより引き続き同じ鍵が生成されます。そのため、両方のサーバーインスタンスが同じ証明書を使用していた場合は、データをあるサーバーインスタンスから別のサーバーインスタンスにトランスポートできます。つまりエクスポートしてからトランスポートできます。

属性の暗号化とパフォーマンス

属性を暗号化することでデータのセキュリティーは向上しますが、システムのパフォーマンスに影響が生じます。どの属性を暗号化する必要があるかを十分に検討し、特に機密にするべきと考えられる属性のみを暗号化します。

機密データはインデックスファイルから直接アクセスできるため、暗号化された属性に対応するインデックスキーを暗号化して、それらの属性を完全に保護する必要があります。インデックス付け自体がすでに Directory Server のパフォーマンスに影響しているため (インデックスキーの暗号化による負荷は含まれない)、データをインポートする、またはデータベースに初めて追加する前に属性暗号化を設定してください。こうすることで、暗号化された属性には最初からインデックスが付けられます。

属性暗号化の使用に関する注意点

属性暗号化機能を実装するときは、次の点を考慮してください。

- 一般に、属性暗号化の設定を変更するときは、データをエクスポートし、変更を加えた上で、新たに設定されたデータをインポートする必要があります。

  これにより、機能が喪失することなく、すべての設定変更が全体として適用されます。このような方法で行わなかった場合、一部の機能が喪失し、データのセキュリティーが危険にさらされる可能性があります。

- 既存のデータベースに対する属性暗号化の設定を変更すると、システムのパフォーマンスに大きく影響することがあります。

  たとえば、既存のデータが格納されたデータベースインスタンスについて考えてみましょう。このデータベースには、`mySensitiveAttribute` という属性を持つエントリが格納されています。その属性の値は、データベースとインデックスファイルにクリアテキストで格納されているものとします。この状態で、あとから、`mySensitiveAttribute` 属性を暗号化しようとすると、属性暗号化の設定を適用するためにサーバーがデータベースとインデックスファイルを更新する必要があり、データベースインスタンス内のすべてのデータはエクスポートされ、データベースに再びインポートされます。これにより、パフォーマンスに大きな影響が生じます。この結果生じるパフォーマンスの問題は、もし、この属性を最初から暗号化していれば回避できるはずです。

- 復号化された形式でデータをエクスポートするときに、誤ったパスワードを使用するとエクスポートが拒否されます。

  データを復号化された形式でエクスポートしようとすると、セキュリティー保護のために、ユーザーにパスワードの入力を求めるプロンプトが表示されます。ユーザーが誤ったパスワードを入力すると、サーバーにより復号化されたデータのエクスポート操作が拒否されます。パスワードは、直接入力するか、パスワードが含まれているファイルへのパスを指定することで入力できます。このファイルには、SSL パスワードファイルと同じ構文があります。120 ページの「証明書データベースパスワードの設定」を参照してください。

- アルゴリズムの変更はサポートされていますが、正しく作成されていない場合、インデックス付け機能は失われる可能性があります。

  データの暗号化に使用するアルゴリズムを変更するには、データをエクスポートし、属性の暗号化設定を変更してから、データをインポートします。この手順どおりに行わない場合、内部暗号化アルゴリズムに基づいて作成されたインデックスは機能しなくなります。

  暗号化された属性は使用する暗号化アルゴリズムを指定する暗号化方式タグの前に置かれるため、データのインポートは内部サーバー操作が処理します。このため、Directory Server では、アルゴリズムを変更する前に、データを暗号化された形式でエクスポートできます。

- サーバーの SSL 証明書を変更すると、暗号化されたデータを復号化できなくなります。

サーバーの SSL 証明書は、属性暗号化機能で独自の鍵の生成に使用され、そのあと暗号化および復号化操作の実行に使用されます。このため、SSL 証明書は暗号化されたデータの復号化に必要です。前もってデータを復号化しないで証明書を変更すると、データは復号化できません。これを回避するには、復号化された形式でデータをエクスポートし、証明書を変更してからデータをインポートしてください。

- 暗号化されたデータを伝送する、つまり、サーバーインスタンス間でエクスポートとインポートを行うには、両方のサーバーインスタンスが同じ証明書を使用する必要があります。

  詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 管理ガイド』の「属性値の暗号化」を参照してください。

▼ 属性の暗号化を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 属性の暗号化を設定するサフィックスに何らかのエントリが含まれるときは、最初にそのサフィックスの内容を **LDIF** ファイルにエクスポートします。

   暗号化されている属性がサフィックスに含まれていて、エクスポートされた LDIF ファイルを使用してサフィックスを再初期化する場合は、エクスポートされた LDIF ファイルで属性を暗号化されたままにすることができます。

2. 属性の暗号化を有効にするには、次のコマンドを使用します。

   ```
   $ dsconf create-encrypted-attr -h host -p port suffix-DN attr-name cipher-name
   ```

   cipher-name は、次のいずれかになります。

   - `des`: DES ブロック暗号化方式
   - `des3`: トリプル DES ブロック暗号化方式
   - `rc2` - RC2 ブロック暗号化方式
   - `rc4` - RC4 ストリーム暗号化方式

   次に例を示します。

   ```
   $ dsconf create-encrypted-attr -h host1 -p 1389 dc=example,dc=com uid rc4
   ```

3. 暗号化された属性を元の状態に戻すには、次のコマンドを使用します。

   ```
   $ dsconf delete-encrypted-attr -h host -p port suffix-DN attr-name
   ```

4 1 つ以上の属性を暗号化するように設定を変更していて、インポート操作の前にそれらの属性に値が含まれている場合は、データベースキャッシュをクリアし、ログを削除します。

暗号化されていない値は、データベースキャッシュとデータベースログには表示されません。

注 – これらのファイルを削除すると、一部の追跡情報が失われます。また、これらのファイルを削除すると、サーバーが復旧モードになり、再起動に時間がかかる場合があります。

データベースキャッシュをクリアし、ログを削除するには、次のように操作します。

a. 59 ページの「Directory Server インスタンスの起動、停止、および再起動」で説明するように、Directory Server を停止します。

b. root または管理者権限を持つユーザーとして、ファイルシステムの次の場所にあるデータベースキャッシュファイルを削除します。

```
# rm instance-path/db/__db.*
```

c. ファイルシステムからデータベースログファイルを削除します。

```
# rm instance-path/db/log.0000000001
```

d. Directory Server を再起動します。

サーバーは、新しいデータベースキャッシュファイルを自動的に作成します。ふたたびキャッシュがいっぱいになるまで、このサフィックスでの操作のパフォーマンスは、若干の影響を受ける可能性があります。

5 212 ページの「サフィックスの初期化」で説明する方法で、LDIF ファイルを使用してサフィックスを初期化します。

このファイルが読み込まれ、対応するインデックスが作成されるときに、指定した属性の値はすべて暗号化されます。

◆ ◆ ◆ 第 5 章

Directory Server のセキュリティー

Directory Server は、ネットワークを介してセキュリティーが確保された、信頼できる通信を行ういくつかのメカニズムをサポートしています。LDAPS は LDAP の標準プロトコルで、SSL (Secure Socket Layer) 上で実行されます。LDAPS はデータを暗号化し、オプションとして認証のために証明書を使用します。この章で SSL という用語を使用する場合、サポートされているプロトコル SSL2、SSL3、および TLS 1.0 を指します。

Directory Server は StartTLS (Start Transport Layer Security) 拡張処理もサポートしているため、元は暗号化されていない LDAP 接続でも TLS を有効にできます。

さらには、SASL (Simple Authentication and Security Layer) を介した GSSAPI (Generic Security Service API) にも対応しています。GSSAPI により、Solaris オペレーティングシステム (Solaris OS) で Kerberos Version 5 セキュリティープロトコルを利用できます。アイデンティティーマッピングメカニズムによって、Kerberos 主体とディレクトリ内のアイデンティティーが関連付けられます。

セキュリティー情報の詳細については、
http://www.mozilla.org/projects/security/pki/nss/
(`http://www.mozilla.org/projects/security/pki/nss/`) の NSS の Web サイトを参照してください。

この章では、SSL によってセキュリティーを設定する手順について説明します。ACI については、第 6 章を参照してください。ユーザーアクセスとパスワードについては、第 7 章を参照してください。

この章の内容は次のとおりです。

Directory Server での SSL の使用

SSL (Secure Sockets Layer) は暗号化された通信と、オプションとして Directory Server とクライアントの間の認証を提供します。SSL は LDAP 上または DSML-over-HTTP で使用できます。SSL は、LDAP 上でデフォルトで有効になりますが、DSML-over-HTTP を使用している場合、簡単に SSL を有効にできます。さらに、サーバー間のセキュリティー保護された通信に SSL を使用するよう、レプリケーションを設定できます。

単純認証 (バインド DN とパスワード) で SSL を使用すると、サーバーとやり取りしたデータがすべて暗号化されます。暗号化によって、機密性とデータの完全性が保証されます。必要に応じて、クライアントは証明書を使用して Directory Server への接続の認証、および SASL (Simple Authentication and Security Layer) を利用したサードパーティー製のセキュリティーメカニズムへの接続の認証ができます。証明書ベースの認証では、公開鍵暗号方式を使用してクライアントまたはサーバーを偽装したり、認証されているユーザーになりすますことはできなくなります。

Directory Server では、SSL による通信と SSL を使用しない通信を別々のポートで同時に実行できます。また、セキュリティーのためにすべての通信をセキュリティー保護された LDAP ポートに限定することもできます。クライアント認証も設定できます。クライアント認証を必要とする、または許可するように設定できます。この設定によって、実行するセキュリティーのレベルが決定されます。

SSL によって、通常の LDAP 接続をセキュリティー保護する Start TLS 拡張処理への対応も有効になります。クライアントが通常の LDAP ポートにバインドされても、TLS (Transport Layer Security) プロトコルを使用して接続を保護できます。Start TLS 処理では、クライアントに一層の柔軟性が与えられ、ポートの割り当ても簡素化できます。

SSL が提供する暗号化メカニズムは、属性の暗号化にも使用されます。SSL を有効にすることで、サフィックスでの属性の暗号化を設定し、ディレクトリに格納するときにデータを保護することができます。詳細については、106 ページの「属性値の暗号化」を参照してください。

これ以外のセキュリティーとして、アクセス制御命令 (ACI) を使用してディレクトリの内容にアクセス制御を設定できます。ACI は、特定の認証メソッドを必要としデータがセキュリティー保護されたチャネルでのみ送信します。SSL と証明書の使用を補完するように ACI を設定します。詳細については、第 6 章を参照してください。

SSL は LDAP 上ではデフォルトで有効になります。DSML-over-HTTP では簡単に SSL を有効にできます。さらに、次に説明するように、一部の SSL 設定は変更が可能です。

証明書を管理する

この節では、Directory Server で SSL 証明書を管理する方法について説明します。

Directory Server 上で SSL を実行するには、自己署名付き証明書または公開鍵インフラストラクチャー (PKI) ソリューションを使用する必要があります。

PKI ソリューションには、外部の認証局 (CA) が関係します。PKI ソリューションの場合、CA 署名付きサーバー証明書が必要です。これには、公開鍵と非公開鍵の両方が含まれます。この証明書は、Directory Server に固有のものです。また、公開鍵を含む信頼できる CA 証明書も必要です。信頼できる CA 証明書は、CA からのサーバー証明書がすべて信頼できることを示します。この証明書は CA ルート鍵またはルート証明書と呼ばれることもあります。

注 – テスト目的で証明書を使用している場合、自己署名付き証明書を使用できます。しかし、本番で自己署名付き証明書を使用することはあまり安全ではありません。本番では、信頼できる証明書発行局 (CA) の証明書を使用してください。

この節で示す手順では、`dsadm` コマンドと `dsconf` コマンドを使用します。これらのコマンドについては、`dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

この節では、Directory Server での証明書設定に関する次の情報について説明します。

- 113 ページの「デフォルトの自己署名付き証明書を表示する」
- 114 ページの「自己署名済み証明書を管理する」
- 114 ページの「CA 署名付きサーバー証明書を要求する」
- 116 ページの「CA 署名付きサーバー証明書と信頼できる CA 証明書を追加する」
- 119 ページの「有効期限の切れた CA 署名付きサーバー証明書を更新する」
- 119 ページの「CA 署名付きサーバー証明書をエクスポートおよびインポートする」
- 120 ページの「証明書データベースパスワードの設定」
- 121 ページの「Directory Server の証明書データベースのバックアップと復元」

▼ デフォルトの自己署名付き証明書を表示する

Directory Server インスタンスを初めて作成した場合、これには、デフォルトの自己署名付き証明書が含まれています。自己署名付き証明書は、公開鍵と非公開鍵のペアで、公開鍵が非公開鍵によって署名されています。自己署名付き証明書は、3 か月間有効です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- デフォルトの自己署名付き証明書を表示するには、次のコマンドを使用します。

  ```
  $ dsadm show-cert instance-path defaultCert
  ```

▼ 自己署名済み証明書を管理する

Directory Server インスタンスを作成すると、デフォルトの自己署名付き証明書が自動的に用意されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. デフォルト設定以外の自己署名付き証明書を作成するには、次のコマンドを使用します。

   ```
   $ dsadm add-selfsign-cert instance-path cert-alias
   ```

   ここで、*cert-alias* は、証明書を特定するために付ける名前です。

   このコマンドのオプションをすべて確認するには、`dsadm`(1M) のマニュアルページまたはコマンド行のヘルプを参照してください。

   ```
   $ dsadm add-selfsign-cert --help
   ```

2. 自己署名付き証明書の期限が切れた場合は、サーバーインスタンスを停止して、証明書を更新します。

   ```
   dsadm stop instance-path
   $ dsadm renew-selfsign-cert instance-path cert-alias
   ```

3. サーバーインスタンスを再起動します。

   ```
   $ dsadm start instance-path
   ```

▼ CA 署名付きサーバー証明書を要求する

この手順は、Directory Server とともに使用するために CA 署名付きサーバー証明書を要求し、インストールする方法を説明しています。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **CA** 署名付きサーバー証明書要求を生成します。

```
$ dsadm request-cert [-W cert-pwd-file] {-S DN | --name name [--org org] [--org-unit org-unit \
  [--city city] [--state state] [--country country]} [-o output-file] [-F format] instance-path
```

たとえば、Example 社の CA 署名付きサーバー証明書を要求するには、次のコマンドを使用します。

```
$ dsadm request-cert --name host1 --org Example --org-unit Marketing \
 -o my_cert_request_file /local/ds
```

サーバーを完全に特定するために、認証局はこの例で示す属性すべてを必要とする場合があります。各属性の説明については、dsadm(1M) のマニュアルページを参照してください。

dsadm request-cert を使用して証明書を要求すると、出力形式として ASCII を指定しない限り、作成される証明書要求はバイナリ証明書要求になります。ASCII を指定すると、作成される証明書要求は、PEM 形式の PKCS #10 証明書要求になります。PEM (Privacy Enhanced Mail) は、RFC 1421 ～ 1424 (http://www.ietf.org/rfc/rfc1421.txt) で指定されている形式で、US-ASCII 文字を使用した base64 形式で符号化されます。要求の内容は、次の例のようになります。

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrjCCARcCAQAwbjELMAkGA1UBhMCVXMxEzARBgNVBAgTCkNBElGT1JOSUExLD
AqBgVBAoTI25ldHNjYXBlIGNvb11bmljYXRpb25zIGNvcnBvcmF0aWuMRwwGgYDV
QQDExNtZWxsb24umV0c2NhcGUuY29tMIGfMA0GCSqGSIb3DQEBAUAA4GNADCBiQK
BgCwAbskGh6SKYOgHy+UCSLnm3ok3X3u83Us7u0EfgSLR0f+K41eNqqWRftGR83e
mqPLDOf0ZLTLjVGJaHJn4l1gG+JDf/n/zMyahxtV7+T8GOFFigFfuxJaxMjr2j7I
vELlxQ4IfZgwqCm4qQecv3G+N9YdbjveMVXW0v4XwIDAQABAAwDQYJKoZIhvcNAQ
EEBQADgYEAZyZAm8UmP9PQYwNy4Pmypk79t2nvzKbwKVb97G+MT/gw1pLRsuBoKi
nMfLgKp1Q38K5Py2VGW1E47/rhm3yVQrIiwV+Z8Lcc=
-----END NEW CERTIFICATE REQUEST-----
```

2 手順に従って、証明書要求を認証局に転送します。

認証局証明書を入手するプロセスは、使用する認証局によって異なります。商用 CA のなかには、証明書を自動的にダウンロードできる Web サイトを備えているものもあります。その他の CA は、要求に応じて電子メールで証明書を送信します。

証明書要求を送信したら、証明書に関する CA からの回答を待つ必要があります。要求に対する回答が届くまでの時間は、状況によって異なります。たとえば、CA が社内にある場合は、要求に対する回答は 1 ～ 2 日しかかからないこともあります。CA が社外にある場合は、数週間かかることもあります。

3 認証局から受け取った証明書を保存します。

証明書を安全な場所にバックアップします。これにより、証明書を失っても、このバックアップファイルから証明書を再インストールできます。証明書はテキストファイルに保存できます。次に、PEM 形式の PKCS #11 証明書の例を示します。

```
-----BEGIN CERTIFICATE-----
MIICjCCAZugAwIBAgICCEEwDQYJKoZIhKqvcNAQFBQAwfDELMAkGA1UEBhMCVVMx
IzAhBgNVBAoGlBhbG9a2FWaWxsZGwSBXaWRnZXRzLCBJbmMuMR0wGwYDVQQLExRX
aWRnZXQgTW3FrZXJzICdSJyBVczEpMCcGAx1UEAxgVGVzdCBUXN0IFRlc3QgVGVz
```

```
dCBUZXN0IFlc3QgQ0EswHhcNOTgwMzEyMDIzMzUWhcNOTgwMzI2MDIzMpzU3WjBP
MQswCYDDVQQGEwJVUzEoMCYGA1UEChMfTmV0c2NhcGUgRGlyZN0b3J5VIFB1Ymxp
Y2F0aW9uczEWMB4QGA1UEAxMNZHVgh49dq2tLNvbjTBaMA0GCSqGSIb3DQEBAQUA
A0kAMEYkCQCksMR/aLGdfp4m0OiGgijG5KgOsyRNvwGYW7kfW+8mmijDtZaRjYNj
jcgpF3Vnlbxbc1X9LVjjNLC5737XZdAgEDozYwpNDARBglghkgBhvhCEAQEEBAMC
APAwHkwYDVR0jBBgwFAU67URjwCaGqZHUpSpdLxlzwJKiMwDQYJKoZIhQvcNAQEF
BQADgYEAJ+BfVem3vBOPBveNdLGfjlb9hucgmaMcQa9FA/db8qimKT/ue9UGOJqL
bwbMKBBopsDn56p2yV3PLIsBgrcuSoBCuFFnxBnqSiTS7YiYgCWqWaUA0ExJFmD6
6hBLseqkSWulk+hXHN7L/NrViO+7zNtKcaZLlFPf7d7j2MgX4Bo=
-----END CERTIFICATE-----
```

▼ CA 署名付きサーバー証明書と信頼できる CA 証明書を追加する

この手順は、Directory Server で使用するために、CA 署名付きサーバー証明書と信頼できる CA 証明書をインストールする方法を説明しています。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **CA 署名付きサーバー証明書を追加します。**

   ```
   $ dsadm add-cert --ca instance-path cert-alias cert-file
   ```

   ここで、*cert-alias* は証明書を特定するために付ける名前です。*cert-file* は、PEM 形式の PKCS #11 証明書を含むテキストファイルです。

   たとえば、CA 署名付きサーバー証明書をインストールするには、次のようなコマンドを使用します。

   ```
   $ dsadm add-cert --ca /local/ds server-cert /local/safeplace/serv-cert-file
   ```

   これで、証明書がインストールされますが、まだ信頼されていません。CA 署名付きサーバー証明書を信頼するには、認証局証明書をインストールする必要があります。

2. **信頼できる認証局証明書を追加します。**

   ```
   $ dsadm add-cert --ca -C instance-path cert-alias cert-file
   ```

   `-C` オプションは、証明書が信頼できる認証局証明書であることを示します。

   たとえば、認証局からの信頼できる証明書をインストールするには、次のようなコマンドを使用します。

   ```
   $ dsadm add-cert --ca -C /local/ds CA-cert /local/safeplace/ca-cert-file
   ```

3. **(省略可能) インストールした証明書を確認します。**

- サーバー証明書をすべて一覧表示して、有効期限とエイリアスを表示するには、次のように入力します。

  ```
  $ dsadm list-certs instance-path
  ```

  次に例を示します。

  ```
  $ dsadm list-certs /local/ds1
  Enter the certificate database password:
  Alias       Valid from Expires on Self-   Issued by          Issued to
                                    signed?
  ----------- ---------- ---------- ------- -----------------  -----------------
  serverCert  2000/11/10 2011/02/10 n       CN=CA-Signed Cert, CN=Test Cert,
              18:13      18:13              OU=CA,O=com        dc=example,dc=com
  defaultCert 2006/05/18 2006/08/18 y       CN=host1,CN=DS,    Same as issuer
              16:28      16:28              dc=example,dc=com
  2 certificates found
  ```

  デフォルトで、Directory Proxy Server のインスタンスには、`defaultCert` と呼ばれるデフォルトのサーバー証明書が含まれます。`Same as issuer` は、デフォルト証明書が自己署名付きサーバー証明書であることを示します。

- 信頼できる CA 証明書を一覧表示するには、次のように入力します。

  ```
  $ dsadm list-certs -C instance-path
  ```

  次に例を示します。

  ```
  $ dsadm list-certs -C /local/ds1
  Enter the certificate database password:
  Alias   Valid from Expires on Self-   Issued by           Issued to
                                signed?
  ------- ---------- ---------- ------- -----------------   -------------
  CA-cert 2000/11/10 2011/02/10 y       CN=Trusted CA Cert, Same as issuer
          18:12      18:12              OU=CA,O=com
  1 certificate found
  ```

- 証明書の有効期限を含む、証明書の詳細を表示するには、次のように入力します。

  ```
  $ dsadm show-cert instance-path cert-alias
  ```

  たとえば、サーバー証明書を表示するには、次のように入力します。

  ```
  $ dsadm show-cert /local/ds1 "Server-Cert"
  Enter the certificate database password:
  Certificate:
      Data:
          Version: 3 (0x2)
  ```

```
        Serial Number: 2 (0x2)
        Signature Algorithm: PKCS #1 MD5 With RSA Encryption
        Issuer:
            "CN=Server-Cert,O=Sun,C=US"
        Validity:
            Not Before: Fri Nov 10 18:12:20 2000
            Not After : Thu Feb 10 18:12:20 2011
        Subject:
            "CN=CA Server Cert,OU=ICNC,O=Sun,C=FR"
        Subject Public Key Info:
            Public Key Algorithm: PKCS #1 RSA Encryption
            RSA Public Key:
                Modulus:
                    bd:76:fc:29:ca:06:45:df:cd:1b:f1:ce:bb:cc:3a:f7:
                    77:63:5a:82:69:56:5f:3d:3a:1c:02:98:72:44:36:e4:
                    68:8c:22:2b:f0:a2:cb:15:7a:c4:c6:44:0d:97:2d:13:
                    b7:e3:bf:4e:be:b5:6a:df:ce:c4:c3:a4:8a:1d:fa:cf:
                    99:dc:4a:17:61:e0:37:2b:7f:90:cb:31:02:97:e4:30:
                    93:5d:91:f7:ef:b0:5a:c7:d4:de:d8:0e:b8:06:06:23:
                    ed:5f:33:f3:f8:7e:09:c5:de:a5:32:2a:1b:6a:75:c5:
                    0b:e3:a5:f2:7a:df:3e:3d:93:bf:ca:1f:d9:8d:24:ed
                Exponent: 65537 (0x10001)
    Signature Algorithm: PKCS #1 MD5 With RSA Encryption
    Signature:
        85:92:42:1e:e3:04:4d:e5:a8:79:12:7d:72:c0:bf:45:
        ea:c8:f8:af:f5:95:f0:f5:83:23:15:0b:02:73:82:24:
        3d:de:1e:95:04:fb:b5:08:17:04:1c:9d:9c:9b:bd:c7:
        e6:57:6c:64:38:8b:df:a2:67:f0:39:f9:70:e9:07:1f:
        33:48:ea:2c:18:1d:f0:30:d8:ca:e1:29:ec:be:a3:43:
        6f:df:03:d5:43:94:8f:ec:ea:9a:02:82:99:5a:54:c9:
        e4:1f:8c:ae:e2:e8:3d:50:20:46:e2:c8:44:a6:32:4e:
        51:48:15:d6:44:8c:e6:d2:0d:5f:77:9b:62:80:1e:30
    Fingerprint (MD5):
        D9:FB:74:9F:C3:EC:5A:89:8F:2C:37:47:2F:1B:D8:8F
    Fingerprint (SHA1):
        2E:CA:B8:BE:B6:A0:8C:84:0D:62:57:85:C6:73:14:DE:67:4E:09:56

    Certificate Trust Flags:
        SSL Flags:
            Valid CA
            Trusted CA
            User
            Trusted Client CA
        Email Flags:
            User
        Object Signing Flags:
            User
```

▼ 有効期限の切れた CA 署名付きサーバー証明書を更新する

CA 署名付きサーバー証明書 (公開鍵と非公開鍵) の有効期限が切れた場合は、次の手順に従って更新します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 認証局から更新された CA 署名付きサーバー証明書を入手します。
2. 更新された証明書を受け取ったら、サーバーインスタンスを停止して、証明書をインストールします。

```
$ dsadm stop instance-path
$ dsadm renew-cert instance-path cert-alias cert-file
```

3. サーバーインスタンスを再起動します。

```
$ dsadm start instance-path
```

▼ CA 署名付きサーバー証明書をエクスポートおよびインポートする

場合によって、あとで証明書をインポートできるように、証明書の公開鍵と非公開鍵をエクスポートすることがあります。たとえば、別のサーバーで使用する証明書が必要な場合があります。

この手順のコマンドは、`"cn=*,o=example"` のようなワイルドカードを含む証明書で使用できます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 証明書をエクスポートします。

```
$ dsadm export-cert [-o output-file] instance-path cert-alias
```

次に例を示します。

```
$ dsadm export-cert -o /tmp/first-certificate /local/ds1 "First Certificate"
$ dsadm export-cert -o /tmp/first-ca-server-certificate /local/ds1/ defaultCert
Choose the PKCS#12 file password:
Confirm the PKCS#12 file password:
```

```
$ ls /tmp
first-ca-server-certificate
```

2 証明書をインポートします。

```
$ dsadm import-cert instance-path cert-file
```

たとえば、host1 上のサーバーインスタンスに証明書をインポートするには、次のように入力します。

```
$ dsadm import-cert -h host1 /local/ds2 /tmp/first-ca-server-certificate
Enter the PKCS#12 file password:
```

3 (省略可能) サーバーに証明書をインポートしたら、インポートした証明書を使用するようサーバーを設定します。

```
$ dsconf set-server-prop -e -h host -p port -w - ssl-rsa-cert-name:server-cert
```

証明書データベースパスワードの設定

デフォルトで、Directory Server は保存されたパスワードを使用して SSL 証明書データベースのパスワードを内部的に管理します。証明書を管理する場合に、ユーザーは証明書パスワードを入力したり、パスワードファイルを指定したりする必要はありません。パスワードは暗号化されているのではなく、非表示になっているだけなので、このオプションはあまり安全ではありません。

より安全に証明書を使用したい場合には、コマンド行でユーザーがパスワード入力を求められるようにサーバーを設定できます。この場合、ユーザーは autostart、backup、disable-service、enable-service、 info、reindex、restore、および stop 以外の dsadm のすべてのサブコマンドに対して証明書データベースのパスワードを入力する必要があります。証明書データベースは、ディレクトリ *instance-path*/alias にあります。

▼ ユーザーが証明書のパスワード入力を求められるようにサーバーを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サーバーを停止します。

```
$ dsadm stop instance-path
```

2 パスワードプロンプトフラグを on に設定します。

```
$ dsadm set-flags instance-path cert-pwd-prompt=on
```

証明書に対してパスワードを設定するよう求められます。

3 次のように入力して、**Server** を起動します。

```
$ dsadm start instance-path
```

Directory Server の証明書データベースのバックアップと復元

Directory Server のインスタンスをバックアップする場合、Directory Server の設定と証明書をバックアップします。バックアップされた証明書は *archive-path*/alias ディレクトリに格納されます。

Directory Server のバックアップと復元の方法については、220 ページの「障害回復用のバックアップを作成する」を参照してください。

SSL 通信の設定

この節では、SSL の有効化と無効化に関する手順について説明します。

セキュリティー保護されていない接続の無効化

サーバーインスタンスが作成されると、デフォルトで LDAP クリアポートとセキュア LDAP ポート (LDAPS) が作成されます。しかし、サーバーが SSL を介してのみ通信するように SSL 以外の通信を無効にする場合もあります。

SSL 接続は、デフォルトの自己署名付き証明書で有効になります。希望する場合は、自分の証明書をインストールできます。サーバーの起動後の証明書の管理と、SSL の無効化の手順については、第 5 章を参照してください。証明書、証明書データベース、CA 署名付きサーバー証明書の入手の概要については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

▼ LDAP クリアポートを無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **LDAP** クリアポートを無効にします。

セキュリティー保護されていないポートを無効にするには、LDAP セキュアポートにバインドします。この例は、ホストサーバー `host1` 上のデフォルトの LDAP セキュアポート `1636` へのバインドを示しています。

```
$ dsconf set-server-prop -h host1 -P 1636 ldap-port:disabled
```

2 変更内容を有効にするために、サーバーを再起動します。

```
$ dsadm restart /local/ds
```

これで、セキュリティー保護されていないポートにバインドすることはできなくなります。

暗号化方式の選択

暗号化方式は、データを暗号化、復号化するために使用するアルゴリズムです。一般に、暗号化に使用するビット数が多いほど、強度と安全性は高まります。SSL の暗号化方式は、使用するメッセージ認証のタイプによっても識別されます。メッセージ認証は、データの整合性を保証するチェックサムを計算する別のアルゴリズムです。

クライアントがサーバーとの SSL 接続を開始するときは、情報の暗号化にどの暗号を使用するかについて、クライアントとサーバーが合意する必要があります。双方向の暗号化プロセスでは、必ず、送信側と受信側の両方が同じ暗号化方式を使用する必要があります。使用する暗号化方式は、サーバーが保存している暗号化方式の一覧の現在の順序によって決まります。サーバーは、その一覧内でクライアントに提示された暗号化方式に一致する最初の暗号化方式を選択します。Directory Server のデフォルトの暗号化方式値は `all` です。これは、背後の SSL ライブラリにサポートされている既知のセキュリティー保護されたすべての暗号化方式を意味します。ただし、この値は特定の暗号化方式のみを受け入れるように変更できます。

Directory Server で使用できる暗号化方式の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

▼ 暗号化方式を選択する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サーバーに対して **SSL** が有効であることを確認します。

121 ページの「SSL 通信の設定」を参照してください。

2 使用可能な SSL 暗号化方式を表示します。

```
$ dsconf get-server-prop -h host -p port ssl-supported-ciphers
ssl-supported-ciphers  :  TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
ssl-supported-ciphers  :  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
ssl-supported-ciphers  :  TLS_DHE_RSA_WITH_AES_256_CBC_SHA
ssl-supported-ciphers  :  TLS_DHE_DSS_WITH_AES_256_CBC_SHA
...
```

3 (省略可能) 暗号化されていないデータのコピーを維持する場合は、SSL 暗号化方式を設定する前にデータをエクスポートします。

208 ページの「LDIF へのエクスポート」を参照してください。

4 SSL 暗号化方式を設定します。

```
$ dsconf set-server-prop -h host -p port ssl-cipher-family:cipher
```

たとえば、暗号ファミリを SSL_RSA_WITH_RC4_128_MD5 と SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA に設定するには、次のように入力します。

```
$ dsconf set-server-prop -h host1 -P 1636 ssl-cipher-family:SSL_RSA_WITH_RC4_128_MD5 \
 ssl-cipher-family:SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
Enter "cn=Directory Manager" password:
Before setting SSL configuration, export Directory Server data.
Do you want to continue [y/n] ? y
Directory Server must be restarted for changes to take effect.
```

5 (省略可能) 既存のリストに SSL 暗号化方式を追加します。

指定した暗号化方式のリストがすでに存在し、そのリストに暗号化方式を追加する場合は、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port ssl-cipher-family+:cipher
```

たとえば、SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA 暗号化方式を追加するには、次のように入力します。

```
$ dsconf set-server-prop -h host1 -P 1636 \
 ssl-cipher-family+:SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
```

6 変更内容を有効にするために、サーバーを再起動します。

```
$ dsadm restart /local/ds
```

資格レベルと認証方法の設定

クライアントに適用されるセキュリティーモデルは、資格レベルと認証方法の組み合わせで定義されます。

Directory Server では、次の資格レベルがサポートされています。

- 匿名。ディレクトリの特定部分に匿名アクセスを許可することは、そのディレクトリへのアクセス権を持つすべてのユーザーに読み取りアクセス権を与えることを意味します。

  匿名資格レベルを使用する場合、すべての LDAP ネームエントリおよび属性に読み取りアクセスを許可する必要があります。

注意 – 匿名でのディレクトリへの書き込みアクセスは許可しないでください。これを許可すると、どのユーザーでも、書き込みアクセス権のある DIT 内の情報 (別のユーザーのパスワードや自分自身の識別情報など) の変更が可能になります。

- プロキシ。クライアントは、プロキシアカウントを使用してディレクトリへの認証またはバインドを行います。

  このプロキシアカウントには、ディレクトリへのバインドを許可されるエントリを設定できます。このプロキシアカウントは、ディレクトリでネームサービス機能を実行するための十分なアクセス権を必要とします。プロキシ資格レベルを使用して、すべてのクライアントで `proxyDN` および `proxyPassword` を設定する必要があります。暗号化された `proxyPassword` はクライアントのローカルに格納されます。

- 匿名プロキシ。複数の資格レベルが定義された複数値エントリ。

  匿名プロキシレベルを割り当てられたクライアントは、最初にそのプロキシ識別情報を使用して認証を試みます。ユーザーのロックアウトやパスワードの有効期限切れなど何らかの理由でプロキシユーザーとして認証できなかった場合、クライアントは匿名アクセスを使用します。この場合、ディレクトリの構成によっては、別のサービスレベルに移行する可能性があります。

クライアント認証は、クライアントのアイデンティティーを確認するためのサーバーのメカニズムです。

クライアント認証は、次のいずれかの方法で実行できます。

- DN とパスワードを提供する。
- クライアントによって提供された証明書を使用する。

  証明書ベースの認証では、SSL プロトコルを介して入手したクライアント証明書を使用して、ユーザーエントリの識別情報が検出されます。証明書ベースの認証では、クライアントが外部メカニズムを指定する SASL バインド要求を送信します。バインド要求は、すでに確立された SSL 認証メカニズムに依存します。
- SASL ベースのメカニズムを使用する。
  - すべてのオペレーティングシステムで、DIGEST-MD5 による SASL。
  - Solaris オペレーティングシステムで、Kerberos V5 によるクライアント認証が可能である、GSSAPI メカニズムによる SASL。

  2 つのうちどちらの SASL メカニズムを使用する場合も、アイデンティティーのマッピングを行うようにサーバーを設定する必要があります。SASL 資格は、主体と呼ばれます。主体の内容のバインド DN を決定するために、各メカニズムに特定のマッピングが必要です。主体が 1 つのユーザーエントリにマッピングされ、SASL メカニズムがそのユーザーのアイデンティティーを検証すると、ユーザーの DN がその接続のバインド DN となります。
- SSL クライアント認証モードを使用する。

  SSL 層ですべてのクライアントが認証されるようにするには、SSL クライアント認証を使用します。クライアントアプリケーションは、SSL 証明書をサーバーに送信することで認証を行います。`SSL-client-auth-mode` フラグを使用して、サーバーが SSL クライアント認証を許可する、要求する、または許可しないよう指定します。デフォルトでは、クライアントは認証を許可されます。

この節では、Directory Server での 2 つの SASL メカニズムの設定に関する次の情報について説明します。

- 125 ページの「Directory Server での SASL 暗号化レベルの設定」
- 127 ページの「DIGEST-MD5 を利用した SASL 認証」
- 129 ページの「GSSAPI を利用した SASL 認証 (Solaris OS のみ)」

セキュリティーの設定の詳細については、132 ページの「LDAP クライアントでセキュリティーを使用するための設定」を参照してください。

Directory Server での SASL 暗号化レベルの設定

SASL メカニズムを設定する前に、暗号化が必要かどうかを指定する必要があります。SASL 暗号化の要件は、最大および最小 SSF (Strength Security Factor) によって設定されます。

属性 `dsSaslMinSSF`(5dsat) および `dsSaslMaxSSF`(5dsat) は、暗号化鍵の長さを示します。これらは、`cn=SASL, cn=security, cn=config` に保存されています。

サーバーに対しては、暗号化しないという選択肢も含めて、すべてのレベルの暗号化を設定できます。つまり、Directory Server は 256 より大きい dsSaslMinSSF 値と dsSaslMaxSSF 値を受け入れます。しかし、現在 SASL メカニズムは 128 より大きい SSF をサポートしません。Directory Server はこれらの値のネゴシエーションを行い最大限の SSF 値 (128) にします。このため、使用されるメカニズムによっては、実際の最大 SSF 値は、設定された最大値より小さい可能性があります。

SASL セキュリティーファクタ認証は、次の 2 つの主要な項目に基づきます。サーバーとクライアントアプリケーションによって要求される最小ファクタと最大ファクタ、および使用可能な暗号化メカニズム。これらは背後のセキュリティーコンポーネントによって提供されます。つまり、サーバーとクライアントは両方で設定された最大ファクタ以下で、両方で設定された最小ファクタ以上の、使用可能な中で最も大きいセキュリティーファクタを使用しようとします。

Directory Server に対するデフォルトの最小 SASL セキュリティーファクタである dsSaslMinSSF は 0 で、保護されていないことを示します。実際の最小値は、Directory Server の最小値を変更しない限り、クライアント設定によって変わります。実際は、サーバーとクライアントに使用する最低レベルに最小値を設定します。サーバーとクライアントが最低要件を満たすメカニズムのネゴシエーションに失敗した場合、接続は確立されません。

▼ SASL 暗号化を要求する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- SASL 暗号化を要求するには dsSaslMinSSF 値を最低限必要な暗号化に設定します。

```
$ ldapmodify -h host -p port -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=SASL, cn=security, cn=config
changetype: modify
replace: dsSaslMinSSF
dsSaslMinSSF: 128
^D
```

▼ SASL 暗号化を許可しない

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- SASL 暗号化を許可しない場合は dsSaslMinSSF と dsSaslMaxSSF の両方の値をゼロに設定します。

```
$ ldapmodify -h host -p port -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=SASL, cn=security, cn=config
changetype: modify
```

```
replace: dsSaslMinSSF
dsSaslMinSSF: 0

replace: dsSaslMaxSSF
dsSaslMaxSSF: 0
^D
```

DIGEST-MD5 を利用した SASL 認証

DIGEST-MD5 メカニズムは、クライアントによって送信されたハッシュされた値をユーザーのパスワードのハッシュと比較することによって、クライアントを認証します。ただし、このメカニズムはユーザーパスワードを読み取る必要があり、DIGEST-MD5 による認証を希望するすべてのユーザーは、ディレクトリ内に {CLEAR} パスワードを持つ必要があります。{CLEAR} パスワードをディレクトリに保存する場合に、第 6 章で説明されているように、パスワード値へのアクセスが ACI によって正しく制限されていることを確認する必要があります。さらに、106 ページの「属性値の暗号化」で説明しているように、サフィックスで属性の暗号化を設定する必要があります。

▼ DIGEST-MD5 メカニズムを設定する

次の手順は、DIGEST-MD5 を使用するように Directory Server を設定する方法を説明しています。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 DIGEST-MD5 がルートエントリ上で supportedSASLMechanisms 属性の値であることを確認するには、ldapsearch コマンドを使用します。

たとえば、次のコマンドはどの SASL メカニズムが有効であるかを表示します。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -s base -b "" "(objectclass=*)" supportedSASLMechanisms
Enter bind password:
dn:
supportedSASLMechanisms: EXTERNAL
supportedSASLMechanisms: DIGEST-MD5
supportedSASLMechanisms: GSSAPI
^D
```

2 DIGEST-MD5 が有効でない場合は、有効にします。

```
$ ldapmodify -h host -p port -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=SASL, cn=security, cn=config
changetype: modify
```

```
add: dsSaslPluginsEnable
dsSaslPluginsEnable: DIGEST-MD5
-
replace: dsSaslPluginsPath
dsSaslPluginsPath: SASL-library
^D
```

ここで、*SASL-library* は次のいずれかです。

JES installation `/usr/lib/mps/sasl2`

Zip installation *install-path*`/dsee6/private/lib`

3 **DIGEST-MD5** のデフォルトのアイデンティティーマッピングを使用するか新規作成します。

詳細については、128 ページの「DIGEST-MD5 アイデンティティーマッピング」を参照してください。

4 **DIGEST-MD5** を使用する **SSL** 経由でサーバーにアクセスするすべてのユーザーのパスワードが `{CLEAR}` に含まれていることを確認します。

パスワード保存スキーマについては、第 7 章を参照してください。

5 **SASL** 設定エントリまたは **DIGEST-MD5** アイデンティティーマッピングエントリの **1** つを変更した場合は、**Directory Server** を再起動します。

DIGEST-MD5 アイデンティティーマッピング

SASL メカニズムの アイデンティティーマッピングは、SASL アイデンティティーの資格をディレクトリ内のユーザーエントリと一致させようとします。マッピングによって、SASL アイデンティティーに対応する DN が見つからなかったときは、認証は失敗します。このメカニズムの詳細な説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

SASL アイデンティティーは、*Principal* という文字列です。これは、各メカニズムに固有の形式でユーザーを表します。DIGEST-MD5 では、クライアントは、`dn:` プレフィックスと LDAP DN、または `u:` プレフィックスの後にクライアントが決定するテキストを続けた情報のいずれかが含まれる主体を作成すべきです。マッピング時に、クライアントが送信した主体は、`${Principal}`プレースホルダで使用されます。

サーバー設定内の次のエントリは、DIGEST-MD5 のデフォルト アイデンティティーマッピングです。

```
dn: cn=default,cn=DIGEST-MD5,cn=identity mapping,cn=config
objectClass: top
objectClass: nsContainer
objectClass: dsIdentityMapping
```

```
objectClass: dsPatternMatching
cn: default
dsMatching-pattern: \${Principal}
dsMatching-regexp: dn:(.*)
dsMappedDN: \$1
```

このアイデンティティーマッピングは、主体の dn フィールドに、ディレクトリ内の既存ユーザーの正確な DN が含まれていることを前提としています。

▼ DIGEST-MD5 用の独自のアイデンティティーマッピングを定義する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. cn=DIGEST-MD5,cn=identity mapping,cn=config の下でデフォルトのマッピングエントリを編集するか、新しいマッピングエントリを作成します。

   次のコマンドは、このマッピングを定義する方法を示しています。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=unqualified-username,cn=DIGEST-MD5,cn=identity mapping
cn=config
objectclass: dsIdentityMapping
objectclass: dsPatternMatching
objectclass: nsContainer
objectclass: top
cn: unqualified-username
dsMatching-pattern: \${Principal}
dsMatching-regexp: u:(.*)@(.*)\\.com
dsSearchBaseDN: dc=\$2
dsSearchFilter: (uid=\$1)
```

2. **Directory Server** を再起動して、新しいマッピングを有効にします。

GSSAPI を利用した SASL 認証 (Solaris OS のみ)

SASL 上の GSSAPI (Generic Security Service API) では、クライアントを認証するために、Kerberos V5 などのサードパーティーのセキュリティーシステムを使用できます。GSSAPI ライブラリは、Solaris OS SPARC® プラットフォームの場合にのみ使用できます。Sun Enterprise Authentication Mechanism™ 1.0.1 サーバーに Kerberos V5 実装をインストールすることをお勧めします。

サーバーは、GSSAPI を使用してユーザーの識別情報を検証します。次に、SASL メカニズムは GSSAPI マッピングルールを適用して、この接続中のすべての操作のバインド DN となる DN を取得します。

▼ Keberos システムを設定する

製造元の指示に従って、Kerberos ソフトウェアを設定します。Sun Enterprise Authentication Mechanism 1.0.1 サーバーを使用している場合、次の手順を使用します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. `/etc/krb5`内のファイルを設定します。
2. ユーザーとサービスを保存するために **Kerberos** データベースを作成します。
3. データベース内で **LDAP** サービスの主体を作成します。

   ```
   $ ldap/server-FQDN@realm
   ```

   ここで、*server-FQDN* は Directory Server の完全修飾ドメイン名です。
4. **Kerberos** デーモンプロセスを開始します。

注 – DNS は、ホストマシンに設定されている必要があります。

各手順の詳細については、ソフトウェアのマニュアルを参照してください。136 ページの「GSSAPI と SASL を使用した Kerberos 認証の設定例」も参照してください。

▼ GSSAPI メカニズムを設定する

次の手順は、Solaris OS 上で GSSAPI を使用するよう Directory Server を設定する方法を説明しています。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. 131 ページの「GSSAPI アイデンティティーマッピング」での説明に従って、**GSSAPI** のデフォルトアイデンティティーマッピングと任意のカスタムマッピングを作成します。

2 サービスキーを保存するために鍵タブを作成します。
LDAP サービスキーは、鍵タブに保存されます。

a. 鍵タブは必ず **Directory Server** ユーザーのみが読み取れるようにします。

b. ファイル名をデフォルトの `/etc/krb5/krb5.keytab` から変更します。

c. デフォルトの鍵タブではなく、必ず新しい鍵タブが使用されるように、環境変数 `KRB5_KTNAME` を設定します。

3 **SASL** 設定エントリまたは **GSSAPI** アイデンティティーマッピングエントリの **1** つを変更した場合は、**Directory Server** を再起動します。
DNS は、ホストマシンに設定されている必要があります。

GSSAPI アイデンティティーマッピング

SASL メカニズムのアイデンティティーマッピングは、SASL アイデンティティーの資格をディレクトリ内のユーザーエントリと一致させようとします。マッピングによって、SASL アイデンティティーに対応する DN が見つからなかったときは、認証は失敗します。

SASL アイデンティティーは、*Principal* という文字列です。これは、各メカニズムに固有の形式でユーザーを表します。Kerberos で GSSAPI を使用した主体は *uid* [/*instance*][@*realm*] という形式のアイデンティティーです。*uid* にはオプションの *instance* ID を含め、オプションの *realm* を続けることができます。これはドメイン名の場合があります。次の例は、いずれも有効なユーザー主体です。

```
bjensen
bjensen/Sales
bjensen@EXAMPLE.COM
bjensen/Sales@EXAMPLE.COM
```

最初は、ディレクトリ内には GSSAPI マッピングは定義されていません。デフォルトのマッピングを定義し、使用する主体をクライアントがどのように定義するかに応じてカスタムマッピングを定義する必要があります。

▼ GSSAPI 用のアイデンティティーマッピングを定義する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. `cn=GSSAPI,cn=identity mapping, cn=config` の下に新しいマッピングエントリを作成します。

   アイデンティティーマッピングエントリ内の属性の定義については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。GSSAPI のマッピングの例は、*instance-path*`/ldif/identityMapping_Examples.ldif` にあります。

   このファイル内のデフォルト GSSAPI マッピングは、主体にユーザー ID のみが含まれていることを前提としています。このマッピングは、ディレクトリの固定ブランチでユーザーを決定します。

   ```
   dn: cn=default,cn=GSSAPI,cn=identity mapping,cn=config
   objectclass: dsIdentityMapping
   objectclass: nsContainer
   objectclass: top
   cn: default
   dsMappedDN: uid=\${Principal},ou=people,dc=example,dc=com
   ```

   このファイルに含まれるもう 1 つの例は、既知のレルムを含む主体にユーザー ID が記録されている場合に、ユーザー ID を決定する方法を示しています。

   ```
   dn: cn=same_realm,cn=GSSAPI,cn=identity mapping,cn=config
   objectclass: dsIdentityMapping
   objectclass: dsPatternMatching
   objectclass: nsContainer
   objectclass: top
   cn: same_realm
   dsMatching-pattern: \${Principal}
   dsMatching-regexp: (.*)@EXAMPLE.COM
   dsMappedDN: uid=\$1,ou=people,dc=EXAMPLE,dc=COM
   ```

2. **Directory Server** を再起動して、新しいマッピングを有効にします。

LDAP クライアントでセキュリティーを使用するための設定

次に、Directory Server とセキュリティー保護された接続を確立する LDAP クライアント内で SSL を設定および使用する方法を説明します。SSL 接続では、サーバーがクライアントに証明書を送信します。クライアントは、まず最初に証明書を信頼することで、サーバーが信頼できるものであることを確認します。次に、必要に応じてクライアントは独自の証明書または SASL メカニズム (2 つのうち 1 つ) の情報を送信することで、いずれかのクライアント認証メカニズムを開始できます。SASL メカニズムは、Kerberos V5 を使用した DIGEST-MD5 および GSSAPI です。

次の各項では、SSL が有効な LDAP クライアントの例として、`ldapsearch` ツールを使用します。

他の LDAP クライアントに SSL 接続を設定する方法については、アプリケーションに付属するマニュアルを参照してください。

注 – クライアントアプリケーションによっては、SSL を実装しても、信頼された証明書がサーバーにあるかどうかを検証しません。これらのクライアントアプリケーションは SSL プロトコルを使用してデータの暗号化を行いますが、機密の保護を保証することも第 3 者がユーザーとして認証されることを防止することもできません。

次の項では、セキュリティーを使用するために LDAP クライアントを設定する方法を説明します。

クライアントでの SASL DIGEST-MD5 の使用

クライアントで DIGEST-MD5 メカニズムを使用している場合、ユーザー証明書をインストールする必要はありません。ただし、暗号化された SSL 接続を利用するには、113 ページの「証明書を管理する」で説明した方法で、サーバー証明書を信頼する必要があります。

レルムの指定

レルムは、認証アイデンティティーが選択される名前空間を定義します。DIGEST-MD5 認証では、特定のレルムに対して認証を行う必要があります。

Directory Server は、DIGEST-MD5 のデフォルトレルムとして、マシンの完全修飾ホスト名を使います。サーバーは、`nsslapd-localhost` 設定属性に含まれる小文字のホスト名を使用します。

レルムを指定しない場合、サーバーが提供するデフォルトのレルムが適用されます。

環境変数の指定

UNIX 環境では、LDAP ツールが DIGEST-MD5 ライブラリを見つけることができるように、`SASL-PATH` 環境変数を設定する必要があります。DIGEST-MD5 ライブラリは、SASL プラグインに動的に読み込まれる共有ライブラリです。`SASL_PATH` 環境変数を次のように設定します。

```
export SASL_PATH=SASL-library
```

このパスは、Directory Server が LDAP ツールが起動されたホストと同じホストにインストールされていることを前提としています。

ldapsearch コマンドの例

SSL を使用せずに DIGEST-MD5 クライアント認証を実行することができます。次の例は、デフォルトの DIGEST-MD5 アイデンティティーマッピングを使用してバインド DN を決定します。

```
$ ldapsearch -h host1 -p 1389 \
 -o mech=DIGEST-MD5 [ \
 -o realm="example.com"] \
 -o authid="dn:uid=bjensen,dc=example,dc=com" \
 -w - \
 -o authzid="dn:uid=bjensen,dc=example,dc=com" \
 -o secProp="minssf=56,maxssf=256,noplain" \
 -b "dc=example,dc=com" "(givenname=Richard)"
```

上の例は、-o (小文字の o) オプションを使用して SASL オプションを指定しています。レルムの指定は省略できますが、指定する場合は、サーバーホストマシンの完全修飾ドメイン名を指定する必要があります。プロキシ操作を対象とする authzid は使用されませんが、authid と authzid はどちらも必要であり、同じ値を指定する必要があります。-w パスワードオプションは authid に適用されます。

authid の値は、アイデンティティーマッピングで使用される主体です。authid には、ディレクトリ内の有効なユーザー DN が後に続く dn: プレフィックスか、クライアントが決定した任意の文字列が後に続く u: プレフィックスが含まれているはずです。このように authid を使用すると、128 ページの「DIGEST-MD5 アイデンティティーマッピング」で説明しているマッピングを使用することができます。

最も一般的な設定は、クライアント認証のために LDAPS セキュアポートと DIGEST-MD5 を使用して暗号化を行う SSL 接続に対する設定です。次の例は、同じ処理を SSL 経由で実行します。

```
$ ldapsearch -h host1 -P 1636 \
 -Z -P .mozilla/bjensen/BJE6001.slt/cert8.db \
 -N "cert-example" -w - \
 -o mech=DIGEST-MD5 [-o realm="example.com"] \
 -o authid="dn:uid=bjensen,dc=example,dc=com" \
 -o authzid="dn:uid=bjensen,dc=example,dc=com" \
 -o secProp="minssf=0,maxssf=0,noplain" \
 -b "dc=example,dc=com" "(givenname=Richard)"
```

この例では、SSL を経由して処理を行う場合に ldapsearch コマンドに -N オプションと -w オプションが必要です。ただし、これらのオプションはクライアント認証には

使用されません。その代わりに、サーバーは authid の値に含まれる主体の DIGEST-MD5 アイデンティティーマッピングを行います。

クライアントでの Kerberos SASL GSSAPI の使用

クライアントで GSSAPI メカニズムを使用する場合、ユーザー認証をインストールする必要はありませんが、Kerberos V5 セキュリティーシステムを設定する必要があります。また、暗号化された SSL 接続を使用する場合、113 ページの「証明書を管理する」で説明しているように、サーバー証明書を信頼します。

▼ ホスト上で Kerberos V5 を設定する

LDAP クライアントを実行するホストマシンで Kerberos V5 を設定する必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 インストール手順に従って **Kerberos V5** をインストールします。

Sun Enterprise Authentication Mechanism 1.0.1 クライアントソフトウェアをインストールすることをお勧めします。

2 **Kerberos** ソフトウェアを設定します。

Sun Enterprise Authentication Mechanism ソフトウェアを使用して、/etc/krb5 の下のファイルを設定します。この設定は、kdc サーバーを設定し、デフォルトレルムと Kerberos システムが必要とするその他の設定を定義します。

3 kerberos_v5 に関する行が先頭になるように、必要に応じて /etc/gss/mech ファイルを編集します。

▼ Kerberos 認証の SASL オプションを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **GSSAPI** メカニズムで有効にするクライアントアプリケーションを使用する前に、ユーザーの主体で **Kerberos** セキュリティーシステムを起動します。

```
$ kinit user-principal
```

ここで、*user-principal* は、お使いの SASL アイデンティティーです。たとえば、bjensen@example.com となります。

2 **Kerberos の使用を設定する SASL オプションを指定します。**

UNIX 環境では、SASL_PATH 環境変数を SASL ライブラリの正しいパスに設定します。Korn シェルでの設定例は次のようになります。

```
$ export SASL_PATH=SASL-library
```

このパスは、Directory Server が LDAP ツールが起動されたホストと同じホストにインストールされていることを前提としています。

次に示す ldapsearch ツールの例は、-o (小文字の o) オプションを使用して Kerberos の使用を設定する SASL オプションを指定する方法を示しています。

```
$ ldapsearch -h www.host1.com -p 1389 -o mech=GSSAPI -o authid="bjensen@EXAMPLE.COM" \
 -o authzid="bjensen@EXAMPLE.COM" -b "dc=example,dc=com" "(givenname=Richard)"
```

authid は、kinit コマンドによって初期化された Kerberos キャッシュに含まれるので、省略することができます。authid を指定する場合、プロキシ操作を対象とする authzid は使用されませんが、authid と authzid にはどちらにも同じ値を指定する必要があります。authid の値は、アイデンティティーマッピングで使用される主体です。レルムを含み、主体はすべて完全な主体にする必要があります。131 ページの「GSSAPI アイデンティティーマッピング」を参照してください。

GSSAPI と SASL を使用した Kerberos 認証の設定例

Directory Server に対する Kerberos の設定は、複雑な場合があります。最初に Kerberos のマニュアルを参照してください。

さらに詳細について知りたい場合は、次に示す手順例を参考にしてください。ただし、この手順は例であることを忘れないでください。自分の設定と環境に合わせて手順を変更してください。

Solaris OS での Kerberos の設定と使用の詳細については、『System Administration Guide: Security Services』を参照してください。このマニュアルは Solaris マニュアルセットの一部です。マニュアルページを参照することもできます。

この例についての情報と使用する手順は、次のとおりです。

1. 137 ページの「この例の前提」
2. 138 ページの「すべてのマシン : Kerberos クライアント設定ファイルの編集」
3. 139 ページの「すべてのマシン : 管理サーバー ACL 設定ファイルの編集」
4. 139 ページの「KDC マシン : KDC サーバー設定ファイルの編集」
5. 140 ページの「KDC マシン : KDC データベースの作成」
6. 140 ページの「KDC マシン : 管理の主体と鍵タブの作成」
7. 141 ページの「KDC マシン : Kerberos デーモンの開始」
8. 141 ページの「KDC マシン : KDC マシンと Directory Server マシンに対するホスト主体の追加」
9. 142 ページの「KDC マシン : Directory Server に対する LDAP 主体の追加」

10. 142 ページの「KDC マシン : KDC へのテストユーザーの追加」
11. 142 ページの「Directory Server マシン : Directory Server のインストール」
12. 143 ページの「Directory Server マシン : GSSAPI を有効にするための Directory Server の設定」
13. 144 ページの「Directory Server マシン : Directory Server 鍵タブの作成」
14. 145 ページの「Directory Server マシン : Directory Server へのテストユーザーの追加」
15. 145 ページの「Directory Server マシン : テストユーザーとしての Kerberos チケットの取得」
16. 146 ページの「クライアントマシン : GSSAPI による Directory Server に対する認証」

この例の前提

この手順例では、1 つ目のマシンを KDC (Key Distribution Center) として操作し、2 つ目のマシンでは Directory Server を実行できるように設定する処理について説明します。この手順の結果として、ユーザーは GSSAPI によって Kerberos 認証を実行できるようになります。

同じマシン上で KDC と Directory Server の両方を実行することもできます。両方を同じマシン上で実行することを選択した場合にも同じ手順を使用できます。この場合、KDC マシンと Directory Server マシンで重複する手順は、一度行うだけで済みます。

この手順では、使用される環境に関する多くの前提条件が発生します。手順例を使用する場合は、環境に合わせて値を変更してください。前提は次のとおりです。

- このシステムには、推奨される最新のパッチクラスタのインストールされた最新の Solaris 9 ソフトウェアをインストールします。適切な Solaris パッチがインストールされていない場合、Directory Server に対する Kerberos 認証は失敗する可能性があります。

  マニュアルに記述された手順は Solaris 10 とほとんど同じですが、いくつかの違いがあります。設定ファイルの形式が少しだけ異なるため、いくつかのコマンドの出力が同じでない場合もあります。
- Kerberos デーモンを実行するマシンには、`kdc.example.com` という完全修飾ドメイン名を付けます。このマシンは、ネームサービスに DNS を使用するように設定する必要があります。この設定は、Kerberos の要件です。`file` など、ほかのネームサービスを代わりに使用すると、特定の操作が失敗することもあります。
- Directory Server を実行するマシンには、`directory.example.com` という完全修飾ドメイン名を付けます。このマシンも、ネームサービスに DNS を使用するように設定する必要があります。
- Directory Server マシンは、Kerberos によって Directory Server に対する認証を行うためのクライアントシステムとしての役割を果たします。この認証は、Directory Server と Kerberos デーモンの両方と通信できる任意のシステムから実行できま

す。しかし、この例で必要なコンポーネントはすべて、Directory Server によって提供され、認証はこのシステムから実行されます。

- Directory Server のユーザーには、`uid=`*username*`,ou=People,dc=example,dc=com` という形式の DN があります。対応する Kerberos 主体は、*username*`@EXAMPLE.COM` です。別のネーミングスキームを使用する場合は、別の GSSAPI アイデンティティーマッピングを使用する必要があります。

すべてのマシン: Kerberos クライアント設定ファイルの編集

`/etc/krb5/krb5.conf` 設定ファイルは、KDC と通信するために Kerberos クライアントが必要とする情報を提供しています。

Kerberos を使用して Directory Server に対する認証を行う KDC マシン、Directory Server マシン、および任意のクライアントマシン上の `/etc/krb5/krb5.conf` 設定ファイルを編集します。

- `"___default_realm___"` をすべて `"EXAMPLE.COM"` に置き換えます。
- `"___master_kdc___"` をすべて `"kdc.example.com"` に置き換えます。
- `"___slave_kdcs___"` の含まれる行を削除して、Kerberos サーバーが 1 つしか存在しないようにします。
- `"___domain_mapping___"` を `".example.com = EXAMPLE.COM"` に置き換えます (`.example.com` の最初のピリオドに注意)。

更新された `/etc/krb5/krb5.conf` 設定ファイルは、次の例の内容のようになります。

例 5–1 編集後の kerberos クライアント設定ファイル `/etc/krb5/krb5.conf`

```
#pragma ident   "@(#)krb5.conf  1.2     99/07/20 SMI"
# Copyright (c) 1999, by Sun Microsystems, Inc.
# All rights reserved.
#
# krb5.conf template
# In order to complete this configuration file
# you will need to replace the __<name\>__ placeholders
# with appropriate values for your network.
#

[libdefaults]
        default_realm = EXAMPLE.COM
[realms]
        EXAMPLE.COM = {
                kdc = kdc.example.com
                admin_server = kdc.example.com
        }
[domain_realm]
```

例 5–1 編集後の kerberos クライアント設定ファイル /etc/krb5/krb5.conf (続き)

```
        .example.com = EXAMPLE.COM
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log
        kdc_rotate = {

# How often to rotate kdc.log. Logs will get rotated no more
# often than the period, and less often if the KDC is not used
# frequently.
                period = 1d

# how many versions of kdc.log to keep around (kdc.log.0, kdc.log.1, ...)
                versions = 10
        }

[appdefaults]
        kinit = {
                renewable = true
                forwardable= true
        }
        gkadmin = {
                help_url =
 http://docs.sun.com:80/ab2/coll.384.1/SEAM/@AB2PageView/1195
        }
```

すべてのマシン:管理サーバー ACL 設定ファイルの編集

/etc/krb5/kadm5.acl 設定ファイル内で、"___default_realm___" を "EXAMPLE.COM" に置き換えます。更新されたファイルは、次の例のようになります。

例 5–2 編集後の管理サーバー ACL 設定ファイル

```
#
# Copyright (c) 1998-2000 by Sun Microsystems, Inc.
# All rights reserved.
#
# pragma ident   "@(#)kadm5.acl  1.1     01/03/19 SMI"
*/admin@EXAMPLE.COM *
```

KDC マシン:KDC サーバー設定ファイルの編集

/etc/krb5/kdc.conf ファイルで、"___default_realm___" を "EXAMPLE.COM" に置き換えます。更新されたファイルは、次の例のようになります。

例 5–3 編集後の KDC サーバー設定ファイル /etc/krb5/kdc.conf

```
# Copyright 1998-2002 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "@(#)kdc.conf   1.2     02/02/14 SMI"

[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                admin_keytab = /etc/krb5/kadm5.keytab
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                default_principal_flags = +preauth
        }
```

KDC マシン: KDC データベースの作成

```
$ /usr/sbin/kdb5_util create -r EXAMPLE.COM -s
Initializing database ’/var/krb5/principal’ for realm ’EXAMPLE.COM’,
master key name ’K/M@EXAMPLE.COM’
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key: password
Re-enter KDC database master key to verify: password
$
```

KDC マシン: 管理の主体と鍵タブの作成

次のコマンドを使用して、kws/admin@EXAMPLE.COM という主体の管理ユーザーと、管理デーモンによって使用されるサービス鍵を作成します。

```
$ /usr/sbin/kadmin.local
kadmin.local:  add_principal kws/admin
Enter password for principal "kws/admin@EXAMPLE.COM": secret
Re-enter password for principal "kws/admin@EXAMPLE.COM": secret
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/kdc.example.com
Entry for principal kadmin/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab changepw/kdc.example.com
```

```
Entry for principal changepw/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/changepw
Entry for principal kadmin/changepw with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  quit$
```

KDC マシン : Kerberos デーモンの開始

次のコマンドを実行して、KDC および管理のデーモンを開始します。

```
$ /etc/init.d/kdc start
$ /etc/init.d/kdc.master start
$
```

KDC プロセスはプロセス一覧に `/usr/lib/krb5/krb5kdc` と表示されます。管理デーモンは、`/usr/lib/krb5/kadmind` と表示されます。

Solaris 10 OS では、デーモンは SMF (Service Management Facility) フレームワークによって管理されます。Solaris 10 OS でデーモンを起動します。

```
$ svcadm disable network/security/krb5kdc
$ svcadm enable network/security/krb5kdc
$ svcadm disable network/security/kadmin
$ svcadm enable network/security/kadmin
$
```

KDC マシン : KDC マシンと Directory Server マシンに対するホスト主体の追加

KDC の Kerberos データベースと Directory Server マシンにホスト主体を追加するには、次の一連のコマンドを使用します。ホスト主体は、`klist` などの特定の Kerberos ユーティリティーによって使用されます。

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  add_principal -randkey host/kdc.example.com
Principal "host/kdc.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/kdc.example.com
Entry for principal host/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  add_principal -randkey host/directory.example.com
Principal "host/directory.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/directory.example.com
Entry for principal host/directory.example.com with kvno 3, encryption type
```

```
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  quit
$
```

KDC マシン : Directory Server に対する LDAP 主体の追加

Directory Server で認証中のユーザーが持っている Kerberos チケットを検証できるようにするには、Directory Server が独自の主体を持っている必要があります。現在、Directory Server は `ldap/`*fqdn*`@`*realm* の主体を要求するためにハードコードされています。ここで、*fqdn* は Directory Server の完全修飾ドメイン名で、*realm* は Kerberos レルムです。*fqdn* は、Directory Server のインストール時に設定される完全修飾名と一致させる必要があります。この場合、Directory Server の主体は、`ldap/directory.example.com@EXAMPLE.COM` となります。

Directory Server の LDAP 主体を作成するには、次の一連のコマンドを使用します。

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin: add_principal -randkey ldap/directory.example.com
Principal "ldap/directory.example.com@EXAMPLE.COM" created.
kadmin: quit
$
```

KDC マシン : KDC へのテストユーザーの追加

Kerberos 認証を実行するには、Kerberos データベース内にユーザー認証が存在している必要があります。この例では、ユーザーのユーザー名を `kerberos-test` にします。つまり Kerberos 主体が `kerberos-test@EXAMPLE.COM` になります。

この例の一連のコマンドを使用してユーザーを作成します。

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  add_principal kerberos-test
Enter password for principal "kerberos-test@EXAMPLE.COM": secret

Re-enter password for principal "kerberos-test@EXAMPLE.COM": secret

Principal "kerberos-test@EXAMPLE.COM" created.
kadmin:  quit
$
```

Directory Server マシン : Directory Server のインストール

Directory Server 6.0 と最新のパッチをインストールします。設定例は次のとおりです。

変数のタイプ	値の例
完全修飾コンピュータ名	directory.example.com
インストールディレクトリ	/opt/SUNWdsee
インスタンスのパス	/local/ds
サーバーユーザー	unixuser
サーバーグループ	unixgroup
サーバーポート	389
サフィックス	dc=example,dc=com

Directory Server マシン : GSSAPI を有効にするための Directory Server の設定

最初に、ファイル /data/ds/shared/bin/gssapi.ldif を作成して、主体に基づいて認証を行う Kerberos ユーザーを識別するために Directory Server によって使用されるマッピングを定義します。次の例と同じ内容のファイルを作成します。

例 5–4 gssapi.ldif ファイルの内容

```
dn: cn=GSSAPI,cn=identity mapping,cn=config
changetype: add
objectClass: top
objectClass: nsContainer
cn: GSSAPI
dn: cn=default,cn=GSSAPI,cn=identity mapping,cn=config
changetype: add
objectClass: top
objectClass: nsContainer
objectClass: dsIdentityMapping
objectClass: dsPatternMatching
cn: default
dsMatching-pattern: \${Principal}
dsMatching-regexp: (.*)@EXAMPLE.COM
dsMappedDN: uid=\$1,ou=People,dc=example,dc=com

dn: cn=SASL,cn=security,cn=config
changetype: modify
replace: dsSaslPluginsPath
dsSaslPluginsPath: /usr/lib/mps/sasl2/libsasl.so
```

次に、ldapmodify コマンドを使用して、適切なマッピングで GSSAPI を有効にするために Directory Server を更新します。次の例を参照してください。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - -a -f /data/ds/shared/bin/gssapi.ldif
adding new entry cn=GSSAPI,cn=identity mapping,cn=config
adding new entry cn=default,cn=GSSAPI,cn=identity mapping,cn=config
modifying entry cn=SASL,cn=security,cn=config
$
```

Directory Server マシン : Directory Server 鍵タブの作成

これまでに説明したように、GSSAPI によって Kerberos ユーザーを認証するには、Directory Server は KDC 内に独自の主体が必要です。認証が正しく行われるためには、主体情報が Directory Server マシン上の Kerberos 鍵タブ内にある必要があります。この情報は、Directory Server が動作するユーザーアカウントが読み取れるファイル内にある必要があります。

正しいプロパティーを持つ鍵タブファイルを作成するには、次の一連のコマンドを使用します。

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  ktadd -k //local/ds/config/ldap.keytab ldap/directory.example.com
Entry for principal ldap/directory.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab
 WRFILE:/local/ds/config/ldap.keytab.
kadmin:  quit
$
```

このカスタム鍵タブのアクセス権と所有権を変更します。鍵タブを Directory Server を実行するために使用されるユーザーアカウントの所有にし、そのユーザーしか読み取れないようにします。

```
$ chown unixuser:unixgroup /local/ds/config /ldap.keytab
$ chmod 600 /local/ds/config/ldap.keytab
$
```

Directory Server は、デフォルトではファイル `/etc/kerb5/krb5.keytab` 内にある標準の Kerberos の鍵タブを使用しようとします。しかし、このファイルを Directory Server ユーザーが読めるようにすると、セキュリティー上のリスクが発生する可能性があります。これが、Directory Server 用のカスタム鍵タブを作成した理由です。

新しいカスタム鍵タブを使用するよう Directory Server を設定します。これは、`KRB5_KTNAME` 環境変数を設定して行います。

最後に Directory Server を再起動してこれらの変更を有効にします。

```
$ KRB5_KTNAME=/etc/krb5/ldap.keytab dsadm restart /local/ds
```

Directory Server マシン : Directory Server へのテストユーザーの追加

Kerberos ユーザーを Directory Server に対して認証するには、そのユーザーの Kerberos 主体に対応する、ユーザーのディレクトリエントリが必要です。

これまでの手順で、kerberos-test@EXAMPLE.COM という主体を持つテストユーザーが Kerberos データベースに追加されました。ディレクトリに追加されたアイデンティティーマッピング設定のために、そのユーザーに対応するディレクトリエントリには、uid=kerberos-test,ou=People,dc=example,dc=com という DN が必要です。

ユーザーをディレクトリに追加する前に、次の内容でファイル testuser.ldif を作成する必要があります。

例 5–5 新しい testuser.ldif ファイル

```
dn: uid=kerberos-test,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: kerberos-test
givenName: Kerberos
sn: Test
cn: Kerberos Test
description: An account for testing Kerberos authentication through GSSAPI
```

次に、ldapmodify を使用して、このエントリをサーバーに追加します。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - -f testuser.ldif
adding new entry uid=kerberos-test,ou=People,dc=example,dc=com
$
```

Directory Server マシン : テストユーザーとしての Kerberos チケットの取得

テストユーザーは、Kerberos データベース、Directory Server、および KDC 内に存在します。このため、Directory Server のテストユーザーとして GSSAPI を経由した Kerberos によって認証を行うことができるようになります。

まず、kinit コマンドを使用してユーザーの Kerberos チケットを取得します。次の例を参照してください。

```
$ kinit kerberos-test
Password for kerberos-test@EXAMPLE.COM: secret
$
```

次に、klist コマンドを使用して、このチケットに関する情報を表示します。

```
$ klist
Ticket cache: /tmp/krb5cc_0
Default principal: kerberos-test@EXAMPLE.COM

Valid starting           Expires                  Service principal
Sat Jul 24 00:24:15 2004  Sat Jul 24 08:24:15 2004  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until Sat Jul 31 00:24:15 2004
$
```

クライアントマシン : GSSAPI による Directory Server に対する認証

最後の手順は GSSAPI を使用した Directory Server に対する認証です。Directory Server の提供する ldapsearch ユーティリティーは、GSSAPI、DIGEST-MD5、および EXTERNAL メカニズムを含む SASL 認証をサポートしています。しかし、GSSAPI を使用してバインドするために、SASL ライブラリへのパスをクライアントに設定する必要があります。SASL_PATH 環境変数を lib/sasl ディレクトリに設定してパスを提供します。

```
$ SASL_PATH=SASL-library
$ export SASL_PATH
$
```

ldapsearch を使用して Directory Server に対して実際に Kerberos ベースの認証を実行するには、-o mech=GSSAPI 引数と -o authzid=*principal* 引数を含める必要があります。

また、ここで -h directory.example.com と表示している完全修飾ホスト名も指定する必要があります。これは、サーバーに対する cn=config 上の nsslapd-localhost 属性の値と一致する必要があります。GSSAPI 認証プロセスでは、クライアントから提供されたホスト名がサーバーから提供されたホスト名と一致する 必要があるため、ここでは -h オプションを使用する必要があります。

次の例では、これまでに作成した Kerberos テストユーザーアカウントとして認証を行なって、dc=example,dc=comエントリを取得しています。

```
$ ldapsearch -h directory.example.com -p 389 -o mech=GSSAPI \
 -o authzid="kerberos-test@EXAMPLE.COM" -b "dc=example,dc=com" -s base "(objectClass=*)"
version: 1
dn: dc=example,dc=com
dc: example
objectClass: top
objectClass: domain
$
```

正常に認証されたかどうか Directory Server のアクセスログを確認します。

```
$ tail -12 /local/ds/logs/access

[24/Jul/2004:00:30:47 -0500] conn=0 op=-1 msgId=-1 - fd=23 slot=23 LDAP
        connection from 1.1.1.8 to 1.1.1.8
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - RESULT err=0 tag=97
     nentries=0 etime=0 dn="uid=kerberos-test,ou=people,dc=example,dc=com"
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - SRCH base="dc=example,dc=com"
     scope=0 filter="(objectClass=*)" attrs=ALL
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - RESULT err=0 tag=101 nentries=1
     etime=0
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=5 - UNBIND
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=-1 - closing - U1
[24/Jul/2004:00:30:48 -0500] conn=0 op=-1 msgId=-1 - closed.
$
```

この例は、バインドが 3 つの手順によるプロセスであることを示しています。最初の 2 つの手順で LDAP 結果 14 (SASL バインド実行中) を返し、3 番目の手順はバインドが成功したことを示します。`method=sasl` タグと `mech=GSSAPI` タグは、このバインドに GSSAPI SASL メカニズムが使用されたことを示しています。成功したバインド応答の最後の `dn="uid=kerberos-test,ou=people,dc=example,dc=com"` は、このバインドが適切なユーザーとして実行されたことを示しています。

パススルー認証

パススルー認証 (PTA) は、バインド要求がバインド DN によってフィルタされるメカニズムです。ある Directory Server (委任者) がバインド要求を受け取り、フィルタに基づいて別の Directory Server (被委任者) にバインド要求を認証するよう求めることができます。この機能の一部として、PTA プラグインによって委任者である Directory Server がローカルのデータベースに保存されているとは限らないエントリに対する単純なパスワードベースのバインド操作を受け入れられるようになります。

PTA プラグインは、サーバーとの非公開の通信のために DSCC にも使用されます。サーバーインスタンスが DSCC に登録されている場合、PTA プラグインが有効になり、DSCC URL が引数として追加されます。

```
$ dsconf get-plugin-prop -h host -p port "Pass Through Authentication" enabled argument
argument  :  ldap://DSCC_URL:DSCC_PORT/cn=dscc
enabled   :  on
```

注 – 可能なかぎり、PTA プラグインを変更しないようにしてください。PTA プラグインを変更すると、DSCC のアクセス問題が発生する可能性があります。

PTA プラグインを変更しなければならない場合は、次の手順に従う必要があります。

- `enabled` プロパティーは `on` のままにします。
- `argument` プロパティーにほかの値を追加できますが、引数内で DSCC URL を維持します。

PTA プラグインが無効になったり、DSCC URL が引数から削除されると、サーバーインスタンスが DSCC 内に `inaccessible` として表示されます。この場合、DSCC が PTA プラグインをリセットするオプションを自動的に提供します。

◆ ◆ ◆ 第 6 章

Directory Server のアクセス制御

安全なディレクトリを作成する上で、ディレクトリの内容へのアクセスを制御することはもっとも重要です。この章では、ディレクトリに対してどのようなアクセス権をユーザーに許可するかを決定する ACI (アクセス制御命令) について説明します。

ディレクトリ配備の計画段階では、全体的なセキュリティーポリシーとして利用できるアクセス制御戦略を定義します。アクセス制御戦略の計画のヒントは、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

ACI の構文やバインドルールなど、ACI のその他の情報については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

この章の内容は次のとおりです。

ACI の作成、表示、および変更

ACI は、Directory Service Control Center (DSCC) を使用するかコマンド行を使用することで作成できます。どちらの方法を選択するとしても、たいていの場合、新しい ACI を最初から作成するよりも、既存の ACI 値を表示しコピーするほうが簡単です。

`aci` 属性値は、DSCC で表示および変更できます。DSCC による ACI の変更方法については、DSCC のオンラインヘルプを参照してください。

▼ ACI を作成、変更、および削除する

コマンド行を使用して ACI を作成するには、最初に LDIF 文を使ってファイル内に ACI を作成します。次に、`ldapmodify` コマンドを使用して ACI をディレクトリツリーに追加します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **LDIF ファイル内に ACI を作成します。**

```
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target)(version 3.0; acl "name";permission bindrules;)
```

この例は ACI の追加方法を示しています。ACI を変更または削除する場合は、`add` を `replace` または `delete` に置き換えます。

よく使われるその他の ACI の例については、151 ページの「アクセス制御の使用例」を参照してください。

2 **LDIF ファイルを使って変更を加えます。**

```
$ ldapmodify -h host -p port -D cn=admin,cn=Administrators,cn=config -w - -f ldif-file
```

▼ ACI 属性値を表示する

ACI はエントリの `aci` 属性の値として格納されます。`aci` 属性は、複数の値を持つオペレーショナル属性であり、ディレクトリユーザーはこの属性の読み取りや変更を行うことができます。このため、ACI 属性自体が ACI で保護される必要があります。通常、管理ユーザーには、`aci` 属性へのすべてのアクセス権が与えられます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 次の `ldapsearch` コマンドを実行して、エントリの **ACI** 属性値を表示します。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -b entryDN -s base "(objectclass=*)" aci
```

このコマンドで得られた LDIF テキストを、新しい LDIF ACI 定義にコピーして編集できます。ACI の値は長い文字列なので、`ldapsearch` 操作からの出力は複数行にわたって表示されることがあります。この場合、最初の空白は継続マーカーになります。LDIF の出力に継続マーカーを入れないようにするには、`-T` オプションを使用します。LDIF の出力をコピーおよびペーストする場合は、出力形式について考慮してください。

注 – 値 aci が権限を与えるか拒否するかを確認する場合は、165 ページの「実行権限の表示」を参照してください。

▼ ACI をルートレベルで表示する

サフィックスを作成すると、いくつかのデフォルトの ACI が最上位またはルートレベルで作成されます。これらの ACI により、デフォルトの管理者ユーザー cn=admin,cn=Administrators,cn=config が、ディレクトリマネージャーと同じディレクトリデータへのアクセス権限を持つことができます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- デフォルトのルートレベル **ACI** を表示します。

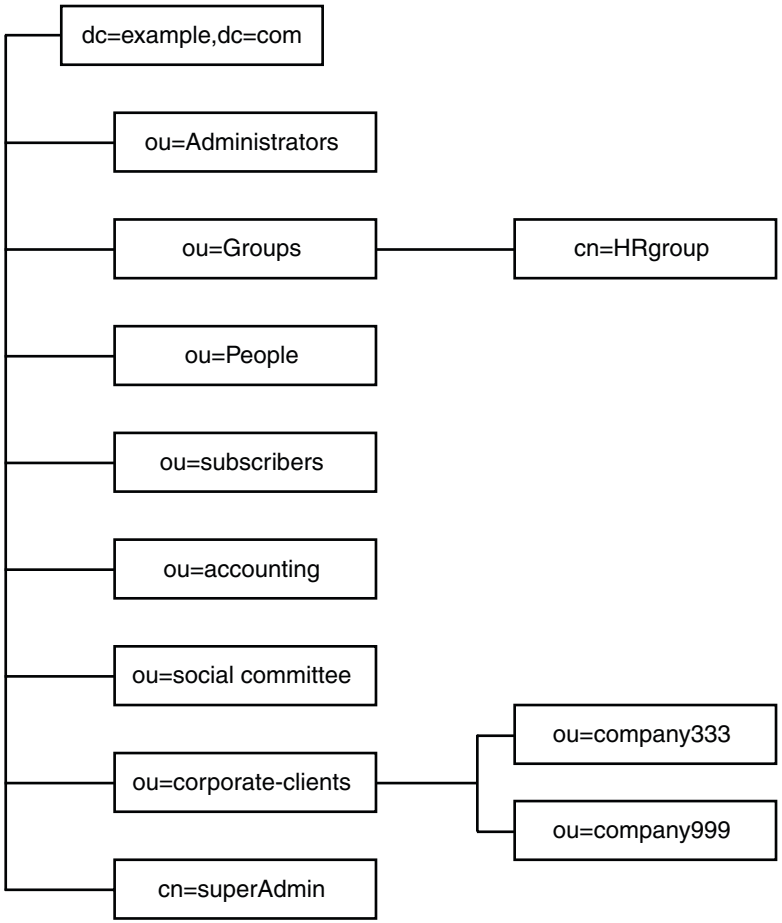
```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -b "" -s base "(objectclass=*)" aci
```

アクセス制御の使用例

この節で示す例では、架空の ISP である Example.com 社が、アクセス制御ポリシーを決定していきます。

また、インストールに付属のサンプルの LDIF ファイルに、*install_path*/ds6/ldif/Example.ldif というサンプルの ACI もあります。

すべての例では、LDIF ファイルを使用して、与えられたタスクをどのように処理するかを説明しています。次の図で、example.com 社のディレクトリ情報ツリーを示します。



Example.com 社の業務は、Web ホスティングサービスとインターネットアクセスの提供です。Example.com の Web ホスティングサービスには、クライアント企業のディレクトリのホスティングが含まれます。Example.com は実際に 2 つの中規模企業のディレクトリ Company333 と Company999 をホストし、部分的に管理を行なっています。また、多数の個人加入者にインターネットへのアクセスを提供しています。

現在、Example.com 社は、次のようなアクセス規則を設定しようとしています。

- Example.com 社の社員に、Example.com ツリー全体を対象とした読み取り、検索、および比較のための匿名アクセス権を与える。153 ページの「匿名アクセスの許可」を参照。
- Example.com 社の社員に、`homeTelephoneNumber`、`homeAddress` などの個人情報への書き込みアクセス権を与える。154 ページの「個人のエントリへの書き込みアクセス権の許可」を参照。

- Example.com 社の契約者に、会社の連絡先情報のエントリ `dc=example,dc=com` の読み取り権限を与える。ただし、その下のエントリの読み取り権限は与えない。156 ページの「特定のレベルへのアクセスの許可」を参照。
- Example.com 社の社員が個人のエントリにロールを追加するアクセス権を与える。ただし、一部の重要なロールは除く。156 ページの「重要なロールに対するアクセスの制限」を参照。
- 特定の管理者に、サフィックスに関してディレクトリマネージャーと同じ権限を与える。157 ページの「サフィックス全体に対するすべてのアクセス権のロールへの許可」を参照。
- Example.com 社の人事部グループに、People 分岐のエントリを対象としたすべての権限を与える。158 ページの「サフィックスに対するすべてのアクセス権のグループへの許可」を参照。
- Example.com 社のすべての社員に対し、ディレクトリの Social Committee エントリの下にグループエントリを作成し、社員が所有するグループエントリを削除するアクセス権を与える。158 ページの「グループエントリの追加および削除権限の許可」を参照。
- Example.com 社のすべての社員に対し、Social Committee エントリの下のグループエントリに、自身を追加するアクセス権を与える。160 ページの「ユーザー自身の操作によるグループへの参加とグループからの退会」を参照。
- 一定の条件付きで、ディレクトリツリーのそれぞれのエントリへのアクセス権を Company333 および Company999 のディレクトリ管理者 (ロール) に与える。これらの条件には、SSL 認証、日時の制約、位置の指定などが含まれる。160 ページの「グループまたはロールへの条件付きアクセスの許可」を参照。
- 個人契約者に対し、個人のエントリへのアクセス権を与える。154 ページの「個人のエントリへの書き込みアクセス権の許可」を参照。
- 個人契約者が個人のエントリ内の課金情報にアクセスできないようにする。161 ページの「アクセスの拒否」を参照。
- 世界のユーザーに対し、個人契約者のサブツリーへの匿名アクセス権を与える。ただし、非公開を希望している契約者は除く。ディレクトリのこの部分は、必要に応じてファイアウォール外部から読み取り専用サーバーになることがあり、毎日 1 回更新される。153 ページの「匿名アクセスの許可」および164 ページの「フィルタを使用したターゲットの設定」を参照。

匿名アクセスの許可

ほとんどのディレクトリは、読み取り、検索、または比較を行うために、少なくとも 1 つのサフィックスに匿名でアクセスできるように設定されています。社員が検索できる電話帳のような、企業内の個人情報を収めたディレクトリを管理している場合、そのためのアクセス権の設定が必要になることがあります。これは Example.com 社内のケースであり、154 ページの「ACI「Anonymous Example.com」」にその例が示されています。

Example.com 社では、ISP として、世界中からアクセス可能な公開電話帳を作成し、契約者全員の連絡先情報を公開することも計画しています。これについては、154 ページの「ACI「Anonymous World」」で例を示しています。

ACI「Anonymous Example.com」

Example.com 社の社員に Example.com ツリー全体を対象とした読み取り、検索、および比較アクセス権を与えるには、LDIF で次のような文を作成します。

```
aci: (targetattr !="userPassword")(version 3.0; acl "Anonymous
 example"; allow (read, search, compare)
 userdn= "ldap:///anyone") ;)
```

この例では、`aci` を `dc=example,dc=com` エントリに追加することを仮定しています。`userPassword` 属性は ACI の対象に含まれていません。

注 – 機密属性や表示すべきではない属性は、前の例でパスワード属性を保護しているのと同じように、「`(targetattr !="`*attribute-name*`")`」の構文を用いて保護してください。

ACI「Anonymous World」

個人契約者サブツリーの読み取りおよび検索アクセス権を世界中に与え、非公開にする契約者の情報へのアクセスを拒否するには、LDIF で次のような文を作成します。

```
aci: (targetfilter= "(!(unlistedSubscriber=yes))")
 (targetattr="homePostalAddress || homePhone || mail")
 (version 3.0; acl "Anonymous World"; allow (read, search)
 userdn="ldap:///anyone";)
```

この例では、ACI を `ou=subscribers,dc=example, dc=com` エントリに追加することを仮定しています。また、各契約者のエントリには、`yes` または `no` の値を持つ `unlistedSubscriber` 属性が設定されているものとします。非公開契約者は、この属性値に基づいて、ターゲット定義のフィルタによって除外されます。フィルタ定義については、164 ページの「フィルタを使用したターゲットの設定」を参照してください。

個人のエントリへの書き込みアクセス権の許可

多くの場合、内部ユーザーが個人で変更できるエントリの属性は、ディレクトリ管理者によって一部だけに制限されています。Example.com 社のディレクトリ管理者は、ユーザーが変更できる対象を、パスワード、自宅の電話番号、自宅住所だけに制限しようとしています。これについては、155 ページの「ACI「Write Example.com」」で例を示しています。

また、Example.com 社には、契約者がディレクトリに対して SSL 接続を確立することを条件に、Example.com ツリー内にある個人情報を更新できるようにするというポリシーもあります。これについては、155 ページの「ACI「Write Subscribers」」 で例を示しています。

ACI「Write Example.com」

注 – このアクセス権を設定することによって、ユーザーは属性値の削除アクセス権も与えられます。

Example.com 社の社員が、個人の自宅の電話番号、自宅住所を変更できるようにするには、LDIF で次のような文を作成します。

```
aci: (targetattr="homePhone ||
 homePostalAddress")(version 3.0; acl "Write Example.com";
 allow (write) userdn="ldap:///self" ;)
```

この例では、ACI を `ou=People,dc=example,dc=com`エントリに追加することを仮定しています。

ACI「Write Subscribers」

注 – このアクセス権を設定することによって、ユーザーは属性値の削除アクセス権も与えられます。

Example.com 社の契約者が個人の自宅の電話番号を変更できるようにするには、LDIF で次のような文を作成します。

```
aci: (targetattr="homePhone")
 (version 3.0; acl "Write Subscribers"; allow (write)
 userdn= "ldap://self" and authmethod="ssl";)
```

この例では、`aci` を `ou=subscribers,dc=example, dc=com` エントリに追加し、ユーザーは SSL を使用してバインドする必要があると仮定しています。

Example.com 社の契約者は、その住所の属性を削除する可能性があるため、住所への書き込みアクセス権は与えられていません。住所は Example.com 社からの請求に重要な情報です。

特定のレベルへのアクセスの許可

ディレクトリツリー内のさまざまなレベルに影響を及ぼす ACI の範囲を設定して、許可するアクセスのレベルを微調整できます。対象の ACI の範囲は、次のいずれかに設定できます。

`base` エントリ自体

`onelevel` エントリ自体と 1 レベル下のすべてのエントリ

`subtree` エントリ自体と、そのエントリの下のすべてのエントリ

ACI 「Read Example.com only」

Example.com 社の契約者に、会社の連絡先情報についてのエントリ `dc=example,dc=com` の読み取り権限は与えても、その下にあるエントリへのアクセスは許可しないようにするには、LDIF で次のような文を作成します。

```
aci: (targetscope="base") (targetattr="*")(version 3.0;
 acl "Read Example.com only";  allow (read,search,compare)
 userdn="ldap:///cn=*,ou=subscribers,dc=example,dc=com";)
```

この例では、ACI を `dc=example,dc=com` エントリに追加することを仮定しています。

重要なロールに対するアクセスの制限

ディレクトリ内のロール定義を使用して、ネットワークやディレクトリの管理などの業務に重要な機能を特定できます。

たとえば、国際的な企業のサイトで特定の時間と曜日に有効なシステム管理者のサブセットを指定する `superAdmin` ロールを作成する必要があるかもしれません。あるいは、特定のサイト上に、応急手当のトレーニングを受けたすべてのスタッフを含む `First Aid` ロールの作成が必要になることもあるかもしれません。ロール定義の作成については、226 ページの「ロールの管理」を参照してください。

ロールによって、業務上あるいはビジネス上重要な機能に関するユーザー特権を与える場合は、そのロールに対するアクセス制限を考慮してください。たとえば、次の例で示すように、Example.com の社員は、`superAdmin` ロール以外の任意のロールを個人のエントリに追加できます。

ACI 「Roles」

Example.com の社員が、`superAdmin` 以外の任意のロールを個人のエントリに追加できるようにするには、LDIF で次のような文を作成します。

```
aci: (targetattr="*") (targattrfilters="add=nsRoleDN:
 (nsRoleDN !="cn=superAdmin, dc=example, dc=com")")
 (version 3.0; acl "Roles"; allow (write)
 userdn= "ldap:///self" ;)
```

この例では、ACI を ou=People,dc=example,dc=comエントリに追加することを仮定しています。

サフィックス全体に対するすべてのアクセス権のロールへの許可

一部のユーザーに、サフィックスに対してディレクトリマネージャーと同じ権限を許可すると、便利な場合があります。Example.com 社の Kirsten Vaughan は Directory Server の管理者です。彼女は superAdmin のロールを持っています。このロールには、次のような利点があります。

- 管理者としてバインドし、SSL のような強力な認証を強制的に使用できることによって、セキュリティーが向上する
- ディレクトリマネージャーパスワードを知っている人が減ることによって、セキュリティーが向上する
- ログによる追跡容易性の向上

注 – Kirsten Vaughan を cn=Administrators,cn=config グループに追加すると、ディレクトリマネージャーと同じ権限が与えられます。

サーバー全体に対してディレクトリマネージャーと同じ権限をユーザーに与える際は、70 ページの「ルートアクセス権を持つ管理ユーザーを作成する」の手順に従ってください。

ACI「Full Access」

Kirsten Vaughan という管理者にディレクトリマネージャーと同じ権限を許可するには、LDIF で次のような文を使用します。

```
aci: (targetattr="*") (version 3.0; acl "Full Access";
 allow (all) groupdn= "ldap:///cn=SuperAdmin,dc=example,dc=com"
 and authmethod="ssl" ;)
```

この例では、ACI がルートエントリ "" (テキストなし) に追加されると仮定しています。

サフィックスに対するすべてのアクセス権のグループへの許可

ほとんどのディレクトリには、業務上の固有の職務を特定するためのグループがあります。グループには、ディレクトリのすべてまたは一部に対してアクセス権を与えることができます。グループにアクセス権限を与えることにより、グループメンバーに個別にアクセス権限を設定する必要がなくなります。代わりに、グループにメンバーを追加することで、アクセス権限をそのメンバーに与えることができます。

たとえば、Directory Server インスタンスを作成すると、ディレクトリへのすべてのアクセス権を持つ `cn=Administrators,cn=config` という管理者グループがデフォルトで作成されます。

Example.com 社の人事部グループには、ディレクトリの `ou=People` エントリへのすべてのアクセス権が許可されています。これによって、このグループのメンバーは、158 ページの「ACI 「HR」」に示すように社員のディレクトリを更新できます。

ACI 「HR」

ディレクトリの employee エントリに対するすべての権限を HR のグループに与えるには、LDIF で次のような文を作成します。

```
aci: (targetattr="*") (version 3.0; acl "HR"; allow (all)
  groupdn= "ldap:///cn=HRgroup,ou=Groups,dc=example,dc=com";)
```

この例では、ACI を次のエントリに追加することを仮定しています。

```
ou=People,dc=example,dc=com
```

グループエントリの追加および削除権限の許可

一部の企業では、業務の効率化や企業全体の活力向上につなげるため、社員自身がツリー内にエントリを作成できるようにしています。たとえば、Example.com 社には、テニス、水泳、スキー、演劇などのさまざまなクラブが組織された社内委員会があります。

Example.com 社の社員はだれでも、159 ページの「ACI「Create Group」」に示すように、新しいクラブを表すグループエントリを作成できます。

160 ページの「ユーザー自身の操作によるグループへの参加とグループからの退会」に示すように、Example.com 社の社員であれば、これらのグループのどれか 1 つのメンバーになることができます。

159 ページの「ACI「Delete Group」」に示すように、グループエントリの変更や削除ができるのは、グループの所有者のみです。

ACI「Create Group」

Example.com 社の社員が ou=Social Committee エントリの下にグループエントリを作成できるようにするには、LDIF で次のような文を作成します。

```
aci: (targetattr="*") (targattrfilters="add=objectClass:
(|(objectClass=groupOfNames)(objectClass=top))")
(version 3.0; acl "Create Group"; allow (read,search,add)
userdn= "ldap:///uid=*,ou=People,dc=example,dc=com")
and dns="*.Example.com";)
```

この例では、ACI を ou=Social Committee, dc=example,dc=com エントリに追加することを仮定しています。

注 –

- この ACI は、書き込みアクセス権を与えないので、エントリを変更できません。
- サーバーが top という値を追加するので、targattrfilters キーワードで objectClass=top を指定する必要があります。
- この ACI は、example.com ドメイン内のクライアントマシンにのみ適用されます。

ACI「Delete Group」

Example.com 社の社員が ou=Social Committee エントリの下に属しているグループエントリを変更または削除できるようにするには、LDIF で次のような文を作成します。

```
aci: (targetattr = "*") (targattrfilters="del=objectClass:
(objectClass=groupOfNames)")
 (version 3.0; acl "Delete Group"; allow (write,delete)
 userattr="owner#GROUPDN";)
```

この例では、aci を ou=Social Committee, dc=example,dc=com エントリに追加しています。

DSCC を使用してこの ACI を作成すると、手動編集モードでのターゲットフィルタの作成とグループ所有権の確認が必要なので、あまり効率的ではありません。

ユーザー自身の操作によるグループへの参加とグループからの退会

多くのディレクトリの ACI は、ユーザーがメーリングリストなどのグループへの参加とグループからの退会を自分で設定できるようになっています。

Example.com 社では、160 ページの「ACI「Group Members」」に示すように、社員であれば `ou=Social Committee` サブツリーの下のどのグループエントリにも参加できます。

ACI「Group Members」

Example.com 社の社員が自分でグループへの参加を設定できるようにするには、LDIF で次のような文を作成します。

```
aci: (targettattr="member")(version 3.0; acl "Group Members";
 allow (selfwrite)
 (userdn= "ldap:///uid=*,ou=People,dc=example,dc=com") ;)
```

この例では、ACI を `ou=Social Committee,dc=example,dc=com` エントリに追加することを仮定しています。

グループまたはロールへの条件付きアクセスの許可

多くの場合、ディレクトリへのアクセス特権をグループやロールに与える場合、それらの特権が、特権ユーザーになりすました侵入者から保護されていることを確認する必要があります。したがって、多くの場合、グループまたはロールへの重要なアクセス権を与えるようなアクセス制御規則には、数多くの条件が付けられます。

たとえば、Example.com 社では、ホスティングサービスの提供先企業である Company333 および Company999 に対して、それぞれ Directory Administrator ロールを作成しました。Example.com 社では、侵入者からデータを保護するために、それぞれの企業が各自でデータを管理し、独自のアクセス制御規則を決定することを求めています。

このため、Company333 と Company999 は、ディレクトリツリーのそれぞれのエントリに関してすべての権限を持っていますが、このアクセス権を行使するには次の条件を満たす必要があります。

- 証明書を使用して、SSL 経由の接続が認証されること
- アクセス要求は月曜日から木曜日の午前 8 時から午後 6 時までの間に限ること

- それぞれの企業に割り当てられた特定の IP アドレスからアクセスが要求されること

これらの条件は、各社の ACI である「Company333」と「Company999」に示されています。これらの ACI の内容は同等なので、「Company333」という ACI だけを次に示します。

ACI「Company333」

Company333 に対して、前述した条件に従ったディレクトリの自社のエントリへのすべてのアクセス権を与えるには、LDIF で次のような文を作成します。

```
aci: (targetattr = "*") (version 3.0; acl "Company333"; allow (all)
 (roledn="ldap:///cn=DirectoryAdmin,ou=Company333,
 ou=corporate clients,dc=example,dc=com") and (authmethod="ssl")
 and (dayofweek="Mon,Tues,Wed,Thu") and (timeofday >= "0800" and
 timeofday <= "1800") and (ip="255.255.123.234"); )
```

この例では、ACI を `ou=Company333,ou=corporate-clients,dc=example,dc=com` エントリに追加することを仮定しています。

アクセスの拒否

サフィックスの大半の部分に対してのアクセスをすでに許可している場合に、既存の ACI の配下にあるサフィックスの限定された範囲に対してアクセスを拒否させることもできます。

注 – アクセスを拒否すると、アクセス制御の振る舞いが予期しないものや複雑なものになる可能性があるため、可能な限り回避してください。アクセスは、範囲、属性リスト、ターゲットフィルタなどの組み合わせを使って制限してください。

また、アクセス拒否 ACI を削除しても、権限が削除されるのではなく、ほかの ACI で設定された権限の幅が広げられます。

Directory Server は、アクセス権限を評価するとき、最初に `deny` 権限を読み取り、次に `allow` 権限を読み取ります。

次の例で、Example.com 社では、すべての契約者に対し、契約者自身のエントリにある接続時間や料金内訳などの課金情報の読み取りアクセス権を与えています。また、Example.com 社では、その情報に対する書き込みアクセス権は拒否するようにしています。読み取りアクセス権については、162 ページの「ACI「Billing Info Read」」で例を示しています。`deny` アクセス権については、162 ページの「ACI「Billing Info Deny」」で例を示しています。

ACI「Billing Info Read」

個人のエントリ内にある課金情報の読み取りアクセス権を契約者に与えるには、LDIF で次のような文を作成します。

```
aci: (targetattr="connectionTime || accountBalance")
 (version 3.0; acl "Billing Info Read"; allow (search,read)
  userdn="ldap:///self";)
```

この例は、関連する属性がスキーマ内で作成済みであり、ACI を `ou=subscribers,dc=example,dc=com` エントリに追加しています。

ACI「Billing Info Deny」

各契約者に対し、契約者個人のエントリ内にある課金情報の変更アクセス権を拒否するには、LDIF で次のような文を作成します。

```
aci: (targetattr="connectionTime || accountBalance")
 (version 3.0; acl "Billing Info Deny";
 deny (write) userdn="ldap:///self";)
```

この例は、関連する属性がスキーマ内で作成済みであり、ACI を `ou=subscribers,dc=example,dc=com` エントリに追加しています。

プロキシ承認

プロキシ承認方式は、特殊な形式の認証です。自分のアイデンティティーでディレクトリにバインドしたユーザーに、プロキシ承認を使用して他のユーザの権限が与えられます。

プロキシ要求を許可するように Directory Server を設定するには、次のことを行う必要があります。

- 管理者には、ほかのユーザーとしてのプロキシ権限を与える。
- 一般ユーザーには、アクセス制御ポリシーで定義されている通常のアクセス権限を与える。

注 – ディレクトリマネージャーを除く、ディレクトリのすべてのユーザーにプロキシ権限を与えることができます。また、ディレクトリマネージャーの DN をプロキシ DN として使用することはできません。プロキシ権限により、すべての DN (ディレクトリマネージャー DN を除く) をプロキシ DN として指定する権限が与えられるので、プロキシ権限を与える場合には十分な注意が必要です。同じ操作中に Directory Server が複数のプロキシ認証の制御を受け取った場合は、クライアントアプリケーションにエラーが返され、操作の試行は失敗します。

プロキシ認証の例

Example.com 社は、`MoneyWizAcctSoftware` としてバインドするクライアントアプリケーションに、Accounting Administrator と同じ LDAP データへのアクセス権限を与えようとしています。

次の条件が適用されます。

- クライアントアプリケーションのバインド DN は `uid=MoneyWizAcctSoftware, ou=Applications,dc=example,dc=com`。
- クライアントアプリケーションがアクセスを要求するターゲットサブツリーは `ou=Accounting,dc=example,dc=com`。
- ディレクトリ内に、`ou=Accounting,dc=example,dc=com` サブツリーへのアクセス権を持つ Accounting Administrator が存在する。

クライアントアプリケーションが Accounting サブツリーへのアクセス権を取得するには、Accounting Administrator と同じアクセス権を使用して、次の条件を満たす必要があります。

- Accounting Administrator は、`ou=Accounting,dc=example,dc=com` サブツリーへのアクセス権を持っている必要がある。たとえば、次の ACI は Accounting Administrator エントリに対するすべての権限を与えます。

```
aci: (targetattr="*") (version 3.0; acl "allowAll-AcctAdmin"; allow
 (all) userdn="ldap:///uid=AcctAdministrator,ou=Administrators,
 dc=example,dc=com";)
```

- クライアントアプリケーションに対するプロキシ権限を与える次の ACI が、ディレクトリ内に存在する必要がある。

```
aci: (targetattr="*") (version 3.0; acl "allowproxy- accountingsoftware";
 allow (proxy) userdn= "ldap:///uid=MoneyWizAcctSoftware,ou=Applications,
dc=example,dc=com";)
```

この ACI が設定されていれば、`MoneyWizAcctSoftware` クライアントアプリケーションがディレクトリにバインドしてから、プロキシ DN のアクセス権限を要求する `ldapsearch` や `ldapmodify` などの LDAP コマンドを送信できます。

この例で、クライアントが ldapsearch コマンドを実行する場合は、このコマンドに次の制御が含まれます。

```
$ ldapsearch -D "uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com" -w - \
 -y "uid=AcctAdministrator,ou=Administrators,dc=example,dc=com" ...
```

クライアントはそのままバインドしますが、プロキシエントリの特権が与えられます。クライアントには、プロキシエントリのパスワードは必要ありません。

フィルタを使用したターゲットの設定

ディレクトリ内に分散した多数のエントリに対して、アクセス制御の設定が必要な場合は、フィルタを使用してターゲットを設定することもできます。

フィルタを使って HR のすべてのユーザーに従業員エントリへのアクセスを許可するには、LDIF で次のような文を作成します。

```
aci: (targetattr="*") (targetfilter=(objectClass=employee))
 (version 3.0; acl "HR access to employees";
 allow (all) groupdn= "ldap:///cn=HRgroup,ou=People,dc=example,dc=com";)
```

この例では、ACI を ou=People,dc=example,dc=comエントリに追加することを仮定しています。

注 – 検索フィルタは、アクセス管理の対象となるオブジェクトを直接指定するわけではないので、誤ったオブジェクトへのアクセスを許可または拒否しないようにしてください。ディレクトリ構造が複雑になるほど、誤ったオブジェクトへのアクセスを許可または拒否してしまうリスクが大きくなります。さらに、フィルタによって、ディレクトリ内のアクセス制御に関する問題解決が難しくなる場合もあります。

コンマを含む **DN** のアクセス権の定義

DN にコンマが含まれている場合、LDIF ACI 文の中で特別な処理が必要です。ACI 文のターゲット部分とバインドルール部分で、1 つの円記号 (\) を使用して、コンマをエスケープする必要があります。次に、この構文の例を示します。

```
dn: o=Example.com Bolivia\, S.A.
objectClass: top
objectClass: organization
aci: (target="ldap:///o=Example.com Bolivia\,S.A.") (targetattr="*")
(version 3.0; acl "aci 2"; allow (all) groupdn =
```

```
"ldap:///cn=Directory Administrators, o=Example.com Bolivia\, S.A.";)
```

実行権限の表示

ディレクトリのエントリに対するアクセスポリシーを管理するとき、定義した ACI がセキュリティーに与える影響を把握しておく必要があります。Directory Server は、ACI が特定のエントリに対して特定のユーザーに与える実行権限を確認することで、既存の ACI を評価します。

この実行権限の取得制御は検索操作で使用でき、Directory Server はそれに対して応答します。この制御に対する応答として、エントリと属性に対する実行権限の情報が検索結果の中で返されます。この追加情報としては、各エントリとその中の各属性に対する読み取り権限および書き込み権限などがあります。検索に使用されるバインド DN や任意の DN では権限を要求することができます。これを選択することで、管理者はディレクトリユーザーの権限を検査できます。

実行権限を表示する機能は、LDAP 制御を利用しています。リモートサーバーとのバインドに使用されるプロキシアイデンティティーにも、実行権限の属性へのアクセスが許可されていることを確認してください。

実行権限取得制御へのアクセスの制限

実行権限を表示するという操作はディレクトリ操作であり、保護し、適切に制限する必要があります。

実行権限情報へのアクセスを制限するには、`getEffectiveRights` 属性のデフォルトの ACI を変更します。次に、`getEffectiveRightsInfo` 属性に新しい ACI を作成します。

たとえば、次の ACI では、ディレクトリ管理者グループのメンバーのみが実行権限取得へのアクセスを許可されます。

```
aci: (targetattr != "aci")(version 3.0; acl
 "getEffectiveRights"; allow(all) groupdn =
 "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

実行権限情報を取得するには、実行権限制御を使用するためのアクセス制御権と、`aclRights` 属性に対する読み取りアクセス権が必要です。このような二重の層に成すアクセス制御により、基本的なセキュリティーを必要に応じて微調整できます。プロキシ承認のように、`aclRights` に対する読み取りアクセス権があれば、エントリと属性に対するどのユーザーの権限に関する情報でも要求することができます。つま

り、リソースを管理するユーザーは、だれがそのリソースに対する権限を持つかを決定できます。これは、そのユーザーが実際にはその権限を使って管理していない場合も同様です。

権限情報を照会しようとしているユーザーが実行権限制御を使用する権限を持たない場合、操作は失敗し、エラーメッセージが返されます。一方、権限情報を照会しようとしているユーザーがこの制御を使用する権限は持つが、`aclRights` 属性に対する読み取りアクセス権を持たない場合は、返されるエントリから `aclRights` 属性が省略されます。この動作は、Directory Server の通常の検索動作を反映しています。

実行権限の取得制御の使用

実行権限の取得制御を指定するには、`ldapsearch` コマンドに `-J"1.3.6.1.4.1.42.2.27.9.5.2"` オプションを指定して実行します。デフォルトでは、エントリと属性に対してバインド DN エントリが持っている実行権限が検索結果の中で返されます。

デフォルトの動作を変更するには、次のオプションを使用します。

- `-c "dn:` *bind DN* `"` — 検索結果には、指定された *DN* にバインドされているユーザーの実行権限が表示されます。管理者はこのオプションを使用して、別のユーザーの実行権限を確認できます。`-c "dn:"` オプションを指定すると、匿名認証用の実行権限が表示されます。
- `-X "`*attributeName ...*`"`— 検索結果には、指定された属性に対する実行権限も表示されます。このオプションは、検索結果に表示されない属性を指定する場合に使用します。たとえば、このオプションを使用すると、現在はエントリに存在していない属性について、ユーザーがその属性を追加する権限を持っているかどうかを調べることができます。
- `-c` オプションまたは `-X` オプション、あるいはその両方を使用するときは、`-J` オプションに実行権限の取得制御の OID が暗黙的に指定されるため、このオプションを指定する必要はありません。実行権限制御に NULL 値を指定すると、現在のユーザーの権限を取得できます。また、現在の `ldapsearch` 操作によって返される属性とエントリの権限も取得できます。

  次に、表示する情報の種類を選択する必要があります。権限だけを表示するか、権限がどのように許可または拒否されているかを示す詳細なログ情報を表示するか、いずれかを選択します。情報の種類を指定するには、検索結果で返す属性として `aclRights` または `aclRightsInfo` を追加します。両方の属性を要求すると、実行権限の情報をすべて取得できます。ただし、単純な権限情報は基本的には詳細なログ情報の写しです。

注 – aclRights 属性と aclRightsInfo 属性は、仮想オペレーショナル属性のように動作します。これらの属性はディレクトリには格納されず、明示的に要求された場合以外は返されません。これらの属性は、実行権限の取得制御に対する応答として Directory Server で生成されます。

このため、これらの属性を、フィルタや何らかの検索操作に使用することはできません。

実行権限機能は、アクセス制御に影響を与えるその他のパラメーターを継承します。これらのパラメータには、時刻、認証方法、マシンアドレス、名前が含まれます。

次の例は、Carla Fuente というユーザーがディレクトリでの自身の権限を確認する方法を示しています。結果の中で、1 は権限が与えられていることを示し、0 は拒否されていることを示します。

```
$ ldapsearch -J "1.3.6.1.4.1.42.2.27.9.5.2 -h host1.Example.com -p 389 \
 -D "uid=cfuente,ou=People,dc=example,dc=com" -w - -b "dc=example,dc=com" \
 "(objectclass=*)" aclRights
Enter bind password:
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

この結果は、Carla Fuente にはディレクトリ内のエントリに少なくとも読み取り権限が与えられていて、自分のエントリを変更できることを示しています。実行権限制御は、通常のアクセス権をバイパスしないため、ユーザーは読み取り権限が与えられていないエントリを見ることはできません。次の例で、ディレクトリマネージャーは、Carla Fuente に読み取り権限が与えられていないエントリを確認できます。

```
$ ldapsearch -h host1.Example.com -p 389 -D cn=admin,cn=Administrators,cn=config -w - \
 -c "dn: uid=cfuente,ou=People,dc=example,dc=com" -b "dc=example,dc=com" \
```

```
 "(objectclass=*)" aclRights
Enter bind password:
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Directory Administrators, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=Special Users,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

上記の出力で、ディレクトリマネージャーは、Carla Fuente がディレクトリツリーの Special Users エントリと Directory Administrators エントリのどちらも表示できないことを確認できます。次の例では、ディレクトリマネージャーは、Carla Fuente が自身のエントリの `mail` 属性と `manager` 属性を変更できないことを確認できます。

```
$ ldapsearch -h host1.Example.com -p 389 -D cn=admin,cn=Administrators,cn=config -w - \
 -c "dn: uid=cfuente,ou=People,dc=example,dc=com" -b "dc=example,dc=com" \
 "(uid=cfuente)" aclRights "*"
Enter bind password:
version: 1
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;attributeLevel;mail: search:1,read:1,compare:1,
 write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
mail: cfuente@Example.com
aclRights;attributeLevel;uid: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
uid: cfuente
aclRights;attributeLevel;givenName: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
givenName: Carla
aclRights;attributeLevel;sn: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
sn: Fuente
aclRights;attributeLevel;cn: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
cn: Carla Fuente
```

```
aclRights;attributeLevel;userPassword: search:0,read:0,
 compare:0,write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
userPassword: {SSHA}wnbWHIq2HPiY/5ECwe6MWBGx2KMiZ8JmjF80Ow==
aclRights;attributeLevel;manager: search:1,read:1,compare:1,
 write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
manager: uid=bjensen,ou=People,dc=example,dc=com
aclRights;attributeLevel;telephoneNumber: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
telephoneNumber: (234) 555-7898
aclRights;attributeLevel;objectClass: search:1,read:1,compare:1,
 write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
```

高度なアクセス制御: マクロ ACI の使用

同じようなディレクトリツリー構造を持つ組織では、マクロによってディレクトリ内で使用する ACI の数を最適化できます。ディレクトリツリー内の ACI の数を減らすことによって、アクセス制御ポリシーの管理が簡単になります。また、ACI によるメモリー使用の効率も向上します。

マクロは、ACI の中で DN、または DN の一部を表現するために使用される可変部分です。マクロを使用すると、ACI のターゲット部分またはバインドルール部分、あるいはその両方の DN を表すことができます。実際の処理では、Directory Server が LDAP 操作を受け取ると、LDAP 操作のターゲットとなるリソースに対して ACI マクロのマッチングが行われます。このマッチングは、一致する部分文字列の存在を確認するために行われます。一致が検出された場合は、一致した部分文字列を使用してバインドルール側のマクロが展開され、その展開バインドルールを評価してリソースにアクセスします。

この節では、マクロ ACI の例とマクロ ACI 構文についての情報を示します。

マクロ ACI の例

マクロ ACI の利点ともっとも効果的に機能させる方法を、例を示しながら説明します。図 6–1 は、全体的な ACI の数を減らすために、マクロ ACI を効果的に利用しているディレクトリツリーです。

この例では、同じツリー構造のサブドメインが同じパターンで繰り返されています (`ou=groups,ou=people`)。Example.com ディレクトリツリーには、

dc=hostedCompany2,dc=example,dc=com および dc=hostedCompany3,dc=example,dc=com という 2 つのサフィックスが格納されているので、このパターンはツリー内でも繰り返されています。ただし、図には示されていません。

ディレクトリツリーにある ACI でも、同じパターンが繰り返されています。たとえば、次の ACI は dc=hostedCompany1,dc=example,dc=com ノード上に置かれています。

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))(version 3.0;
 acl "Domain access"; allow (read,search) groupdn=
 "ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,
 dc=example,dc=com";)
```

この ACI は、dc=hostedCompany1,dc=example,dc=com ツリー内のすべてのエントリに対する読み取りおよび検索権限を DomainAdmins グループに与えます。

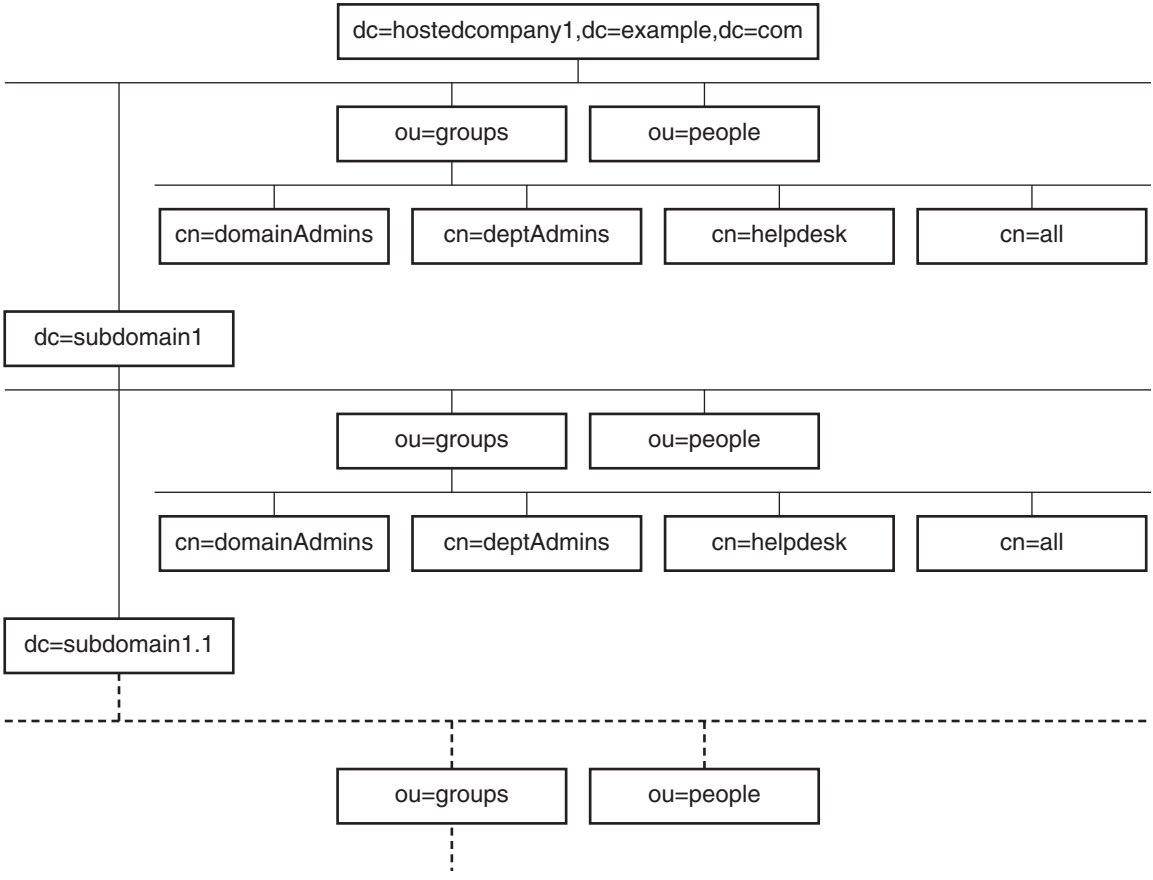


図 6–1 マクロ ACI のディレクトリツリーの例

次の ACI は、dc=hostedCompany1,dc=example,dc=com ノード上に置かれています。

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

次の ACI は、dc=subdomain1,dc=hostedCompany1, dc=example,dc=com ノード上に置かれています。

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,
  dc=example,dc=com";)
```

次の ACI は、dc=hostedCompany2,dc=example,dc=com ノード上に置かれています。

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany2, dc=example,dc=com";)
```

次の ACI は、dc=subdomain1,dc=hostedCompany2, dc=example,dc=com ノード上に置かれています。

```
aci: (targetattr="*")
 (targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany2,
 dc=example,dc=com";)
```

前述の 4 つの ACI の違いは、groupdn キーワード内で指定されている DN だけです。DN 用のマクロを使用することによって、これらの ACI を、1 つの ACI にまとめてルートツリーの dc=example,dc=com ノードに置くことができます。このマクロ ACI は次のようになります。

```
aci: (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
 (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

前述の個々の ACI では使われていなかった target キーワードをここでは新たに使っていることに注意してください。

この例では、ACI の数が 4 つから 1 つに減っています。ただし、本当の利点はそれだけではなく、ディレクトリツリー全体で複数繰り返されているパターンをまとめることができるところにあります。

マクロ ACI の構文

ここでは、わかりやすくするために、userdn、roledn、groupdn、userattr などのバインド資格を与えるために使用される ACI キーワードをまとめて、「サブジェクト」と呼びます。サブジェクトは、ACI の適用対象を決定します。

次の表に、特定の ACI キーワードの置換に使用できるマクロを示します。

表 6–1 マクロ ACI キーワード

マクロ	説明	ACI キーワード
($dn)	target 文でマッチング要素として用いられます。サブジェクト内では、実際に {$dn} と一致した文字列に置き換えられます。	target、targetfilter、userdn、roledn、groupdn、userattr
[$dn]	サブジェクトのサブツリー内のあらゆる RDN に置き換えられます。	targetfilter、userdn、roledn、groupdn、userattr
($attr.*attrName*)	ターゲットエントリの *attrName* 属性に設定されている値に置き換えられて、サブジェクトに設定されます。	userdn、roledn、groupdn、userattr

マクロ ACI キーワードには、次のような制限が適用されます。

- サブジェクトで ($dn) マクロや [$dn] マクロを使用するときは、($dn) マクロを含む target 文を定義する必要があります。
- サブジェクトで ($attr.*attrName*) マクロと ($dn) マクロを組み合わせることはできますが、[$dn] マクロと組み合わせることはできません。

ターゲットエントリでの ($dn) とのマッチング処理

ACI の target 文に含まれる ($dn) マクロを、LDAP 要求のターゲットとなるエントリと比較することによって、実際に置き換えられる値が決定します。たとえば、このエントリをターゲットとする LDAP 要求があるとします。

```
cn=all,ou=groups,dc=subdomain1, dc=hostedCompany1,dc=example,dc=com
```

また、次のような target 文を定義している ACI もあるとします。

```
(target="ldap:///ou=Groups,($dn),dc=example,dc=com")
```

この場合、(`$dn`) マクロは `dc=subdomain1, dc=hostedCompany1` と一致します。ACI のサブジェクト内で (`$dn`) はこの部分文字列に置き換えられます。

サブジェクト内での (`$dn`) の置換

ACI のサブジェクト内では、(`$dn`) マクロはターゲット内で一致する部分文字列全体に置き換えられます。次に例を示します。

```
groupdn="ldap:///cn=DomainAdmins,ou=Groups,($dn),dc=example,dc=com"
```

サブジェクトは次のようになります。

```
groupdn="ldap:///cn=DomainAdmins,ou=Groups,
 dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"
```

マクロが展開されると、通常のプロセスに続いて Directory Server が ACI を評価し、アクセス権が与えられるかどうかを決定します。

注 – 標準の ACI とは異なり、マクロ置換を使った ACI はターゲットエントリの子へのアクセスを許可する必要はありません。これは、子の DN がターゲットとなった場合に、置換によってサブジェクト文字列内に有効な DN が作成されない可能性があるためです。

サブジェクト内での [`$dn`] の置換

[`$dn`] の置換メカニズムは (`$dn`) のものと少し異なります。一致する対象が見つかるまで、一番左にある RDN コンポーネントを外しながらマクロの展開を継続して行い、ターゲットリソースの DN の検証を繰り返します。

たとえば、`cn=all,ou=groups, dc=subdomain1,dc=hostedCompany1,dc=example,dc=com` サブツリーをターゲットとする LDAP 要求で、次のような ACI があるとします。

```
aci: (targetattr="*")
 (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],
 dc=example,dc=com";)
```

サーバーは次のように処理を続け、この ACI を展開します。

1. target 文の (`$dn`) が `dc=subdomain1,dc=hostedCompany1` に一致します。
2. サブジェクトの `[$dn]` を `dc=subdomain1,dc=hostedCompany1` で置き換えます。

   この結果、サブジェクトは `groupdn="ldap:///cn=DomainAdmins,ou=Groups, dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"` になります。バインド DN がそのグループのメンバーであるためにアクセスが許可される場合は、マクロの展開を停止し、ACI を評価します。バインド DN がメンバーでない場合は、処理を継続します。
3. サブジェクトの `[$dn]` を `dc=hostedCompany1` で置き換えます。

   この結果、サブジェクトは `groupdn="ldap:///cn=DomainAdmins,ou=Groups, dc=hostedCompany1,dc=example,dc=com"` になります。バインド DN がこのグループのメンバーであるかどうかが再び検証され、メンバーである場合は ACI が完全に評価されます。バインド DN がメンバーでない場合は、マクロの展開は最後に一致した RDN で置き換えたところで終了し、この ACI に対する評価は終了します。

`[$dn]` マクロの利点は、ドメインレベルの管理者に対して、ディレクトリツリー内のすべてのサブドメインへのアクセス権を柔軟な方法で与えることができることです。したがって、`[$dn]` マクロは、ドメイン間の階層的な関係を表す場合に便利です。

たとえば、次のような ACI があるとします。

```
aci: (target="ldap:///ou=*,($dn),dc=example,dc=com") (targetattr="*")
(targetfilter=(objectClass=nsManagedDomain))
(version 3.0; acl "Domain access"; allow (read,search) groupdn=
"ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

この ACI は、`cn=DomainAdmins,ou=Groups, dc=hostedCompany1,dc=example,dc=com` のすべてのメンバーに対して、 `dc=hostedCompany1` の下にあるすべてのサブドメインへのアクセス権を与えます。したがって、そのグループに属する管理者は、サブツリー `ou=people,dc=subdomain1.1,dc=subdomain1` などにアクセスできます。

ただし、同時に、`cn=DomainAdmins,ou=Groups, dc=subdomain1.1` のメンバーの `ou=people,dc=subdomain1, dc=hostedCompany1` および `ou=people,dc=hostedCompany1` ノードに対するアクセスは拒否されます。

(`$attr.`*attrName*`)` に対するマクロマッチング

(`$attr.`*attrname*) マクロは、常に DN のサブジェクト部分で使用されます。たとえば、次のような `roledn` を定義できます。

```
roledn = "ldap:///cn=DomainAdmins,($attr.ou),dc=HostedCompany1,dc=example,dc=com"
```

ここで、サーバーが次のエントリをターゲットとする LDAP 操作を受け取ったとします。

```
dn: cn=Babs Jensen,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Babs Jensen
sn: Jensen
ou: Sales
...
```

ACI の roledn 部分を評価するために、サーバーはターゲットエントリの ou 属性の値を読み取ります。そのあと、サブジェクト内でこの値を置換してマクロを展開します。この例では、roledn は次のように展開されます。

```
roledn = "ldap:///cn=DomainAdmins,ou=Sales,dc=HostedCompany1,dc=example,dc=com"
```

続いて、通常の ACI 評価アルゴリズムに従って、Directory Server が ACI を評価します。

マクロ内で指定された属性が複数の値を持つ場合は、それぞれの値でマクロが展開されます。最初にマッチングに成功した値が使用されます。

アクセス制御情報のログ

エラーログに記録されているアクセス制御に関する情報を取得するには、適切なログレベルを設定する必要があります。

▼ ACI のログを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- **ACI の処理を考慮に入れるようにログレベルを設定します。**

  ```
  $ dsconf set-log-prop -h host -p port error level:err-acl
  ```

TCP ラップによるクライアントホストのアクセス制御

接続が TCP レベルで受け入れまたは拒否されるホストや IP アドレスは、TCP ラッパーを使用して制御できます。TCP ラップにより、クライアントホストのアクセスを制限できます。これにより、Directory Server への初期の TCP 接続に対する、ホストベースではない保護が可能になります。

Directory Server に対して TCP ラップを設定することはできますが、TCP ラップは、特にサービス拒否攻撃の際に、パフォーマンスを著しく低下させる可能性がありま

す。最良のパフォーマンスは、Directory Server の外部で保守されるホストベースのファイアウォールを使用するか、IP ポートのフィルタリングによって得られます。

▼ TCP ラップを使用可能にする

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 インスタンスパス内のどこかに `hosts.allow` ファイルまたは `hosts.deny` ファイルを作成します。

たとえば、このファイルを *instance-path*`/config` 内に作成します。作成するファイルの形式は、必ず `hosts_access`(4) に従うようにしてください。

2 アクセスファイルへのパスを設定します。

```
$ dsconf set-server-prop -h host -p port host-access-dir-path:path-to-file
```

次に例を示します。

```
$ dsconf set-server-prop -h host -p port host-access-dir-path:/local/ds1/config
"host-access-dir-path" property has been set to "/local/ds1/config".
The "/local/ds1/config" directory on host1 must contain valid hosts.allow
and/or hosts.deny files.
Directory Server must be restarted for changes to take effect.
```

▼ TCP ラップを使用不可にする

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- ホストアクセスパスを `""` に設定します。

```
$ dsconf set-server-prop -h host -p port host-access-dir-path:""
```

◆ ◆ ◆ 第 7 章

Directory Server のパスワードポリシー

ユーザーが Directory Server に接続すると、認証されます。ディレクトリは、認証時に設定された識別情報に応じて、ユーザーに対してアクセス権限を与え、リソースを制限することができます。この章でのアカウントとは、大まかにはユーザーエントリを指しています。アカウントは、ユーザーがディレクトリに対して実行する操作の権限も示します。ここでのパスワードポリシーの説明では、すべてのアカウントがユーザーエントリとパスワードに関連付けられています。

さらに、この章では、パスワードポリシーの一面であるアカウントのアクティブ化についても説明します。ディレクトリ管理者は、パスワードポリシーと関係なく、アカウントを直接ロックおよびロック解除できます。

この章では、認証方法については取り上げていません。SASL GSSAPI やクライアント SSL 証明書ベースの認証などの認証方法では、パスワードを使用しないこともあります。この章のパスワードポリシーに関する情報はそのような認証方法には適用しません。認証メカニズムの設定については、第 5 章を参照してください。

また、この章では、Directory Server 6.1 と以前のバージョンの Directory Server とのパスワードポリシーの互換性についても取り上げていません。 Directory Server 6.1 のインスタンスを作成すると、以前のバージョンからのアップグレードを容易にするために、パスワードポリシー実装はデフォルトで Directory Server 5 互換モードに設定されます。この章で説明するパスワードポリシー機能を十分に活用するには、パスワードポリシー互換モードを変更する必要があります。パスワードポリシー互換モードの設定の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Migration Guide』の「Password Policy Compatibility」を参照してください。

この章の内容は次のとおりです。

パスワードポリシーとワークシート

この節ではパスワードポリシー設定について説明し、要件に合うパスワードポリシーの定義に役立つワークシートを提供します。

注 – デフォルトのパスワードポリシーを使用するには、184 ページの「デフォルトのパスワードポリシーの管理」を参照してください。

パスワードポリシー設定

Directory Server でパスワードポリシーを指定する場合、オブジェクトクラス `pwdPolicy`(5dsoc) を含むエントリを変更するか作成します。

特定のタイプのユーザーのパスワードポリシーを定義する場合は、次のことを考慮する必要があります。

- 侵入者がパスワードをクラックしようとしているように見える場合にアカウントをロックアウトする方法。

  詳細については、179 ページの「アカウントロックアウトのポリシー」を参照してください。
- パスワードの変更方法。

  詳細については、180 ページの「パスワード変更のポリシー」を参照してください。
- 許可するパスワードの値。

  詳細については、180 ページの「パスワードコンテンツのポリシー」を参照してください。
- パスワードの有効期限の処理方法。

  詳細については、181 ページの「パスワード有効期限のポリシー」を参照してください。
- 最後に認証に成功した時間をサーバーが記録するかどうか。

  182 ページの「最後の認証時間の追跡のポリシー」を参照してください。

この章のあとの節で、パスワードポリシーのこれらの点の処理方法を説明します。実装を計画している各パスワードポリシーを明確にするには、182 ページの「パスワードポリシーを定義するためのワークシート」を使用します。

アカウントロックアウトのポリシー

この節では、アカウントロックアウトを制御するポリシー属性について説明します。

Directory Server のアカウントとは、大まかにはユーザーのエントリとそのユーザーがディレクトリに対して操作を実行するために必要な権限を指します。各アカウントはバインド DN とユーザーパスワードに関連付けられています。侵入者がパスワードをクラックしようとしているように見える場合、Directory Server でアカウントをロックする必要があります。ロックにより、侵入者はそのアカウントを使用してバインドできなくなります。さらに、ロックにより、侵入者が攻撃を続けることができなくなります。

管理者は、ロールを共有するすべてのユーザーのアカウントを手動で非アクティブ化することもできます。手順については、201 ページの「アカウントの手動でのロック」を参照してください。さらに、パスワードポリシーを指定する際に重要なことは、管理者の介入なしに、Directory Server が自動的にアカウントをロックできる環境にすることです。

まず、バインドの失敗が著しく多く発生する場合は、Directory Server で `pwdLockout`(5dsat) を使用して、自動的にアカウントがロックされるように指定する必要があります。Directory Server は連続して失敗したアカウントへのバインド試行を追跡します。`pwdMaxFailure`(5dsat) を使用して、何回連続して失敗したら Directory Server がアカウントをロックするかを指定します。

Directory Server はパスワードポリシーに従って、厳密にアカウントをロックします。操作は完全に機械的です。アカウントは、侵入者がアカウントに対して攻撃を仕掛けようとしているためでなく、ユーザーが誤ったパスワードを入力したことによりロックされることもあります。`pwdFailureCountInterval`(5dsat) を使用して、失敗した試行の間隔がどれだけ空いたら Directory Server がそれまでの失敗した試行の記録を消去するかを指定できます。`pwdLockoutDuration`(5dsat) を使用してDirectory Server がアカウントのロックを自動的に解除するまで、ロックアウトを継続する期間を指定します。管理者は、悪意なく正当なミスをしたユーザーのアカウントのロックを解除するために介入する必要はありません。

ユーザーデータがレプリケーショントポロジにレプリケートされる場合、ロックアウト属性は、ほかのエントリデータとともにレプリケートされます。`pwdIsLockoutPrioritized`(5dsat) 属性のデフォルト設定は `TRUE` です。この設定では、ロックアウト属性の更新は高い優先順位でレプリケートされます。したがって、ユーザーが、ロックアウトされるまでにレプリカの 1 つへのバインド試行を連続して失敗できるのは、`pwdMaxFailure` で設定された回数までです。ユーザーのバインド試行の失敗がほかのレプリカに対してもカウントされることで、全体として試行できる回数はさらに少なくなる可能性があります。ユーザーが、レプリケートされたトポロジ全体でロックアウトされるまで、`pwdMaxFailure` で設定された回数だけ試行できるようにする方法の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「グローバルアカウントロックアウトを使用した認証の防止」を参照してください。

パスワード変更のポリシー

この節では、パスワードの変更を制御するポリシーについて説明します。

多くの配備で、Directory Server はアイデンティティーデータのリポジトリになります。pwdAllowUserChange(5dsat) を有効にし、ユーザーが自身のパスワードを変更できるようにすることをお勧めします。これにより、管理者が各ユーザーのパスワードを変更する必要がなくなります。

ユーザーが自分のパスワードを変更できるようにしたら、次に必要な作業として、ユーザーがどのように自分のパスワードを変更できるかを制御することが考えられます。pwdSafeModify(5dsat) を使用して、パスワード変更の前に、ユーザーに既存のパスワードを正しく入力することを求めるように設定できます。パスワードの変更方法の例については、196 ページの「pwdSafeModify が TRUE の場合のコマンド行からのパスワードの変更」を参照してください。pwdInHistory(5dsat) を使用して、Directory Server が記憶するパスワード数を指定して、ユーザーがパスワードを再利用できないようにすることができます。さらに、pwdMinAge(5dsat) を設定して、ユーザーが著しく頻繁にパスワードを変更できないようにすることもできます。

管理者でも、または管理するアプリケーションでも、たいていの場合、ユーザーエントリをディレクトリに作成します。ユーザーが初めて新しいアカウントにバインドしたときに変更するユーザーパスワード値を割り当てることができます。さらに、ユーザーパスワードのリセットが必要になる場合もあります。リセット後、ユーザーは次にアカウントを使用するときにパスワードを変更する必要があります。Directory Server には、特定の属性 pwdMustChange(5dsat) があります。これを使用して、他のユーザーによってパスワード値がリセットされた後に、ユーザーがパスワードを変更する必要があるかどうかを指定することができます。

さらに、passwordRootdnMayBypassModsChecks(5dsat) を設定して、ディレクトリ管理者がパスワードを変更する場合に、ポリシーが適用されないように指定することもできます。

パスワードコンテンツのポリシー

この節では、パスワードコンテンツを制御するポリシー属性について説明します。

ディレクトリ検索で一般にパスワード値は返されませんが、攻撃者はディレクトリデータベースへのアクセス権を取得する可能性があります。そのため、パスワード値は一般に、passwordStorageScheme(5dsat) で指定した、サポートされるハッシュ形式で保存します。

pwdMinLength(5dsat) を設定して、パスワードを指定した文字数以上にすることを強制できます。

さらに、pwdCheckQuality(5dsat) を設定して、パスワードが最低限のパスワード品質の定義を満たしているかをチェックすることもできます。チェックを実行すると、Directory Server はパスワードが最小文字数以上であることをチェックします。さら

に、サーバーはパスワードに cn、 givenName、mail、ou、sn、または uid 属性の値が含まれていないことをチェックします。また、強力なパスワードチェックプラグインが有効にされている場合、Directory Server はパスワードにプラグインが使用する辞書ファイルの文字列が含まれていないことをチェックします。さらに、パスワードにさまざまなタイプの文字が適切に混在して含まれていることもチェックします。

強力なパスワードチェックを有効にするには、dsconf set-server-prop コマンドを使用します。pwd-strong-check-enabled プロパティーを使用して、プラグインを有効にし、サーバーを再起動して、変更を有効にします。パスワードに含める必要がある文字セットを指定するには、pwd-strong-check-require-charset プロパティーを使用します。pwd-strong-check-require-charset プロパティーは次の値のマスクを使用します。

lower	新しいパスワードには、小文字を含める必要があります。
upper	新しいパスワードには、大文字を含める必要があります。
digit	新しいパスワードには、数字を含める必要があります。
special	新しいパスワードには、特殊文字を含める必要があります。
any-two	新しいパスワードには、上記の 2 つ以上の文字セットから、それぞれ 1 文字以上含める必要があります。
any-three	新しいパスワードには、上記の 3 つ以上の文字セットから、それぞれ 1 文字以上含める必要があります。

pwd-strong-check-require-charset プロパティーのデフォルトの設定は lower && upper && digit && special です。

パスワード有効期限のポリシー

この節では、パスワードの有効期限を制御するポリシー属性について説明します。

ユーザーがパスワードを定期的に変更するように、Directory Server で、パスワードが特定の経過時間に達すると、有効期限が切れるように設定できます。このためには、pwdMaxAge(5dsat) を設定します。

パスワードの有効期限が切れそうであることをユーザーに通知する必要があります。バインドに使用するパスワードの有効期限が切れそうであるという警告を返すように、Directory Server を設定できます。pwdExpireWarning(5dsat) を使用して、有効期限のどれくらい前からクライアントがバインドしたときに警告を返すかを定義します。クライアントアプリケーションが警告を受け取ることに注意してください。ユーザーが直接警告を受け取るわけではありません。クライアントアプリケーションは、パスワードの有効期限が切れそうであるという警告を受け取ったら、エンドユーザーに通知する必要があります。

ユーザーが有効期限切れのパスワードで 1 回以上バインドを試みることを許可するには、pwdGraceAuthNLimit(5dsat) を設定します。ユーザーは、期限内にパスワードの変更に失敗した場合でも、パスワードを変更するためにバインドできます。この猶予認証でログインした場合、パスワードの有効期限が切れていないときと同様にユーザーはすべての操作を実行できます。

Directory Server は、エントリのパスワードが変更されるたびにオペレーショナル属性 pwdChangedTime(5dsat) を更新します。結果として、パスワードの有効期限を有効にするために待機している場合、パスワードの有効期限を有効にすると、すでに古くなったユーザーパスワードはすぐに期限切れとなります。この動作が意図と反する場合は、ユーザーに猶予認証でのログインを許可し、すぐにパスワードを変更させるようにしてください。

最後の認証時間の追跡のポリシー

この節では、パスワードポリシー属性 pwdKeepLastAuthTime(5dsat) の使用について説明します。

pwdKeepLastAuthTime を設定すると、Directory Server はユーザーが認証するたびに、最後にバインドに成功した時間を追跡します。時間は、ユーザーのエントリの pwdLastAuthTime(5dsat) オペレーショナル属性に記録されます。

この操作によって、バインド操作が成功するたびに更新が追加されるため、pwdKeepLastAuthTime 機能はデフォルトで無効にされています。配備でこの機能を使用する場合は、明示的に有効にする必要があります。

パスワードポリシーを定義するためのワークシート

ワークシートは、コマンド行インタフェースまたは Directory Service Control Center (DSCC) を使用して実装するパスワードポリシーの定義に役立つように設計されています。パスワードポリシーごとに 1 つのワークシートを使用します。

パスワードポリシーエントリの DN を記録したら、各ポリシー領域の属性の設定についての決定を記録します。それらの設定の理由も記録します。

パスワードポリシーワークシート			
パスワードポリシーエントリ識別名			
dn: cn=			

ポリシー領域	属性	ここに設定を記入してください	ここに設定の理由を記入してください
アカウントのロックアウト	pwdFailureCountInterval(5dsat)		
	pwdIsLockoutPrioritized(5dsat)		
	pwdLockout(5dsat)		
	pwdLockoutDuration(5dsat)		
	pwdMaxFailure(5dsat)		
パスワードの変更	passwordRootdnMayBypassModsChecks(5dsat)		
	pwdAllowUserChange(5dsat)		
	pwdInHistory(5dsat)		
	pwdMinAge(5dsat)		
	pwdMustChange(5dsat)		
	pwdSafeModify(5dsat)		
パスワードの内容	passwordStorageScheme(5dsat)		
	pwdCheckQuality(5dsat)		
	pwdMinLength(5dsat)		
パスワードの有効期限	pwdExpireWarning(5dsat)		
	pwdGraceAuthNLimit(5dsat)		
	pwdMaxAge(5dsat)		
最後の認証時間の追跡	pwdKeepLastAuthTime(5dsat)		

注 – pwdCheckQuality 属性を 2 に設定すると、サーバーは追加のチェックを実行できます。パスワードチェックプラグインも有効にすると、新しいパスワードの値をチェックする際に、プラグインの設定内容も考慮されます。

デフォルトのパスワードポリシーの管理

デフォルトのパスワードポリシーは、専用のポリシーが定義されていない、ディレクトリインスタンスのすべてのユーザーに適用されます。ただし、デフォルトのパスワードポリシーはディレクトリマネージャーには適用されません。ポリシーの範囲の詳細については、187 ページの「どのパスワードポリシーを適用するか」を参照してください。

デフォルトのパスワードポリシーは、`dsconf` コマンドを使用して設定できるポリシーの 1 つです。デフォルトのパスワードポリシーを表示するには、`cn=Password Policy,cn=config`を読み取ります。

この節では、各ポリシー領域のポリシー属性と関連の `dsconf` サーバープロパティーについて説明します。さらに、デフォルトのパスワードポリシー設定を表示して変更する方法についても説明します。

パスワードポリシー属性と dsconf サーバープロパティーの相関関係

次の表に、各パスワードポリシー領域のパスワードポリシー属性と関連する `dsconf` サーバープロパティーを示します。

ポリシー領域	ポリシー属性	dsconf サーバープロパティー
アカウントのロックアウト	pwdFailureCountInterval	pwd-failure-count-interval
	pwdLockout	pwd-lockout-enabled
	pwdLockoutDuration	pwd-lockout-duration
	pwdMaxFailure	pwd-max-failure-count
パスワードの変更	passwordRootdnMayBypassModsChecks	pwd-root-dn-bypass-enabled
	pwdAllowUserChange	pwd-user-change-enabled
	pwdInHistory	pwd-max-history-count
	pwdMinAge	pwd-min-age
	pwdMustChange	pwd-must-change-enabled
	pwdSafeModify	pwd-safe-modify-enabled

ポリシー領域	ポリシー属性	dsconf サーバープロパティー
パスワードの内容	pwdCheckQuality	pwd-check-enabled、pwd-accept-hashed-password-enabled、pwd-strong-check-dictionary-path、pwd-strong-check-enabled 、pwd-strong-check-require-charset
	pwdMinLength	pwd-min-length
	passwordStorageScheme	pwd-storage-scheme
パスワードの有効期限	pwdExpireWarning	pwd-expire-warning-delay
	pwdGraceAuthNLimit	pwd-grace-login-limit
	pwdMaxAge	pwd-max-age
最後の認証時間の追跡	pwdKeepLastAuthTime	pwd-keep-last-auth-time-enabled

注 – pwdCheckQuality に関連するプロパティーはパスワードチェックプラグインを設定します。そのため、サーバーインスタンス全体に 5 つのプロパティーが適用されます。さらに、この 5 つのプロパティーは pwdCheckQuality: 2 であるほかのパスワードポリシーにも適用されます。

▼ デフォルトのパスワードポリシー設定を表示する

dsconf コマンドを使用して、デフォルトのパスワードポリシー設定を表示できます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- デフォルトのパスワードポリシー設定を読み取ります。

```
$ dsconf get-server-prop -h host -p port | grep ^pwd-
pwd-accept-hashed-pwd-enabled      :  N/A
pwd-check-enabled                  :  off
pwd-compat-mode                    :  DS5-compatible-mode
pwd-expire-no-warning-enabled      :  on
pwd-expire-warning-delay           :  1d
pwd-failure-count-interval         :  10m
pwd-grace-login-limit              :  disabled
pwd-keep-last-auth-time-enabled    :  off
pwd-lockout-duration               :  1h
```

```
pwd-lockout-enabled                 :  off
pwd-lockout-repl-priority-enabled   :  on
pwd-max-age                         :  disabled
pwd-max-failure-count               :  3
pwd-max-history-count               :  disabled
pwd-min-age                         :  disabled
pwd-min-length                      :  6
pwd-mod-gen-length                  :  6
pwd-must-change-enabled             :  off
pwd-root-dn-bypass-enabled          :  off
pwd-safe-modify-enabled             :  off
pwd-storage-scheme                  :  SSHA
pwd-strong-check-dictionary-path    :  /local/ds6/plugins/words-english-big.txt
pwd-strong-check-enabled            :  off
pwd-strong-check-require-charset    :  lower
pwd-strong-check-require-charset    :  upper
pwd-strong-check-require-charset    :  digit
pwd-strong-check-require-charset    :  special
pwd-supported-storage-scheme        :  CRYPT
pwd-supported-storage-scheme        :  SHA
pwd-supported-storage-scheme        :  SSHA
pwd-supported-storage-scheme        :  NS-MTA-MD5
pwd-supported-storage-scheme        :  CLEAR
pwd-user-change-enabled             :  on
```

▼ デフォルトのパスワードポリシー設定を変更する

デフォルトのパスワードポリシーを変更するには、`dsconf` コマンドを使用して、サーバーのプロパティーを設定します。

注 – この手順を実行する前に、182 ページの「パスワードポリシーを定義するためのワークシート」を参照して、記入してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 ワークシートの設定を元にして、`dsconf` コマンドでどのプロパティーを設定すればよいかを確認します。

2 `dsconf set-server-prop` コマンドを使用して、デフォルトのパスワードポリシープロパティーを適切に変更します。

たとえば、次のコマンドを使用すると、ディレクトリマネージャーがパスワードを変更するときに、デフォルトのポリシーに違反してもよいことになります。

```
$ dsconf set-server-prop -h host -p port pwd-root-dn-bypass-enabled:on
```

特別なパスワードポリシーの管理

特別なパスワードポリシーは `pwdPolicy`(5dsoc) エントリに定義します。ポリシーはディレクトリツリーの任意の場所に定義できますが、一般にポリシーで管理するアカウントでレプリケートされるサブツリーに定義します。ポリシーは `cn=` *policy name*、*subtree* の形式の DN を持ちます。

パスワードポリシーを定義したら、目的のユーザーエントリに `pwdPolicySubentry`(5dsat) 属性を設定して、パスワードポリシーを割り当てます。

ここで説明する内容は次のとおりです。

どのパスワードポリシーを適用するか

Directory Server では、複数のパスワードポリシーを設定できます。この節では、デフォルトのパスワードポリシーと特別なパスワードポリシーについて説明します。さらに、この節では、特定のアカウントに複数のパスワードポリシーを適用できる場合に、どのポリシーを強制するかについても説明します。

初めて Directory Server インスタンスを作成すると、そのインスタンスにはデフォルトのパスワードポリシーが適用されます。デフォルトのパスワードポリシーは、設定エントリ `cn=PasswordPolicy,cn=config` に示されています。デフォルトのパスワードポリシーはディレクトリマネージャーを除くディレクトリのすべてのアカウントに適用されます。

すべての Directory Server パスワードポリシーと同様に、`cn=PasswordPolicy,cn=config` はオブジェクトクラス `pwdPolicy`(5dsoc) とオブジェクトクラス `sunPwdPolicy`(5dsoc) を持ちます。

注 – Directory Server インスタンスを作成すると、パスワードポリシー属性は Directory Server 5 互換モードのままであるため、以前のバージョンからのアップグレードが簡単です。Directory Server 5 互換モードでは、Directory Server はオブジェクトクラス passwordPolicy(5dsoc) を持つパスワードポリシーエントリも処理します。

アップグレードが完了すると、『Sun Java System Directory Server Enterprise Edition 6.1 Migration Guide』で説明するように、新しいパスワードポリシーの機能をすべて使用できます。管理上の移動は、ディレクトリアプリケーションには認識されません。

この章では、新しいパスワードポリシー機能を使用したパスワードポリシー設定について説明します。

デフォルトのパスワードポリシーを変更して、デフォルトの設定を上書きできます。dsconf(1M) コマンドを使用して、デフォルトのパスワードポリシーにサーバーのプロパティーを設定できます。それらのサーバープロパティー名は一般に pwd- プレフィックスから始まります。それらのプロパティーの設定を変更する場合、インスタンスのデフォルトのパスワードポリシーを上書きします。ただし、レプリケーションでは変更がレプリカにコピーされません。デフォルトのパスワードポリシーの変更は、ディレクトリデータではなく、インスタンスの設定に含まれます。

デフォルトのパスワードポリシーを設定するほかに、特別なパスワードポリシーも設定できます。特別なパスワードポリシーは、ディレクトリツリーのエントリによって定義します。特別なパスワードポリシーエントリは、デフォルトのパスワードポリシーと同じオブジェクトクラス pwdPolicy(5dsoc) を持つため、同じポリシー属性を持ちます。特別なパスワードポリシーは、正規のディレクトリエントリであるため、通常のディレクトリエントリと同じ方法でポリシーエントリがレプリケートされます。

ユーザーエントリは、オペレーショナル属性 pwdPolicySubentry(5dsat) の値によって特別なパスワードポリシーを参照します。ユーザーエントリによって参照した場合、特別なパスワードポリシーはインスタンスのデフォルトのパスワードポリシーを上書きします。多くの配備で、ユーザーロールを割り当てます。pwdPolicySubentry 値を設定して、サービスクラス (CoS) と連携して、ユーザーアカウントに適用するパスワードポリシーを決定するようにロールを設定できます。ロールによってパスワードポリシーセットを上書きするには、そのユーザーのエントリディレクトリの pwdPolicySubentry 値を変更します。

この節を要約すると、最初にデフォルトのパスワードポリシーが適用されます。デフォルトのパスワードポリシーを変更して、デフォルトを上書きできます。次に、特別なパスワードポリシーエントリを作成して、デフォルトのパスワードポリシーを上書きできます。ロールと CoS によってパスワードポリシーを割り当てる場合に、各エントリにパスワードポリシーを指定して、CoS によって割り当てられたポリシーを上書きできます。

▼ パスワードポリシーを作成する

他のディレクトリエントリの作成や変更と同じ方法で、特別なパスワードポリシーを作成、変更できます。次の手順では、テキストエディタを使用して、LDIF にパスワードポリシーエントリを書き込む方法を示します。次に `-a` オプションを使用して、`ldapmodify` コマンドを実行し、ディレクトリにパスワードポリシーエントリを追加します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　ほかに指示がない限り、ここに示すデータの例は `Example.ldif` から抜粋したものです。

1. 作成するポリシーについて、パスワードポリシーワークシートを完成させます。
   サンプルについては、182 ページの「パスワードポリシーを定義するためのワークシート」を参照してください。

2. ワークシートに基づいて、パスワードポリシーエントリを **LDIF** に書き込みます。
   たとえば、次のポリシーエントリは、Example.com の臨時従業員のパスワードポリシーを指定します。この従業員のサブツリーのルートは `dc=example,dc=com` です。

```
dn: cn=TempPolicy,dc=example,dc=com
objectClass: top
objectClass: pwdPolicy
objectClass: sunPwdPolicy
objectClass: LDAPsubentry
cn: TempPolicy
pwdAttribute: userPassword
pwdCheckQuality: 2
pwdLockout: TRUE
pwdLockoutDuration: 300
pwdMaxFailure: 3
pwdMustChange: TRUE
```

   デフォルトのパスワードポリシー設定に加えて、ここに示すポリシーは追加の動作を指定します。パスワード品質チェックを実行します。3 回連続してバインドが失敗すると、アカウントは 5 分間 (300 秒) ロックされます。パスワードのリセット後に、パスワードを変更する必要があります。ポリシーをユーザーアカウントに割り当てると、ここに明示的に指定した設定で、デフォルトのパスワードポリシーが上書きされます。

3 ディレクトリにパスワードポリシーエントリを追加します。

たとえば、次のコマンドは Example.com の臨時従業員のパスワードポリシーを dc=example,dc=com の下に追加します。パスワードポリシーは pwp.ldif というファイルに保存されています。

```
$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f pwp.ldif
Enter bind password:
adding new entry cn=TempPolicy,dc=example,dc=com

$ ldapsearch -D uid=kvaughan,ou=people,dc=example,dc=com -w --b dc=example,dc=com \
"(&(objectclass=ldapsubentry)(cn=temppolicy))"
Enter bind password:
version: 1
dn: cn=TempPolicy,dc=example,dc=com
objectClass: top
objectClass: pwdPolicy
objectClass: LDAPsubentry
cn: TempPolicy
pwdCheckQuality: 2
pwdLockout: TRUE
pwdLockoutDuration: 300
pwdMaxFailure: 3
pwdMustChange: TRUE
$
```

Example.ldif に示すように、kvaughan は dc=example,dc=com エントリを変更するアクセス権を持つ人事マネージャーです。Example.ldif に示すように、Vaughan のバインドパスワードは bribery です。

参照 定義したポリシーによって管理されるユーザーアカウントを定義するには、190 ページの「各アカウントにパスワードポリシーを割り当てる」または 191 ページの「ロールと CoS を使用してパスワードポリシーを割り当てる」を参照してください。

▼ 各アカウントにパスワードポリシーを割り当てる

次の手順で、既存のパスワードポリシーを 1 つのユーザーアカウントに割り当てます。

注 – この手順を実行するには、特別なパスワードポリシーを作成している必要があります。189 ページの「パスワードポリシーを作成する」を参照してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

ここに示す例は、ほかに指示がない限り、Example.ldif から抜粋したものです。

- ユーザーエントリの pwdPolicySubentry 属性の値にパスワードポリシー **DN** を追加します。

  たとえば、次のコマンドは、189 ページの「パスワードポリシーを作成する」 で定義しているパスワードポリシーを David Miller のエントリに割り当てます。David Miller の DN は uid=dmiller,ou=people,dc=example,dc=com です。

```
$ cat pwp.ldif
dn: uid=dmiller,ou=people,dc=example,dc=com
changetype: modify
add: pwdPolicySubentry
pwdPolicySubentry: cn=TempPolicy,dc=example,dc=com

$ ldapmodify -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f pwp.ldif
Enter bind password:
modifying entry uid=dmiller,ou=people,dc=example,dc=com

$ ldapsearch -D uid=kvaughan,ou=people,dc=example,dc=com -w - -b dc=example,dc=com \
"(uid=dmiller)" pwdPolicySubentry
Enter bind password:
version: 1
dn: uid=dmiller, ou=People, dc=example,dc=com
pwdPolicySubentry: cn=TempPolicy,dc=example,dc=com
$
```

  Example.ldif に示すように、kvaughan は人事マネージャーで、dc=example,dc=com エントリを変更するアクセス権を持ちます。Example.ldif に示すように、Vaughan のバインドパスワードは bribery です。

▼ ロールと CoS を使用してパスワードポリシーを割り当てる

次の手順では、ロールとサービスクラス (CoS) を適用して、既存の特別なパスワードポリシーをユーザーのセットに割り当てます。ロールと CoS の詳細については、第 9 章を参照してください。

注 – この手順を実行するには、特別なパスワードポリシーを作成している必要があります。189 ページの「パスワードポリシーを作成する」を参照してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

ここに示す例は、ほかに指示がない限り、Example.ldif から抜粋したものです。

1 パスワードポリシーによって管理されるエントリのロールを作成します。

たとえば、次のコマンドはExample.comの臨時従業員のフィルタを適用されたロールを作成します。

```
$ cat tmp.ldif
dn: cn=TempFilter,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: TempFilter
nsRoleFilter: (&(objectclass=person)(status=contractor))
description: filtered role for temporary employees

$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f tmp.ldif
Enter bind password:
modifying entry cn=TempFilter,ou=people,dc=example,dc=com

$
```

Example.ldifに示すように、kvaughanは人事マネージャーで、dc=example,dc=comエントリを変更するアクセス権を持ちます。Example.ldifに示すように、Vaughanのバインドパスワードはbriberyです。

2 パスワードポリシーエントリの**DN**を生成するサービスクラスを作成します。

このDNは、作成したロールを持つユーザーのpwdPolicySubentry属性の値です。

たとえば、次のコマンドは、Example.comの臨時従業員のフィルタを適用したロールを作成します。コマンドはロールを持つユーザーに
cn=TempPolicy,dc=example,dc=comを割り当てます。

```
$ cat cos.ldif
dn: cn=PolTempl,dc=example,dc=com
objectclass: top
objectclass: nsContainer

dn: cn="cn=TempFilter,ou=people,dc=example,dc=com",
 cn=PolTempl,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: LDAPsubentry
objectclass: costemplate
cosPriority: 1
pwdPolicySubentry: cn=TempPolicy,dc=example,dc=com

dn: cn=PolCoS,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
```

```
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDN: cn=PolTempl,dc=example,dc=com
cosSpecifier: nsRole
cosAttribute: pwdPolicySubentry operational

$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f cos.ldif
Enter bind password:
modifying entry cn=TempFilter,ou=people,dc=example,dc=com

$
```

これにより、ステータスが `contractor` であるユーザーは、`cn=TempPolicy,dc=example,dc=com` パスワードポリシーの対象となります。

▼ 初回ログインパスワードポリシーを設定する

多くの配備で、新しいアカウントに適用するパスワードポリシーは、既存のアカウントに適用するパスワードポリシーと異なります。この節では、初回ログインパスワードポリシーについて説明します。このポリシーでは、ユーザーが新しく作成されたアカウントを 3 日間使用でき、アカウントがロックされる前に、自身の新しいパスワードを設定するようにします。このポリシーは、パスワードがリセットされたユーザーについても同様に機能するように設計されています。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 新しく作成されたアカウント用に特別なパスワードポリシーを作成します。

たとえば、有効期限を 3 日間、つまり 259,200 秒に設定するパスワードポリシーエントリを追加します。さらに、このパスワードポリシーでは、`pwdMustChange`(5dsat) に `TRUE` を設定し、ユーザーが初めてバインドしたときに、自身のパスワードを変更する必要があることを示します。

```
$ cat firstLogin.ldif
dn: cn=First Login,dc=example,dc=com
objectClass: top
objectClass: LDAPsubentry
objectClass: pwdPolicy
objectClass: sunPwdPolicy
cn: First Login
passwordStorageScheme: SSHA
pwdAttribute: userPassword
pwdInHistory: 0
pwdExpireWarning: 86400
pwdLockout: TRUE
pwdMinLength: 6
pwdMaxFailure: 3
```

```
pwdMaxAge: 259200
pwdFailureCountInterval: 600
pwdAllowUserChange: TRUE
pwdLockoutDuration: 3600
pwdMinAge: 0
pwdCheckQuality: 2
pwdMustChange: TRUE

$ ldapmodify -a -D cn=admin,cn=Administrators,cn=config -w - -f firstLogin.ldif
Enter bind password:
adding new entry cn=First Login,dc=example,dc=com

$
```

2 新しく作成されたすべてのアカウントを含むロールを作成します。

このロールの作成では、新しく作成したアカウントを既存のアカウントと区別するいくつかの方法を設定します。

a. pwdReset**(5dsat)** 属性が TRUE に設定されているアカウントが新しいアカウントであると定義します。

ユーザーのパスワードがパスワード管理者などの別のユーザーによって変更された場合、 pwdReset が TRUE に設定されます。

b. 新しいアカウントを識別するロールを作成します。

たとえば、次のコマンドは、パスワードがリセットされたアカウントのロールを作成します。

```
$ cat newRole.ldif
dn: cn=First Login Role,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: First Login Role
nsRoleFilter: (pwdReset=TRUE)
description: Role to assign password policy for new and reset accounts

$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f newRole.ldif
Enter bind password:
adding new entry cn=First Login Role,ou=people,dc=example,dc=com

$
```

3 サービスクラスによって、新しく作成したアカウントのパスワードポリシーを割り当てます。

```
$ cat newCoS.ldif
dn: cn=First Login Template,dc=example,dc=com
objectClass: top
objectClass: nsContainer

dn: cn="cn=First Login Role,ou=people,dc=example,dc=com",
 cn=First Login Template,dc=example,dc=com
objectClass: top
objectClass: extensibleObject
objectClass: LDAPSubEntry
objectClass: CoSTemplate
cosPriority: 1
pwdPolicySubentry: cn=First Login,dc=example,dc=com

dn: cn=First Login CoS,dc=example,dc=com
objectClass: top
objectClass: LDAPSubEntry
objectClass: CoSSuperDefinition
objectClass: CoSClassicDefinition
cosTemplateDN: cn=First Login Template,dc=example,dc=com
cosSpecifier: nsRole
cosAttribute: pwdPolicySubentry operational

$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -f newCoS.ldif
Enter bind password:
adding new entry cn=First Login Template,dc=example,dc=com

adding new entry cn="cn=First Login Role,ou=people,dc=example,dc=com",
 cn=First Login Template,dc=example,dc=com

adding new entry cn=First Login CoS,dc=example,dc=com

$
```

例 7–1 パスワードポリシーの割り当ての確認

追加したロールに適合する新しいユーザーを追加します。ユーザーを追加してみて、新しいユーザーが新しいパスワードポリシーの対象となり、既存のユーザーが対象とならないことを確認します。

```
$ cat quentin.ldif
dn: uid=qcubbins,ou=People,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
```

```
objectclass: inetOrgPerson
uid: qcubbins
givenName: Quentin
sn: Cubbins
cn: Quentin Cubbins
mail: quentin.cubbins@example.com
userPassword: ch4ngeM3!
description: New account

$ ldapmodify -a -D uid=kvaughan,ou=people,dc=example,dc=com -w - -f quentin.ldif
Enter bind password:
adding new entry uid=qcubbins,ou=People,dc=example,dc=com

$ ldapsearch -D uid=kvaughan,ou=people,dc=example,dc=com -w - \
-b dc=example,dc=com uid=qcubbins nsrole pwdPolicySubentry
Enter bind password:
version: 1
dn: uid=qcubbins,ou=People,dc=example,dc=com
nsrole: cn=first login role,ou=people,dc=example,dc=com
pwdPolicySubentry: cn=First Login,dc=example,dc=com
$ ldapsearch -b dc=example,dc=com uid=bjensen nsrole pwdPolicySubentry
version: 1
dn: uid=bjensen, ou=People, dc=example,dc=com
$
```

Barbara Jensen の既存のアカウントにはデフォルトのパスワードポリシーが適用されます。しかし、Quentin Cubbins の新しいアカウントには、定義したパスワードポリシーが適用されます。

pwdSafeModify が TRUE の場合のコマンド行からのパスワードの変更

ユーザーのパスワードポリシーの pwdSafeModify が TRUE に設定されている場合、パスワードを変更するには、新しいパスワードと共に古いパスワードを指定する必要があります。コマンド dsconf set-server-prop pwd-safe-modify-enabled:on は、デフォルトのパスワードポリシーと同じ効果を持ちます。

ldappasswd(1) コマンドを使用してパスワードを変更できます。このコマンドは、安全なパスワードの変更をサポートしています。このコマンドは RFC 3062「LDAP Password Modify Extended Operation (http://www.ietf.org/rfc/rfc3062.txt)」を実装します。

ldapmodify(1) コマンドを使用して、パスワードを変更できます。その場合に ldapmodify コマンドに渡す LDIF は次のようにする必要があります。

```
dn: パスワードを変更するユーザーの DN
changetype: modify
delete: userPassword
userPassword: 古いパスワード
-
add: userPassword
userPassword: 新しいパスワード
```

さらに、LDAP パスワード変更拡張操作を使用することもできます。拡張操作のサポートの設定については、197 ページの「パスワード変更拡張操作によりパスワードをリセットする」で説明しています。

期限切れパスワードのリセット

パスワードポリシーによってパスワードの有効期限を適用している場合、期間内にパスワードを変更しないユーザーもいます。この節では、期限切れになったパスワードを変更する方法を示します。

注 – Directory Server は、エントリのパスワードが変更されるたびにオペレーショナル属性 pwdChangedTime(5dsat) を更新します。結果として、パスワードの有効期限を有効にするために待機している場合、パスワードの有効期限を有効にすると、すでに古くなったユーザーパスワードはすぐに期限切れとなります。この動作が意図と反する場合は、ユーザーに猶予認証でのログインを許可し、すぐにパスワードを変更させるようにしてください。

この節では、パスワード変更拡張操作によってパスワードをリセットする手順と、パスワードの期限が切れた場合に、猶予認証を許可する手順を説明します。

この節で説明するメカニズムは、管理者か、または実際のユーザーとディレクトリとのやり取りを処理するアプリケーションで使用することを意図しています。一般に、エンドユーザーに、意図したとおりにメカニズムを実際に使用させるのは、アプリケーションの役割です。

▼ パスワード変更拡張操作によりパスワードをリセットする

パスワードの有効期限が切れると、ユーザーアカウントがロックされます。パスワードをリセットすると、アカウントのロックが解除されます。パスワードは管理者などのほかのユーザーによってリセットできます。パスワードをリセットすると、Directory Server によってユーザーアカウントのロックが解除されます。Directory Server では、RFC 3062「LDAP Password Modify Extended Operation

(http://www.ietf.org/rfc/rfc3062.txt)」をサポートしています。拡張操作を使用して、ディレクトリ管理者またはディレクトリアプリケーションがパスワードのリセットによりアカウントのロックを解除することを許可することができます。

次の手順に示すように、パスワード変更拡張操作の使用を許可する場合は、注意してください。信頼する管理者とアプリケーションにのみアクセスを制限します。ネットワーク上でパスワードをテキストで送受信しないでください。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 パスワード管理者またはパスワード管理アプリケーションにユーザーアクセス権を与えます。

2 パスワード管理者がパスワード変更拡張操作にアクセスして使用できるようにします。

次のコマンドは、Password Managers ロールのメンバーが SSL で接続した場合にパスワード変更拡張操作を使用できるようにする ACI を設定します。

```
$ cat exop.ldif
dn: oid=1.3.6.1.4.1.4203.1.11.1,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid: 1.3.6.1.4.1.4203.1.11.1
cn: Password Modify Extended Operation
aci: (targetattr != "aci")(version 3.0;
 acl "Password Modify Extended Operation
 "; allow( read, search, compare, proxy ) (roledn = "
 ldap:///cn=Password Managers,dc=example,dc=com" and authmethod = "SSL");)

$ ldapmodify -a -D cn=admin,cn=Administrators,cn=config -w - -f exop.ldif
Enter bind password:
adding new entry oid=1.3.6.1.4.1.4203.1.11.1,cn=features,cn=config

$
```

cn=features,cn=config の下のエントリにより、パスワード変更拡張操作を使用する操作へのアクセスを管理します。

3 パスワード管理者にユーザーパスワードをリセットしてもらいます。

この手順は、ユーザーアカウントのロックを解除し、ldappasswd(1) コマンドで実行できます。

4 (省略可能) ユーザーがパスワードを変更する必要がある場合は、パスワード管理者にユーザーに通知してもらいます。

ユーザーエントリを管理するパスワードポリシーに pwdMustChange: TRUE が含まれる場合、リセット後にユーザーは自身のパスワードを変更する必要があります。

▼ パスワードの期限が切れた場合に猶予認証を許可する

次の手順では、ユーザーに猶予認証を与え、ユーザーが期限切れになったパスワードを変更できるようにする方法を説明します。

猶予認証は、パスワードポリシーの要求および応答コントロールを処理するアプリケーションによって管理することを意図しています。この手順では、アプリケーションでコントロールを使用する方法の単純な例を示します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. ユーザーがパスワードポリシーの要求および応答コントロールを使用するアプリケーションに対するアクセス権を持つことを確認します。

   アプリケーションでは、ユーザーが猶予認証を適切に処理していることを確認する必要があります。

2. アプリケーションでパスワードポリシーコントロールを使用できるようにします。

   次のコマンドは、`Password Managers` ロールのメンバーがパスワードポリシーコントロールを使用できるようにする ACI を設定します。

```
$ cat ctrl.ldif
dn: oid=1.3.6.1.4.1.42.2.27.8.5.1,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid: 1.3.6.1.4.1.42.2.27.8.5.1
cn: Password Policy Controls
aci: (targetattr != "aci")(version 3.0; acl "Password Policy Controls
 "; allow( read, search, compare, proxy ) roledn = "
 ldap:///cn=Password Managers,dc=example,dc=com";)

$ ldapmodify -a -D cn=admin,cn=Administrators,cn=config -w - -f ctrl.ldif
Enter bind password:
adding new entry oid=1.3.6.1.4.1.42.2.27.8.5.1,cn=features,cn=config

$
```

   `cn=features,cn=config` の下のエントリの目的は、パスワードポリシーの要求および応答コントロールを使用する操作へのアクセスを管理できるようにすることだけです。

3. パスワードポリシーの `pwdGraceAuthNLimit` 属性で、パスワードの期限が切れたあとに猶予認証によるログインを何回許可するかを設定します。

4 アプリケーション側では、ユーザーに対して猶予認証の許可されている回数内で期限の切れているパスワードをすみやかに変更しなければならないことを指示する必要があります。

アカウントプロパティーの設定

次の各節で、アカウントの検索制限、サイズ制限、時間制限、およびアイドルタイムアウトの設定方法について説明します。

▼ アカウントに対する検索制限を設定する

- ldapmodify コマンドを使用して、nsLookThroughLimit の値を設定します。
  次のコマンドで、Barbara Jensen の検索制限が解除されます。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=people,dc=example,dc=com
changetype: modify
add: nsLookThroughLimit
nsLookThroughLimit: -1
^D
modifying entry uid=bjensen,ou=people,dc=example,dc=com

^D
$
```

▼ アカウントに対するサイズ制限を設定する

- ldapmodify コマンドを使用して、nsSizeLimit の値を設定します。
  次のコマンドで、Barbara Jensen のサイズ制限が解除されます。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=people,dc=example,dc=com
changetype: modify
add: nsSizeLimit
nsSizeLimit: -1
^D
modifying entry uid=bjensen,ou=people,dc=example,dc=com

^D
$
```

▼ アカウントに対する時間制限を設定する

- `ldapmodify` コマンドを使用して、`nsTimeLimit` の値を設定します。
  次のコマンドで、Barbara Jensen の時間制限が解除されます。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=people,dc=example,dc=com
changetype: modify
add: nsTimeLimit
nsTimeLimit: -1
^D
modifying entry uid=bjensen,ou=people,dc=example,dc=com

^D
$
```

▼ アカウントに対するアイドルタイムアウトを設定する

- `ldapmodify` コマンドを使用して、`nsIdleTimeout` の値を設定します。
  次のコマンドで、Barbara Jensen のアイドルタイムアウトが 5 分 (300 秒) に設定されます。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bjensen,ou=people,dc=example,dc=com
changetype: modify
add: nsIdleTimeout
nsIdleTimeout: 300
^D
modifying entry uid=bjensen,ou=people,dc=example,dc=com

^D
$
```

アカウントの手動でのロック

Directory Server では、指定した回数のバインド試行が失敗した後に、アカウントを強制的にロックアウトするパスワードポリシーを設定できます。詳細については、179 ページの「アカウントロックアウトのポリシー」を参照してください。この節では、ディレクトリマネージャーが使用できる手動のアカウントロックおよびアクティブ化ツールについて説明します。

ディレクトリマネージャーは、ロックアウト期間タイマーを使用せずに、アカウントロックアウトを管理できます。ロックされたアカウントは、パスワードを手動でリセットするまで、ロックされます。ディレクトリマネージャーは、無期限で特定のアカウントを無効化することもできます。

この節では、アカウントのステータスを確認し、アカウントを無効化して、再びアクティブ化する方法を説明します。

▼ アカウントのステータスを確認する

次に示すように、アカウントのステータスを確認します。

注–ディレクトリマネージャーとしてバインドする必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- アカウントまたはロールのステータスを確認するには、`ns-accountstatus` コマンドを使用します。

  次のコマンドは、Barbara Jensen のアカウントのステータスを確認します。

```
$ ns-accountstatus -D "cn=Directory Manager" -j pwd.txt \
 -I uid=bjensen,ou=people,dc=example,dc=com
uid=bjensen,ou=people,dc=example,dc=com activated.
$
```

  詳細については、`ns-accountstatus`(1M) のマニュアルページを参照してください。

▼ アカウントを無効化する

次に示すように、アカウントまたはロールを無効化します。

注–ディレクトリマネージャーとしてバインドする必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- アカウントまたはロールを無効化するには、`ns-inactivate` コマンドを使用します。

  次のコマンドでは、Barbara Jensen のアカウントを無効化します。

```
$ ns-inactivate -D "cn=Directory Manager" -j pwd.txt \
-I uid=bjensen,ou=people,dc=example,dc=com
```

```
uid=bjensen,ou=people,dc=example,dc=com inactivated.
$
```

詳細については、ns-inactivate(1M) のマニュアルページを参照してください。

▼ アカウントをふたたびアクティブ化する

次に示すように、アカウントまたはロールのロックを解除します。

注 – ディレクトリマネージャーとしてバインドする必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- アカウントまたはロールをふたたびアクティブ化するには、ns-activate コマンドを使用します。

  次のコマンドは、Barbara Jensen のアカウントをふたたびアクティブ化します。

```
$ ns-activate -D "cn=Directory Manager" -j pwd.txt \
-I uid=bjensen,ou=people,dc=example,dc=com
uid=bjensen,ou=people,dc=example,dc=com activated.
$
```

  詳細については、ns-activate(1M) のマニュアルページを参照してください。

◆ ◆ ◆ 第 8 章

Directory Server のバックアップと復元

Directory Server で管理されるデータは、まとめてインポートされることがよくあります。Directory Server Enterprise Edition には、サフィックス全体のインポートとエクスポートを行うツールが用意されています。また、一度にすべてのサフィックスのバックアップを作成したり、すべてのデータをバックアップから復元したりするツールも用意されています。

バックアップや復元の操作を始める前に、現在の状況に合うバックアップおよび復元戦略を設計するようにしてください。さまざまなバックアップオプション、考慮事項、バックアップおよび復元戦略を計画する際のガイドラインの詳細については『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「バックアップと復元のポリシーの設計」を参照してください。

この章の内容は次のとおりです。

バイナリバックアップ

この節では、ディレクトリデータのバイナリバックアップを実行する方法について説明します。この節で説明するバイナリバックアップ手順以外にも、サフィックスをレプリケーショントポロジで初期化するためのバイナリコピーを作成できます。266 ページの「バイナリコピーを使用したレプリケートされたサフィックスの初期化」を参照してください。

ディレクトリデータのみのバックアップ

バイナリデータのバックアップでは、あとでデータベースファイルが破損したり削除されたりする場合に使用できる、ディレクトリデータのコピーを保存できます。この操作では、設定データはバックアップされません。障害回復のために Directory Server 全体をバックアップする場合は、219 ページの「障害からの回復」を参照してください。

注意 – バックアップの処理中には、サーバーを停止しないでください。

バックアップは、「削除の遅延」よりも頻繁に実行する必要があります。`nsDS5ReplicaPurgeDelay` 属性によって指定される削除の遅延は、更新履歴ログに対して内部削除操作を開始するまでの期間 (秒単位) です。デフォルトの削除の遅延は 604800 秒 (1 週間) です。更新履歴ログは、レプリケートが完了している、またはレプリケートが完了していない更新の記録を保持しています。

更新の頻度が削除の遅延より低い場合、バックアップを行う前に更新履歴ログの内容がクリアされてしまう可能性があります。この場合、バックアップからデータを復元しようとしても、バックアップ前にクリアされた更新記録は失われています。

この節で説明するバックアップ手順では、サーバーファイルのコピーがデフォルトで同じホスト上に格納されます。セキュリティー強化のために、このバックアップを別のマシンや別のファイルシステムにコピーして格納してください。

▼ ディレクトリデータをバックアップする

Directory Server を停止して `dsadm backup` コマンドを実行する必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ディレクトリデータをバックアップします。

```
$ dsadm backup instance-path archive-dir
```

次に例を示します。

```
$ dsadm backup /local/ds /local/tmp/20051205
```

注 – サーバーの稼働中は、dsconf backup コマンドを使用してディレクトリデータをバックアップできます。ただし、バックアップの実行中にディレクトリデータに変更を加えると、適切に復旧することが難しくなります。dsconf backup の使用時にこの問題を回避するには、レプリケーションのリフェラルを設定するか、サーバーを読み取り専用にします。

dsadm コマンドと dsconf コマンドの詳細については、dsadm(1M) および dsconf(1M) のマニュアルページを参照してください。

▼ dse.ldif ファイルをバックアップする

サーバーを復元する場合、dse.ldif 設定ファイルに、サーバーのバックアップ時と同じ設定情報を指定する必要があります。

- dse.ldif 設定ファイルをバックアップします。

  ```
  $ cp instance-path/config/dse.ldif archive-dir
  ```

  次の操作を実行すると、Directory Server は自動的に dse.ldif 設定ファイルのバックアップをディレクトリ *instance-path*/config に作成します。

  - Directory Server を起動すると、dse.ldif ファイルのバックアップが、dse.ldif.startOK というファイルに作成されます。
  - cn=config ブランチの内容を変更する場合は、サーバーが dse.ldif ファイルに変更を書き込む前に、ファイルが onfigディレクトリの dse.ldif.bak ファイルにバックアップされます。

ファイルシステムのバックアップ

この手順では、「凍結モード」機能を使用します。凍結モードでは、ディスク上のデータベース更新を停止できるため、ファイルシステムのスナップショットを安全に撮ることができます。堅固なバックアップを確実にするため、凍結モードを追加の手段として使用できます。

ファイルシステムのバックアップの進行中は、サーバーでディスク上のユーザーデータを書き込むことはできません。特定の時間内に更新が行われないことが確実な場合は、この時間内にバックアップを行います。更新が行われないことを保証できない場合は、バックアップを行う前にサーバーを凍結モードにします。

凍結モードがオンの場合、設定されたすべてのデータベースはオフラインになります。進行中の内部操作には、オフラインになるデータベースが通知されます。進行中の LDAP 操作は完了し、データベース環境はフラッシュします。ユーザーデータの検索を含むそれ以降の着信操作は、凍結モードがオフに設定されるまで拒否されます。ただし、凍結モードがオンの場合でも設定パラメータの検索はできます。

シングルサーバーのトポロジでは、凍結モードがオンの場合に受信される操作によって LDAP エラーが返されます。ログに記録されるエラーメッセージは、オフラインのデータベースに対する標準的なエラーです。レプリケートされたトポロジでは、リフェラルが返されます。凍結モードを適切に機能させるには、データベース上でほかのタスクを実行しないようにしてください。

▼ ファイルシステムをバックアップする

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1. (省略可能) サーバーを凍結モードにします。

   ```
   $ dsconf set-server-prop -h host -p port read-write-mode:frozen
   ```

2. ファイルシステムのタイプに合うツールを使って、ファイルシステムをバックアップします。

3. サーバーが凍結モードになっている場合は、再度サーバーを読み書き可能にします。

   ```
   $ dsconf set-server-prop -h host -p port read-write-mode:read-write
   ```

   サーバーがレプリケーションの更新を別のサーバーから受け取った場合は、凍結モードがオフになるとすぐにレプリケーションの更新が始まります。

LDIF へのバックアップ

LDIF へのバックアップにより、ディレクトリデータを書式付き LDF ファイルにバックアップできます。

LDIF へのエクスポート

LDIF でサフィックスの内容をエクスポートすることで、ディレクトリデータをバックアップできます。データのエクスポートは、次のような場合に便利です。

- サーバー上のデータのバックアップ。
- 他のディレクトリサーバーへのデータのコピー。
- 他のアプリケーションへのデータのエクスポート。
- ディレクトリトポロジ変更後のサフィックスの再生成。

エクスポート処理を実行しても、設定情報 (`cn=config`) はエクスポートされません。

注意 – エクスポート処理の実行中には、サーバーを停止しないでください。

▼ サフィックスを LDIF にエクスポートする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- サフィックスを LDIF ファイルにエクスポートするには、次のコマンドのいずれかを使用します。
  - サーバーがローカルにあり、停止している場合は、次のように入力します。

    ```
    $ dsadm export instance-path suffix-DN LDIF-file
    ```
  - サーバーがリモートにあり、実行中の場合は、次のように入力します。

    ```
    $ dsconf export -h host -p port suffix-DN LDIF-file
    ```

  次の例では、`dsconf export` を使用して、2 つのサフィックスを単一の LDIF ファイルにエクスポートします。

  ```
  $ dsconf export -h host1 -p 1389 ou=people,dc=example,dc=com \
   ou=contractors,dc=example,dc=com /local/ds/ldif/export123.ldif
  ```

  `dsadm export` コマンドと `dsconf export` コマンドでは、`--no-repl` オプションを指定して、レプリケーション情報がエクスポートされないように指定することもできます。デフォルトでは、レプリケートされたサフィックスはレプリケーション情報とともに LDIF ファイルにエクスポートされます。結果として作成される LDIF ファイルには、レプリケーションメカニズムで使用される属性サブタイプが含まれています。あとでこの LDIF ファイルをコンシューマサーバーにインポートして、コンシューマレプリカを初期化できます。この手順については、261 ページの「レプリカの初期化」を参照してください。

  これらのコマンドの詳細については、`dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

バイナリ復元

次に示す手順では、サフィックスをディレクトリに格納する方法を示します。サーバーは、206 ページの「ディレクトリデータのみのバックアップ」で説明した手順で、あらかじめバックアップされている必要があります。レプリケーションアグリーメントに関連するサフィックスを復元する前に、214 ページの「レプリケートされたサフィックスの復元」をお読みください。

注意 – 復元の処理中には、サーバーを停止しないでください。サーバーを復元すると既存のデータベースファイルが上書きされるため、バックアップのあとに行なった変更は失われます。

▼ サーバーを復元する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- サーバーを復元するには、次のコマンドのいずれかを使用します。
  - サーバーがローカルにあり、停止している場合は、次のように入力します。

    ```
    $ dsadm restore instance-path archive-dir
    ```

    たとえば、バックアップをバックアップディレクトリから復元するには、次のように入力します。

    ```
    $ dsadm restore /local/ds/ local/ds/bak/2006_07_01_11_34_00
    ```
  - サーバーがリモートにあり、実行中の場合は、次のように入力します。

    ```
    $ dsconf restore -h host -p port archive-dir
    ```

    たとえば、バックアップをバックアップディレクトリから復元するには、次のように入力します。

    ```
    $ dsconf restore -h host1 -p 1389 /local/ds/bak/2006_07_01_11_34_00
    ```

  これらのコマンドの詳細については、`dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

dse.ldif 設定ファイルの復元

Directory Server では、次のディレクトリ内に、`dse.ldif` ファイルのバックアップコピーが 2 つ作成されます。

instance-path`/config`

`dse.ldif.startOK` ファイルには、サーバー起動時に `dse.ldif` ファイルのコピーが記録されます。`dse.ldif.bak` ファイルには、`dse.ldif` ファイルに加えられた最新の変更内容のバックアップが含まれます。最新の変更内容を含むファイルを自分のディレクトリにコピーします。

▼ dse.ldif 設定ファイルを作成する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1. サーバーを停止します。

   ```
   $ dsadm stop instance-path
   ```

2. 設定ファイルが入っているディレクトリに移動します。

   ```
   $ cd instance-path/config
   ```

3. 有効であることがわかっているバックアップ設定ファイルで `dse.ldif` ファイルを上書きします。たとえば、次のように入力します。

   ```
   $ cp dse.ldif.startOK dse.ldif
   ```

4. 次のコマンドでサーバーを起動します。

   ```
   $ dsadm start instance-path
   ```

LDIF ファイルからのデータのインポート

次のような方法で、データを Directory Server サフィックスにインポートできます。

- LDIF ファイルからサフィックスを初期化します。この操作により、サフィックス内の現在のデータが削除され、LDIF ファイルの内容と置き換えられます。
- `ldapadd`、`ldapmodify`、または `ldapdelete` 操作をまとめて実行するには、LDIF ファイルを使用します。これにより、ディレクトリ内の任意のサフィックスについて、そのエントリをまとめて追加、変更、削除できます。

次の表は、サフィックスの初期化と、エントリの一括の追加、変更、削除の違いを示しています。

表 8–1 サフィックスの初期化とデータの一括インポートの比較

比較ドメイン	サフィックスの初期化	エントリの一括の追加、変更、削除
内容の上書き	内容の上書き	内容を上書きしない
LDAP 処理	追加のみ	追加、変更、削除
性能	高速	低速

表 8–1 サフィックスの初期化とデータの一括インポートの比較 (続き)

比較ドメイン	サフィックスの初期化	エントリの一括の追加、変更、削除
サーバーの障害への対応	不可 (障害が発生するとすべての変更内容は失われる)	ベストエフォート (障害発生時までの変更内容はそのまま残る)
LDIF ファイルの位置	クライアントまたはサーバーと同じマシン上	クライアントマシン上
設定情報のインポート (`cn=config`)	設定情報をインポートする	設定情報をインポートしない

コマンド (Commands)	サーバーがローカルにあり、停止している場合:
	`dsadm import`
	サーバーがリモートにあり、実行中の場合:
	`dsconf import`
		`ldapmodify -B`

サフィックスの初期化

サフィックスを初期化すると、サフィックスに含まれている既存のデータが、追加するエントリだけを含む LDIF ファイルの内容によって上書きされます。

サフィックスを初期化するユーザーは、ディレクトリマネージャーまたは管理者としての認証を受けている必要があります。

サーバーが実行中の場合、ルートエントリを含む LDIF ファイルをインポートできるのは、ディレクトリマネージャーと管理者のみです。セキュリティー上の理由により、これらのユーザーのみが、たとえば `dc=example,dc=com.` のようなサフィックスのルートエントリへのアクセス権を持ちます。

レプリケーションアグリーメントに関連するサフィックスを復元する前に、214 ページの「レプリケートされたサフィックスの復元」をお読みください。

▼ サフィックスを初期化する

注 – インポートする LDIF ファイルでは、UTF-8 文字セットエンコードが使用されている必要があります。

サフィックスを初期化するときは、ルートエントリと、対応するサフィックスのすべてのディレクトリツリーノードが LDIF ファイルに含まれている必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 次のコマンドのいずれかを使用して、**LDIF** ファイルからサフィックスを初期化します。つまり、データベースの内容を **LDIF** ファイルにインポートします。

注意 – 次のコマンドで、サフィックスのデータを上書きします。

  - サーバーがローカルにあり、停止している場合は、次のように入力します。

```
$ dsadm import instance-path LDIF-file suffix-DN
```

    次の例では、`dsadm import` コマンドを使用して、LDIF ファイルを 1 つのサフィックスにインポートします。

```
$ dsadm import /local/ds /local/file/example/demo1.ldif \
 /local/file/example/demo2.ldif dc=example,dc=com
```

  - サーバーがリモートにあり、実行中の場合は、次のように入力します。

```
$ dsconf import -h host -p port LDIF-file suffix-DN
```

    次の例では、`dsconf import` を使用して LDIF ファイルをインポートします。このコマンドを実行するために root 権限は必要ありませんが、ディレクトリマネージャーなどの root 権限を持つユーザーとして認証される必要があります。

```
$ dsconf import -h host1 -p 1389 /local/file/example/demo1.ldif \
 ou=People,dc=example,dc=com
```

注 – `dsconf import` か `dsconf reindex` のいずれか、または複数のサフィックスで並行して両方のコマンドを実行すると、トランザクションログが大きくなり、パフォーマンスに悪影響を及ぼすことがあります。

これらのコマンドの詳細については、`dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

エントリの一括の追加、変更、削除

`ldapmodify` 操作を実行すると、エントリをまとめて追加、変更、削除できます。エントリは、既存のエントリを変更または削除するための更新文を含む LDIF ファイルに指定されています。この操作では、すでに存在しているエントリは消去されません。

変更されたエントリは、Directory Server で管理されるサフィックスの対象となることがあります。エントリを追加するほかの処理と同様に、インポートされた新しいエントリすべてにインデックスが付けられます。

`ldapmodify` コマンドにより、LDAP によって LDIF ファイルがインポートされ、このファイルに含まれるすべての操作が実行されます。このコマンドを使用すると、すべてのディレクトリサフィックスのデータを同時に変更できます。

レプリケーションアグリーメントに関連するサフィックスを復元する前に、214 ページの「レプリケートされたサフィックスの復元」を参照してください。

▼ エントリをまとめて追加、変更、削除する

注 – インポートする LDIF ファイルでは、UTF-8 文字セットエンコードが使用されている必要があります。

LDIF をインポートするときは、ディレクトリ内に親エントリが存在するか、ファイルから親エントリを最初にコピーする必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **LDIF ファイルからまとめて追加、変更、または削除します。**

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - -B baseDN -f LDIF-file
```

次の例では、`ldapmodify` コマンドを使用してインポートが実行されます。このコマンドを実行するために root 権限は必要ありませんが、`cn=Directory Manager` や `cn=admin,cn=Administrators,cn=config` などの root 権限を持つユーザーとして認証される必要があります。最後のパラメータは、インポートする LDIF ファイルの名前を指定するものです。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - \
 -B dc=example,dc=com -f /local/ds/ldif/demo.ldif
```

レプリケートされたサフィックスの復元

サプライヤサーバーとコンシューマサーバーの間でレプリケートされるサフィックスを復元する場合は、特別な注意が必要です。可能な場合は、サフィックスをバックアップから復元するのではなく、レプリケーションメカニズムにより更新するようにしてください。

サプライヤまたはハブインスタンスを復元する場合、サーバーはバックアップ時と同じ設定にする必要があります。これを確実に実行するには、Directory Server データを復元する前に、`dse.ldif` ファイルを復元します。210 ページの「`dse.ldif` 設定ファイルの復元」を参照してください。

ここでは、レプリカを復元すべき場合とその方法、および復元後にほかのレプリカとの同期を確保する方法について説明します。レプリカの初期化については、261 ページの「レプリカの初期化」を参照してください。

レプリケートされたサフィックスが大規模で、多数のエントリを追加し、レプリケーションの更新を確実に正しく追加されるようにする場合は、271 ページの「大量のレプリケートされたサフィックスへの多数のエントリの段階的追加」を参照してください。

この節では、次の情報について説明します。

シングルマスターモデルでのサプライヤの復元

シングルマスターサプライヤであるサフィックスには、レプリケーショントポロジ全体に対して重要なデータ (authoritative data) が含まれています。したがって、このサフィックスを復元することは、トポロジ全体のすべてのデータを初期化し直すことと同じです。シングルマスターを復元するのは、復元するバックアップの内容ですべてのデータを初期化し直す場合に限定してください。

エラーのためにシングルマスターのデータを復旧できない場合は、コンシューマ上のデータを使用することも検討してください。これは、バックアップされたデータより新しい更新がコンシューマ上のデータに含まれている可能性があるためです。この場合は、コンシューマレプリカから LDIF ファイルにデータをエクスポートし、この LDIF ファイルを使用してマスターを初期化し直します。

バックアップから復元する場合でも、LDIF ファイルをインポートする場合でも、このマスターレプリカから更新を受け取るすべてのハブレプリカとコンシューマレプリカをあとで初期化し直す必要があります。コンシューマの再初期化が必要であることを示すメッセージが、サプライヤサーバーのログファイルに記録されます。

マルチマスターモデルでのサプライヤの復元

マルチマスターレプリケーションでは、ほかの各マスターも、レプリケートされるデータに対してコピーする権限を持っています。現在のレプリカの内容が反映されていない可能性があるため、古いバックアップを復元することはできません。可能な場合は、レプリケーションメカニズムにより、ほかのマスターの内容を使用してマスターを更新するようにしてください。

それが不可能な場合は、次のどちらかの方法でマルチマスターレプリカを復元してください。

- もっとも簡単な方法として、バックアップから復元する代わりに、ほかのマスターの 1 つを使用して目的のマスターを初期化し直します。これにより、目的のマスターに最新のデータが送られ、データはすぐにレプリケートできる状態になります。262 ページの「LDIF からのレプリカの初期化」を参照してください。
- 数百万のエントリを持つレプリカの場合は、バイナリコピーを作成して、ほかのマスターの 1 つから作成したより新しいバックアップから復元することで、時間を短縮できます。266 ページの「バイナリコピーを使用したレプリケートされたサフィックスの初期化」を参照してください。
- このマスターのバックアップが、ほかのどのマスターに対しても更新履歴ログの最長保存期間を過ぎていない場合は、このバックアップを使用してマスターを復元できます。更新履歴ログの保存期間については、254 ページの「マスターレプリカの更新履歴ログ設定を変更する」を参照してください。このようにバックアップからマスターを復元すると、ほかのマスターはそれぞれの更新履歴ログを使用して、このマスターを更新します。これにより、バックアップの作成時以降に加えられた変更内容がすべてこのマスターに反映されます。

復元や再初期化の方法にかかわらず、初期化後のマスターレプリカは読み取り専用モードになります。この動作により、このレプリカとほかのマスターとの同期をとったあとに、書き込み操作を許可できます。詳細は、218 ページの「マルチマスターモデルでのマスターの復元」を参照してください。

復元または初期化し直したマスターに書き込み操作を許可する前に、すべてのレプリカを反映させることができるので、ハブサーバーやコンシューマサーバーを初期化し直すことが不要になるという利点があります。

ハブの復元

この節の内容は、レプリケーションメカニズムで自動的にハブレプリカを更新できない場合だけに適用されます。このような状況として、データベースファイルが破損した場合や、レプリケーションが長時間にわたって中断された場合が含まれます。このような場合は、次のいずれかの方法で、ハブレプリカを復元または初期化し直す必要があります。

- もっとも簡単な方法として、バックアップから復元する代わりに、ほかのマスターレプリカの 1 つを使用してハブを初期化し直します。これにより、ハブに最新のデータが送られ、データはすぐにレプリケートできる状態になります。212 ページの「サフィックスの初期化」を参照してください。
- 数百万のエントリを持つレプリカの場合は、バイナリコピーを作成して、別のハブのレプリカサフィックスから作成したより新しいバックアップから復元することで、所要時間を短縮できます。266 ページの「バイナリコピーを使用したレプリケートされたサフィックスの初期化」を参照してください。コピーできるほかのハブレプリカがない場合は、前の項目で説明した方法でハブを初期化し直すか、可能な場合は、次の項目で説明するようにハブを復元します。
- このハブのバックアップが、そのサプライヤ (ハブレプリカまたはマスターレプリカ) のどちらに対しても更新履歴ログの最長保存期間を過ぎていない場合は、このバックアップを使用してハブを復元できます。ハブが復元されると、そのサプライヤはそれぞれの更新履歴ログを使用して、このハブを更新します。これにより、バックアップの作成時以降に加えられた変更内容が、すべてこのハブに反映されます。

注 – ハブレプリカの復元や再初期化の方法にかかわらず、あとでこのハブのコンシューマをすべて初期化し直す必要があります。ほかのレベルのハブもすべて初期化し直す必要があります。

専用コンシューマの復元

この節の内容は、レプリケーションメカニズムで自動的に専用コンシューマレプリカを更新できない場合だけに適用されます。このような状況として、データベースファイルが破損した場合や、レプリケーションが長時間にわたって中断された場合が含まれます。このような場合は、次のいずれかの方法で、コンシューマを復元または初期化し直す必要があります。

- もっとも簡単な方法として、バックアップから復元する代わりに、そのサプライヤの 1 つ (マスターレプリカまたはハブレプリカ) を使用してコンシューマを初期化し直します。これにより、コンシューマに最新のデータが送られ、データはすぐにレプリケートできる状態になります。262 ページの「LDIF からのレプリカの初期化」を参照してください。
- 数百万のエントリを持つレプリカの場合は、バイナリコピーを作成して、別のコンシューマのレプリカサフィックスから作成したより新しいバックアップから復元することで、所要時間を短縮できます。266 ページの「バイナリコピーを使用したレプリケートされたサフィックスの初期化」を参照してください。コピーできるほかのコンシューマレプリカがない場合は、前の項目で説明した方法でレプリカを初期化し直すか、可能な場合は、次の項目で説明するようにハブを復元します。
- このコンシューマのバックアップが、そのサプライヤ (ハブレプリカまたはマスターレプリカ) のどちらに対しても更新履歴ログの最長保存期間を過ぎていない場合は、このバックアップを使用してこのコンシューマを復元できます。このようにコンシューマを復元すると、そのサプライヤはそれぞれの更新履歴ログを使用して、コンシューマを更新します。これにより、バックアップの作成時以降に加えられた変更内容が、すべてこのコンシューマに反映されます。

マルチマスターモデルでのマスターの復元

マルチマスターレプリケーションでは、あるマスターの復元中に他のマスターが変更を処理することもあります。このため、復元が完了した時点で、新しいマスターは復元データに含まれていなかった新しい更新を受け取る必要があります。マスターの復元には時間がかかるため、その間に発生する未適用の更新の数も問題となることがあります。

これらの未適用更新が適用されるように、新たに復元されたマスターは、復元後、クライアント側からの操作に対して自動的に読み取り専用モードに設定されます。これは、コマンド行で LDIF ファイルからデータをインポートするか、バックアップを使用してバイナリコピーを実行することで、マスターを復元する場合のみに当てはまります。

したがって、マルチマスター設定の復元後のマスターは、レプリケーションの更新を処理し、クライアントからの読み取り操作を受け付けますが、すべての書き込み操作に対してはリフェラルを返します。

更新を許可する前に、新しいマスターがほかのマスターと完全に同期していることを確認するには、初期化されたマスターを手動で更新できるようにします。

注 – この新しい対応方法によってマスターレプリカがリフェラルを送信する場合、書き込み処理を待機しているクライアントのホップ回数が、制限回数に達してしまうことも考えられます。利用可能なマスターにアクセスできるように、クライアントのホップ制限の設定を変更する必要があるかもしれません。すべてのマスターレプリカを初期化または再初期化するときは、どのレプリカもクライアントからの更新を受け付けられないため、すべての書き込み処理が失敗します。

サーバーの応答を最大化するには、いかなる場合も初期化したマスターを注意深く監視し、リフェラルの属性を適切に設定してください。

▼ コマンド行による更新の受け付けを開始する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

次のコマンドは、マルチマスターレプリカの初期化プロセスを自動化するスクリプトで使用できます。

1. `insync` ツールを使用して、レプリカの状態が他のすべてのマスターと一致していることを確認します。

   すべてのサーバーで変更の遅れがゼロである場合、またはそのレプリカに適用する更新がなかった場合 (遅れが `-1` となる場合) は、すべてのレプリカが同期しています。詳細については、`insync`(1) のマニュアルページを参照してください。

2. 更新の受け付けを始めます。

   ```
   $ dsconf set-suffix-prop -h host -p port suffix-DN repl-accept-client-update-enabled:on
   ```

   このコマンドは、サーバーを自動的に読み書きモードに設定します。

障害からの回復

Directory Server を障害回復の目的でバックアップまたは復元する場合は、次の手順に従います。

▼ 障害回復用のバックアップを作成する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1. `dsadn backup` コマンドまたは `dsconf backup` コマンドを使用して、データベースファイルのバックアップを作成します。
   205 ページの「バイナリバックアップ」で説明する手順に従い、バックアップファイルを安全な場所に保管します。
2. 設定ディレクトリ *instance-path* `/config` を安全な場所にコピーします。
3. スキーマディレクトリ *instance-path*`/config/schema` を安全な場所にコピーします。
4. 別名ディレクトリ *instance-path*`/alias` を安全な場所にコピーします。

▼ 障害回復のために復元する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1. すでにホスト上にあるものと同じバージョンの **Directory Server** をインストールします。
2. `dsadm create` コマンドを使用して、サーバーインスタンスを作成します。
   バックアップ時に使用したものと同じインスタンスを使用します。61 ページの「サフィックスの作成」を参照してください。
3. 設定ディレクトリ *instance-path* `/config` を復元します。
4. スキーマディレクトリ *instance-path*`/config/schema` を復元します。
5. 別名ディレクトリ *instance-path*`/alias` を復元します。
6. 復元されたサーバーの設定が正しいことを確認します。
   たとえば、ディレクトリ構造とプラグイン設定は、バックアップサーバー上のものと同じものにする必要があります。

7 `dsconf restore` コマンドを使用して、データベースファイルを復元します。
209 ページの「バイナリ復元」で説明した手順に従います。

◆ ◆ ◆ 第 9 章

9 Directory Server のグループ、ロール、および CoS

ディレクトリのデータの階層構造を超えて、ユーザーを表すエントリを管理するために、多くの場合、共通の属性値を共有するグループを作成する必要があります。Directory Server には、グループ、ロール、サービスクラス (CoS) による高度なエントリ管理機能が備えられています。

この章の内容は次のとおりです。

グループ、ロール、およびサービスクラスについて

グループ、ロール、および CoS は次のように定義されます。

- グループは、メンバーのリストまたはメンバーを定義するフィルタで表されるエントリです。メンバーのリストから構成されるグループの場合、Directory Server はユーザーエントリごとに `isMemberOf` 属性の値を生成します。そのため、ユーザーエントリの `isMemberOf` 属性には、そのエントリが属するすべてのグループが示されます。
- ロールは、ロールの各メンバーに対して `nsrole` 属性を生成するメカニズムによって、グループと同等またはそれ以上の機能を提供します。
- CoS は計算された属性を生成します。これにより、各エントリに属性を格納しなくても、エントリで共通の属性値を共有できます。

  共通の計算された属性値から自動的にスタティックグループのメンバー全員に継承させる目的で `isMemberOf` 属性を使用することはできません。

Directory Server には、ロール、グループ、CoS の計算された属性の値に基づいて検索を実行する機能があります。操作で使用するフィルタ文字列には、`nsRole` 属性また

はCoS定義によって生成された任意の属性を含めることができます。さらに、フィルタ文字列を使用して、この属性の値の比較操作を実行することもできます。ただし、計算されたCoS属性にインデックスを作成することはできません。そのため、CoSによって生成された属性を含む検索では、時間とメモリーの面で、大量のリソースを消費する可能性があります。

ロール、グループ、およびサービスクラスが提供する機能を活用するには、ディレクトリの配備を計画する段階で、ディレクトリのトポロジを決定しておく必要があります。これらの機能と、それらによってトポロジを簡単にする方法については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「ディレクトリエントリのグループ化と属性の管理」を参照してください。

ロールとグループの仕組みの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第8章「Directory Server Groups and Roles」を参照してください。CoSの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第9章「Directory Server Class of Service」を参照してください。

グループの管理

グループにより、エントリを関連付けて管理を簡単にすることができます。たとえば、グループを使用すると、アクセス制御命令(ACI)を簡単に定義できます。グループ定義は特別なエントリで、スタティックなリストにメンバーの名前を指定するか、またはダイナミックなエントリセットを定義するフィルタを指定します。

グループに含めることが可能なメンバーの範囲は、グループ定義エントリの位置に関係なく、ディレクトリ全体となります。管理を簡略化するために、すべてのグループ定義エントリは、通常、1か所に格納されます。通常は、ルートサフィックスの下の `ou=Groups` に格納されます。

グループにはスタティックグループとダイナミックグループの 2 つのタイプがあります。

- スタティックグループ。スタティックグループを定義するエントリは、groupOfNames または groupOfUniqueNames オブジェクトクラスから継承されます。グループのメンバーは、1 個以上の DN のリストであり、各 DN は、member または uniqueMember 属性値で表されます。

  または、スタティックグループに isMemberOf 属性を使用することができます。isMemberOf 属性は、検索の開始時に計算され、ユーザーエントリに追加されます。そして、検索の終了後に削除されます。この機能により、グループの管理が簡単になり、読み取りアクセスが高速になります。
- ダイナミックグループ。groupOfURLs オブジェクトクラスから継承されるダイナミックグループを定義するエントリ。グループのメンバーシップは、複数値属性 memberURL に指定された、1 つまたは複数のフィルタによって定義されます。フィルタが評価されたときにそのどれかに一致するエントリが、ダイナミックグループのメンバーとなります。

▼ 新しいスタティックグループを作成する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 新しいスタティックグループを作成するには、ldapmodify コマンドを使用します。

たとえば、System Administrators という新しいスタティックグループを作成し、メンバーを追加するには、次のコマンドを使用できます。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=System Administrators, ou=Groups, dc=example,dc=com
cn: System Administrators
objectclass: top
objectclass: groupOfNames
ou: Groups
member: uid=kvaughan, ou=People, dc=example,dc=com
member: uid=rdaugherty, ou=People, dc=example,dc=com
member: uid=hmiller, ou=People, dc=example,dc=com
```

2 新しいグループが作成され、メンバーが追加されたことを確認します。

たとえば、Kirsten Vaughan が新しい System Administrators グループに含まれているかを確認するには、次のように入力します。

```
$ ldapsearch -b "dc=example,dc=com" uid=kvaughan isMemberOf
uid=kvaughan,ou=People,dc=example,dc=com
isMemberOf: cn=System Administrators, ou=Groups, dc=example,dc=com
isMemberOf: cn=HR Managers,ou=groups,dc=example,dc=com
```

▼ 新しいダイナミックグループを作成する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- 新しいダイナミックグループを作成するには、`ldapmodify` コマンドを使用します。

  たとえば、Database Administrators という新しいダイナミックグループを作成し、Jensen という姓のメンバーを追加するには、次のコマンドを使用できます。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=Database Administrators, ou=Groups, dc=example,dc=com
cn: Database  Administrators
objectclass: top
objectclass: groupOfUrls
ou: Groups
memberURL: ldap:///dc=example,dc=com??sub?(sn=Jensen)
```

ロールの管理

ロールは、アプリケーションでより効率的かつ簡単に使用できるように設計された代替のグループ化メカニズムです。ロールはグループと同様に定義し、管理しますが、各メンバーエントリの生成されたロール属性は自動的にエントリのロールを示します。たとえば、アプリケーションでは、グループを選択してメンバーリストを参照しなくても、エントリのロールを読み取ることができます。

デフォルトでは、ロールの適用範囲は、その範囲が定義されているサブツリーに限定されます。ただし、入れ子のロールの範囲は拡張できます。ほかのサブツリーにあるロールを入れ子にしたり、ディレクトリの任意の場所のメンバーを含めたりすることができます。詳細については、230 ページの「ロールの範囲を拡張する」および 229 ページの「入れ子のロール定義の例」を参照してください。

この節では、ロールを安全に使用する方法とコマンド行からロールを管理する方法を説明します。

ロールの安全な使い方

ロールを安全に使用するには、アクセス制御命令 (ACI) を設定して、該当する属性を保護する必要があります。たとえば、ユーザー A が、管理ロール MR を所有しているとします。管理ロールはスタティックグループと同じで、`nsRoleDN` 属性をエントリに追加することによって、各メンバーエントリにロールを明示的に割り当てます。MR ロールは、コマンド行からアカウントの無効化を使用して、ロックされています。つまり、ユーザー A の `nsAccountLock` 属性は「true」として計算されるので、ユーザー A はサーバーにバインドできません。ただし、ユーザーがバインド済みで、MR ロールに関して現在ロックされているという通知を受けたとします。ユー

ザーが nsRoleDN 属性に書き込みアクセスできないようにする ACI が存在しなければ、ユーザーは自身のエントリから nsRoleDN 属性を削除し、自身のロックを解除できます。

ユーザーが nsRoleDN 属性を削除できないようにするには、ACI を適用する必要があります。フィルタを適用したロールを使用する場合、ユーザーが属性を変更してフィルタが適用されたロールを放棄することを防ぐために、フィルタの一部を保護する必要があります。フィルタが適用されたロールで使用されている属性をユーザーが追加、削除、または変更できないようにする必要があります。同様に、フィルタ属性の値を計算する場合、フィルタ属性の値を変更できるすべての属性を保護する必要があります。入れ子のロールには、フィルタを適用したロールと管理ロールが含まれることがあるため、入れ子のロールに含まれる各ロールについても上記注意点を考慮する必要があります。

セキュリティー目的で ACI を設定する詳細な手順については、第 6 章を参照してください。

コマンド行からのロールの管理

ロールは、ディレクトリ管理者がコマンド行ユーティリティーを使用してアクセスできるようにエントリに定義されます。ロールの作成が完了したら、次のようにロールにメンバーを割り当てます。

- 管理ロールのメンバーのエントリに、nsRoleDN 属性を含めます。
- フィルタを適用したロールのメンバーは、nsRoleFilter 属性で指定したフィルタに一致するエントリとなります。
- 入れ子のロールのメンバーは、入れ子のロール定義エントリの nsRoleDN 属性で指定したロールのメンバーとなります。

すべてのロール定義は LDAPsubentry および nsRoleDefinition オブジェクトクラスから継承されます。次の例に、各ロールタイプに固有のその他のオブジェクトクラスと関連付けられた属性を示します。

管理ロール定義の例

マーケティング担当者全員のロールを作成するには、次の ldapmodify コマンドを使用します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=Marketing,ou=marketing,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsSimpleRoleDefinition
objectclass: nsManagedRoleDefinition
```

```
cn: Marketing
description: managed role for marketing staff
```

nsManagedRoleDefinition オブジェクトクラスは、LDAPsubentry、nsRoleDefinition、および nsSimpleRoleDefinition オブジェクトクラスから継承されます。

Bob という名前のマーケティング担当者のメンバーにロールを割り当てるには、次のようにエントリを更新します。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=Bob Arnold,ou=marketing,ou=People,dc=example,dc=com
changetype: modify
add: nsRoleDN
nsRoleDN: cn=Marketing,ou=marketing,ou=People,dc=example,dc=com
```

nsRoleDN 属性は、エントリが管理ロールのメンバーであることを示します。管理ロールはそのロール定義の DN によって識別します。ユーザーが自身の nsRoleDN 属性を変更できるようにするが、nsManagedDisabledRole を追加または削除できないようにするには、次の ACI を追加します。

```
aci: (targetattr="nsRoleDN")(targattrfilters="add=nsRoleDN:
(!(nsRoleDN=cn=AdministratorRole,dc=example,dc=com)),
del=nsRoleDN:(!(nsRoleDN=cn=nsManagedDisabledRole,dc=example, dc=com)")
(version3.0;aci "allow mod of nsRoleDN by self except for critical values";
allow(write) userdn="ldap:///self";)
```

フィルタを適用したロール定義の例

営業マネージャーのフィルタを適用したロールを設定するには、全員が isManager 属性を持つものとして、次の ldapmodify コマンドを使用します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=ManagerFilter,ou=sales,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: ManagerFilter
nsRoleFilter: (isManager=True)
Description: filtered role for sales managers
```

nsFilteredRoleDefinition オブジェクトクラスは、LDAPsubentry、nsRoleDefinition、および nsComplexRoleDefinition オブジェクトクラスから継承されます。nsRoleFilter 属性は、下位組織を持つ ou=sales 組織のすべての従業員を検索するフィルタを指定します。たとえば、次のように指定します。

```
$ ldapsearch -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w - \
 -b "ou=People,dc=example,dc=com" -s sub "(cn=*Fuentes)"
dn: cn=Carla Fuentes,ou=sales,ou=People,dc=example,dc=comcn: Carla Fuentes
isManager: TRUE...
nsRole: cn=ManagerFilter,ou=sales,ou=People,
dc=example,dc=com
```

注– フィルタを適用したロールのフィルタ文字列には、任意の属性を使用できます。ただし、CoS メカニズムによって生成される計算された属性は使用できません。

フィルタを適用したロールのメンバーがユーザーエントリである場合、それらが自身をロールに追加または削除する機能を制限することができます。ACI によってフィルタを適用した属性を保護します。

入れ子のロール定義の例

入れ子のロール内に入れ子にするロールは nsRoleDN 属性を使用して指定します。前の例で作成したロールのマーケティング担当者と営業マネージャーのメンバーの両方を含むロールを作成するには、次のコマンドを使用します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=MarketingSales,ou=marketing,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsNestedRoleDefinition
cn: MarketingSales
nsRoleDN: cn=ManagerFilter,ou=sales,ou=People,dc=example,dc=com
nsRoleDN: cn=Marketing,ou=marketing,ou=People,dc=example,dc=com
nsRoleScopeDN: ou=sales,ou=People,dc=example,dc=com
```

nsNestedRoleDefinition オブジェクトクラスは LDAPsubentry、nsRoleDefinition、および nsComplexRoleDefinition オブジェクトクラスから継承されます。nsRoleDN 属性には、マーケティングの管理ロールとセールスマネージャーのフィルタが適用されたロールの DN が含まれます。前述の例のユーザー Bob と Carla は、どちらもこの新しい入れ子のロールのメンバーになります。

このフィルタの範囲には、フィルタが存在するサブツリーと nsRoleScopeDN 属性の値以下のサブツリーであるデフォルトの範囲が含まれます。この例では、ManagerFilter が ou=sales,ou=People,dc=example,dc=com サブツリーにあります。このサブツリーを範囲に追加する必要があります。

ロールの範囲拡張

Directory Server では、ロールの範囲をロール定義エントリのサブツリーを超えて拡張するための属性を使用できます。これは、nsRoleScopeDN という 1 つの値からなる属性で、既存のロールに追加する範囲の DN を含みます。nsRoleScopeDN 属性を追加できるのは、入れ子のロールだけです。229 ページの「入れ子のロール定義の例」を参照してください。

▼ ロールの範囲を拡張する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

nsRoleScopeDN 属性により、あるサブツリーのロールの範囲を拡張して、別のサブツリーにエントリを含めることができます。たとえば、example.com ディレクトリツリーに次の 2 つのメインサブツリーがあるとします。 o=eng,dc=example,dc=com (エンジニアリングサブツリー) および o=sales,dc=example,dc=com (販売サブツリー)。エンジニアリングサブツリーのユーザーには、販売サブツリーのロール (SalesAppManagedRole) で管理される販売アプリケーションに対するアクセス権が必要です。ロールの範囲を拡張するには、次を実行します。

1. エンジニアリングサブツリーのユーザーのロールを作成します。

   たとえば、EngineerManagedRole のロールを作成します。この例では、管理されるロールを使用していますが、フィルタが適用されたロールや入れ子のロールであってもかまいません。

2. 販売サブツリーに、たとえば SalesAppPlusEngNestedRole のような入れ子のロールを作成し、新たに作成した EngineerManagedRole と、元からある SalesAppManagedRole を格納します。

3. SalesAppPlusEngNestedRole に nsRoleScopeDN 属性を追加します。属性値には、追加するエンジニアリングサブツリーの範囲の **DN** を指定します。この例では、o=eng,dc=example,dc=com を指定します。

   エンジニアリングユーザーには、SalesAppPlusEngNestedRole ロールにアクセスして販売アプリケーションにアクセスできるよう、適切なアクセス権を与える必要があります。さらに、ロールの範囲全体をレプリケートする必要があります。

注 – 拡張する範囲を入れ子のロールに制限することは、以前に、あるドメインのロールを管理していた管理者は、その他のドメインに既に存在するロールの使用権限しか持たないことを意味します。管理者はその他のドメインに任意のロールを作成することはできません。

サービスクラス

サービスクラス (CoS) メカニズムは、クライアントアプリケーションがエントリを取り出すときに計算された属性を生成し、エントリの管理を簡単にして、必要なストレージ容量を削減します。CoS メカニズムにより、エントリ間で属性を共有できます。グループやロールの場合と同様に、CoS はヘルパーエントリに依存しています。

配備で CoS を使う方法については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「サービスクラスによる属性の管理」を参照してください。

CoS を Directory Server に実装する方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 9 章「Directory Server Class of Service」を参照してください。

注 – すべての検索操作で、CoS で生成された属性の有無を調べたり、属性の値を比較したりできます。フィルタを適用したロールに使用されている内部フィルタを除き、クライアントの検索操作のすべてのフィルタ文字列に、計算された属性の名前を使用できます。

CoS の安全な使い方

次に、各 CoS エントリのデータに読み取りおよび書き込み保護を設定する際の一般的な原則について説明します。各アクセス制御命令 (ACI) を定義する詳細な手順については、第 6 章で説明しています。

CoS 定義のエントリの保護

CoS 定義のエントリには、生成された属性の値は含まれません。このエントリは、値を検索するための情報を提供します。CoS 定義エントリを読み取ると、値を含むテンプレートエントリを見つける方法が分かります。このエントリに書き込むと、計算された属性の生成方法が変更されます。

したがって、CoS 定義のエントリに読み取りと書き込みの両方の ACI を定義する必要があります。

CoS テンプレートエントリの保護

CoS テンプレートエントリには、生成された CoS 属性の値が含まれます。したがって、少なくともテンプレートの CoS 属性の読み取りと更新を ACI によって 保護する必要があります。

- ポインタ *CoS* の場合は、1 つのテンプレートエントリの名前の変更が禁止されています。通常、テンプレートエントリ全体を保護するのがもっとも簡単な方法です。
- クラシック *CoS* では、すべてのテンプレートエントリは、定義エントリで指定された共通の親を持ちます。この親エントリにテンプレートを格納するだけで、親エントリに対するアクセス制御によってテンプレートが保護されます。ただし、親の下のほかのエントリにアクセスする場合は、テンプレートエントリを個別に保護する必要があります。
- 間接 *CoS* の場合は、アクセスする必要があるユーザーエントリを含む、ディレクトリ内の任意のエントリにテンプレートを指定できます。必要に応じて、ディレクトリ全体の CoS 属性に対するアクセスを制御するか、またはテンプレートとして使用される各エントリの CoS 属性のセキュリティーを保護することができます。

CoS のターゲットエントリの保護

計算された CoS 属性が生成される、CoS 定義の適用範囲内のすべてのエントリも値の算出に役立ちます。

CoS 属性がターゲットエントリにすでに存在する場合は、デフォルトでは、CoS メカニズムはこの値を上書きしません。この動作を変更する場合は、ターゲットエントリを上書きするように CoS を定義するか、すべてのターゲットエントリで CoS 属性を保護します。

間接 CoS とクラシック CoS は、ターゲットエントリの specifier 属性に依存します。この属性は、使用するテンプレートエントリの DN または RDN を指定します。ACI を使用してこの属性を保護する場合は、CoS の適用範囲全体でグローバルに保護するか、または各ターゲットエントリで必要に応じて個別に保護する必要があります。

その他の従属関係の保護

計算された CoS 属性は、ほかの生成された CoS 属性やロールを基に定義できます。計算された CoS 属性を保護するには、これらの従属関係を理解し、保護する必要があります。

たとえば、ターゲットエントリの CoS 指定子属性が `nsRole` になることがあります。したがって、ロール定義も ACI によって保護する必要があります。

一般に、計算された属性値の算出に関係する属性またはエントリには、読み取りおよび書き込みアクセス制御の ACI を設定します。このため、複雑な従属関係は、十

分に計画してから設定するか、以後のアクセス制御の実装の複雑さを軽減できるように簡素化する必要があります。その他の計算された属性との従属関係を最小限に抑えると、ディレクトリのパフォーマンスを向上させ、管理作業を削減することができます。

コマンド行からの CoS の管理

設定情報とテンプレートデータはすべてディレクトリ内にエントリとして格納されるので、LDAP コマンド行ツールを使用して CoS 定義を設定、管理できます。ここでは、コマンド行を使用して CoS 定義エントリと CoS テンプレートエントリを作成する方法について説明します。

コマンド行からの CoS 定義のエントリの作成

すべての CoS 定義エントリは `LDAPsubentry` オブジェクトクラスを持ち、`cosSuperDefinition` オブジェクトクラスから継承されます。さらに、CoS の各タイプは、特定のオブジェクトクラスから継承され、対応する属性を含みます。次の表に、各タイプの CoS 定義エントリに関連付けられたオブジェクトクラスと属性を一覧表示します。

表 9–1　CoS 定義エントリのオブジェクトクラスと属性

CoS のタイプ	CoS 定義のエントリ

ポインタ CoS	`objectclass: top`
	`objectclass: LDAPsubentry`
	`objectclass: cosSuperDefinition`
	`objectclass: cosPointerDefinition`
	`cosTemplateDN:` *DN*
	`cosAttribute:` *attributeName override merge*

間接 CoS	`objectclass: top`
	`objectclass: LDAPsubentry`
	`objectclass: cosSuperDefinition`
	`objectclass: cosIndirectDefinition`
	`cosIndirectSpecifier:` *attributeName*
	`cosAttribute:` *attributeName override merge*

表 9–1 CoS 定義エントリのオブジェクトクラスと属性 (続き)

CoS のタイプ	CoS 定義のエントリ

クラシック CoS	`objectclass: top`
	`objectclass: LDAPsubentry`
	`objectclass: cosSuperDefinition`
	`objectclass: cosClassicDefinition`
	`cosTemplateDN:` *DN*
	`cosSpecifier:` *attributeName*
	`cosAttribute:` *attributeName override merge*

`cosAttribute` は複数値属性です。各値は CoS メカニズムによって生成される属性を定義します。

CoS 定義エントリには次の属性を使用できます。これらの各属性の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』の各属性を参照してください。

表 9–2 CoS 定義のエントリの属性

属性	CoS 定義のエントリ内の目的
`cosAttribute` *attributeName override merge*	値を生成する対象となる計算された属性の名前を定義します。この属性は複数値属性です。それぞれの値は属性の名前を表し、この属性値はテンプレートから生成されます。*override* 修飾子と *merge* 修飾子により、次の表に示す特殊な場合での CoS 属性値の算出方法を指定します。 *attributeName* にサブタイプを含めることはできません。サブタイプを持つ属性値は無視されますが、`cosAttribute` のその他の値は処理されます。
`cosIndirectSpecifier` *attributeName*	ターゲットエントリの属性名を定義します。間接 CoS は、この属性の値を使用してテンプレートエントリを識別します。名前が指定された属性は指示子と呼ばれ、各ターゲットエントリに完全 DN 文字列を含める必要があります。この属性には値を 1 つしか指定できませんが、*attributeName* に複数値属性を指定して複数のテンプレートを指定できます。
`cosSpecifier` *attributeName*	ターゲットエントリの属性名を定義します。クラシック CoS は、この属性の値を使用してテンプレートエントリを識別します。名前が指定された属性は指示子と呼ばれ、ターゲットエントリの RDN になる文字列を含める必要があります。この属性には値を 1 つしか指定できませんが、*attributeName* に複数値属性を指定して複数のテンプレートを指定できます。
`cosTemplateDN` *DN*	ポインタ CoS 定義用にテンプレートエントリの完全 DN、またはクラシック CoS 用にテンプレートエントリのベース DN を指定します。この属性は単一の値です。

注 – isMemberOf 属性を CosSpecifier として使用することで、共通の計算された属性値から自動的にスタティックグループのメンバー全員に継承させることはできません。

cosAttribute 属性には、CoS 属性の名前のあとに、override 修飾子と merge 修飾子の 2 つの修飾子を指定できます。

override 修飾子は CoS によって動的に生成された属性が、すでに物理的にエントリに存在する場合の動作を示します。*override* 修飾子は次のいずれかを指定できます。

- default (または修飾子なし): エントリに計算された属性と同じタイプの実際の属性が存在する場合、サーバーはエントリに格納されている実際の属性値を上書きしません。
- override: 値がすでにエントリに格納されている場合でもサーバーは CoS によって生成された値を常に返すことを示します。
- operational: 検索で属性が明示的に要求されている場合にのみ、属性を返すことを示します。operational 属性の場合は、この属性を取得するために、スキーマ検査を渡す必要はありません。operational 修飾子の動作は override 修飾子と同じです。

  属性を operational にすることができるのは、その属性がスキーマ内でも operational と定義されている場合だけです。たとえば、description 属性は、スキーマ内で operational としてマークされていないので、CoS を使用して description 属性の値を生成する場合は、operational 修飾子を使用できません。

merge 修飾子には何も指定しないか、merge-schemes を指定するかのどちらかです。この修飾子は複数のテンプレートまたは複数の CoS 定義から、計算された CoS 属性に複数の値を指定できます。詳細については、236 ページの「複数の値を持つ CoS 属性」を参照してください。

実際の属性値の上書き

override 修飾子を含むポインタ CoS 定義のエントリの作成例を次に示します。

```
dn: cn=pointerCoS,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=exampleUS,cn=data
cosAttribute: postalCode override
```

このポインタ CoS 定義のエントリでは、このポインタ CoS が、postalCode 属性の値を生成するテンプレートエントリ cn=exampleUS,cn=data に関連付けられています。

override 修飾子が指定されているので、この値がターゲットエントリに存在する場合は、その postalCode 属性値よりも、この値が優先されます。

注 – CoS 属性に operational または override 修飾子を定義すると、CoS 適用範囲内のエントリでは、その属性の「実際」の値に対して書き込み操作を行うことはできなくなります。

複数の値を持つ CoS 属性

merge-schemes 修飾子を指定した場合、生成される CoS 属性には、次の 2 つの方法で複数の値を指定できます。

- 間接 CoS またはクラシック CoS では、ターゲットエントリの specifier 属性に複数の値を指定できます。この場合、それぞれの値によってテンプレートが決定され、各テンプレートの値は生成された値の一部になります。
- 任意のタイプの複数の CoS 定義のエントリで、cosAttribute に同じ属性名を含めることができます。この場合、すべての定義に merge-schemes 修飾子が含まれているときは、各定義によって算出されたすべての値が生成された属性に含まれます。

2 つの状況が同時に発生したり、さらに多くの値を定義する場合もあります。ただし、どの場合でも、重複した値が生成された属性に返されるのは 1 度だけです。

merge-schemes 修飾子を指定しない場合は、テンプレートエントリの cosPriority 属性を使用して、生成された属性のすべてのテンプレートの中から 1 つの値を決定します。この状況については、次の節で説明します。

merge-schemes 修飾子は、ターゲットに定義された「実際」の値とテンプレートから生成された値をマージしません。*merge* 修飾子は *override* 修飾子に依存しません。あらゆる組み合わせが可能で、それぞれが示す動作は有効です。また、修飾子は属性名のあとに任意の順序で指定できます。

注 – 同じ属性に複数の CoS 定義が存在する場合は、そのすべての定義に同じ *override* 修飾子および *merge* 修飾子を指定する必要があります。CoS 定義に指定された修飾子の組み合わせが異なる場合は、すべての定義から任意の 1 つの組み合わせが選択されます。

CoS 属性の優先順位

複数の CoS 定義または複数値指示子が存在するが、merge-schemes 修飾子が存在しない場合、Directory Server は優先順位属性を使用して、計算された属性の 1 つの値を定義する 1 つのテンプレートを選択します。

cosPriority 属性は、対象となるすべてのテンプレートの中での特定のテンプレートのグローバルな優先順位を表します。優先順位 0 は、優先順位がもっとも高いことを示します。cosPriority 属性を含まないテンプレートは、もっとも優先順位が低いとみなされます。2 つ以上のテンプレートによって属性値が指定されているが、優先順位が同じまたは設定されていない場合は、任意の値が選択されます。

merge-schemes 修飾子を使用する場合は、テンプレートの優先順位は考慮されません。マージするときに、テンプレートで定義する優先順位に関係なく、対象となるすべてのテンプレートが値を定義します。次の節で説明するように、cosPriority 属性は CoS テンプレートエントリに対して定義されます。

注 – cosPriority 属性には負の値を指定できません。また、間接 CoS が生成する属性は優先順位をサポートしていません。間接 CoS 定義のテンプレートエントリでは、cosPriority を使用しないでください。

コマンド行からの CoS テンプレートエントリの作成

ポインタ CoS またはクラシック CoS を使用する場合、テンプレートエントリには LDAPsubentry および cosTemplate オブジェクトクラスが含まれます。このエントリは、特に CoS 定義用に作成する必要があります。CoS テンプレートエントリを LDAPsubentry オブジェクトクラスのインスタンスにすることで、設定エントリの影響を受けずに、通常の検索を実行できるようになります。

間接 CoS メカニズムのテンプレートは、ディレクトリ内の任意の既存テンプレートエントリです。事前にターゲットを指定する必要はなく、LDAPsubentry オブジェクトクラスを指定する必要もありませんが、ターゲットに任意の cosTemplate オブジェクトクラスが含まれている必要があります。間接 CoS テンプレートには、CoS を評価して計算された属性とその値を生成する場合にだけアクセスします。

どのような場合でも CoS テンプレートエントリには、ターゲットエントリ上の CoS によって生成された属性と値を含める必要があります。属性名は、CoS 定義のエントリの cosAttribute 属性に指定されています。

次の例は、postalCode 属性を生成するポインタ CoS の優先順位がもっとも高いテンプレートエントリを示します。

```
dn: cn=ZipTemplate,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: extensibleobject
objectclass: cosTemplate
postalCode: 95054
cosPriority: 0
```

次の節では、テンプレートエントリの例と CoS 定義のエントリの各タイプの例を紹介します。

ポインタ CoS の例

次のコマンドは cosPointerDefinition オブジェクトクラスを持つポインタ CoS 定義エントリを作成します。この定義エントリでは、前の節の例で示したCoS テンプレートエントリを使用して、ou=People,dc=example,dc=com ツリーのすべてのエントリで共通の郵便番号を使用します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=pointerCoS,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=ZipTemplate,ou=People,dc=example,dc=com
cosAttribute: postalCode
```

ここで作成した CoS テンプレートエントリ cn=ZipTemplate,ou=People,dc=example,dc=com は、ou=People,dc=example,dc=com サフィックスの下に置かれているすべてのエントリに対して、その postalCode 属性に格納されている値を提供します。同じサブツリーで郵便番号を持たないエントリを検索すると、生成される属性の値は次のようになります。

```
$ ldapsearch -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w - \
 -b "ou=People,dc=example,dc=com" -s sub "(cn=*Jensen)"
dn: cn=Babs Jensen,ou=People,dc=example,dc=com
cn: Babs Jensen
...
postalCode: 95054
```

間接 CoS の例

間接 CoS は cosIndirectSpecifier 属性の属性に名前を付けて、各ターゲットに固有のテンプレートを特定します。間接 CoS のテンプレートエントリには、その他のユーザーエントリを含むディレクトリ内のすべてのエントリを指定できます。この例の間接 CoS は、ターゲットエントリの manager 属性を使用して、CoS テンプレートエントリを識別するものです。テンプレートエントリはマネージャーのユーザーエントリです。マネージャーのユーザーエントリには、生成する属性の値が含まれます。この例では、値は departmentNumber の値です。

次のコマンドは cosIndirectDefinition オブジェクトクラスを含む間接 CoS 定義エントリを作成します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=generateDeptNum,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
```

```
objectclass: cosIndirectDefinition
cosIndirectSpecifier: manager
cosAttribute: departmentNumber
```

次に、テンプレートエントリに cosTemplate オブジェクトクラスを追加し、生成する属性が定義されていることを確認します。この例では、すべてのマネージャーエントリはテンプレートです。

```
$ ldapmodify -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=Carla Fuentes,ou=People,dc=example,dc=com
changetype: modify
add: objectclass
objectclass: cosTemplate
-
add: departmentNumber
departmentNumber: 318842
```

この CoS では、manager 属性を含むターゲットエントリ (ou=People,dc=example,dc=com の下のエントリ) は、自動的にマネージャーの部署番号を持ちます。ターゲットエントリの departmentNumber 属性がサーバーに存在しないため、計算されます。ただし、departmentNumber 属性はターゲットエントリの一部として返されます。たとえば、Babs Jensen のマネージャーを Carla Fuentes として定義した場合、このマネージャーの部署番号は次のように表示されます。

```
$ ldapsearch -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w - \
 -b "ou=People,dc=example,dc=com" -s sub "(cn=*Jensen)"
dn: cn=Babs Jensen,ou=People,dc=example,dc=com
cn: Babs Jensen
...
manager: cn=Carla Fuentes,ou=People,dc=example,dc=com
departmentNumber: 318842
```

クラシック CoS の例

この例では、クラシック CoS によって住所を生成する方法を示します。生成される値は、CoS 定義の cosTemplateDN とターゲットエントリの cosSpecifier 属性の値の組み合わせで検索されるテンプレートエントリに指定されます。次のコマンドは cosClassicDefinition オブジェクトクラスを使用して、定義エントリを作成します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=classicCoS,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: ou=People,dc=example,dc=com
cosSpecifier: building
cosAttribute: postalAddress
```

同じコマンドを使用して、各ビルの住所を持つテンプレートエントリを作成します。

```
dn: cn=B07,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: extensibleobject
objectclass: cosTemplate
postalAddres: 7 Old Oak Street, Anytown, CA 95054
```

この CoS では、`building` 属性を含むターゲットエントリ (`ou=People,dc=example,dc=com` の下のエントリ) は、自動的に対応する住所を持ちます。CoS メカニズムは、RDN 内に specifier 属性値を持つテンプレートエントリを検索します。この例では、Babs Jensen に B07 ビルが割り当てられていれば、住所は次のように表示されます。

```
$ ldapsearch -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w - \
 -b "ou=People,dc=example,dc=com" -s sub "(cn=*Jensen)"
dn: cn=Babs Jensen,ou=People,dc=example,dc=com
cn: Babs Jensen
...
building: B07
postalAddress: 7 Old Oak Street, Anytown, CA 95054
```

ロールに基づく属性の作成

エントリで所有されているロールに基づいたエントリの属性値を生成するクラシック CoS スキーマを作成できます。たとえば、ロールに基づく属性を使用して、サーバーの検索制限をエントリごとに設定できます。

ロールに基づく属性を作成するには、クラシック CoS の CoS 定義のエントリ内で `cosSpecifier` として `nsRole` 属性を使用します。`nsRole` 属性には複数の値を指定できるので、複数の使用可能なテンプレートエントリを含む CoS スキーマを定義できます。使用するテンプレートエントリを明確に決定するには、`cosPriority` 属性を CoS テンプレートエントリに追加します。

たとえば、マネージャーロールのメンバーであれば、標準のメールボックス容量の割り当てを超えて使用できるようにする CoS を作成できます。次のようなマネージャーロールがあるとします。

```
dn: cn=ManagerRole,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
```

```
cn: ManagerRole
nsRoleFilter: (isManager=True)
Description: filtered role for managers
```

次のようなクラシック CoS 定義のエントリが作成されます。

```
dn: cn=generateManagerQuota,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: cn=managerCOS,ou=People,dc=example,dc=com
cosSpecifier: nsRole
cosAttribute: mailboxquota override
```

CoS テンプレートの名前は、`cosTemplateDn` と、`nsRole` の値 (ロールの DN) の組み合わせである必要があります。次に例を示します。

```
dn:cn="cn=ManagerRole,ou=People,dc=example,dc=com",\
 cn=managerCOS,ou=People,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: extensibleobject
objectclass: cosTemplate
mailboxquota: 1000000
```

CoS テンプレートエントリは、`mailboxquota` 属性値を提供します。追加で指定した `override` 修飾子は、CoS がターゲットエントリ内にある既存のすべての `mailboxquota` 属性値を上書きするように指定します。ロールのメンバーであるターゲットエントリは、たとえば次のような、ロールと CoS が生成する計算された属性を持ちます。

```
$ ldapsearch -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -\
 -b "ou=People,dc=example,dc=com" -s sub "(cn=*Fuentes)"
dn: cn=Carla Fuentes,ou=People,dc=example,dc=comcn: Carla Fuentes
isManager: TRUE...nsRole: cn=ManagerRole,ou=People,dc=example,dc=com
mailboxquota: 1000000
```

注 – ロールエントリおよび CoS 定義のエントリは、適用範囲内に同じターゲットエントリを指定できるように、ディレクトリツリーの同じ位置に置く必要があります。CoS ターゲットエントリも、検索や管理を簡単に実行できるように、同じ位置に置く必要があります。

CoS プラグインの監視

Directory Server では、CoS プラグインの特定の面を監視できます。CoS 監視属性は cn=monitor,cn=Class of Service,cn=plugins,cn=config エントリに格納されます。このエントリの各属性の詳細とそれらが提供する情報については、『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』を参照してください。

CoS ログの設定

ディレクトリサーバーは、複数の該当する定義エントリ間で何らかの区別を付ける必要がある場合に、警告メッセージを記録します。それらの警告メッセージは次の形式になります。

```
Definition /defDN1/ and definition /defDN2/ compete to provide attribute
 '/type/' at priority /level/
```

さらに、ディレクトリサーバーが複数の該当する可能性のある定義エントリ間で何らかの区別を付ける必要がある場合に、情報メッセージを記録するように、サーバーを設定することもできます。このためには、プラグインからのメッセージを含めるようにエラーログを設定します。

注 – 追加のログレベルを設定すると、ログの負荷が増す可能性があるため、本稼働用サーバーではログを設定しない方がよいです。

情報メッセージの内容は次の形式になります。

```
Definition /defDN1/ and definition /defDN2/ potentially compete
to provide attribute '/type/' at priority /level/
```

次に、定義エントリに CoS の優先順位を適切に設定することによって、CoS のあいまいな状況を解決するかどうかを選択できます。

参照の完全性の管理

参照の完全性はエントリ間の関係を維持するためのプラグインメカニズムです。グループのメンバーシップなど、一部のタイプの属性には別のエントリの DN が含まれています。参照の完全性を利用することで、エントリを削除したときに、そのエントリの DN を含むすべての属性も削除できます。

たとえば、参照の完全性が有効になっているときに、あるユーザーのエントリがディレクトリから削除されると、そのユーザーは、所属しているあらゆるグループからも削除されます。参照の完全性が無効な状態では、管理者はグループからユー

ザーを手動で削除する必要があります。これは、Directory Server を、ユーザーとグループの管理にディレクトリを使用するほかの Sun Java System 製品に統合する場合に重要な機能です。

参照の完全性の仕組み

参照の完全性プラグインが有効になっているときに削除操作や名前変更、または移動の操作を実行すると、指定された属性に対する完全性更新がただちに実行されます。ただし、デフォルトでは、参照の完全性プラグインは無効になっています。

ディレクトリ内のユーザーエントリまたはグループエントリの削除、名前の変更、移動を行なった場合、常に操作が参照の完全性のログファイルに記録されます。

instance-path/logs/referint

更新間隔と呼ばれる指定した時間が経過すると、参照の完全性が有効になっているすべての属性が検索され、検索結果のエントリと、ログファイル内に記録された削除または変更されたエントリの DN が照合されます。特定のエントリが削除されたことがログファイルに記録されている場合は、対応する属性が削除されます。特定のエントリが変更されたことがログファイルに記録されている場合は、対応する属性値が記録に従って変更されます。

参照の完全性プラグインのデフォルトの設定が有効になっている場合に、削除、名前変更、移動操作を行うと、ただちに member、uniquemember、 owner、seeAlso、および nsroledn 属性に対する完全性更新が実行されます。ただし、参照の完全性プラグインの動作は、次のような用途に合わせてユーザーが自由に設定できます。次の動作を設定できます。

- 参照の完全性の更新を別のファイルに記録する。
- 更新間隔を変更する。

  参照の完全性の更新がシステムに与える影響を軽減するために、更新間隔を長くする。
- 参照の完全性を適用する属性を選択する。

  DN 値を含む属性を使用または定義するために、参照の完全性プラグインを使用してそれを監視する。

▼ 参照の完全性プラグインを設定する

注 – 参照の完全性プラグインで使用される全データベースのすべての属性に、インデックスを設定する必要があります。インデックスはすべてのデータベースの設定内で作成する必要があります。旧バージョン形式の更新履歴ログが有効になっている場合、cn=changelog サフィックスにインデックスを設定する必要があります。詳細については、第 12 章を参照してください。

レプリケートされた環境では、特定の制限が参照の完全性プラグインの使用に関連付けられています。これらの制限の一覧については、272 ページの「レプリケーションと参照の完全性」を参照してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. すべてのレプリカが設定され、すべてのレプリケーションアグリーメントが定義されていることを確認します。
2. 参照の完全性を維持する一連の属性を定義し、マスターサーバーで使用する更新間隔を決定します。
3. 同じ属性セットと同じ更新間隔を使用して、すべてのマスターサーバーで参照の完全性プラグインを有効にします。
   - 参照の完全性の属性を定義するには、次のコマンドを使用します。

     ```
     $ dsconf set-server-prop -h host -p port ref-integrity-attr:attribute-name \
      ref-integrity-attr:attribute-name
     ```
   - 既存の属性リストに参照の完全性属性を追加するには、次のコマンドを使用します。

     ```
     $ dsconf set-server-prop -h host -p port ref-integrity-attr+:attribute-name
     ```
   - 参照の完全性の更新間隔を定義するには、次のコマンドを使用します。

     ```
     $ dsconf set-server-prop -h host -p port ref-integrity-check-delay:duration
     ```
   - 参照の完全性を有効にするには、次のコマンドを使用します。

     ```
     $ dsconf set-server-prop -h host -p port ref-integrity-enabled:on
     ```
4. すべてのコンシューマサーバー上で参照の完全性プラグインが無効になっていることを確認します。

◆ ◆ ◆ 第 10 章

10 Directory Server のレプリケーション

レプリケーションは、Directory Server のディレクトリの内容を別の 1 つまたは複数の Directory Server に自動的にコピーするメカニズムです。すべての書き込み操作が自動的に他の Directory Server にミラー化されます。レプリケーションの概念、レプリケーションの導入例、特定のディレクトリ配備でのレプリケーションの計画方法の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

レプリケーショントポロジでは、一般にサーバー上のサフィックスとサーバー上の別のサフィックス間でレプリケートします。このため、レプリカ、レプリケートされたサフィックス、レプリケートされたサーバーという語は同じ意味で使うことができます。

この章では、コマンド行を使用したレプリケーションのさまざまな導入例の設定作業について説明します。説明する内容は次のとおりです。

レプリケーション配備の計画

無限の数のマスターによるレプリケーション配備を設定できます。配備にハブやコンシューマを含める必要はありません。ハブやコンシューマのレプリケーションを設定する手順も、この章で説明しますが、必須の作業ではありません。

レプリケーションの設定を始める前に、組織でレプリケーションを配備する方法を十分に理解している必要があります。『Sun Java System Directory Server Enterprise Edition 6.1 Reference』で説明するレプリケーションの概念を理解する必要があります。さらに、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』で説明している設計ガイドラインを使用して、今後のレプリケーション設定を慎重に計画することも必要です。

レプリケーションの設定と管理に推奨されるインタフェース

最も簡単にレプリケーションを設定し、管理する方法は、Directory Service Control Center (DSCC) を使用することです。DSCC を使用すると、自動的にレプリケーションを設定できます。レプリケーショントポロジの設定に必要な自動化のレベルを選択できます。たとえば、レプリケーションの設定時にサフィックスを初期化するかどうかを選択できます。DSCC には、エラーを回避できるチェックも備えられています。さらに、DSCC ではレプリケーショントポロジをグラフィカルに表示します。

DSCC のオンラインヘルプに、DSCC を使用してレプリケーションを設定する手順を説明しています。

注 – この章で説明するコマンド行の手順は、レプリケーションの設定に DSCC を使用できない場合にのみ使用してください。

レプリケーションの設定手順の概要

247 ページの「レプリケーションの設定手順の概要」では、1 つのサフィックスをレプリケートすることを前提としています。複数のサフィックスをレプリケートする場合は、各サーバーでサフィックスを並行して設定する必要があります。つまり、複数サフィックスのレプリケーションを設定するには、各手順を繰り返す必要があります。

この章の後半で、レプリケーションの設定方法を詳しく説明します。

▼ レプリケーションの設定手順の概要

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

レプリケーショントポロジを設定するには、次の手順で説明するような一般的な手順に従います。

1 すべてのサーバーで次の操作を実行し、サーバー上に専用のコンシューマレプリカを作成します。

   a. コンシューマのレプリカサフィックス用の空のサフィックスを作成します。
   249 ページの「コンシューマのレプリカサフィックスを作成する」を参照してください。

   b. コンシューマのレプリカサフィックスを有効にします。
   250 ページの「コンシューマレプリカを有効にする」を参照してください。

   c. (省略可能) コンシューマの詳細設定を行います。
   250 ページの「コンシューマの詳細設定を行う」を参照してください。

2 ハブの設定が必要な場合は、すべてのサーバーで次の手順を実行し、ハブのレプリカサフィックスをサーバー上に作成します。

   a. ハブのレプリカサフィックス用の空のサフィックスを作成します。
   252 ページの「ハブのレプリカサフィックスを作成する」を参照してください。

   b. ハブのレプリカサフィックスを有効にします。
   252 ページの「ハブレプリカを有効にする」を参照してください。

c. (省略可能) ハブの詳細設定を行います。
252 ページの「ハブレプリカの更新履歴ログ設定を変更する」を参照してください。

3 すべてのサーバーで次の手順を実行し、マスターのレプリカサフィックスをサーバー上に作成します。

a. マスターのレプリカサフィックス用のサフィックスを作成します。
253 ページの「マスターレプリカのサフィックスを作成する」を参照してください。

b. マスターのレプリカサフィックスを有効にします。
253 ページの「マスターレプリカを有効にする」を参照してください。

c. (省略可能) マスターの詳細設定を行います。
254 ページの「マスターレプリカの更新履歴ログ設定を変更する」を参照してください。

注 – レプリケーションアグリーメントを作成する前に、すべてのレプリカを有効にし、レプリケーションアグリーメントの作成後すぐにコンシューマレプリカを初期化できるようにします。コンシューマの初期化は、常にレプリケーションの設定の最後の段階で実行します。

---

4 レプリケーションマネージャーの設定が完了していることを確認します。

- デフォルトのマネージャーを使用する場合は、すべてのサーバーでデフォルトのレプリケーションマネージャーのパスワードを設定します。256 ページの「デフォルトのレプリケーションマネージャーパスワードを変更する」を参照してください。
- デフォルト以外のレプリケーションマネージャーを使用する場合は、すべてのサーバーで代わりのレプリケーションマネージャーエントリを定義します。254 ページの「デフォルト以外のレプリケーションマネージャーの使用」を参照してください。

5 次のようにして、すべてのマスターレプリカにレプリケーションアグリーメントを作成します。

a. マルチマスタートポロジのマスター間

b. マスターと専用コンシューマの間

c. マスターとハブレプリカの間

257 ページの「レプリケーションアグリーメントの作成と変更」を参照してください。

6 (省略可能) 部分レプリケーションを使用する場合は、ここで設定します。
258 ページの「部分レプリケーション」を参照してください。

7 (省略可能) レプリケーションの優先順位を使用する場合は、ここで設定します。
260 ページの「レプリケーションの優先順位」を参照してください。

8 ハブレプリカとそのコンシューマとの間のレプリケーションアグリーメントを設定します。
257 ページの「レプリケーションアグリーメントの作成と変更」を参照してください。

9 マルチマスターレプリケーションでは、データのオリジナルコピーを含むマスターレプリカから順にすべてのマスターを初期化します。
261 ページの「レプリカの初期化」を参照してください。

10 ハブとコンシューマレプリカを初期化します。
261 ページの「レプリカの初期化」を参照してください。

専用コンシューマ上でのレプリケーションの有効化

専用コンシューマは、レプリケートされたサフィックスの読み取り専用コピーです。専用コンシューマは、レプリケーションマネージャーとしてバインドされたサーバーから更新を受け取り、変更を行います。コンシューマサーバーの設定では、レプリカサフィックス用に空のサフィックスを準備し、そのサフィックスのレプリケーションを有効にします。オプションの詳細設定では、リフェラルの設定、削除の遅延の変更、プロパティーの変更などが含まれます。

次の節では、サーバー上で、専用コンシューマ用にレプリケートされたサフィックスを設定する方法について説明します。専用コンシューマ用にレプリケートされたサフィックスを設定したいすべてのサーバーに対して、同じ手順を繰り返してください。

▼ コンシューマのレプリカサフィックスを作成する

- 空のサフィックスをまだ作成していない場合は、レプリケーションの対象となるマスターレプリカと同じ **DN** を使用してコンシューマに空のサフィックスを作成します。
  手順については、61 ページの「サフィックスの作成」を参照してください。

注意 – すでにサフィックスが存在し、それが空でない場合は、マスターからレプリケートされたサフィックスが初期化されたときにそのサフィックスの内容は失われます。

▼ コンシューマレプリカを有効にする

空のサフィックスを作成したら、コンシューマのレプリカサフィックスを有効にする必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- コンシューマのレプリカサフィックスを有効にします。

```
$ dsconf enable-repl -h host -p port consumer suffix-DN
```

次に例を示します。

```
$ dsconf enable-repl -h host1 -p 1389 consumer dc=example,dc=com
```

▼ コンシューマの詳細設定を行う

高度な機能を使用するため、コンシューマのレプリカサフィックスを設定する場合は、ここで実行します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. リフェラルに **SSL** を使用する場合は、セキュリティー保護されたリフェラルを設定します。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN referral-url:ldaps://servername:port
```

次に例を示します。

```
$ dsconf set-suffix-prop -h host1 -p 1389 dc=example,dc=com \
 referral-url:ldaps://server2:2389
```

レプリケーションのメカニズムでは、レプリケーショントポロジに含まれるすべての既知のマスターのリフェラルを返すようにコンシューマを自動的に設定します。これらのデフォルトリフェラルは、クライアントが標準的な接続で簡単な認証を使うことを前提としています。安全な接続のために SSL を使用してマスターにバインドするオプションをクライアントに提供するには、`ldaps://` *servername* `:` *port* という

形式でリフェラルを追加します。*port* にはセキュリティー保護された接続に使うポート番号を指定します。マスターがセキュリティー保護された接続のみに設定されていれば、URL はデフォルトでセキュリティー保護されたポートを指定します。

リフェラルとして 1 つまたは複数の LDAP URL を追加したときは、コンシューマがマスターレプリカのリフェラルではなく、これらの LDAP URL のリフェラルを送信するようにすることができます。たとえば、クライアントが常にデフォルトのポートではなく、マスターサーバーのセキュリティー保護されたポートを参照するようにするとします。これらのセキュリティー保護されたポートの LDAP URL のリストを作成し、これらのリフェラルを使用して、プロパティーを設定します。すべての更新を処理する特定のマスターまたは Directory Server プロキシを指定する場合も、排他的なリフェラルを使用することができます。

2 コンシューマのレプリケーション削除の遅延を変更する場合は、次のコマンドを使用します。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN repl-purge-delay:time
```

たとえば、削除の遅延を 2 日に設定するには、次のように入力します。

```
$ dsconf set-suffix-prop -h host1 -p 1389 edc=example,dc=com repl-purge-delay:2d
```

コンシューマサーバーは、レプリケートされたサフィックスの内容に加えられる変更に関する内部情報を格納し、削除の遅延はこの情報を保持する期間を決定します。削除の遅延は、コンシューマとマスター間のレプリケーションを中断して、なお正常に回復できる期間を決定する要素の一つとなります。この値は、サプライヤサーバーの更新履歴ログの `MaxAge` パラメータと関連付けられています。これらの 2 つのパラメータのうち、短いほうの設定が、2 つのサーバー間のレプリケーションが無効になった、またはダウンした場合でも正常な状態に復元できる最長期間を決定します。ほとんどの場合には、デフォルトの 7 日間が適当です。

ハブ上でのレプリケーションの有効化

ハブレプリカは、コンシューマとしてだけではなく、マスターとしても機能し、レプリケートされたデータをより多くのコンシューマに配信します。このため、レプリケーションの更新をそれぞれのサプライヤから受信して、レプリケーションの更新をそれぞれのコンシューマに送信します。ハブレプリカは変更を受け付けませんが、マスターにリフェラルを返します。

ハブサーバーの設定では、レプリカサフィックス用に空のサフィックスを準備し、そのサフィックスのレプリケーションを有効にします。必要に応じて、異なるレプリケーションマネージャーの選択、リフェラルの設定、削除の遅延の設定、更新履歴ログパラメータの変更など、詳細な設定も行うことができます。

次の節では、1 つのハブサーバーを設定する方法を説明します。ハブのレプリカサフィックスを含むすべてのサーバーで、同じ手順を繰り返してください。

▼ ハブのレプリカサフィックスを作成する

- 空のサフィックスをまだ作成していない場合は、レプリケーションの対象となるマスターレプリカと同じ **DN** を使用してハブサーバーに空のサフィックスを作成します。

  手順については、61 ページの「サフィックスの作成」を参照してください。

  すでにサフィックスが存在し、それが空でない場合は、マスターからレプリケートされたサフィックスが初期化されたときにそのサフィックスの内容は失われます。

▼ ハブレプリカを有効にする

ハブレプリカがある場合は、それらをここで有効にします。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ハブのレプリカサフィックスを有効にします。

  ```
  $ dsconf enable-repl -h host -p port hub suffix-DN
  ```

  次に例を示します。

  ```
  $ dsconf enable-repl -h host1 -p 1389 hub dc=example,dc=com
  ```

▼ ハブレプリカの更新履歴ログ設定を変更する

ハブの詳細設定で、変更する必要があるパラメータは更新履歴ログに関するものだけです。サプライヤとして、ハブサーバーは更新履歴ログが必要です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ハブの更新履歴ログ設定を変更するには、次のいずれかのコマンドを使用します。

  ```
  $ dsconf set-server-prop -h host -p port suffix-DN repl-cl-max-age:value
  ```

  ```
  $ dsconf set-server-prop -h host -p port suffix-DN repl-cl-max-entry-count:value
  ```

マスターレプリカ上でのレプリケーションの有効化

マスターレプリカにはデータのマスターコピーが含まれ、更新を他のすべてのレプリカに配信する前に、すべての変更を集中的に管理します。マスターはすべての変更を記録し、関連する各コンシューマの状態を確認して、必要に応じて更新を送信します。マルチマスターレプリケーションでは、マスターレプリカが他のマスターから更新を受け取ることもあります。

マスターサーバーを設定するときは、マスターレプリカを含むサフィックスを決定し、マスターレプリカを有効にします。また、必要に応じてレプリケーションの詳細設定を行います。

次の節では、1 つのマスターサーバーを設定する方法を説明します。特定のマスターのレプリカサフィックスを含むすべてのサーバーで、同じ手順を繰り返してください。

▼ マスターレプリカのサフィックスを作成する

- レプリケートするエントリを保存するマスターサーバー上でサフィックスを選択、または作成します。

  手順については、61 ページの「サフィックスの作成」を参照してください。

  マルチマスター設定と初期化を正しく実行するため、データを含む 1 つのマスターのみを読み込みます。他のレプリケートされたサフィックスのデータは上書きされます。

▼ マスターレプリカを有効にする

マスターのレプリケーションを有効にする場合は、レプリケーション ID を割り当てる必要があります。レプリケーション ID は、更新ステートメントの所有者を区別し、マルチマスターレプリケーションで発生する可能性のある競合を解決するために使用します。そのため、このサフィックスのすべてのマスターレプリカで、レプリケーション ID が一意である必要があります。レプリケーション ID は設定すると変更できません。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- マスターのレプリカサフィックスを有効にします。

```
$ dsconf enable-repl -h host -p port -d ReplicaID master suffix-DN
```

ReplicaID は 1 から 65534 までの整数です。

たとえば、レプリカ ID 1 でマスターのレプリカサフィックスを作成するには、次のコマンドを使用します。

```
$ dsconf enable-repl -h host1 -p 1389 -d 1 master dc=example,dc=com
```

▼ マスターレプリカの更新履歴ログ設定を変更する

マスターの詳細設定では、更新履歴ログ設定を変更する場合があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- マスターの更新履歴ログ設定を変更する場合は、次のいずれかのコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port suffix-DN repl-cl-max-age:value
```

```
$ dsconf set-server-prop -h host -p port suffix-DN repl-cl-max-entry-count:value
```

レプリケーションマネージャーの設定

この節では、デフォルト以外のレプリケーションマネージャーを設定する方法とデフォルトのレプリケーションマネージャーパスワードを設定する方法を説明します。

デフォルト以外のレプリケーションマネージャーの使用

レプリケーションマネージャーは、サプライヤがレプリケーション更新を送信する場合に、コンシューマサーバーにバインドするために使用するユーザーです。更新を受け取るサフィックスを持つすべてのサーバーでは、少なくとも 1 つのレプリケーションマネージャーエントリが必要です。

Directory Server にはデフォルトのレプリケーションマネージャーエントリがあり、特に単純なレプリケーション事例の場合に、すべてのサーバーで使用できます。`cn=replication manager,cn=replication,cn=config`。レプリケーションメカニズムは、このユーザーを使用してコンシューマレプリカを自動的に設定するので、レプリカを簡単に配備できます。

複雑なレプリケーション事例の場合は、レプリケートされたサフィックスごとに異なるパスワードを持つ複数のレプリケーションマネージャーが必要になることがあります。既存のデフォルトのレプリケーションマネージャーを 1 つまたは複数の新しいレプリケーションマネージャーで置き換えることができます。

注意 – レプリケーションマネージャーエントリの DN とパスワードを使用して、バインドを実行したり、サーバー上で処理を行うことはできません。レプリケーションマネージャーはレプリケーションメカニズムでのみ使用します。他の用途では、レプリカの再初期化が必要になることがあります。

ディレクトリマネージャーをレプリケーションマネージャーとして使用することはできません。cn=admin,cn=Administrators,cn=config エントリは、他の管理作業でも使用するため、管理者グループのこのユーザーまたは他のすべてのユーザーもレプリケーションマネージャーとして使用できません。

各コンシューマのレプリケーションマネージャーを選択したら、選択または作成したレプリケーションマネージャーの DN を覚えておきます。この DN とそのパスワードは、あとからこのコンシューマのサプライヤとの間でレプリケーションアグリーメントを作成するときに必要です。

▼ デフォルト以外のレプリケーションマネージャーを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 すべてのコンシューマ (ターゲット) のレプリカサフィックスに対して、新しいレプリケーションマネージャーとパスワードを作成します。

```
$ ldapmodify -a -h host -p port -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn:"cn=new-replication-manager,cn=replication,cn=config"
objectclass: top
objectclass: person
userpassword:password
sn:new-replication-manager
```

次に例を示します。

```
$ ldapmodify -a -h host1 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn:"cn=ReplicationManager3,cn=replication,cn=config"
objectclass: top
objectclass: person
userpassword:secret
sn:ReplicationManager3
```

2 すべてのコンシューマ (ターゲット) のレプリカサフィックスに対して、レプリケーションマネージャーバインド **DN** を設定します。

```
$ dsconf set-suffix-prop -h host -p port suffix-DN \
 repl-manager-bind-dn:"cn=new-replication-manager,cn=replication,cn=config"
```

次に例を示します。

```
$ dsconf set-suffix-prop -h host1 -p 1389 dc=example,dc=com \
 repl-manager-bind-dn:"cn=ReplicationManager3,cn=replication,cn=config"
```

3 すべてのサプライヤ(ソース)のレプリカサフィックスに作成したすべてのレプリケーションアグリーメントに、レプリケーションマネージャーバインド **DN** を設定します。

a. 新しいレプリケーションマネージャーパスワードを設定する一時ファイルを作成します。

このファイルが一度読み取られ、パスワードは将来使用するために格納されます。

```
$ echo password > password-file
```

b. 更新の実行時に、レプリケーションメカニズムで使用するレプリケーションマネージャーバインド **DN** とパスワードを設定します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN host:port \
 auth-bind-dn:"cn=new-replication-manager,cn=replication,cn=config" \
 auth-pwd-file:password-file
```

次に例を示します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389 \
 auth-bind-dn:"cn=ReplicationManager3,cn=replication,cn=config" \
 auth-pwd-file:pwd.txt
```

c. 一時パスワードファイルを削除します。

```
$ rm password-file
```

▼ デフォルトのレプリケーションマネージャーパスワードを変更する

1 レプリケーションマネージャーパスワードを設定するための一時ファイルを作成します。

このファイルが一度読み取られ、パスワードは将来使用するために格納されます。

```
$ echo password > password-file
```

2 レプリケーショントポロジのすべてのコンシューマ(ターゲット)サーバーに、レプリケーションマネージャーバインドパスワードを設定します。

```
$ dsconf set-server-prop -h host -p port def-repl-manager-pwd-file:password-file
```

次に例を示します。

```
$ dsconf set-server-prop -h host1 -p 1389 def-repl-manager-pwd-file:pwd.txt
```

3 一時パスワードファイルを削除します。

```
$ rm password-file
```

レプリケーションアグリーメントの作成と変更

レプリケーションアグリーメントはサプライヤのパラメータのセットで、指定したコンシューマに更新を送信する方法を設定し、制御します。コンシューマに更新を送信するサプライヤのレプリカサフィックスには、レプリケーションアグリーメントを作成する必要があります。更新するすべてのコンシューマのサプライヤにレプリケーションアグリーメントを作成する必要があります。

▼ レプリケーションアグリーメントを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用して、新しいレプリケーションアグリーメントを作成する場合、既存のレプリケーションアグリーメントから一部またはすべてのレプリケーションアグリーメント設定をコピーすることができます。

1 マスターサーバーから、レプリケートする各コンシューマのレプリケーションアグリーメントを作成します。

```
$ dsconf create-repl-agmt -h host -p port suffix-DN consumer-host:consumer-port [consumer-host:consumer-port]
```

次に例を示します。

```
$ dsconf create-repl-agmt -h host1 -p 1389 dc=example,dc=com host2:1389
```

コマンド行を使用して、既存のレプリケーションアグリーメントを表示するには、`dsconf list-repl-agmts` コマンドを使用します。

注 – レプリケーションの実行中にマスター上のポート番号を変更しても、サーバーを初期化し直す必要はありません。ただし、古いアドレス (*host*: *old-port*) をポイントしていた古いレプリケーションアグリーメントは使用できなくなります。ポート番号を変更する前と同じようにレプリケーションを続行させる場合は、新しいアドレス (*host*:*new-port*) で新しいアグリーメントを作成する必要があります。

---

2 レプリケーションアグリーメントが正しく作成されていることを確認します。

```
$ dsconf show-repl-agmt-status -h host -p port suffix-DN consumer-host:consumer-port
```

3 認証状態が **OK** でない場合は、`dsconf accord-repl-agmt` コマンドを実行します。

注 – コマンド `dsconf accord-repl-agmt` は、デフォルトのレプリケーションマネージャーを使用する場合にのみ使用します。新しいレプリケーションマネージャーを作成した場合は、このコマンドによって、必要な設定が上書きされるため、このコマンドを使わないでください。

`dsconf accord-repl-agmt` コマンドにより、サプライヤとターゲットサーバーの両方が同じレプリケーション認証設定を共有します。

```
$ dsconf accord-repl-agmt -h host -p port suffix-DN consumer-host:consumer-port
```

次に例を示します。

```
$ dsconf accord-repl-agmt -h host2 -p 1389 dc=example,dc=com host1:1389
```

▼ レプリケーションアグリーメントの対象を変更する

この手順では、既存のレプリケーションアグリーメントで指定されたリモートレプリカを変更します。既存のアグリーメントのサフィックス DN と設定は同じままです。

- レプリケーションアグリーメントのリモートレプリカのホスト名とポート番号を変更します。

  ```
  $ dsconf change-repl-dest -h host -p port suffix-DN host:port new-host:new-port
  ```

  このコマンドで `-A` *protocol* オプションを指定して実行すると、レプリケーションで使用される認証プロトコルを変更できます。

部分レプリケーション

デフォルトでは、レプリケーション操作によって、レプリケートされたサフィックスに含まれるすべてのエントリがコンシューマレプリカにコピーされます。部分レプリケーション機能を用いると、選択したサフィックスのどの属性をレプリケーションの対象とし、どの属性を対象外とするかを選択できます。部分レプリケーションはレプリケーションアグリーメントに設定されるので、マスターのコン

シューマのレプリカサフィックスごとに属性セットを定義できます。配信するデータを制御し、レプリケーションの帯域幅とコンシューマリソースをより効率的に利用できます。

たとえば、photo、jpegPhoto、audio のように一般に値が大きい属性をレプリケーションの対象外とすることで、レプリケーションの帯域幅を節約できます。この場合、コンシューマではこれらの属性を利用できなくなります。また、認証に必要な uid 属性と userpassword 属性だけをコンシューマサーバーにレプリケートすることもできます。

部分レプリケーションに関する注意点

注 – 部分レプリケーションは、Directory Server 5.2 より前のバージョンの製品では使用できません。部分レプリケーションアグリーメントを設定する場合は、マスターとコンシューマレプリカが共に Directory Server 5.2 以上を使用している必要があります。

属性の部分的なセットを有効化または変更するには、コンシューマレプリカを初期化し直す必要があります。このため、配備前に部分レプリケーションの必要性を検討し、レプリケートされたサフィックスを初期化する前に属性セットを設定しておく必要があります。

ACI、ロール、CoS などの複雑な機能が特定の属性に依存する小規模な属性セットをレプリケートするときは、慎重な対応が必要です。さらに、ACI、ロール、または CoS メカニズムの指示子やフィルタで示されているその他の属性をレプリケートしないと、データのセキュリティーが損なわれる可能性があります。レプリケートしないと、検索で返される属性セットが異なる可能性もあります。レプリケーションの対象に含める属性のリストを管理するよりも、除外する属性のリストを管理する方法が安全であり、人的なミスも少なくなります。

レプリケートするすべてのエントリがスキーマに準拠しない属性セットをレプリケートするときは、コンシューマサーバーのスキーマチェックを無効にする必要があります。スキーマに準拠しないエントリをレプリケートしても、レプリケーションメカニズムはコンシューマ上でのスキーマチェックを行わないため、エラーは発生しません。しかし、スキーマに準拠しないエントリがコンシューマに含まれるようになるので、クライアントにとって一貫した状態にする必要があるためスキーマチェックを無効にする必要があります。

部分レプリケーションは、ハブと専用コンシューマのマスターレプリカのレプリケーションアグリーメントに設定します。マルチマスターレプリケーション環境の 2 つのマスターレプリカ間での部分レプリケーションは設定できません。また、複数のマスターが同じレプリカとの間でレプリケーションアグリーメントを持つ場合、同じ属性セットをレプリケートするようにすべてのアグリーメントを設定する必要があります。

▼ 部分レプリケーションを設定する

部分レプリケーションを設定するには、サフィックスを指定し、そのサフィックスの属性を含めるか除外するかを決定して、含めるかまたは除外する属性を選択します。サフィックスの属性を除外するように選択すると、他のすべての属性が自動的に含まれます。同様に、サフィックスの特定の属性を含めるように選択すると、他のすべての属性が自動的に除外されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ソースサーバーに存在するレプリケーションアグリーメントに部分レプリケーションを設定します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN consumer-host:consumer-port property:value
```

property は `repl-fractional-exclude-attr` または `repl-fractional-include-attr` です。

たとえば、JPEG と TIFF の写真をサフィックス `dc=example,dc=com` でレプリケートされる属性から除外する部分アグリーメントを設定する場合、次のコマンドを使用します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389
 repl-fractional-exclude-attr:jpegPhoto repl-fractional-exclude-attr:tiffPhoto
```

除外する属性の既存リストに属性を追加するには、次のコマンドを使用します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN consumer-host:consumer-port repl-fractional-exclude-attr+:attribute
```

レプリケーションの優先順位

レプリケーションの優先順位の指定は必須の作業ではありません。ユーザーパスワードの更新などの特定の変更を高い優先順位でレプリケートするように指定するレプリケーションルールを作成できます。レプリケーションルールに指定されたすべての変更は、高い優先順位でレプリケートされ、ほかのすべての変更は、通常の優先順位でレプリケートされます。

注 – レプリケーションの優先順位ルールは、マスターサーバーにのみ作成する必要があります。ハブやコンシューマの設定は必要ありません。

▼ レプリケーションの優先順位を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- マスターに新しいレプリケーションの優先順位ルールを作成するには、次のコマンドを使用します。

  ```
  $ dsconf create-repl-priority -h host -p port suffix-DN priority-name property:value
  ```

  次のプロパティーを使用して、レプリケーションの優先順位を設定できます。

  - 操作のタイプ、`op-type`
  - バインド DN、`bind-dn`
  - ベース DN、`base-dn`
  - 属性タイプ、`attr`

  priority-name はユーザーが自由に定義できます。

  たとえば、ユーザーパスワードの変更を高い優先順位でレプリケートするように指定するレプリケーションルールを作成するには、次のコマンドを使用します。

  ```
  $ dsconf create-repl-priority -h host2 -p 1389 dc=example,dc=com pw-rule \
   attr:userPassword
  ```

  現在のレプリケーションルールを表示するには、`dsconf list-repl-priorities -v` コマンドを使用します。このコマンドを `-v` オプションを付けて使用した場合、優先順位付きレプリケーションルールに関する追加情報が表示されます。

  ```
  $ dsconf list-repl-priorities -h host2 -p 1389 -v
  ```

  詳細は、`dsconf`(1M) のマニュアルページを参照してください。

レプリカの初期化

レプリケーションアグリーメントを作成し、両方のレプリカを設定したら、レプリケーションを開始する前に、コンシューマのレプリカサフィックスを初期化する必要があります。初期化時は、サプライヤのレプリカサフィックスからコンシューマのレプリカサフィックスにデータが物理的にコピーされます。

さらに、特定のエラーが発生した場合、または設定を変更した場合は、レプリカを初期化し直す必要があります。たとえば、何らかの理由で 1 つのマスターのレプリカサフィックスをバックアップから復元した場合、そのレプリカが更新するすべてのレプリカを初期化し直す必要があります。

初期化し直したときは、コンシューマ側のレプリケートされたサフィックスは削除され、マスター側のサフィックスの内容に置き換えられます。これにより、レプリ

カの同期が確保され、レプリケーションの更新が再開されます。この節で説明するどの方法で初期化を行なっても、コンシューマレプリカのインデックスは自動的にふたたび作成されるため、クライアントからの読み取り要求にもただちに正しく対応できます。

マルチマスターレプリケーションでは、トポロジのほかのマスターによって更新されたコンシューマであれば、初期化し直す必要がない場合もあります。

▼ レプリケートされたサフィックスをリモート(サプライヤ)サーバーから初期化する

既存のレプリケーションアグリーメントを使用して、リモートサーバーからサフィックスを初期化できます。この初期化方法は、他の方法より簡単なため、可能な限りこの方法を使用します。データが大量でインポートに時間がかかりすぎる場合にのみ他の方法を使用してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用したレプリケートされたサフィックスのオンライン初期化は、コンシューマを初期化または再初期化する簡単な方法です。ただし、大量のエントリを初期化する場合、この処理には時間がかかることがあります。この場合は、コマンド行によるコンシューマのオフライン初期化の方が効率的な場合があります。

1 レプリカを初期化します。

```
$ dsconf init-repl-dest -h host -p port suffix-DN destination-host:destination-port [destination-host:destination-port]
```

destination-host:*destination-port* はホストおよびターゲットサーバーのポートで、リモートサーバーから初期化します。

2 (省略可能)各アグリーメントで、サフィックスが初期化済みとなっていることを確認します。

```
$ dsconf show-repl-agmt-status -h host -p port suffix-DN destination-host:destination-port
```

LDIF からのレプリカの初期化

▼ LDIF からレプリケートされたサフィックスを初期化する

次の手順では、LDIF ファイルからレプリケートされたサフィックスを初期化するために使用する一般的な手順を説明します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用したレプリケートされたサフィックスのオンライン初期化は、コンシューマを初期化または再初期化する簡単な方法です。ただし、大量のエントリを初期化する場合、この処理には時間がかかることがあります。この場合は、コマンド行によるコンシューマのオフライン初期化の方が効率的な場合があります。

1 レプリケーションアグリーメントを設定していることを確認します。

この操作は、レプリカを初期化する前に実行する必要があります。

2 マスターのレプリカサフィックスから、元のサフィックスデータのコピーを **LDIF** ファイルにエクスポートします。

264 ページの「レプリケートされたサフィックスを LDIF にエクスポートする」を参照してください。

マルチマスターレプリケーション環境では、オリジナルマスターからエクスポートした LDIF ファイルを使用して他のマスターとコンシューマの両方を初期化できます。カスケード型のレプリケーションでは、同じファイルを使用してハブレプリカとそのコンシューマを初期化できます。

どの場合にも、設定が完了しているマスターレプリカからエクスポートした LDIF ファイルから開始する必要があります。これ以外の任意の LDIF ファイルにはレプリケーションメタデータが含まれないため、これを使用してすべてのレプリカを初期化することはできません。

3 部分レプリカを初期化する場合、ファイルをフィルタして、レプリケートされる属性のみを維持し、そのファイルをすべてのコンシューマサーバーに転送します。

265 ページの「部分レプリケーションのための LDIF ファイルのフィルタリング」を参照してください。

4 レプリカを初期化します。

次のいずれかの操作を行います。

- オフライン (停止している) サーバーで高速に初期化する場合、`dsadm import` コマンドを使用します。

  ```
  $ dsadm import instance-path LDIF_file suffix-DN
  ```

- LDIF ファイルからレプリカをオンラインで初期化するには、 `dsconf import` コマンドを使用します。

  ```
  $ dsconf import -h host -p port LDIF_file suffix-DN
  ```

  `dsconf import` を使用すると、`dsadm import` を使用した場合よりも遅くなりますが、インポート操作中にサーバーを停止する必要がありません。

サフィックスの初期化の詳細と例については、212 ページの「サフィックスの初期化」を参照してください。コマンドの詳細な使い方については、`dsadm`(1M) および `dsconf`(1M) を参照してください。

5 (省略可能) 各アグリーメントで、サフィックスが初期化済みとなっていることを確認します。

```
$ dsconf show-repl-agmt-status -h host -p port suffix-DN destination-host:destination-port
```

▼ レプリケートされたサフィックスを LDIF にエクスポートする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 次のいずれかのコマンドを使用して、レプリケートされたサフィックスの内容を **LDIF** ファイルにエクスポートします。

  - オフラインエクスポートの場合、次のように入力します。

    ```
    $ dsadm export instance-path suffix-DN LDIF_file
    ```

  - オンラインエクスポートの場合、次のように入力します。

    ```
    $ dsconf export -h host -p port suffix-DN LDIF_file
    ```

  次の例では、レプリケートされたサフィックス `dc=example,dc=com` 全体とレプリケーション情報をファイル `example_replica_export.ldif` にエクスポートします。

  ```
  $ dsconf export -h host2 -p 1389 dc=example,dc=com  \
   /local/ds/ldif/example_export_replica.ldif
  ```

  詳細については、208 ページの「LDIF へのバックアップ」 および `dsadm`(1M) および `dsconf`(1M) のマニュアルページを参照してください。

部分レプリケーションのための LDIF ファイルのフィルタリング

DSCC を使った場合、部分レプリケーションが設定されたレプリカの初期化は透過的に行われます。初期化時に、選択されている属性だけがコンシューマに送られます。

部分レプリケーションを設定した場合、エクスポートされた LDIF ファイルをコンシューマサーバーにコピーする前に、未使用の属性をフィルタで除外します。Directory Server にはこの目的で `fildif` ツールがあります。このツールは、指定した LDIF ファイルをフィルタリングし、レプリケーションアグリーメントに定義されている属性セットが許可する属性だけを残します。

このツールはサーバーの設定を読み取り、属性セットの定義を決定します。設定ファイルを読み取るには、`fildif` ツールを root として実行するか、プロセスおよびファイルを所有するユーザー (`nsslapd-localuser` 属性によって指定) として実行する必要があります。たとえば、次のコマンドは、前の例で `dc=example,dc=com` サフィックスからエクスポートされたファイルをフィルタリングします。

```
$ fildif -i /local/ds1/ldif/example_master.ldif \
 -o /local/ds1/ldif/filtered.ldif -b "cn=host2.example.com:1389, \
 cn=replica,cn=\\"dc=example,dc=com\\",cn=mapping tree,cn=config" -p /local/ds1
```

`fildif` コマンドの場所については、35 ページの「コマンドの場所」を参照してください。

`-i` オプションと `-o` オプションは、それぞれ入力ファイルと出力ファイルです。`-b` オプションは、部分レプリケーションが定義されているレプリケーションアグリーメントの DN です。次のコマンドを使用して、この DN を検索できます。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -b "cn=config" "(&(objectclass=nsds5replicationagreement) (nsDS5ReplicaPort=replica-port) \
 (nsDS5ReplicaHost=replica-host))" dn
```

次に例を示します。

```
$ ldapsearch -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w - \
 -b "cn=config" "(&(objectclass=nsds5replicationagreement) \
 (nsDS5ReplicaPort=2090)(nsDS5ReplicaHost=host2))" dn
Enter bind password:
version: 1
dn: cn=host2:1389,cn=replica,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config
```

`fildif` ツールのすべてのコマンド行構文については、`fildif`(1) のマニュアルページを参照してください。

`fildif` ツールを使用して作成した `filtered.ldif` ファイルを使用して、このレプリケーションアグリーメントの対象となるコンシューマを初期化できます。このファイルをコンシューマサーバーに転送し、211 ページの「LDIF ファイルからのデータのインポート」で説明するとおりにインポートします。

バイナリコピーを使用したレプリケートされたサフィックスの初期化

バイナリコピーにより、サーバーのバイナリバックアップファイルを使用して、別のサーバーに同じディレクトリの内容を復元することで、サーバー全体のクローンを作成できます。バイナリコピーを使用して、マスターまたはハブサーバーのバイナリコピーから任意のサーバーを初期化または再初期化できます。または別のコンシューマサーバーのバイナリコピーからコンシューマを初期化または再初期化できます。

注 – この高度な手順では、Directory Server 上のデータベースファイルとの間で情報をやり取りします。この機能は、経験が豊富な管理者以外は使用しないでください。

この機能にはある種の制限が適用されるため、処理時間の短縮を見込めるのは、たとえば百万件単位のエントリを含むレプリカなど、大容量のデータベースファイルを持つレプリカだけです。

レプリケーションでのバイナリコピーの使用の制限

バイナリコピーは、あるマシンから別のマシンにデータベースファイルを移動するため、次の制限が厳密に適用されます。

- 両方のマシンが同じオペレーティングシステム (サービスパックやパッチも含む) を実行している必要があります。
- 両方のマシンで同じプロセッサアーキテクチャーを使用している必要があります。たとえば、2 台の UltraSPARC® T1 プロセッサ間でバイナリコピーを実行できますが、UltraSPARC T1 プロセッサと AMD Opteron プロセッサ間では実行できません。
- 両方のマシンがビッグエンディアンかリトルエンディアンである必要があります。
- 両方のマシンがメモリーを同じようにマップしている必要があります。たとえば、2 台の 64 ビットシステム上のサーバーインスタンス間でバイナリコピーを実行できますが、32 ビットシステムのサーバーインスタンスと、64 ビットシステムの別のサーバーインスタンス間では実行できません。
- 両方のマシンに同じバージョンの (バイナリ形式 (32 ビットまたは 64 ビット)、サービスパック、パッチも含まれる) Directory Server がインストールされている必要があります。
- 両方のサーバーは、同じサフィックスに分岐する同じディレクトリツリーを持つ必要があります。すべてのサフィックスのデータベースファイルを一緒にコピーする必要があります。サフィックスを個別にコピーすることはできません。

- 両方のサーバーの各サフィックスには、同じインデックス (VLV (仮想リスト表示) インデックスも含む) が設定されている必要があります。サフィックスのデータベースの名前を同じにする必要があります。
- 各サーバーに、レプリカとして同じサフィックスが設定されている必要があります。
- 部分レプリケーションが設定されている場合は、すべてのサーバーが同じように設定されている必要があります。
- どちらのサーバーでも、属性の暗号化は使用できません。
- 属性値の一意性プラグインが有効な場合は、両方のサーバーで同じ設定にします。また、次の手順で、新しいコピーを設定し直す必要があります。

  以下の手順では、バイナリコピーを実行する別の方法について説明します。サーバーを停止する必要がないバイナリコピーと使用ディスク容量が最小ですむバイナリコピー。

サーバーを初期化するためのバイナリコピーの作成

この節では、サーバーを初期化するためのバイナリコピーの作成方法と、使用ディスク容量が最小になるバイナリコピーの作成方法について説明します。

▼ サーバーを初期化するためのバイナリコピーを作成する

次の手順を使用して、レプリケートするサーバーを初期化するためのバイナリコピーを実行します。通常のバックアップ機能を使用して、サーバーのデータベースファイルのコピーを作成するためです。通常のバックアップを実行することで、サーバーを停止しなくても、すべてのデータベースファイルを一定の状態に維持できます。

この手順には特定の制限があります。バックアップと復元の処理によって、同じマシンにデータベースファイルのコピーが作成されるため、各マシンでこれらのファイルが占有するディスクスペースの容量が 2 倍になります。また、これらのファイルに対する実際のコピー処理は、ディレクトリに G バイト単位のデータが含まれる場合、時間がかかることがあります。

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1 新しくレプリケートされたサフィックスのターゲットマシンに **Directory Server** をインストールし、必要に応じてサーバーの新しいインスタンスを作成して、**266** ページの「レプリケーションでのバイナリコピーの使用の制限」に従って、サーバーを設定します。

2 このレプリケートされたサフィックスに関連するレプリケーショントポロジにすべてのレプリケーションアグリーメントを作成します。

サプライヤからこのレプリカにアグリーメントを含めます。このレプリカが専用コンシューマでない場合は、このレプリカからそのコンシューマにアグリーメントを含めます。257 ページの「レプリケーションアグリーメントの作成と変更」を参照してください。

3 初期化するレプリカと同じ種類 (マスター、ハブ、コンシューマのいずれか) の、完全に設定され、初期化されたレプリカを選択し、**205** ページの「バイナリバックアップ」の手順に従って通常のバックアップ処理を行います。

4 バックアップディレクトリからターゲットマシンのディレクトリにファイルをコピーまたは転送します。この操作には、`ftp` コマンドなどを使います。

5 マルチマスターレプリケーションの状況で新しいマスターを初期化した場合、**218** ページの「マルチマスターモデルでのマスターの復元」の手順に従います。

▼ 最小のディスク容量でサーバーの初期化を行うためにバイナリコピーを使用する

この手順では、データベースファイルのバックアップコピーを作成しないため、ディスクスペースの消費が少なく、処理に要する時間も少なくなります。ただし、データベースファイルを一貫した状態に保つため、クローン作成の対象となるサーバーを停止する必要があります。

注意 – マルチマスターレプリケーションにすでに組み込まれているマスターの再初期化に、この手順を使うことはできません。この手順を利用できるのは、コンシューマサーバーの再初期化、または新しいマスターサーバーの初期化だけです。既存のマスターレプリカを再初期化するには、オンライン初期化を使用して、LDIF ファイルをインポートするか、267 ページの「サーバーを初期化するためのバイナリコピーの作成」の手順に従います。

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1 新しくレプリケートされたサフィックスのターゲットマシンに **Directory Server** をインストールし、必要に応じてサーバーの新しいインスタンスを作成して、**266** ページの「レプリケーションでのバイナリコピーの使用の制限」に従って、サーバーを設定します。

2. このレプリカに関連するレプリケーショントポロジにすべてのレプリケーションアグリーメントを作成します。

   サプライヤからこのレプリカにアグリーメントを含めます。このレプリカが専用コンシューマでない場合は、このレプリカからそのコンシューマにアグリーメントを含めます。257 ページの「レプリケーションアグリーメントの作成と変更」を参照してください。

3. 59 ページの「Directory Server インスタンスの起動、停止、および再起動」で説明するように、初期化または再初期化するターゲットサーバーを停止します。

4. 初期化するレプリカと同じ種類 (マスター、ハブ、コンシューマのいずれか) の、完全に設定され、初期化されたレプリカを選択し、このサーバーも停止します。

   マルチマスター設定に組み込まれているマスターレプリカのクローンを作成するときは、それを停止する前に、その他のマスターから最新のすべての変更が完全に反映されていることを確認する必要があります。

5. トランザクションログ、更新履歴ログ、地域ファイル (`__db.`*xxx* ファイル) など、すべてのデータベースファイルをターゲットサーバーから削除します。

   ファイルの位置を変更していないかぎり、データベースファイルとトランザクションログは *instance-path*`/db` ディレクトリに保存されています。

6. `ftp` コマンドなどを使用して、トランザクションログや更新履歴ログを含むすべてのデータベースファイルをソースレプリカマシンからターゲットマシンにコピーするか、転送します。

   ファイルの位置を変更していないかぎり、データベースファイルとトランザクションログは *instance-path*`/db` ディレクトリに保存されています。

   マスターまたはハブレプリカを初期化する場合は、更新履歴ログにあるすべてのファイルもコピーする必要があります。更新履歴ログはデフォルトで *instance-path*`/changelog` にあります。

7. ソースサーバーとターゲットサーバーの両方を再起動します。

カスケード型レプリケーションでのレプリカの初期化

カスケード型レプリケーションの場合、常に次の手順に示す順番でレプリカを初期化します。

▼ カスケード型レプリケーションでレプリカを初期化する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. マルチマスターレプリケーションもある場合、1 つのマスターにレプリケートするすべてのデータセットが存在することを確認し、このマスターを使用して、ほかの各マスターのレプリカを初期化します。
2. それぞれのマスターレプリカから、最初の階層のハブレプリカに属するレプリカを初期化します。
3. ハブの構成が複数の階層に分かれている場合、各階層を上から順に初期化していきます。
4. 最後の階層のハブレプリカから専用コンシューマのレプリカを初期化します。

レプリケートされたサフィックスのインデックス生成

インデックスは、別のサーバーインスタンスに自動的にレプリケートされません。レプリケートされたサフィックスを保持するすべてのサーバーインスタンスの属性のインデックスを生成するには、次のいずれかの操作を実行します。

- DSCC を使用して、レプリケートされたサフィックスを保持するすべてのサーバーインスタンスをサーバーグループとして管理します。グループ内の 1 つのサーバーにインデックスを追加し、「サーバーの設定のコピー」操作を使用して、インデックス設定をグループ内の他のサーバーにコピーします。

  DSCC の詳細については、45 ページの「Directory Service Control Center のインタフェース」を参照してください。
- 第 12 章に説明するように、`dsconf` コマンドを使用して、各サーバーインスタンスのインデックスを管理します。
- 266 ページの「バイナリコピーを使用したレプリケートされたサフィックスの初期化」に説明するように、バイナリコピーを使用して、サフィックスを初期化します。

大量のレプリケートされたサフィックスへの多数のエントリの段階的追加

きわめて大量のエントリを含むディレクトリがあり、大量のエントリを追加する場合、時間がかかりすぎるため、`ldapmodify -a` を使用しないでください。代わりに、`dsconf import` コマンドにレプリケートされるトポロジにエントリを追加するオプションを付けて実行し、新しいエントリを段階的に追加します。エントリをインポートすると、レプリケーションメタデータと共に追加エントリを含む LDIF ファイルが生成されます。次にこの生成された LDIF ファイルを他のレプリカにインポートします。生成された LDIF ファイルにより、データを追加するレプリカ全体で、レプリケーションの同期が定期的に行われます。

▼ 大量のレプリケートされたサフィックスに多数のエントリを追加する

始める前に　この手順により、大きな LDIF ファイルが生成されます。最初の `dsconf import` コマンドを実行する前に、生成される LDIF ファイル用に十分なディスク容量があることを確認します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

注意 – この手順を使用して、サーバーパスに大量のエントリがあるサーバーを初期化することができます。ただし、いずれかのインポートが失敗すると、データベース全体が失われる可能性があります。各インポートの前にデータをバックアップしてください。

1. マスターレプリカで、エントリをインポートします。

   ```
   $ dsconf import -h host -p port -K generated-LDIF-file suffix-DN
   ```

   `-K` オプションにより、既存のデータが削除されません。さらに、新しいエントリとレプリケーションプロセスに必要な情報を格納するファイル *generated-LDIF-file* も生成されます。

2. 他のすべてのレプリカで、前の手順で生成されたファイルをインポートします。

   ```
   $ dsconf import -h host -p port \
   -K -f incremental-output=no generated-LDIF-file suffix-DN
   ```

   オプション `-f incremental-output=no` は追加の LDIF ファイルを生成しないことを示します。この手順で必要となる生成された LDIF ファイルは 1 つだけです。

レプリケーションと参照の完全性

レプリケーションで参照の完全性プラグインを使用する場合、それをすべてのマスターサーバで有効にする必要があります。ハブサーバーまたはコンシューマサーバー上のプラグインを有効にする必要はありません。

次に、レプリケーション環境における参照の完全性の使用に関する制限を示します。

- このプラグインは、マスターレプリカを含むすべてのサーバーで有効にする必要があります。
- すべてのマスターで同じ設定でこのプラグインを有効にする必要があります。
- ハブまたはコンシューマレプリカだけを含むサーバーで有効にしても意味がありません。

参照の完全性プラグインの設定の詳細については、244 ページの「参照の完全性プラグインを設定する」を参照してください。

SSL を経由するレプリケーション

すべてのレプリケーション操作が SSL 接続で行われるように、レプリケーションに関わる Directory Server を構成することができます。

▼ SSL 用にレプリケーション操作を設定する

次の手順に、2 つのマスターを持つレプリケーショントポロジでレプリケーションを設定するコマンド例を示します。

注 – この例では、自己署名証明書を使用した簡単なレプリケーション設定を示しています。本稼働環境に SSL によるレプリケーションを設定する場合、証明機関によって信頼された証明書を使用する方がセキュリティーが向上します。

サプライヤサーバー証明書が、SSL ハンドシェイク時にクライアントとして機能できない SSL サーバー専用証明書である場合、SSL を経由するレプリケーションは失敗します。

レプリケーションが SSL によってセキュリティー保護されていても、レプリケーションマネージャーの認証はまだ簡単なバインドとパスワードを使用して行われます。クライアントベースの認証を使用して、レプリケーションのセキュリティーを完全に保護することができますが、これには、複雑な設定が必要です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 新しいサーバーを作成し、それらを起動します。

```
$ dsadm create -p 1389 -P 1636 /local/ds1
$ dsadm create -p 2389 -P 2636 /local/ds2

$ dsadm start /local/ds1
$ dsadm start /local/ds2
```

2 すべてのサーバーで、空のサフィックスを作成します。

```
$ dsconf create-suffix -e -i -p 1389 dc=example,dc=com
$ dsconf create-suffix -e -i -p 2389 dc=example,dc=com
```

3 すべてのサーバーで、マルチマスターパスワードファイルを設定します。

```
$ dsconf set-server-prop -e -i -h example1.server -p 1389 \
 def-repl-manager-pwd-file:/local/ds1/replmanrpwd1.txt
$ dsconf set-server-prop -e -i -h example2.server -p 2389 \
 def-repl-manager-pwd-file:/local/ds1/replmanrpwd2.txt
```

4 すべてのサーバーで、レプリケーションを有効にします。

```
$ dsconf enable-repl -h example1.server -p 1389 -e -i -d 1 master dc=example,dc=com
$ dsconf enable-repl -h example2.server -p 2389 -e -i -d 2 master dc=example,dc=com
```

5 すべてのサーバーで、既存のデフォルトの証明書を表示します。

```
$ dsadm show-cert -F der -o certfile1 /local/ds1 defaultCert
$ dsadm show-cert -F der -o certfile2 /local/ds2 defaultCert
```

6 すべてのサーバーに、ほかのすべてのサーバーからの **CA** によって信頼された証明書を追加します。

```
$ dsadm add-cert --ca /local/ds1 "ds2 Repl Manager Cert" certfile2
$ dsadm add-cert --ca /local/ds2 "ds1 Repl Manager Cert" certfile1
```

7 すべてのマスターサーバーとハブ(ソース)サーバーで、すべてのコンシューマ(ターゲット)サーバーとのレプリケーションアグリーメントを作成します。

レプリケーションアグリーメントにはセキュリティー保護された LDAP ポートを使用します。

```
$ dsconf create-repl-agmt -h example1.server -p 1389 -e -i \
 --auth-protocol "ssl-simple" dc=example,dc=com example2.server:2636
$ dsconf create-repl-agmt -h example2.server -p 2389 -e -i \
 --auth-protocol "ssl-simple" dc=example,dc=com example1.server:1636
```

8 すべてのレプリケーションアグリーメントで、認証パスワードファイルを、レプリケーションアグリーメント内のコンシューマ (ターゲット) サーバーのレプリケーションマネージャーパスワードファイルとして設定します。

```
$ dsconf set-repl-agmt-prop -h example1.server -p 1389 -e -i \
 dc=example,dc=com example2.server:2636 auth-pwd-file:/local/ds1/replmanrpwd2.txt
$ dsconf set-repl-agmt-prop -h example2.server -p 2389 -e -i \
 dc=example,dc=com example1.server:1636 auth-pwd-file:/local/ds1/replmanrpwd1.txt
```

サフィックスの初期化が完了すると、サプライヤはすべてのレプリケーション更新メッセージを SSL 経由でコンシューマに送信します。証明書を使用するオプションを選んだ場合は、証明書が利用されます。SSL のアグリーメント設定を使用して DSCC からカスタマーの初期化を行う場合も、セキュリティー保護された接続が使われます。

9 すべてのサーバーで、設定の変更を反映するため、サーバーを再起動します。

```
$ dsadm restart /local/ds1
$ dsadm restart /local/ds2
```

10 いずれかのマスターサーバーで、サフィックスを初期化します。

```
$ dsconf import -h example1.server -p 1389 -e -i /tmp/Example.ldif dc=example,dc=com
```

11 まだ初期化されていないすべてのサーバーで、レプリケーションアグリーメントを使用して、サーバーを初期化します。

```
$ dsconf init-repl-dest -e -i -h example1.server -p 1389 \
 dc=example,dc=com example1.server:2636
```

WAN を経由するレプリケーション

Directory Server では、広域ネットワーク (WAN) によって接続されたマシン間のマルチマスターレプリケーションを含むあらゆる形式のレプリケーションを実行できます。このレプリケーションにより、サプライヤサーバーは、待ち時間が大きく、帯域幅が小さいネットワーク経由で、最適な帯域幅を使用することによりコンシューマを初期化し、更新することができます。

注 – WAN 経由でレプリケートするレプリケーショントポロジの配備またはトラブルシューティングを行う場合、ネットワークの速度、待ち時間、およびパケットロスを調べる必要があります。これらのいずれかの点で、ネットワークの問題があると、レプリケーションの遅延が発生する可能性があります。

さらに、レプリケーションデータの転送率は、使用可能な物理媒体が帯域幅に関して許可している転送率を常に下回ります。レプリカ間の更新量を使用可能な帯域幅と物理的に適合させることができない場合、更新負荷が大きくなったときにレプリカ間に差異が生じることを、チューニングによって回避できなくなります。レプリケーションの遅延と更新のパフォーマンスは、さまざまな要因によります。次のような要因がありますが、これだけに限定されません。変更の頻度、エントリサイズ、サーバーハードウェア、エラー率、平均待ち時間、平均帯域幅。

環境でのレプリケーションに関する疑問がある場合は、Sun サービスプロバイダに問い合わせてください。

デフォルトでは、レプリケーションメカニズムの内部パラメータは WAN に合わせて最適化されています。ただし、前述の要因などが原因でレプリケーションが遅くなるときは、ウィンドウサイズとグループサイズのパラメータを調節してみてください。また、ネットワークのピーク時を避けてレプリケーションをスケジュールすることで、ネットワークの全体的な利用率を高めることができます。最後に、Directory Server は、帯域幅の使用を最適化するためにレプリケーションデータの圧縮に対応しています。

ネットワークパラメータの設定

ネットワーク経由でエントリをより効率的に送信するために、レプリケーションメカニズムがエントリをグループ化する方法は、ウィンドウとグループネットワークパラメータによって決定されます。これらのパラメータは、サプライヤとコンシューマがレプリケーション更新メッセージと、その確認応答を交換する方法に影響します。すべてのレプリケーションアグリーメントのパラメータは設定可能であるため、各コンシューマの特定のネットワーク状況に従ってレプリケーションパフォーマンスを調整することができます。

変更の効果を監視して、必要に応じてパラメータを調整します。手順については、288 ページの「レプリケーションの状態の取得」を参照してください。ウィンドウとグループのサイズパラメータを変更するときに、レプリケーションを中断する必要はありません。

ウィンドウサイズの設定

ウィンドウサイズ (デフォルト値は `10`) は、コンシューマからの即時の確認応答なしに送信できる更新メッセージの最大数を表します。

各コンシューマからの確認応答を待機するよりも、短時間に連続して多数のメッセージを送信する方が効果的です。適切なウィンドウサイズを使用することで、レプリケーション更新や確認応答の到着を待機するためにレプリカが費やす時間を排除できます。

コンシューマレプリカがサプライヤよりも遅れている場合、詳細な調整を行う前に、ウィンドウサイズをデフォルトよりも大きい数字 (100 など) に設定して、レプリケーションのパフォーマンスをもう一度確認してみます。レプリケーションの更新頻度が高く、更新間隔が短い場合、ローカルエリアネットワーク (LAN) 接続されたレプリカでもウィンドウサイズを大きくすることでパフォーマンスが向上する可能性があります。

▼ ウィンドウサイズを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ウィンドウサイズを変更します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN consumer-host:consumer-port transport-window-size:value
```

次に例を示します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389 \
 transport-window-size:20
```

グループサイズの設定

グループサイズ (デフォルト値は 1) は 1 つの更新メッセージに入れることのできるデータ修正の最大数を表します。ネットワーク接続がレプリケーションを妨害しているように思われる場合、グループサイズをデフォルトよりも大きい数字 (10 など) に設定して、レプリケーションのパフォーマンスを再確認してみます。

グループサイズを大きくする場合、次のことがあてはまることを確認します。

- ウィンドウサイズがグループサイズよりも大幅に大きい数字に設定されていること
- ウィンドウサイズをグループサイズで割った値が、コンシューマの `cn=config` の `nsslapd-maxThreadsPerConn` の値よりも大幅に上回っていること (通常は 2 倍)

  グループサイズが 1 よりも大きい数に設定されている場合、サプライヤはグループが満たされるのを待たずに、コンシューマに更新を送信します。

▼ グループサイズを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- グループサイズを変更します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN \
 consumer-host:consumer-port transport-group-size:value
```

次に例を示します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389 \
 transport-group-size:10
```

レプリケーションアクティビティーのスケジュール

レプリカ間の即時同期が重要でない場合は、ネットワークの利用率が低い時間にレプリケーションをスケジュールできます。ネットワークを多く利用できるほど、データのレプリケーションが大幅に速く完了するはずです。

日または週単位で、1 日の特定の時間にレプリケーションを開始および終了するようにスケジュールできます。これは、レプリケーションアグリーメントによって、コンシューマごとに個別に実行できます。新しいスケジュールはただちに有効になり、対応するコンシューマに対する次回のデータのレプリケーションは、スケジュールと合致する日時になった時点で行われます。

▼ レプリケーションアクティビティーをスケジュールする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリケーションスケジュールを変更します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN \
 host:port repl-schedule:value
```

たとえば、レプリケーションを毎晩 2:00 から 4:00 までの間に行うように設定する場合、次のように入力します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389 \
 repl-schedule:"0200-0400 0123456"
```

`0123456` は曜日を示し、 `0` は日曜日、`1` は月曜日というように表します。

レプリケーションの圧縮の設定

レプリケーションで使われる帯域幅を節約するために、コンシューマの更新時に送信されるデータを圧縮するようにレプリケーションを設定できます。レプリケー

ションメカニズムは、Zlib 圧縮ライブラリを使用します。圧縮を利用するには、Solaris または Linux プラットフォームでサプライヤとコンシューマの両方が稼動している必要があります。

WAN 環境で予想されるレプリケーションの利用状況に対して、最高の結果が得られる圧縮レベルを実験的にテストし、選択する必要があります。ネットワーク帯域幅が広い LAN ではこのパラメータを設定しないでください。圧縮と圧縮解除の計算により、レプリケーションが遅くなります。

▼ レプリケーションの圧縮を設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- マスターサーバーのレプリケーションアグリーメントエントリにレプリケーションの圧縮を設定します。

```
$ dsconf set-repl-agmt-prop -h host -p port suffix-DN \
 consumer-host:consumer-port transport-compression:level
```

level には `high`、 `medium`、`low`、または `none` を指定できます。

たとえば、レプリケーションの更新を `host1:1389` のコンシューマに送信する場合、最速の圧縮を使用するには、次のように入力します。

```
$ dsconf set-repl-agmt-prop -h host2 -p 1389 dc=example,dc=com host1:1389 \
 transport-compression:high
```

圧縮レベルの設定の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

レプリケーショントポロジの変更

ここでは、既存のレプリケーショントポロジの管理の以下の側面について説明します。

レプリケーションマネージャーの変更

レプリケーションアグリーメントを編集して、コンシューマサーバーへのバインドに使われるレプリケーションマネージャーの識別情報を変更できます。レプリケーションが中断されないように、レプリケーションアグリーメントを変更する前に、

新しいレプリケーションマネージャーエントリまたはコンシューマの証明書エントリを定義する必要があります。ただし、バインドの失敗によってレプリケーションが中断された場合、レプリケーション回復設定の制限内でエラーを修正したときは、レプリケーションメカニズムによって必要なすべての更新が自動的に送信されます。手順については、254 ページの「デフォルト以外のレプリケーションマネージャーの使用」を参照してください。

レプリケーションアグリーメントの管理

レプリケーションアグリーメントを無効化、有効化、または削除できます。

レプリケーションアグリーメントの無効化

レプリケーションアグリーメントを無効にすると、そのアグリーメントに指定されているコンシューマに対してマスターが更新を送信しなくなります。そのサーバーのレプリケーションは停止されますが、アグリーメントに記録されているすべての設定は残されます。あとからまたアグリーメントを有効にすることで、レプリケーションを再開できます。中断後にレプリケーションメカニズムを再開することについては、279 ページの「レプリケーションアグリーメントの有効化」を参照してください。

▼ レプリケーションアグリーメントを無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリケーションアグリーメントを無効にします。

```
$ dsconf disable-repl-agmt -h host -p port suffix-DN consumer-host:consumer-port
```

次に例を示します。

```
$ dsconf disable-repl-agmt -h host2 -p 1389 dc=example,dc=com host1:1389
```

レプリケーションアグリーメントの有効化

レプリケーションアグリーメントを有効にすると、指定のコンシューマのレプリケーションが再開されます。ただし、レプリケーションの回復設定で許容される時間より長くレプリケーションを中断していた場合は、別のサプライヤによるコンシューマの更新が行われないため、コンシューマを初期化し直す必要があります。レプリケーションの回復設定は、最大サイズとこのサプライヤの更新履歴ログの経過時間とコンシューマの削除の遅延です (250 ページの「コンシューマの詳細設定を行う」を参照)。

中断時間が短く、レプリケーションが回復された場合は、アグリーメントがふたたび有効になったときに、マスターが自動的にそのコンシューマを更新します。

▼ レプリケーションアグリーメントを有効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリケーションアグリーメントを有効にします。

  ```
  $ dsconf -h host -p port enable-repl-agmt suffix-DN consumer-host:consumer-port
  ```

  次に例を示します。

  ```
  $ dsconf -h host2 -p 1389 enable-repl-agmt dc=example,dc=com host1:1389
  ```

レプリケーションアグリーメントの削除

レプリケーションアグリーメントを削除すると、対応するコンシューマのレプリケーションは停止され、アグリーメントに関するすべての設定情報が失われます。後日レプリケーションを再開する場合は、279 ページの「レプリケーションアグリーメントの無効化」で説明するように、アグリーメントを無効にします。

▼ レプリケーションアグリーメントを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリケーションアグリーメントを削除します。

  ```
  $ dsconf delete-repl-agmt -h host -p port suffix-DN consumer-host:consumer-port
  ```

  次に例を示します。

  ```
  $ dsconf delete-repl-agmt -h host2 -p 1389 dc=example,dc=com host1:1389
  ```

レプリカの昇格と降格

レプリカの昇格と降格は、レプリケーショントポロジで、レプリカの役割を変更することを意味します。専用コンシューマをハブに変更したり、ハブをマスターに変更したりできます。また、マスターをハブに変更したり、ハブを専用コンシューマに変更したりすることもできます。ただし、マスターを直接コンシューマに格下げしたり、コンシューマを直接マスターに格上げすることはできません。

マルチマスターレプリケーションのメカニズムでレプリカの役割を変更できることで、トポロジがとても柔軟になります。コンシューマレプリカが処理を担当していたサイトの負荷が増え、複数のレプリカを持つハブによる処理が必要になることも

あります。レプリカの内容に対して多数の変更が含まれるときは、ハブをマスターに昇格させることで、ローカルな変更に迅速に対応し、その変更を他のサイトの他のマスターにレプリケートできます。

レプリカを昇格または降格させる場合は、次のことに注意します。

- コンシューマを昇格させると、ハブになります。ハブを昇格させると、マスターになります。サーバーをコンシューマからマスターに直接昇格させることはできません。まずコンシューマをハブに昇格させてから、ハブをマスターに昇格させる必要があります。同様に、マスターをコンシューマに降格させる場合、マスターをハブに降格させてから、ハブをコンシューマに降格させる必要があります。
- マスターをハブに降格させると、レプリカは読み取り専用となり、残りのマスターに対してはリフェラルを送信するように設定されます。新しいハブは、設定されているすべてのコンシューマをハブまたは専用コンシューマとして維持します。
- シングルマスターレプリケーションでマスターをハブに降格させると、マスターレプリカの存在しないトポロジが作成されます。新しいマスターを定義することを前提として、Directory Server でもこのような変更が可能です。ただし、マスターを降格させる前にマルチマスターとして新しいマスターを追加し、初期化できるようにしておくことをお勧めします。
- ハブをコンシューマに降格させる前に、ハブとのすべてのレプリケーションアグリーメントを無効にするか、削除しておく必要があります。これを実行しないと、降格操作が次のエラーによって失敗します: `LDAP_OPERATIONS_ERROR “Unable to demote a hub to a read-only replica if some agreements are enabled”`。

  そのハブのコンシューマが他のハブまたはマスターによって更新されるように設定されていない場合、そのコンシューマは更新されなくなります。これらのコンシューマが更新されるように、残りのハブまたはマスターに新しいアグリーメントを作成する必要があります。
- コンシューマをハブに昇格させると、更新履歴ログが有効になり、コンシューマとの間に新しいアグリーメントを定義できるようになります。
- ハブをマスターに昇格させると、レプリカは更新要求を受け付けるようになり、他のマスター、ハブ、または専用コンシューマとの間に新しいアグリーメントを定義できるようになります。

▼ レプリカの役割を変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリカの役割を変更するには、次のいずれかのコマンドを使用します。

```
$ dsconf promote-repl -h host -p port role suffix-DN
```

```
$ dsconf demote-repl -h host -p port role suffix-DN
```

role は `master`、 `hub`、または `consumer` です。

レプリケートされたサフィックスの無効化

レプリケートされたサフィックスを無効にすると、それはレプリケーショントポロジから除外されます。設定されている役割 (マスター、ハブ、またはコンシューマ) に応じて、そのレプリケートされたサフィックスは更新されなくなり、更新を送信しなくなります。サプライヤサーバーのサフィックスを無効にすると、すべてのレプリケーションアグリーメントが削除されます。そのレプリカをふたたび有効にするときは、これらのアグリーメントを作成し直す必要があります。

▼ レプリケートされたサフィックスを無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- レプリケートされたサフィックスを無効にします。

```
$ dsconf disable-repl -h host -p port suffix-DN
```

次に例を示します。

```
$ dsconf disable-repl -h host2 -p 1389 dc=example,dc=com
```

レプリケートされたサフィックスの同期の維持

定期的に行う保守のために、レプリケーションに関わる Directory Server を停止したあと、オンラインに戻す場合、レプリケーションによってただちに更新されるようにする必要があります。特に、マルチマスター環境のマスターサーバーでは、マルチマスターセットのもう一つのサーバーからディレクトリ情報を更新する必要があります。マルチマスター以外の環境でも、ハブサーバーや専用コンシューマサーバーが保守のためにオフラインになった場合、オンラインに復帰したときは、マスターサーバー側から更新を行う必要があります。

ここでは、レプリケーションの再試行アルゴリズムおよび次の実行まで待たずに、強制的にレプリケーション更新を行う方法について説明します。

注 – ここで説明されている手順を利用できるのは、レプリケーションの設定が完了し、さらにコンシューマを初期化した直後だけです。

レプリケーションの再試行アルゴリズム

ソースレプリカのターゲットへのレプリケーションが失敗すると、増分的間隔で、定期的に再試行されます。再試行の間隔はエラーの種類によって異なります。

ソースレプリカとターゲットレプリカの間で、常に同期をとるレプリケーションアグリーメントを設定していても、オフライン状態の時間が 5 分を超えたレプリカをただちに更新するには、この方法では不十分です。

▼ レプリケーションの更新を強制的に実行する

レプリケーションを停止した場合、ターゲットのサフィックスに対して、レプリケーションの更新を強制的に実行することができます。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- ソースサーバーで、ターゲットサーバーへのレプリケーションの更新を再起動します。

```
$ dsconf update-repl-dest-now -h host -p port suffix-DN destination-host:destination-port
```

次に例を示します。

```
$ dsconf update-repl-dest-now -h host2 -p 1389 dc=example,dc=com host1:1389
```

Directory Server 6.x 以前のリリースでのレプリケーション

ここでは、6.x 以前の Directory Server のリリースでのレプリケーションの設定方法について説明します。

Directory Server 6.x と Directory Server 5.1 または 5.2 間のレプリケート

Directory Server 5.1、5.2、および 6.x は、レプリケーション設定に関して互換性がありますが、次の例外があります。

- Directory Server 6.x 以前のリリースでは、レプリケーションの優先順位がサポートされていません。6.x マスターレプリカでレプリケーションの優先順位を設定する場合、レプリケーションの優先順位は、Directory Server 6.x を実行するコンシューマには転送されますが、Directory Server の以前のバージョンを実行するコンシューマには転送されません。
- Directory Server 5.1 または 5.2 マスターを含むレプリケーショントポロジでは、無限数のマスターはサポートされていません。Directory Server 6.x では、レプリケーショントポロジで無限数のマスターをサポートしていますが、レプリケーショントポロジに Directory Server 5.2 マスターサーバーが含まれる場合、この数は 4 に制限されます。Directory Server 5.1 ではマルチマスターレプリケーションをサポートしていません。

旧バージョン形式の更新履歴ログの使用

旧バージョン形式の更新履歴ログは LDAP クライアントが Directory Server データに対して行われた変更履歴を確認するために使用します。旧バージョン形式の更新履歴ログは、`cn=changelog` というサフィックスの下で、Directory Server の更新履歴ログに対する独立したデータベースに格納されます。

旧バージョン形式の更新履歴ログは、スタンドアロンサーバーまたはレプリケーショントポロジ内の各サーバー上で有効にできます。サーバー上で旧バージョン形式の更新履歴ログが有効になっている場合、デフォルトでは、そのサーバー上のすべてのサフィックスに対する更新がログファイルに記録されます。旧バージョン形式の更新履歴ログは、指定したサフィックスに対する更新のみをログファイルに記録するように設定できます。

レプリケーショントポロジで、旧バージョン形式の更新履歴ログを使用する方法および旧バージョン形式の更新履歴ログの使用の制限については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Replication and the Retro Change Log Plug-In」を参照してください。

旧バージョン形式の更新履歴ログのエントリの属性については、`changeLogEntry`(5dsoc) のマニュアルページを参照してください。

旧バージョン形式の更新履歴ログの変更の詳細については、`dsconf`(1M) のマニュアルページを参照してください。

この節では、旧バージョン形式の更新履歴ログを使用するさまざまな方法について説明します。

▼ 旧バージョン形式の更新履歴ログを有効にする

旧バージョン形式の更新履歴ログを使用するには、ログを有効にする必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 旧バージョン形式の更新履歴ログ設定エントリを変更します。

```
$ dsconf set-server-prop -h host -p port retro-cl-enabled:on
```

2 サーバーを再起動します。

詳細については、59 ページの「Directory Server インスタンスの起動、停止、および再起動」を参照してください。

▼ 指定したサフィックスに対する更新を記録するための旧バージョン形式の更新履歴ログを設定する

サーバー上で旧バージョン形式の更新履歴ログが有効になっている場合、デフォルトでは、そのサーバー上のすべてのサフィックスに対する更新がログファイルに記録されます。この手順では、指定したサフィックスに対する更新のみを記録するように、旧バージョン形式の更新履歴ログを設定する方法について説明します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 旧バージョン形式の更新履歴ログ設定エントリを変更します。

```
$ dsconf set-server-prop -h host -p port retro-cl-suffix-dn:suffix-DN
```

たとえば、`cn=Contractors,dc=example,dc=com` サフィックスと `ou=People,dc=example,dc=com` サフィックスに対する変更のみを記録するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host2 -p 1389 \
 retro-cl-suffix-dn:"cn=Contractors,dc=example,dc=com" \
 retro-cl-suffix-dn:"ou=People,dc=example,dc=com"
```

指定したサフィックスの既存リストにサフィックスを追加するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port retro-cl-suffix-dn+:suffix-DN
```

2 サーバーを再起動します。

詳細については、59 ページの「Directory Server インスタンスの起動、停止、および再起動」を参照してください。

▼ 削除したエントリの属性を記録するための旧バージョン形式の更新履歴ログを設定する

この手順では、エントリが削除されたときにそのエントリの指定された属性を記録するように旧バージョン形式の更新履歴ログを設定する方法について説明します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 記録する属性を指定します。

```
$ dsconf set-server-prop -h host -p port retro-cl-deleted-entry-attr: \
 attribute1 attribute2
```

たとえば、旧バージョン形式の更新履歴ログで、削除されたエントリの UID 属性を記録するように設定するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port retro-cl-deleted-entry-attr:uid
```

指定した属性の既存リストに属性を追加するには、次のコマンドを使用します。

```
$ dsconf set-server-prop -h host -p port retro-cl-deleted-entry-attr+:attribute
```

2 サーバーを再起動します。

詳細については、59 ページの「Directory Server インスタンスの起動、停止、および再起動」を参照してください。

▼ 旧バージョン形式の更新履歴ログを削除する

旧バージョン形式の更新履歴ログのエントリは、指定した期間の経過後に自動的に削除することができます。エントリを自動的に削除する期間を設定するには、`cn=Retro Changelog Plugin, cn=plugins, cn=config` エントリに `nsslapd-changelogmaxage` 設定属性を設定します。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 旧バージョン形式の更新履歴ログが有効にされていることを確認します。

```
$ dsconf get-server-prop -h host -p port retro-cl-enabled
```

2 旧バージョン形式の更新履歴ログが有効にされていなければ、有効にします。

```
$ dsconf set-server-prop -h host -p port retro-cl-enabled:on
```

3 更新履歴ログの最大経過時間を設定します。

```
$ dsconf set-server-prop -h host -p port retro-cl-max-age:duration
```

duration には `undefined` (経過時間の制限なし) または次のいずれかを指定できます。

- `s` (秒)
- `m` (分)
- `h` (時)
- `d` (日)
- `w` (週)

たとえば、旧バージョン形式の更新履歴ログの最大経過時間を 2 日に設定するには、次のように入力します。

```
$ dsconf set-server-prop -h host 2 -p 1389 retro-cl-max-age:2d
```

旧バージョン形式の更新履歴ログは、更新記録ログに対する次回の処理時に削除されます。

アクセス制御と旧バージョン形式の更新履歴ログ

旧バージョン形式の更新履歴ログは検索操作をサポートしています。また、次の形式のフィルタを含む検索用に最適化されています。

```
(&(changeNumber>=X)(changeNumber<=Y))
```

原則として、旧バージョン形式の更新履歴ログに対しては、追加操作や変更操作を行わないでください。ログのサイズを削減するため、エントリを削除できます。旧バージョン形式の更新履歴ログで修正処理を実行する必要があるのは、デフォルトのアクセス制御ポリシーを修正する場合だけです。

旧バージョン形式の更新履歴ログを作成すると、デフォルトで次のアクセス制御規則が適用されます。

- 旧バージョン形式の更新履歴ログのトップエントリ `cn=changelog` に対する読み取り、検索、および比較の権限は、すべての認証ユーザー (`userdn=anyone` のユーザー。`userdn=all` で指定された匿名アクセスとは異なる) に付与されます。
- Directory Manager に対する暗黙の了承を除き、書き込みおよび削除アクセスは付与されません。

  旧バージョン形式の更新履歴ログのエントリにはパスワードなどの重要な情報が含まれている場合があるので、読み取りアクセス権を匿名ユーザーに付与しないでください。認証されたユーザーにも内容の表示が許可されない場合でも、旧バージョン形式の更新履歴ログの内容へのアクセスをさらに制限することが必要なことがあります。

旧バージョン形式の更新履歴ログに対するデフォルトのアクセス制御ポリシーを変更するには、`cn=changelog` エントリの `aci` 属性を変更する必要があります。第 6 章を参照してください。

レプリケーションの状態の取得

DSCC またはコマンド行ツールを使用して、レプリケーションの状態を取得できます。

DSCC でのレプリケーションの状態の取得

「サフィックス」タブを使用して、レプリケーションアグリーメントやレプリケーションの遅延など、レプリケーションをグラフィカルに表示できます。詳細については、DSCC オンラインヘルプを参照してください。

さらに、DSCC を使用して、次の図に示すように、レプリケーショントポロジを表示することができます。

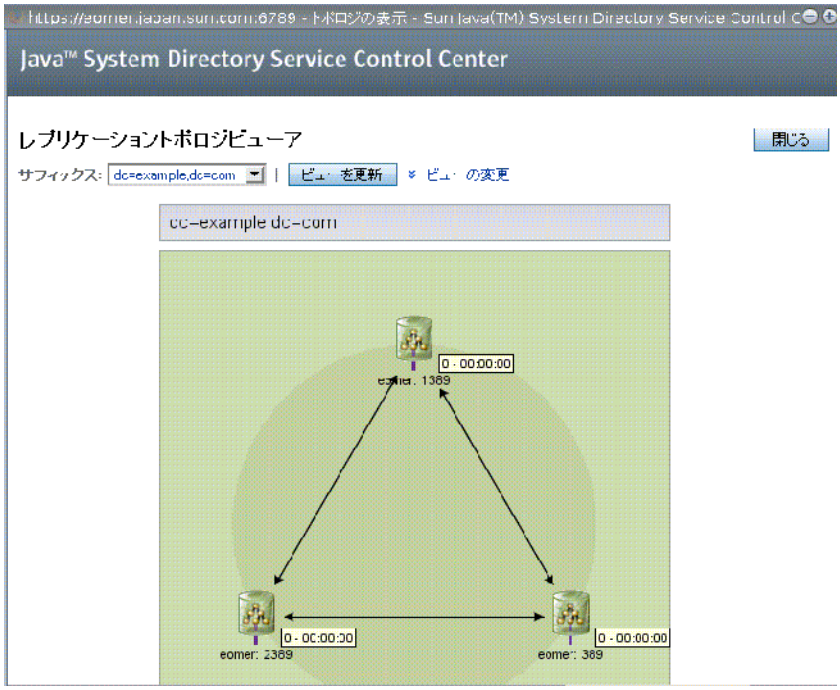


図 10–1 レプリケーショントポロジの例

コマンド行を用いたレプリケーション状態の取得

DSCC を使用できない場合は、コマンド行ツールを使用して、レプリケーション配備に関する情報を取得します。

これらのツールの完全なコマンド行構文と使用例については、マニュアルページを参照してください。

- `repldisc`: レプリケーションの配備に含まれるすべての既知のサーバーを検出し、テーブルを作成します。`repldisc`(1) のマニュアルページを参照してください。
- `insync`: サプライヤレプリカと、1 つまたは複数のコンシューマレプリカの間の同期状態を示します。`insync`(1) のマニュアルページを参照してください。
- `entrycmp`: 複数のレプリカに含まれる同じエントリを比較します。`entrycmp`(1) のマニュアルページを参照してください。

これらのコマンドのあるディレクトリを見つけるには、35 ページの「コマンドの場所」を参照してください。

よく発生するレプリケーションの競合の解決

マルチマスターレプリケーションでは、疎整合型レプリケーションモデル (Loose Consistency Replication Model) を使用します。つまり、同一のエントリを別々のサーバーから同時に変更できます。2 つのサーバー間で更新が送信された場合、更新の競合を解決する必要があります。たいていは自動的に解決されます。たとえば、各サーバーの変更に関するタイムスタンプは、優先される最近の変更によって解決されます。しかし、一部の更新の競合では、解決に至るまでに、手動の調整が必要になることもあります。

この節の内容は、次のとおりです。

DSCC によるレプリケーションの競合の解決

レプリケーションの競合を解決するもっとも簡単な方法は、DSCC を使用することです。詳細については DSCC オンラインヘルプを参照してください。

コマンド行によるレプリケーションの競合の解決

コマンド行を使用して、レプリケーションの競合を解決できます。レプリケーションプロセスで自動的に解決できない更新の競合があるエントリには、競合マーカーとしてのオペレーショナル属性 `nsds5ReplConflict` が含まれます。

競合しているエントリを見つけるには、この属性を含むエントリを定期的に検索します。たとえば、次の `ldapsearch` コマンドを使用して、競合のあるエントリを見つけます。

```
$ ldapsearch -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config \
 -w - -b "dc=example,dc=com" "(nsds5ReplConflict=*)"
```

`nsds5ReplConflict` 属性には、デフォルトでインデックスが設定されています。

ネーミングの競合の解決

サーバーがお互いの変更をレプリケートする前にエントリが作成された場合、別々のマスターに同じ DN を持つエントリが作成される可能性があります。レプリケーション時には、競合解決メカニズムによって 2 番目に作成されたエントリの名前が自動的に変更されます。

DN のネーミングが競合しているエントリは、オペレーショナル属性 nsuniqueid によって指定される一意の識別子を DN 内に含めることで名前を変更します。

たとえば、2 つのマスターで uid=bjensen,ou=People,dc=example,dc=com というエントリが同時に作成されると、レプリケーション後の 2 つのエントリは、次のようになります。

- uid=bjensen,ou=People,dc=example,dc=com
- nsuniqueid=66446001-1dd211b2-66225011-2ee211db+uid=bjensen,dc=example,dc=com

2 つ目のエントリには、有効な DN を指定する必要があります。競合しているエントリを削除し、競合しない名前で追加し直すことができます。ただし、エントリの名前を変更しても、その内容は変更されていません。名前変更の手順は、ネーミング属性が 1 つの値を持つか複数の値を持つかによって異なります。そのための手順は次のとおりです。

▼ 複数の値からなるネーミング属性を持つ競合エントリの名前を変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 古い **RDN** 値を維持しながらエントリの名前を変更します。たとえば、次のように入力します。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: nsuniqueid=66446001-1dd211b2-66225011-2ee211db+uid=bjensen,dc=example,dc=com
changetype: modrdn
newrdn: uid=bj66446001
deleteoldrdn: 0
^D
```

この手順では古い RDN 値を削除することはできません。ここには削除できないオペレーショナル属性 nsuniqueid も含まれているからです。

2 ネーミング属性の古い **RDN** 値と競合マーカー属性を削除します。たとえば、次のように入力します。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: uid=bj66446001,dc=example,dc=com
changetype: modify
delete: uid
uid: bjensen
-
delete: nsds5ReplConflict
^D
```

▼ 1つの値からなるネーミング属性を持つ競合エントリの名前を変更する

dc (ドメインコンポーネント) など、重複したエントリのネーミング属性が 1 つの値の場合は、エントリの名前を単に同じ属性の別の値に変更することはできません。代わりに一時的な名前を付ける必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 別のネーミング属性を使用してエントリの名前を変更し、古い **RDN** を保持しておきます。たとえば、次のように入力します。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: nsuniqueid=66446001-1dd211b2-66225011-2ee211db+dc=HR,dc=example,dc=com
changetype: modrdn
newrdn: o=TempHREntry
deleteoldrdn: 0
^D
```

この手順では古い RDN 値を削除することはできません。ここには削除できないオペレーショナル属性 `nsuniqueid` も含まれているからです。

2 必要なネーミング属性を一意の値に変更し、競合マーカー属性を削除します。たとえば、次のように入力します。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: o=TempHREntry,dc=example,dc=com
changetype: modify
replace: dc
dc: NewHR
delete: nsds5ReplConflict
^D
```

3 エントリの名前を変更し、指定したネーミング属性に戻します。たとえば、次のように入力します。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: dc=NewHR,dc=example,dc=com
changetype: modrdn
newrdn: dc=HR
deleteoldrdn: 1
^D
```

deleteoldrdn 属性の値に 1 を設定すると、一時的な属性と値のペアである o=TempHREntry が削除されます。この属性を保持する場合は、deleteoldrdn 属性の値に 0 を設定します。

親のないエントリの競合の解決

エントリの削除操作がレプリケートされたとき、コンシューマサーバーが削除されるエントリが子エントリを持つことを検出した場合、競合解決処理によって *glue* エントリが作成され、親のないエントリをディレクトリに持つことを回避します。

同様に、エントリの追加後にレプリケーションが実行され、コンシューマサーバーが追加されたエントリの親エントリを検出できなかった場合も、競合解決処理は親を表す glue エントリを作成し、親のないエントリが追加されることを回避します。

glue エントリは、glue および extensibleObject というオブジェクトクラスを持つ一時的なエントリです。glue エントリは、次のさまざまな方法で作成されます。

- 競合の解決の手順で、一致する一意の識別子を持つ削除されたエントリが見つかった場合は、glue エントリはそのエントリが復活されます。さらに、glue オブジェクトクラスと nsds5ReplConflict 属性も含まれます。

  この場合は、glue エントリを修正して glue オブジェクトクラスと nsds5ReplConflict 属性を削除し、通常のエントリに戻すか、または glue エントリとその子エントリを削除します。
- サーバーによって、glue および extensibleObject オブジェクトクラスを持つ必要最小限のエントリが作成されます。

  このような場合は、意味のあるエントリになるようにエントリを修正するか、またはエントリとその子エントリをすべて削除します。

潜在的な相互運用性の問題の解決

メールサーバーのように属性の一意性に依存するアプリケーションとの相互運用性のため、nsds5ReplConflict 属性を持つエントリへのアクセスを制限する必要がある場合があります。これらのエントリへのアクセスを制限しない場合は、1 つの属性だけを要求するアプリケーションが元のエントリと nsds5ReplConflict を含む競合解決エントリの両方を取得し、処理が失敗します。

アクセスを制限するには、次のコマンドを使用して、匿名の読み取りアクセスを許可するデフォルトの ACI を変更する必要があります。

```
$ ldapmodify -h host2 -p 1389 -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: dc=example,dc=com
changetype: modify
```

```
delete: aci
aci: (target ="ldap:///dc=example,dc=com")
 (targetattr !="userPassword"
 (version 3.0;acl "Anonymous read-search  access";
 allow (read, search, compare)(userdn = "ldap:///anyone");)
-
add: aci
aci: (target="ldap:///dc=example,dc=com")
 (targetattr!="userPassword")
 (targetfilter="(!(nsds5ReplConflict=*))")(version 3.0;acl
 "Anonymous read-search access";allow (read, search, compare)
 (userdn="ldap:///anyone");)
^D
```

新しい ACI では、検索結果として返されたものの中から `nsds5ReplConflict` 属性を持つエントリが保持されます。

◆ ◆ ◆ 第 11 章

11 Directory Server のスキーマ

Directory Server には、数多くのオブジェクトクラスおよび属性を持つ標準のスキーマが付属しています。通常の作業では標準のオブジェクトクラスと属性で十分ですが、新しいオブジェクトクラスや属性の作成など、スキーマの拡張が必要となることもあります。標準スキーマの概要と配備に適合するスキーマの設計手順については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

この章では、スキーマを管理する方法について説明します。次の内容について説明します。

スキーマ検査の管理

スキーマ検査を有効にすると、インポート、追加、変更のすべての処理が Directory Server によって、次のように現在定義されているディレクトリスキーマに準拠するようになります。

- 各エントリのオブジェクトクラスと属性は、スキーマに準拠する。
- エントリには、そのエントリに定義されているすべてのオブジェクトクラスに必要なすべての属性が含まれる。
- エントリには、そのエントリのオブジェクトクラスに許可されている属性だけが含まれる。

注 – エントリを変更する場合、Directory Server は、変更する属性だけでなく、エントリ全体に対してスキーマ検査を実行します。このため、エントリのいずれかのオブジェクトクラスまたは属性がスキーマに準拠していない場合、変更処理は失敗します。

ただし、スキーマ検査では構文に関する属性値の有効性は検証されません。

スキーマ検査はデフォルトで有効にされています。一般に、スキーマ検査を有効にして、Directory Server を実行します。多くのクライアントアプリケーションでは、スキーマ検査を有効にしておくことは、すべてのエントリがスキーマに準拠しているものと見なされます。ただし、スキーマ検査を有効にしても、Directory Server はディレクトリの既存の内容を確認しません。ディレクトリのすべての内容がスキーマに確実に準拠するようにするには、エントリを追加する前、またはすべてのエントリを再初期化する前にスキーマ検査を有効にする以外に方法はありません。

スキーマ検査を無効にするのは、スキーマに準拠していることが確実な LDIF ファイルのインポートを高速で処理するときだけです。ただし、スキーマに準拠しないエントリのインポートにはリスクがあります。スキーマ検査が無効にされている場合、スキーマに準拠しないインポートされたエントリが検出されません。

レプリケートされた環境でのスキーマ検査の使用の詳細については、313 ページの「ディレクトリスキーマのレプリケート」を参照してください。

▼ スキーマの準拠の問題を修正する

エントリがスキーマに準拠していない場合は、このエントリを検索することができず、エントリに対する変更処理も失敗することがあります。次の手順に従って、問題を修正します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　スキーマの準拠の問題を修正する必要性を避けるため、配備の前にスキーマを計画し、スキーマの変更を最小にします。詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』を参照してください。

1. エントリが準拠しない理由を判断するには、エントリを取得し、それを現在定義されているスキーマと手動で比較します。

   詳細については、304 ページの「属性タイプを表示する」および 308 ページの「オブジェクトクラスを表示する」を参照してください。

2 スキーマに準拠するようにエントリを変更するか、エントリに準拠するようにスキーマを変更します。

カスタムスキーマについて

ディレクトリのニーズに対して、標準スキーマでは著しく制限される場合に、標準スキーマを拡張できます。スキーマをカスタマイズする場合は、次のガイドラインに従います。

- できるかぎり既存のスキーマ要素を再利用する。
- 各オブジェクトクラスに定義する必須の属性の数を最小限にする。
- 同じ目的で複数のオブジェクトクラスまたは属性を定義しない。
- できるかぎりスキーマを簡潔にする。

スキーマをカスタマイズする場合は、標準スキーマの属性またはオブジェクトクラスの既存の定義の変更、削除、および置換は行わないでください。標準スキーマを修正すると、ほかのディレクトリや LDAP クライアントアプリケーションとの互換性に問題が生じます。

Directory Server 内部オペレーショナル属性は変更しないでください。ただし、外部アプリケーション用に独自のオペレーショナル変数を作成できます。

`objectClass: extensibleObject` を使用する代わりに、常にオブジェクトクラスを定義します。Directory Server は、オブジェクトクラス `extensibleObject` があるエントリのスキーマ検査を実行しないため、エントリに存在する属性を制限したり、検査したりしません。アプリケーションでのタイプミス、たとえば `givenName` 属性タイプを `giveName` と間違えた場合も、Directory Server はその間違いに気づきません。また、Directory Server は、`extensibleObject` エントリの中で未定義の属性は、複数値属性で、大文字小文字に関係ない文字列構文を持つことを前提とします。さらに、特定のオブジェクトクラスを持つエントリに依存するアプリケーションもあります。一般に、オブジェクトクラスへの拡張を必要とするアプリケーションがある場合は、スキーマ管理を放棄しないでください。代わりに、アプリケーションに必要な属性を含む補助のオブジェクトクラスを作成します。

この節では、デフォルトのディレクトリスキーマについてと、カスタマイズした属性とオブジェクトクラスの作成について説明します。

デフォルトの Directory Server スキーマ

Directory Server で提供されるスキーマは、*instance-path* `/config/schema/` ディレクトリに保存されているファイルのセットに記述されています。

このディレクトリには、Directory Server の一般的なすべてのスキーマと関連製品が格納されています。LDAP v3 標準のユーザースキーマと組織スキーマは、`00core.ldif`

ファイルに記述されています。旧バージョンのディレクトリで使用された設定スキーマは、`50ns-directory.ldif` ファイルに記述されています。

注 – サーバーの稼動中は、このディレクトリ内のファイルを変更しないでください。

オブジェクト識別子

各 LDAP オブジェクトクラスまたは属性には、一意の名前とオブジェクト識別子 (OID) が割り当てられている必要があります。スキーマを定義するときは、組織に固有の OID が必要です。1 つの OID ですべてのスキーマ要件に対応できます。属性とオブジェクトクラスのその OID に新しいエントリを追加します。

OID の取得とスキーマでの割り当ては、次の手順で行います。

- IANA (Internet Assigned Numbers Authority) または国内の機関から組織の OID を取得する。

  国によっては、企業にすでに OID が割り当てられています。所属する組織がまだ OID を持っていない場合は、IANA から OID を取得できます。
- OID の割り当てを追跡できるように、OID レジストリを作成する。

  OID レジストリは、ディレクトリスキーマで使用する OID と OID の説明を提供するリストで、作成者が保持します。OID レジストリにより、OID が複数の目的に使用されないようにすることができます。
- スキーマ要素を入れるために、OID ツリーにエントリを作成する。

  OID エントリまたはディレクトリスキーマの下に少なくとも 2 つのエントリ (1 つは属性用の *OID*.1、もう 1 つはオブジェクトクラス用の *OID*.2) を作成します。独自のマッチングルールや制御を定義する場合は、必要に応じて *OID*.3 などの新しいエントリを追加できます。

属性とオブジェクトクラスの命名

新しい属性とオブジェクトクラスの名前を作成する場合、スキーマで使いやすいように、わかりやすい名前を作成します。

作成する要素に固有の接頭辞を付けて、作成したスキーマ要素と既存のスキーマ要素間での名前の衝突を防ぎます。たとえば、Example.com 社では、各カスタムスキーマ要素の前に `Example` という接頭辞を追加します。また、ディレクトリ内の Example.com 社員を識別するために `ExamplePerson` という特別なオブジェクトクラスを追加します。

LDAP では、属性タイプ名とオブジェクトクラス名は、大文字と小文字が区別されません。アプリケーションでは、それらを大文字と小文字を区別しない文字列として扱う必要があります。

新しいオブジェクトクラスを定義する場合

ディレクトリのエントリに格納する必要がある情報の中に既存のオブジェクトクラスがサポートしていないものがある場合は、新しいオブジェクトクラスを追加します。

新しいオブジェクトクラスを作成するには、次の 2 つの方法があります。

- 属性を追加するオブジェクトクラス構造ごとに 1 つずつ、多数の新しいオブジェクトクラスを作成する。
- ディレクトリ用に作成するすべての属性を含む 1 つのオブジェクトクラスを作成する。このオブジェクトクラスは AUXILIARY オブジェクトクラスとして定義して作成する。

  サイトに ExampleDepartmentNumber と ExampleEmergencyPhoneNumber という属性を作成するとします。これらの属性にいくつかのサブセットを許可する複数のオブジェクトクラスを作成できます。ExamplePerson というオブジェクトクラスを作成し、そのオブジェクトクラスが ExampleDepartmentNumber と ExampleEmergencyPhoneNumber を許可するようにします。ExamplePerson の親は inetOrgPerson であるとします。ExampleOrganization というオブジェクトクラスを作成し、そのオブジェクトクラスが ExampleDepartmentNumber と ExampleEmergencyPhoneNumber 属性を許可するようにします。ExampleOrganization の親は organization オブジェクトクラスであるとします。

  新しいオブジェクトクラスは、LDAP v3 スキーマ形式では次のようになります。

```
objectclasses: (1.3.6.1.4.1.42.2.27.999.1.2.3 NAME 'ExamplePerson'
 DESC 'Example Person Object Class' SUP inetorgPerson STRUCTURAL MAY
 (ExampleDepartmentNumber $ ExampleEmergencyPhoneNumber) )
objectclasses: (1.3.6.1.4.1.42.2.27.999.1.2.4 NAME
 'ExampleOrganization' DESC 'Example Organization Object Class' SUP
 organization STRUCTURAL MAY (ExampleDepartmentNumber
 $ ExampleEmergencyPhoneNumber) )
```

  または、これらのすべての属性を許可する 1 つのオブジェクトクラスを作成することができます。属性を使う必要があるエントリで、そのオブジェクトクラスを使用できます。 1 つのオブジェクトクラスは、次のようになります。

```
objectclasses: (1.3.6.1.4.1.42.2.27.999.1.2.5 NAME 'ExampleEntry'
 DESC  'Example Auxiliary Object Class' SUP top  AUXILIARY MAY
 (ExampleDepartmentNumber $ ExampleEmergencyPhoneNumber) )
```

  新しい ExampleEntry オブジェクトクラスには、構造上のオブジェクトクラスに関係なく任意のエントリで使用できることを示す AUXILIARY が付いています。

新しいオブジェクトクラスを実装する方法を決めるときは、次の点に留意します。

- 複数の STRUCTURAL オブジェクトクラスを作成すると、作成および管理するスキーマ要素の数も増える。

  一般に、要素の数が少なければ、管理の手間も少なくて済みます。スキーマに複数のオブジェクトクラスを追加することを計画している場合は、1 つのオブジェクトクラスにまとめた方が簡単な場合があります。
- 複数の STRUCTURAL オブジェクトクラスを作成する場合は、より厳密かつ注意深いデータ設計が必要となる。

  データを厳密に設計するには、個々のデータを配置するオブジェクトクラス構造を考慮する必要があります。この制限は役立つ場合とわずらわしい場合があります。
- 複数のタイプのオブジェクトクラス構造に入れたいデータがある場合は、1 つの AUXILIARY オブジェクトクラスを使用した方がデータ設計が簡単になる。

  たとえば、preferredOS 属性を人のエントリとグループエントリの両方に設定するとします。このような場合は、1 つのオブジェクトクラスを作成して、そのクラスでこの属性が許可されるようにします。
- 目的に合ったグループを構成する実際のオブジェクトとグループ要素に関連するオブジェクトクラスを設計する。
- 新しいオブジェクトクラスに必須の属性を設定しない。

  必須の属性を設定するとスキーマに柔軟性がなくなります。新しいオブジェクトクラスを作成する場合は、必須の属性より許可の属性にするようにします。

  新しいオブジェクトクラスを定義したら、そのオブジェクトクラスの許可された属性と必須の属性、および継承するオブジェクトクラスを決める必要があります。

新しい属性を定義する場合

ディレクトリのエントリに格納する必要がある情報の中に既存の属性がサポートしていないものがある場合は、新しい属性を追加します。できるかぎり、標準属性を使用するようにします。デフォルトのディレクトリスキーマにある属性を探し、それらを新しいオブジェクトクラスに関連付けて使用します。

たとえば、person、organizationalPerson、または inetOrgPerson の各オブジェクトクラスがサポートしている以外の情報を、個人のエントリに格納したい場合があります。ディレクトリに生年月日を格納する場合、Directory Server の標準スキーマには対応する属性がありません。 dateOfBirth という新しい属性を作成できます。この属性を許可する新しい補助クラスを定義して、この属性をエントリで使用できるようにします。

カスタムスキーマファイルを作成する場合

カスタムスキーマファイルを作成するとき、特にレプリケーションを使用する場合は、次の点に注意する必要があります。

- 新たに追加したスキーマ要素をオブジェクトクラスで使用するためには、事前にすべての属性が定義されている必要があります。属性とオブジェクトクラスは同じスキーマファイル内で定義できます。
- 作成する各カスタム属性またはオブジェクトクラスは、1 つのスキーマファイル内でだけ定義されている必要があります。この方法により、サーバーが最近作成されたスキーマをロードするときに、以前の定義の上書きを防ぎます。Directory Server は、最初に数字順、次にアルファベット順でスキーマファイルをロードします。
- 新しいスキーマ定義を手動で作成するときは、一般にその定義を `99user.ldif` ファイルに追加する方法が最も適しています。

  LDAP を使用してスキーマ要素を更新すると、新しい要素が自動的に `99user.ldif` ファイルに書き込まれます。このとき、もしほかのカスタムスキーマファイルも使用していると、そのファイルに対して行なったスキーマ定義の変更が、上書きされる可能性があります。`99user.ldif` ファイルのみを使用すると、スキーマ要素の重複と、スキーマの変更の上書きを防ぐことができます。
- Directory Server はスキーマファイルを英数字順に読み込む、つまり、数字が小さいものから先に読み込むため、カスタムスキーマファイルの名前を次のように指定する必要があります。

  `[00-99]` *filename*`.ldif`

  この数字は、すでに定義されているどのディレクトリ標準スキーマよりも大きな値にする必要があります。

  スキーマファイルの名前を標準のスキーマファイルより小さい数字で指定すると、そのスキーマを読み込むときにサーバーにエラーが発生することがあります。また、標準の属性およびオブジェクトクラスはすべて、カスタムスキーマ要素が読み込まれたあとに読み込まれることになってしまいます。
- Directory Server は内部スキーマ管理用に順番が最後のファイルを使用するため、カスタムスキーマファイル名を数字順またはアルファベット順で、`99user.ldif` より大きくならないようにしてください。

  たとえば、スキーマファイルを作成し、`99zzz.ldif` という名前を付けた場合、次にスキーマを更新すると、`X-ORIGIN` の値が `'user defined'` であるすべての属性が `99zzz.ldif` に書き込まれます。その結果、重複した情報を持つ 2 つの LDIF ファイルが存在し、`99zzz.ldif` ファイル内のいくつかの情報が消去される可能性があります。

- 原則として、追加するカスタムスキーマ要素の識別には、次の 2 つの項目を使用します。
  - カスタムスキーマファイルの X-ORIGIN フィールドに指定されている 'user defined'
  - ほかの管理者カスタムスキーマ要素を理解しやすくするため、X-ORIGIN フィールドの 'Example.com Corporation defined' のような、よりわかりやすいラベル。たとえば、X-ORIGIN ('user defined' 'Example.com Corporation defined') など

    スキーマ要素を手動で追加し、X-ORIGIN フィールドに 'user defined' を使用しない場合、このスキーマ要素は DSCC に読み取り専用で表示されます。

    'user defined' という値は、LDAP または DSCC を使用してカスタムスキーマ定義を追加する場合は、サーバーによって自動的に追加されます。ただし、X-ORIGIN フィールドによりわかりやすい値を追加しないと、どのスキーマが関連しているかをあとで理解することが難しくなります。

変更は自動的にはレプリケートされないため、カスタムスキーマファイルはすべてのサーバーに手動で伝達します。

ディレクトリスキーマを変更すると、サーバーはスキーマがいつ変更されたのかを示すタイムスタンプを記録します。各レプリケーションセッションの最初に、サーバーはコンシューマのタイムスタンプとこのタイムスタンプを比較し、必要であればスキーマの変更をプッシュします。カスタムスキーマファイルについては、サーバーは 99user.ldif ファイルに関連付けられている 1 つのタイムスタンプだけを維持します。つまり、カスタムスキーマファイルに加えた変更、または 99user.ldif 以外のファイルに対する変更は、レプリケートされません。このため、トポロジ全体にすべてのスキーマ情報が行き渡るように、カスタムスキーマファイルをほかのすべてのサーバーに伝達する必要があります。

LDAP 経由での属性タイプの管理

この節では、LDAP 経由で属性タイプを作成、表示、および削除する方法を説明します。

属性タイプの作成

cn=schema エントリは複数の値を持つ属性 attributeTypes があり、ディレクトリスキーマの各属性タイプの定義を格納します。それらの定義は ldapmodify(1) コマンドを使用して追加できます。

新しい属性タイプの定義とユーザー定義属性タイプの変更は、99user.ldif ファイルに保存されます。

各属性タイプ定義には、新しい属性タイプを定義する 1 つ以上の OID を指定する必要があります。新しい属性タイプには、少なくとも次の要素を使用することを考慮してください。

- 属性 **OID**。属性のオブジェクト識別子に相当します。OID はスキーマオブジェクトを一意に識別する文字列で、通常は小数点で区切られた数値です。

  LDAP v3 に厳密に準拠するには、有効な数値 OID を指定する必要があります。OID の詳細または企業のプレフィックスを要求するには、`iana@iana.org` の IANA (Internet Assigned Number Authority) に電子メールを送信するか、IANA Web サイト (`http://www.iana.org`)を参照してください。
- 属性名。属性の一意の名前に相当します。その属性タイプとも呼ばれます。属性名はアルファベットから始まる必要があり、ASCII 文字、数字、ハイフンだけが有効です。

  属性名には大文字を含めることもできますが、LDAP クライアントでは属性を区別するために、大文字と小文字で区別すべきではありません。RFC 4512 (`http://www.ietf.org/rfc/rfc4512.txt`) のセクション 2.5 に従って、属性名は大文字と小文字を区別しないで扱う必要があります。

  オプションで、属性タイプに代替の属性名 (エイリアスとも呼ばれる) を含めることもできます。
- 属性の説明。属性の目的を説明する短い説明文です。
- 構文。OID によって参照され、属性に保持されているデータを説明します。

  OID による属性構文は RFC 4517 (`http://www.ietf.org/rfc/rfc4517.txt`) に示されています。
- 使用できる値の数。デフォルトで、属性は複数の値を持つことができますが、1 つの値に制限することができます。

▼ 属性タイプを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **RFC 4517 (`http://www.ietf.org/rfc/rfc4517.txt`) に指定された構文に従って、属性タイプ定義を準備します。**

2. **`ldapmodify`(1) コマンドを使用して、属性タイプ定義を追加します。**

   Directory Server によって、指定した定義に `X-ORIGIN 'user defined'` が追加されます。

例 11–1 属性タイプの作成

次の例では、ldapmodify コマンドを使用して、ディレクトリ文字列構文で新しい属性タイプを追加します。

```
$ cat blogURL.ldif
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.2.3.4.5.6.7
 NAME ( 'blog' 'blogURL' )
 DESC 'URL to a personal weblog'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
 SINGLE-VALUE )

$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - -f blogURL.ldif
Enter bind password:
modifying entry cn=schema

$
```

本稼働環境では、1.2.3.4.5.6.7 ではなく、有効な一意の OID を指定します。

属性タイプの表示

cn=schema エントリには複数の値を持つ属性 attributeTypes があり、ディレクトリスキーマの各属性タイプの定義を格納します。それらの定義は、ldapsearch(1) コマンドを使用して読み取ることができます。

▼ 属性タイプを表示する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ディレクトリスキーマに現在存在するすべての属性タイプ定義を表示するには、ldapsearch コマンドを使用します。

例 11–2 属性タイプの表示

次のコマンドは、すべての属性タイプの定義を表示します。

```
$ ldapsearch -T -b cn=schema "(objectclass=*)" attributeTypes
```

-T オプションにより、ldapsearch コマンドは LDIF 行を折りたたまないため、grep や sed などのコマンドを使用して、出力を簡単に操作できます。次に、grep コマンドを

使用して、このコマンドの出力をパイプすると、ディレクトリスキーマのユーザー定義拡張のみを表示できます。次に例を示します。

```
$ ldapsearch -T -b cn=schema "(objectclass=*)" attributeTypes | grep "user defined"
 attributeTypes: ( 1.2.3.4.5.6.7 NAME ( 'blog' 'blogURL' )
 DESC 'URL to a personal weblog'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE
 X-ORIGIN 'user defined' )
```

属性タイプの削除

cn=schema エントリには複数の値を持つ属性 attributeTypes があり、ディレクトリスキーマの各属性タイプの定義を格納します。X-ORIGIN 'user defined' を含む定義を削除するには、ldapmodify(1) コマンドを使用します。

スキーマは cn=schema 内の LDAP ビューによって定義されるため、ldapsearch ユーティリティーおよび ldapmodify ユーティリティーを使用してスキーマをオンラインで表示、変更することができます。しかし、削除できるスキーマ要素は、X-ORIGIN フィールドに 'user defined' という値が設定されている要素だけです。サーバーは他の定義を削除しません。

ユーザー定義属性の変更は、ファイル 99user.ldif に保存されます。

▼ 属性タイプを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 削除する属性タイプの定義を表示します。
   詳細については、304 ページの「属性タイプを表示する」を参照してください。
2. ldapmodify(1) コマンドを使用して、スキーマに表示される属性タイプ定義を削除します。

例 11–3 属性タイプの削除

次のコマンドは、例 11–1で作成した属性タイプを削除します。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=schema
changetype: delete
```

```
delete: attributeTypes
attributeTypes: ( 1.2.3.4.5.6.7 NAME ( 'blog' 'blogURL' )
 DESC 'URL to a personal weblog'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE
 X-ORIGIN 'user defined' )
^D
```

Directory Server によって追加された `X-ORIGIN 'user defined'` を含めて、このスキーマ定義を拡張として分類する必要があります。

LDAP 経由でのオブジェクトクラスの管理

この節では、LDAP 経由で、オブジェクトクラスを作成、表示、および削除する方法を説明します。

オブジェクトクラスの作成

`cn=schema` エントリには、ディレクトリスキーマの各オブジェクトクラスの定義を格納し、複数の値を持つ属性 `objectClasses` があります。それらの定義は `ldapmodify`(1) コマンドを使用して追加できます。

新しいオブジェクトクラス定義とユーザー定義オブジェクトクラスへの変更は `99user.ldif` ファイルに保存されます。

ほかのオブジェクトクラスから継承する複数のオブジェクトクラスを作成するときは、最初に親オブジェクトクラスを作成する必要があります。新しいオブジェクトクラスがカスタム属性を使用するときは、その属性も事前に定義しておく必要があります。

各オブジェクトクラス定義に、1 つ以上の OID を指定する必要があります。新しいオブジェクトクラスには少なくとも次の要素を使用することを考慮してください。

- オブジェクトクラス **OID**。オブジェクトクラスのオブジェクト識別子に相当します。OID はスキーマオブジェクトを一意に識別する文字列で、通常は小数点で区切られた数値です。

  LDAP v3 に厳密に準拠するには、有効な数値 OID を指定する必要があります。OID の詳細または企業のプレフィックスを要求するには、`iana@iana.org` の IANA (Internet Assigned Number Authority) に電子メールを送信するか、IANA Web サイト (`http://www.iana.org`)を参照してください。
- オブジェクトクラス名。オブジェクトクラスの一意の名前に相当します。
- 親オブジェクトクラス。このオブジェクトクラスが属性を継承する既存のオブジェクトクラスです。

  このオブジェクトクラスをほかの特定のオブジェクトクラスから継承させない場合は、`top` を使用します。

一般に、ユーザーエントリに対して属性を追加する場合、親オブジェクトは inetOrgPerson オブジェクトクラスになります。企業エントリに対して属性を追加する場合、親オブジェクトは通常 organization または organizationalUnit になります。グループエントリに対して属性を追加する場合、親オブジェクトは通常 groupOfNames または groupOfUniqueNames になります。

- 必須属性。このオブジェクトクラスに存在する必要がある属性を示し、定義します。
- 使用可能な属性。このオブジェクトクラスに存在可能な追加の属性を示し、定義します。

▼ オブジェクトクラスを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **RFC 4517 (**http://www.ietf.org/rfc/rfc4517.txt**)** に指定された構文に従って、オブジェクトクラス定義を準備します。

2. ldapmodify**(1)** コマンドを使用して、オブジェクトクラス定義を追加します。

   Directory Server によって、指定した定義に X-ORIGIN 'user defined' が追加されます。

例 11–4 オブジェクトクラスの作成

次の例では、ldapmodify コマンドを使用して、新しいオブジェクトクラスを追加します。

```
$ cat blogger.ldif
dn: cn=schema
changetype: modify
add: objectClasses
objectClasses: ( 1.2.3.4.5.6.8
 NAME 'blogger'
 DESC 'Someone who has a blog'
 SUP inetOrgPerson
 STRUCTURAL
 MAY blog )

$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w - -f blogger.ldif
Enter bind password:
modifying entry cn=schema

$
```

生産環境では、1.2.3.4.5.6.8 ではなく、有効な一意の OID を指定します。

オブジェクトクラスの表示

cn=schema エントリには、ディレクトリスキーマの各オブジェクトクラスの定義を格納し、複数の値を持つ属性 objectClasses があります。それらの定義は、ldapsearch(1) コマンドを使用して読み取ることができます。

▼ オブジェクトクラスを表示する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ldapsearch コマンドを使用して、ディレクトリスキーマに現在存在するすべてのオブジェクトクラス定義を表示します。

例 11–5 オブジェクトクラスの表示

次のコマンドは、すべてのオブジェクトクラスの定義を表示します。

```
$ ldapsearch -T -b cn=schema "(objectclass=*)" objectClasses
```

-T オプションにより、ldapsearch コマンドは LDIF 行を折りたたまないため、grep や sed などのコマンドを使用して、出力を簡単に操作できます。次に、grep コマンドを使用して、このコマンドの出力をパイプすると、ディレクトリスキーマのユーザー定義拡張のみを表示できます。次に例を示します。

```
$ ldapsearch -T -b cn=schema "(objectclass=*)" objectClasses | grep "user defined"
objectClasses: ( 1.2.3.4.5.6.8 NAME 'blogger'
 DESC 'Someone who has a blog' STRUCTURAL MAY blog
 X-ORIGIN 'user defined' )
$
```

オブジェクトクラスの削除

cn=schema エントリには、ディレクトリスキーマの各オブジェクトクラスの定義を格納し、複数の値を持つ属性 objectClasses があります。X-ORIGIN 'user defined' を含む定義を削除するには、ldapmodify(1) コマンドを使用します。

スキーマは cn=schema 内の LDAP ビューによって定義されるため、ldapsearch ユーティリティーおよび ldapmodify ユーティリティーを使用してスキーマをオンラインで表示、変更することができます。しかし、削除できるスキーマ要素は、X-ORIGIN フィールドに 'user defined' という値が設定されている要素だけです。サーバーは他の定義を削除しません。

ユーザー定義の要素の変更は、 99user.ldif ファイルに保存されます。

▼ オブジェクトクラスを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 削除するオブジェクトクラス定義を表示します。
   詳細については、308 ページの「オブジェクトクラスを表示する」を参照してください。
2. `ldapmodify(1)` コマンドを使用して、スキーマに表示されるオブジェクトクラス定義を削除します。

例 11–6 オブジェクトクラスの削除

次のコマンドは、例 11–4 で作成したオブジェクトクラスを削除します。

```
$ ldapmodify -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=schema
changetype: delete
delete: objectClasses
objectClasses: ( 1.2.3.4.5.6.8 NAME 'blogger' DESC 'Someone who has a blog'
 STRUCTURAL MAY blog X-ORIGIN 'user defined' )
^D
```

Directory Server によって追加された `X-ORIGIN 'user defined'` を含めて、このスキーマ定義を拡張として分類する必要があります。

Directory Server スキーマの拡張

スキーマに新しい属性を追加する場合は、それらの属性を持つオブジェクトクラスを新しく作成する必要があります。必要な属性のほとんどが含まれている既存のオブジェクトクラスに対して、新たに必要となった属性を追加すると、LDAP クライアントとの相互運用性が低下するためです。

Directory Server と既存の LDAP クライアントとの相互運用性は、標準の LDAP スキーマに依存しています。標準スキーマを変更すると、サーバーのアップグレード時にも問題が発生します。同様の理由から、標準スキーマの要素を削除することはできません。

Directory Server スキーマは `cn=schema` エントリの属性に保存されます。設定エントリと同様に、これは、サーバーの起動中にファイルから読み取られる、スキーマの LDAP ビューです。

Directory Server スキーマの拡張に使用する方法は、スキーマ拡張を保存するファイル名を制御するかどうかによって異なります。さらに、レプリケーションによってコンシューマに変更をプッシュするかどうかによっても異なります。次の表を参照して、特定の状況で実行する手順を判断してください。

表 11–1　スキーマの拡張方法

作業	参照先
レプリケーションを使用しない。カスタムスキーマファイルを追加して、スキーマを拡張する。	311 ページの「カスタムスキーマファイルによってスキーマを拡張する」
LDAP を経由してスキーマを拡張する。	312 ページの「LDAP によりスキーマを拡張する」
レプリケーションを使用する。すべてのサーバーでカスタムスキーマファイルのファイル名を維持する。	311 ページの「カスタムスキーマファイルによってスキーマを拡張する」
レプリケーションを使用する。マスターレプリカにカスタムスキーマファイルを追加して、スキーマを拡張する。次に、レプリケーションメカニズムによって、そのスキーマ拡張をコンシューマサーバーにコピーする。	313 ページの「スキーマファイルとレプリケーションを使用してスキーマを拡張する」

オブジェクトクラス、属性、ディレクトリスキーマの詳細と、スキーマの拡張のガイドラインについては、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「ディレクトリスキーマの設計」を参照してください。標準属性およびオブジェクトクラスについては、『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』を参照してください。

この節では、ディレクトリスキーマを拡張する様々な方法を説明します。

カスタムスキーマファイルによるスキーマの拡張

スキーマファイルは LDIF ファイルで *instance-path* `/config/schema/` にあります。*instance-path* は、Directory Server インスタンスが存在するファイルシステムディレクトリに対応します。たとえば、インスタンスは `/local/ds/` などにあります。このファイルは Directory Server と Directory Server に依存するすべてのサーバーが使用する標準スキーマを定義します。ファイルと標準スキーマについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』と『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』に説明しています。

サーバーは、起動時に 1 回だけスキーマファイルを読み取ります。スキーマのメモリー内の LDAP ビューの `cn=schema` 内にファイルの LDIF の内容が追加されます。ス

キーマ定義の順序には意味があるため、スキーマファイルの名前の先頭には番号がつけられ、英数字順に読み込まれます。このディレクトリに含まれるスキーマファイルには、インストール時に定義されたシステムユーザーだけが書き込み処理を実行できます。

スキーマを LDIF ファイルに直接定義するときは、`X-ORIGIN` フィールドの値として `'user defined'` を指定することはできません。この値は、`cn=schema` の LDAP ビューで定義されるスキーマ要素用に予約されており、これは `99user.ldif` ファイルに表示されます。

`99user.ldif` ファイルには、`cn=schema` エントリと、コマンド行または DSCC から追加されたすべてのスキーマ定義の追加 ACI が含まれます。新しいスキーマ定義を追加すると、`99user.ldif` ファイルは上書きされます。このファイルを変更するときは、変更が最新になるように、サーバーを直ちに再起動する必要があります。

他のスキーマファイルに定義されている標準のスキーマを変更しないでください。ただし、新しいファイルを追加して、新しい属性やオブジェクトクラスを定義することはできます。たとえば、複数のサーバーに新しいスキーマ要素を定義するには、`98mySchema.ldif` という名前をファイルにその要素を定義し、このファイルをすべてのサーバーのスキーマディレクトリにコピーします。次にすべてのサーバーを再起動して、新しいスキーマファイルを読み込みます。

▼ カスタムスキーマファイルによってスキーマを拡張する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. `98mySchema.ldif` などの独自のスキーマ定義ファイルを作成します。

   スキーマファイルの定義の構文については、RFC 4517 (`http://www.ietf.org/rfc/rfc4517.txt`) で説明されています。

2. **(省略可能) このスキーマが更新をほかのサーバーに送るマスターレプリカである場合は、レプリケーショントポロジでスキーマ定義ファイルを各サーバーインスタンスにコピーします。**

   レプリケーションメカニズムでは、スキーマを含む LDIF ファイルに直接加えた変更は検出されません。そのため、マスターを再起動したあとでも、変更がコンシューマにレプリケートされません。

3. **スキーマ定義ファイルをコピーした各 Directory Server インスタンスを再起動します。**

   サーバーが再起動すると、スキーマ定義が再ロードされ、変更が有効になります。

LDAP によるスキーマの拡張

スキーマは cn=schema 内の LDAP ビューによって定義されるため、ldapsearch ユーティリティーおよび ldapmodify ユーティリティーを使用してスキーマをオンラインで表示、変更することができます。しかし、変更できるスキーマ要素は、X-ORIGIN フィールドに 'user defined' という値が設定されている要素だけです。サーバーは、その他の定義に対するすべての変更処理を拒否します。

新しい要素の定義とユーザー定義の要素に対する変更は、99user.ldif ファイルに保存されます。

▼ LDAP によりスキーマを拡張する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　コマンド行からのスキーマ定義の変更は、正確な入力が必要な値が長いため、エラーを生じがちです。しかし、ディレクトリスキーマの更新が必要なスクリプトにこの機能を指定することができます。

1. ldapmodify(1) コマンドを使用して、各 attributeTypes 属性値を追加または削除します。

   詳細については、303 ページの「属性タイプを作成する」または 305 ページの「属性タイプを削除する」を参照してください。

2. ldapmodify(1) コマンドを使用して、各 objectClasses 属性値を追加または削除してください。

   詳細については、307 ページの「オブジェクトクラスを作成する」または 309 ページの「オブジェクトクラスを削除する」を参照してください。

参照　いずれかの値を変更するには、特定の値を削除してから、新しい値として値を追加する必要があります。この処理は、属性に複数の値を持つために必要です。詳細については、96 ページの「複数値属性の 1 つの値の変更」を参照してください。

スキーマファイルとレプリケーションを使用したスキーマの拡張

カスタムスキーマファイルについては、310 ページの「カスタムスキーマファイルによるスキーマの拡張」を参照してください。次の手順では、レプリケーションメカニズムを使用して、スキーマ拡張をトポロジのすべてのサーバーに伝達する方法を説明します。

▼ スキーマファイルとレプリケーションを使用してスキーマを拡張する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1. 次のいずれかの方法で、スキーマ拡張を準備します。
   - `98mySchema.ldif` などの独自のスキーマ定義ファイルを作成します。
   - `99user.ldif` にスキーマ拡張を追加します。

   スキーマファイルの定義の構文については、RFC 4517 (`http://www.ietf.org/rfc/rfc4517.txt`) で説明されています。
2. スキーマ定義ファイルを配置するマスターサーバーで、`schema_push` コマンドを実行します。

   このスクリプトは実際にはスキーマをレプリカにプッシュしません。代わりに、このスクリプトは、スキーマファイルがロードされるとすぐにレプリケートされるように、特別な属性をスキーマファイルに書き込みます。詳細については、`schema_push`(1M) のマニュアルページを参照してください。
3. スキーマ定義ファイルを配置したマスターサーバーを再起動します。

   レプリケーションメカニズムでは、スキーマを含む LDIF ファイルに直接加えた変更は検出されません。ただし、`schema_push` の実行後にサーバーを再起動すると、サーバーがすべてのスキーマファイルをロードし、レプリケーションメカニズムによって、新しいスキーマがコンシューマにレプリケートされます。

ディレクトリスキーマのレプリケート

2 つのサーバーの間で 1 つまたは複数のサフィックスのレプリケーションを設定するたびに、スキーマ定義も自動的にレプリケートされます。スキーマ定義の自動レプリケーションにより、すべてのレプリカが、コンシューマにレプリケート可能なすべてのオブジェクトクラスと属性を定義する完全な同一のスキーマになります。マスターサーバーもマスタースキーマを格納します。

ただし、スキーマレプリケーションは、LDAP を経由してスキーマを変更した場合でも、即時に実行されません。スキーマレプリケーションは、ディレクトリデータの更新によって、またはスキーマ変更後の最初のレプリケーションセッションの開始時にトリガーされます。

すべてのレプリカにスキーマを適用するには、少なくともすべてのマスターでスキーマ検査を有効にする必要があります。スキーマは、LDAP 処理が行われるマス

ターでチェックされるため、コンシューマの更新時はチェックの必要はありません。パフォーマンスを向上させるために、レプリケーションメカニズムではコンシューマレプリカでのスキーマ検査を行いません。

注–ハブと専用コンシューマでは、スキーマ検査を無効にしないでください。スキーマ検査は、コンシューマのパフォーマンスに影響を与えません。スキーマ検査は常に有効にして、レプリカの内容がそのスキーマと一致するようにします。

コンシューマの初期化時に、マスターサーバーはスキーマをコンシューマに自動的にレプリケートします。さらに、DSCC またはコマンド行ツールから、スキーマが変更された場合にも、マスターサーバーは自動的にスキーマをレプリケートします。デフォルトで、スキーマ全体がレプリケートされます。コンシューマに存在していない追加のスキーマ要素は、コンシューマで作成され、`99user.ldif` ファイルに保存されます。

たとえば、マスターサーバーの起動時に、サーバーの `98mySchema.ldif` ファイルにスキーマ定義が含まれているとします。さらに、ほかのサーバー、マスター、ハブ、または専用コンシューマのいずれかに対するレプリケーションアグリーメントを定義するとします。このマスターからレプリカを初期化すると、レプリケートされたスキーマには `98mySchema.ldif` からの定義が含まれますが、レプリカサーバー側の `99user.ldif` にもこの定義が格納されます。

コンシューマの初期化時にスキーマがレプリケートされたあとで、マスター側の `cn=schema` でスキーマを変更すると、マスターはスキーマ全体をコンシューマにもレプリケートします。このように、コマンド行ユーティリティーまたは DSCC からマススキーマに加えた変更は、コンシューマにレプリケートされます。これらの変更はマスターの `99user.ldif` に保存され、先述した同じメカニズムによって、変更がコンシューマの `99user.ldif` にも保存されます。

レプリケート環境で整合性のあるスキーマを維持するには、次のガイドラインに留意します。

- コンシューマサーバーのスキーマを変更しない。

  コンシューマサーバーのスキーマを変更すると、レプリケーションエラーが発生する可能性があります。これは、コンシューマのスキーマの違いによって、サプライヤからの更新がコンシューマのスキーマに一致しなくなる可能性があるためです。

- マルチマスターレプリケーション環境では、1 つのマスターサーバーでスキーマを変更する。

  2 つのマスターサーバーのスキーマを変更すると、最後に更新されたマスターのスキーマがコンシューマに伝達されます。コンシューマのスキーマがほかのマスターのスキーマと一致しなくなる可能性があります。

部分レプリケーションを設定する場合は、次のガイドラインにも留意します。

- 部分レプリケーションの設定ではサプライヤがスキーマをプッシュするため、部分コンシューマレプリカのスキーマは、マスターレプリカのスキーマのコピーとなります。このため、適用される部分レプリケーション設定には対応しません。
- 一般に、Directory Server はスキーマ違反を回避するために、スキーマに定義されている各エントリのすべての必須属性をレプリケートします。必要な属性をフィルタで除外する部分レプリケーションを設定する場合、スキーマ検査を無効にする必要があります。
- スキーマ検査で部分レプリケーションが有効になっていると、レプリカをオフラインで初期化できなくなる可能性があります。Directory Server では、必要な属性をフィルタで除外した場合に、LDIF からデータをロードできません。
- 部分コンシューマレプリカで、スキーマ検査を無効にしている場合、その部分コンシューマレプリカが存在するサーバーインスタンス全体に、スキーマ検査が適用されません。その結果、サプライヤレプリカが、部分コンシューマと同じサーバーインスタンスに設定されません。

スキーマレプリケーションの制限

デフォルトでは、レプリケーションメカニズムによってスキーマがレプリケートされるたびに、スキーマ全体がコンシューマに送信されます。スキーマ全体をコンシューマに送信することが望ましくない状況は 2 つあります。

- DSCC またはコマンド行から `cn=schema` に加える変更は、ユーザー定義のスキーマ要素だけに対象が限定され、すべての標準スキーマは変更されません。スキーマを頻繁に変更する場合、未変更のスキーマ要素を含む大規模な要素セットを毎回送信することはパフォーマンスに影響します。ユーザー定義のスキーマ要素だけをレプリケートすることで、レプリケーションとサーバーのパフォーマンスを向上できます。
- Directory Server のマスターが Directory Server 5.1 のコンシューマにレプリケートすると、これらのバージョンの設定属性のスキーマが異なり、競合が発生します。この場合は、ユーザー定義のスキーマ要素のみをレプリケートする必要があります。

注 – Directory Server は `11rfc2307.ldif` スキーマファイルを使用します。このスキーマファイルは、RFC 2307 (`http://www.ietf.org/rfc/rfc2307.txt`) に準拠しています。

Directory Server 5.2 より前の Directory Server のバージョンでは `10rfc2307.ldif` スキーマファイルを使用します。

▼ スキーマレプリケーションを制限する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- ユーザー定義のスキーマのみがレプリケートされるように、スキーマレプリケーションを制限します。

  ```
  $ dsconf set-server-prop -h host -p port repl-user-schema-enabled:on
  ```

  デフォルト値の `off` を使うことで、必要に応じてスキーマ全体をレプリケートできます。

◆ ◆ ◆ 第 12 章

Directory Server のインデックス

書籍の索引と同様に、Directory Server のインデックスを利用することで、検索文字列とディレクトリの内容への参照を関連づけ、検索を速く行うことができます。

インデックスのタイプとインデックスのチューニングについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 6 章「Directory Server Indexing」を参照してください。

この章の内容は次のとおりです。

インデックスの管理

この節では、特定の属性のインデックスを管理する方法について説明します。この節では、インデックスの作成、変更、削除について説明します。仮想リスト表示 (VLV) 操作に固有の手順については、324 ページの「ブラウズインデックスの管理」を参照してください。

▼ インデックスを一覧表示する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 既存のインデックスとそれらのプロパティーを一覧表示するには、次のコマンドを使用します。

```
$ dsconf list-indexes -h host -p port -v suffix-DN
```

▼ インデックスを作成する

注 – 新しいシステムインデックスを作成することはできません。システムインデックスは、Directory Server によって内部的に定義されているものだけが保持されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 新しいインデックス設定を作成します。

`dsconf create-index` コマンド行ユーティリティーを使用して、インデックスを作成する属性を指定し、新しいインデックス情報を設定します。

たとえば `preferredLanguage` 属性のインデックスエントリを作成するには、次のコマンドを使用します。

```
$ dsconf create-index -h host -p port dc=example,dc=com preferredLanguage
```

注 – コマンド `dsconf create-index` はインデックス設定を行いますが、実際に検索に必要なインデックスファイルを作成するわけではありません。インデックスファイルを生成するとパフォーマンスに影響を与える可能性があります。インデックス作成手順をより厳密に制御するには新しいインデックス設定が作成されたあとに、手動でインデックスファイルを生成します。

インデックスを作成する場合は、常に属性の基本名を使用します。属性の別名は使用しないでください。属性の基本名は、スキーマでその属性に一覧表示された最初の名前です。たとえば、`userid` 属性では `uid` が基本名になります。

2 (省略可能) インデックスのプロパティーを設定するには `dsconf set-index-prop` コマンドを使用します。

`dsconf create-index` コマンドはデフォルトのプロパティーでインデックスを作成します。これらのプロパティーを変更する場合は、`dsconf set-index-prop` コマンドを使用します。インデックスのプロパティーの変更の詳細については、319 ページの「インデックスを変更する」を参照してください。

3 インデックスファイルを生成します。

319 ページの「インデックスを生成する」を参照してください。

4 インデックスを作成するすべてのサーバーに対し、前の手順を繰り返します。

▼ インデックスを変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. インデックスのプロパティーを変更します。

   ```
   $ dsconf set-index-prop -h host -p port suffix-DN attr-name property:value
   ```

   たとえば、`preferredLanguage` インデックスの近似インデックス `approx-enabled` を有効にするには、次のコマンドを使用します。

```
$ dsconf set-index-prop -h host -p port dc=example,dc=com preferredLanguage approx-enabled:on
```

   各インデックスについて次のプロパティーを変更できます。

   - `eq-enabled` 等価
   - `pres-enabled` プレゼンス
   - `sub-enabled` 部分文字列

   変更する必要がありそうなプロパティーの 1 つに、オプションの `nsMatchingRule` 属性があります。この属性には、サーバーで既知のマッチングルールの OID が含まれます。これは国際化インデックスの言語照合順序の OID と、`CaseExactMatch` のようなその他のマッチングルールを有効にします。サポートされるロケールとそれらの関連付けられた照合順序の OID の一覧については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

   インデックス設定属性の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』を参照してください。

2. 新しいインデックスを再生成します。

   319 ページの「インデックスを生成する」を参照してください。

3. 変更した属性インデックスを含むすべてのサーバーに対し、前の手順を繰り返します。

▼ インデックスを生成する

次の手順では、新しいインデックスまたは変更したインデックスを検索できるように、インデックスファイルを生成します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- インデックスファイルは、次のいずれかの方法で生成します。

- オンラインで新しいインデックスファイルを生成します。

  ```
  $ dsconf reindex -h host -p port [-t attr] suffix-DN
  ```

  `-t` は、すべての属性ではなく、指定した属性のみインデックスを再生成するように指定します。

  たとえば、`preferredLanguage` インデックスを再生成するには、次のように入力します。

  ```
  $ dsconf reindex -h host -p port -t preferredLanguage dc=example,dc=com
  ```

  `dsconf reindex` コマンドの実行中も、サーバーからサフィックスの内容を使用できます。ただし、コマンドが完了するまで検索のインデックスは作成されません。インデックスの再生成は多くのリソースを消費するタスクであるため、サーバー上のその他の処理のパフォーマンスに影響を生じることがあります。
- オフラインで新しいインデックスファイルを生成します。

  ```
  $ dsadm reindex -t attr instance-path suffix-DN
  ```

  たとえば、`preferredLanguage` インデックスを再生成するには、次のように入力します。

  ```
  $ dsadm reindex -t preferredLanguage /local/ds dc=example,dc=com
  ```
- サフィックスを再初期化することによって、オフラインで速やかにすべてのインデックスを再生成します。

  サフィックスを再初期化すると、すべてのインデックスファイルが自動的に再生成されます。ディレクトリのサイズによりますが、多くの場合、複数の属性のインデックスを再生成するよりサフィックスの再初期化の方が高速です。ただし、初期化時にはサフィックスを使用できません。詳細については、323 ページの「再初期化によるサフィックスのインデックスの再生成」を参照してください。

注 – `dsconf import` か `dsconf reindex` のいずれか、または複数のサフィックスで並行して両方のコマンドを実行すると、トランザクションログが大きくなり、パフォーマンスに悪影響を及ぼすことがあります。

▼ インデックスを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 属性に設定されているすべてのインデックスを削除します。

  ```
  $ dsconf delete-index -h host -p port suffix-DN attr-name
  ```

  たとえば、次のコマンドは `preferredLanguage` 属性のすべてのインデックスを削除します。

  ```
  $ dsconf delete-index -h host -p port dc=example,dc=com preferredLanguage
  ```

  デフォルトのインデックスを削除する場合は、Directory Server の機能に影響する可能性があるため、十分に注意してください。

インデックスリストのしきい値の変更

システムインデックスリストのサイズがインデックスリストのしきい値を超えると、検索が遅くなることがあります。インデックスリストのしきい値は各インデックスキーの値の最大数です。インデックスリストのしきい値のサイズを超えているかどうかを判断するには、アクセスログを調べます。アクセスログ `RESULT` メッセージの末尾の `notes=U` フラグは、インデックスを使用しない検索が実行されたことを示します。同じ接続と操作の前の `SRCH` メッセージは、使用された検索フィルタを示します。次の 2 行の例は、10,000 エントリを返す `cn=Smith` のインデックスを使用しない検索を追跡します。メッセージからタイムスタンプが削除されています。

```
conn=2 op=1 SRCH base="o=example.com" scope=0 filter="(cn=Smith)"
conn=2 op=1 RESULT err=0 tag=101 nentries=10000 notes=U
```

システムで頻繁にインデックスリストのしきい値を超える場合は、しきい値を増加して、パフォーマンスを向上させることを検討してください。次の手順では `dsconf set-server-prop` コマンドを使用して、`all-ids-threshold` プロパティーを変更します。インデックスのチューニングと `all-ids-threshold` プロパティーの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Tuning Indexes for Performance」を参照してください。

▼ インデックスリストのしきい値を変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. インデックスリストのしきい値を調整します。

次のレベルでインデックスリストのしきい値を調整できます。

- インスタンスレベル

  ```
  dsconf set-server-prop -h host -p port all-ids-threshold:value
  ```

- サフィックスレベル

  ```
  dsconf set-suffix-prop -h host -p port suffix-DN all-ids-threshold:value
  ```

- エントリレベル

  ```
  dsconf set-index-prop -h host -p port suffix-DN all-ids-threshold:value
  ```

- 検索のタイプ別インデックスレベル

  ```
  dsconf set-index-prop -h host -p port suffix-DN all-ids-threshold search-type:value
  ```

  search-type は次のいずれかになります。

  - `eq-enabled` 等価
  - `pres-enabled` プレゼンス
  - `sub-enabled` 部分文字列

  `all-ids-threshold` プロパティーは近似インデックスには設定できません。

DSCC を使用して、検索タイプ別にインデックスレベルでしきい値を設定できます。詳細については Directory Server のオンラインヘルプを参照してください。

2 サフィックスインデックスを再生成します。

319 ページの「インデックスを生成する」を参照してください。

3 データベースキャッシュサイズを古い `all IDs` しきい値に合わせて調整しており、サーバーに十分な物理メモリーがある場合は、データベースキャッシュサイズを増やすことをお勧めします。

データベースキャッシュサイズを、`all IDs` しきい値の増加量の 25 パーセント増加します。

つまり、 `all IDs` しきい値を 4000 から 6000 に増加した場合、インデックスリストのサイズの増加を見込んで、データベースキャッシュサイズを約 12 ½ パーセント増加できます。

データベースキャッシュサイズは属性 `dbcachesize` を使用して設定します。業務用サーバーに変更を適用する前に、実験して最適なサイズを見つけてください。

サフィックスのインデックスの再生成

インデックスファイルが壊れた場合、サフィックスのインデックスを再生成して、対応するデータベースディレクトリにインデックスファイルを再作成する必要があ

ります。サフィックスのインデックスを再生成するには、ディレクトリサーバーの実行中にサフィックスのインデックスを再生成するか、サフィックスを初期化します。

ディレクトリサーバーの実行中のサフィックスのインデックスの再生成

サフィックスのインデックスの再生成を行うと、サーバーはサフィックスに含まれるすべてのエントリを調べ、インデックスファイルを再作成します。インデックスの再生成中、サフィックスの内容は読み取り専用になります。サーバーは、インデックスを再生成するすべての属性のサフィックス全体を走査する必要があり、数百万のエントリを持つサフィックスの場合、この処理には数時間かかることがあります。かかる時間も設定するインデックスによって異なります。さらに、サフィックスのインデックスの再生成中は、インデックスを使用できず、サーバーのパフォーマンスに影響があります。

▼ サフィックスのすべてのインデックスを再生成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- サフィックスのすべてのインデックスを再生成します。

```
$ dsconf reindex -h host -p port suffix-DN
```

たとえば、`dc=example,dc=com` サフィックスのすべてのインデックスを初期化するには、次のコマンドを使用します。

```
$ dsconf reindex -h host -p port dc=example,dc=com
```

再初期化によるサフィックスのインデックスの再生成

サフィックスを再初期化すると、新しい内容がインポートされます。つまり、サフィックスの内容が置き換えられ、新しいインデックスファイルが作成されます。サフィックスの再初期化は、エントリのロード時に同時にすべての属性のインデックスが作成されるので、複数の属性のインデックスの再生成よりも速く実行することができます。ただし、再初期化中はサフィックスを使用できません。

▼ 再初期化によりサフィックスのインデックスを再生成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 64 ページの「リフェラルを設定し、サフィックスを読み取り専用にする」で説明するように、サフィックスを読み取り専用に設定します。

2 208 ページの「LDIF へのバックアップ」で説明するように、サフィックス全体を LDIF ファイルにエクスポートします。

3 211 ページの「LDIF ファイルからのデータのインポート」で説明するように、同じ LDIF ファイルをインポートして、サフィックスを再初期化します。

初期化中は、サフィックスを利用することはできません。初期化が完了すると、設定されたすべてのインデックスを利用できるようになります。

4 64 ページの「リフェラルを設定し、サフィックスを読み取り専用にする」で説明するように、サフィックスをふたたび書き込み可能にします。

ブラウズインデックスの管理

ブラウズインデックスは、検索結果に対してサーバー側でのソートを要求する検索処理でのみ使用される特別なインデックスです。『Sun Java System Directory Server Enterprise Edition 6.1 Reference』で、Directory Server のブラウズインデックスの仕組みを説明しています。

クライアント検索用のブラウズインデックス

クライアント検索結果のソート用にカスタマイズしたブラウズインデックスを手動で定義する必要があります。ブラウズインデックス、または仮想リスト表示 (VLX) インデックスを作成するには、次の手順を実行します。この節では、ブラウズインデックスエントリの追加または変更の手順とブラウズインデックスの再生成の手順も説明します。

▼ ブラウズインデックスを作成する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1 新しいブラウズインデックスエントリを追加するか、既存のブラウズインデックスエントリを編集するには、`ldapmodify` コマンドを使用します。

手順については、325 ページの「ブラウズインデックスエントリを追加または変更する」を参照してください。

2 `dsconf reindex` コマンドを実行して、サーバーに保持される新しいブラウズインデックスのセットを生成します。

手順については、326 ページの「ブラウズインデックスを再生成する」を参照してください。

▼ ブラウズインデックスエントリを追加または変更する

ブラウズインデックスは、特定のベースエントリとサブツリーに対して指定された検索ごとに異なります。ブラウズインデックスの設定は、エントリを含むサフィックスのデータベース設定に定義されます。

1. ディレクトリサーバーの各ブラウズインデックスに vlvBase、vlvScope、および vlvFilter 属性を設定します。

   これらの属性は、検索のベース、検索の範囲、検索のフィルタを設定します。これらの属性は vlvSearch オブジェクトクラスを使用します。

2. 各ブラウズインデックスに vlvSort 属性を設定します。

   この属性は、インデックスをソートする属性の名前または属性を指定します。このエントリは先頭のエントリの子で、 vlvIndex オブジェクトクラスを使用して、ソートする属性と順番を指定します。

   次の例は、ldapmodify コマンドを使用して、ブラウズインデックス設定エントリを作成します。

```
$ ldapmodify -a -h host -p port -D cn=admin,cn=Administrators,cn=config -w -
Enter bind password:
dn: cn=people_browsing_index, cn=database-name,
cn=ldbm database,cn=plugins,cn=config
objectClass: top
objectClass: vlvSearch
cn: Browsing ou=People
vlvBase: ou=People,dc=example,dc=com
vlvScope: 1
vlvFilter: (objectclass=inetOrgPerson)

dn: cn=Sort rev employeenumber, cn=people_browsing_index,
 cn=database-name,cn=ldbm database,cn=plugins,cn=config
objectClass: top
objectClass: vlvIndex
cn: Sort rev employeenumber
vlvSort: -employeenumber
^D
```

   vlvScope は次のいずれかです。

   - ベースエントリのみの場合 0
   - ベースの直接の子の場合 1
   - ベースをルートにしたサブツリー全体の場合 2

   vlvfilter は、クライアント検索操作で使われる LDAP フィルタと同じフィルタです。すべてのブラウズインデックスエントリは同じ場所に配置されるため、cn の値にはブラウズインデックスの名前を指定しておく必要があります。

vlvSearch エントリは、それぞれが少なくとも 1 つの vlvIndex エントリを持つ必要があります。vlvSort 属性は、ソートする属性とソート順序を定義する属性名のリストです。属性名の前に付けられたダッシュ (-) は、順序を逆にすることを意味します。複数の vlvIndex エントリを定義することで、検索に複数のインデックスを定義できます。前述の例では、次のエントリを追加できます。

```
$ ldapmodify -a -h host -p port
 -D cn=admin,cn=Administrators,cn=config -w -
dn: cn=Sort sn givenname uid, cn=people_browsing_index,
 cn=database-name,cn=ldbm database,cn=plugins,cn=config
objectClass: top
objectClass: vlvIndex
cn: Sort sn givenname uid
vlvSort: sn givenname uid
^D
```

3. ブラウズインデックス設定を変更するには、対応する vlvSearch エントリまたは対応する vlvIndex エントリを編集します。

4. ブラウズインデックスを削除して、サーバーで維持しないようにするには、各 vlvIndex エントリを削除します。

   または、vlvIndex エントリが 1 つだけ存在する場合は、vlvSearch エントリと vlvIndex エントリの両方を削除します。

▼ ブラウズインデックスを再生成する

- ブラウズインデックスエントリを作成したら、指定した属性の新しいブラウズインデックスを生成します。

  ```
  $ dsadm reindex -l -t attr-index instance-path suffix-DN
  ```

  このコマンドは、ディレクトリの内容をスキャンし、ブラウズインデックス用のデータベースファイルを作成します。

  次の例は、前の項で定義したブラウズインデックスを生成します。

  ```
  $ dsadm reindex -l -b database-name -t Browsing /local/ds \
   ou=People,dc=example,dc=com
  ```

  dsadm reindex コマンドの詳細については、dsadm(1M) のマニュアルページを参照してください。

◆ ◆ ◆ 第 13 章

13 Directory Server の属性値と一意性

UID 一意性検査プラグインによって、特定の属性の値をディレクトリまたはサブツリーのすべてのエントリ内で一意にできます。プラグインは、特定の属性に対して既存の値を含むエントリを追加しようとする動作を停止します。また、ディレクトリにすでに存在する値に属性を変更したり、追加する動作を停止します。

UID 一意性検査プラグインはデフォルトでは無効になっています。有効にすると、デフォルトで UID 属性の一意性を確認します。プラグインの新しいインスタンスを作成して、その他の属性値を一意にすることができます。UID 一意性検査プラグインが属性値の一意性を確認できるのは、1 つのサーバー上だけです。

この章の内容は次のとおりです。

属性値の一意性の概要

UID 一意性検査プラグインは、操作前のプラグインです。サーバーがディレクトリの更新を実行する前に、LDAP の追加、変更、DN の変更操作を確認します。プラグインは、操作によって 2 つのエントリが同じ属性値を持つかどうかを判断します。同じ属性を持つ場合、サーバーは操作を停止して、エラー 19 LDAP_CONSTRAINT_VIOLATION をクライアントに返します。

このプラグインは、ディレクトリ内の 1 つ以上のサブツリーや、特定のオブジェクトクラスのエントリ間で、一意性を確保するように設定できます。この設定により、属性値を一意にするエントリのセットが決まります。

他の属性の一意性を確保する必要がある場合は、UID 一意性検査プラグインの複数のインスタンスを定義します。値を一意にする属性ごとに 1 つのプラグインインスタンスを定義します。同じ属性に複数のプラグインインスタンスを用意すること

で、複数のエントリセットでその属性の一意性を個別に確保できます。サブツリーの各セットで特定の属性値は 1 回しか許可されません。

既存のディレクトリで属性値の一意性を有効にしても、サーバーは既存のエントリ間での一意性をチェックしません。一意性が適用されるのは、エントリを追加する時点、あるいは属性が追加または変更される時点です。

デフォルトでは、UID 一意性検査プラグインは無効になっています。これは、このプラグインがマルチマスターレプリケーションに影響を与えるためです。レプリケーションの使用時に UID 一意性検査プラグインを有効にできますが、331 ページの「レプリケーション使用時の一意性検査プラグインの使用」で説明している動作に気を付けてください。

UID とその他の属性の一意性の適用

この 節では、uid 属性に対するデフォルトの一意性検査プラグインを有効にして、設定する方法と、その他の属性の一意性を適用する方法について説明します。

▼ UID 属性の一意性を適用する

dsconf コマンドを使用して UID 一意性検査プラグインを有効にし、設定する方法について、次の手順で説明します。プラグイン設定エントリの DN は、cn=uid uniqueness,cn=plugins,cn=config です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC の使用時には、デフォルトの UID 一意性検査プラグインを変更して、別の属性の一意性を適用しないでください。UID 一意性検査プラグインを使用しない場合は、プラグインを無効にしたまま、329 ページの「その他の属性の一意性を適用する」での説明に従って、ほかの属性に対して新しいプラグインインスタンスを作成します。

1. プラグインを有効にします。

```
$ dsconf enable-plugin -h host -p port "uid uniqueness"
```

2. 一意性を適用するサブツリーの指定方法に従って、プラグイン引数を変更します。
   - 1 つのサブツリーのベース DN を指定するには、次のように入力します。

```
$ dsconf set-plugin-prop -h host -p port "uid uniqueness" argument:uid argument:subtreeBaseDN
```

次に例を示します。

```
$ dsconf set-plugin-prop -h host1 -p 1389 "uid uniqueness" argument:uid \
 argument:dc=People,dc=example,dc=com
```

- 複数のサブツリーを指定するには、各サブツリーの完全ベース DN を値として指定した引数を追加します。

```
$ dsconf set-plugin-prop -h host -p port "uid uniqueness" argument:uid \
 argument:subtreeBaseDN argument:subtreeBaseDN
```

- ベースエントリのオブジェクトクラスに従ってサブツリーを指定するには、引数に次の値を設定します。*baseObjectClass* を持つ各エントリの下位にあるサブツリーに対して、UID 属性の一意性が適用されます。オプションとして、3 番目の引数に *entryObjectClass* を指定すると、このオブジェクトクラスを持つエントリをターゲットとする操作だけで、一意性を適用することもできます。

```
$ dsconf set-plugin-prop -h host -p port "uid uniqueness" argument:attribute=uid \
 argument:markerObjectClass=baseObjectClass argument:entryObjectClass=baseObjectClass
```

- 既存の引数リストに引数を追加するには、次のコマンドを使用します。

```
$ dsconf set-plugin-prop -h host -p port "uid uniqueness" argument+:argument-value
```

3 変更内容を有効にするために、サーバーを再起動します。

▼ その他の属性の一意性を適用する

UID 一意性検査プラグインを使用すると、すべての属性の一意性を適用できます。ディレクトリの cn=plugins,cn=config の下に新しいエントリを作成することによって、プラグインの新しいインスタンスを作成する必要があります。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 新しいプラグインを作成します。

```
$ dsconf create-plugin -h host -p port -H lib-path -F init-func \
 -Y type plugin-name
```

plugin-name は、属性名を含む短い説明的な名前にしてください。たとえば、メール ID 属性の一意性のためにプラグインを作成するには、次のコマンドを使用します。

```
$ dsconf create-plugin -h host1 -p 1389 -H /opt/SUNWdsee/ds6/lib/uid-plugin.so \
 -F NSUniqueAttr_Init -Y preoperation "mail uniqueness"
```

2 プラグインのプロパティーを設定します。

```
$ dsconf set-plugin-prop -h host -p port plugin-name property:value
```

たとえば、メールの一意性検査プラグインのプロパティーを設定するには、次のように入力します。

```
$ dsconf set-plugin-prop -h host1 -p 1389 "mail uniqueness" \
 desc:"Enforce unique attribute values..." version:6.0 \
 vendor:"Sun Microsystems, Inc." depends-on-type:database
```

3 プラグインを有効にします。

```
$ dsconf enable-plugin -h host -p port plugin-name
```

4 プラグインの引数を指定します。

これらの引数は、一意性が適用されるサブツリーの決定方法によって異なります。

- ベース DN に従って 1 つまたは複数のサブツリーを定義するには、最初の引数は一意の値を持つ属性の名前である必要があります。その後の引数は、サブツリーのベースエントリの完全な DN です。

  ```
  $ dsconf set-plugin-prop -h host -p port plugin-name argument:attribute-name \
   argument:subtreeBaseDN argument:subtreeBaseDN...
  ```

- 既存の引数リストに引数を追加するには、次のコマンドを使用します。

  ```
  $ dsconf set-plugin-prop -h host -p port plugin-name argument+:argument-value
  ```

- ベースエントリのオブジェクトクラスに基づいてサブツリーを定義するには、1 番目の引数に `attribute=` *attribute-name* を追加して、一意の値を持つ属性の名前を指定する必要があります。2 番目の引数には、一意性を適用するサブツリーのベースエントリを決める *baseObjectClass* を指定する必要があります。オプションとして、3 番目の引数に *entryObjectClass* を指定すると、このオブジェクトクラスを持つエントリをターゲットとする操作だけで、一意性を適用することもできます。

```
$ dsconf set-plugin-prop -h host -p port plugin-name argument:attribute=attribute-name \
 argument:markerObjectClass=baseObjectClass argument:requiredObjectClass=entryObjectClass
```

すべてのプラグインの引数で、 = 記号の前後に空白文字を入れることはできません。

5 変更内容を有効にするために、サーバーを再起動します。

レプリケーション使用時の一意性検査プラグインの使用

UID 一意性検査プラグインでは、レプリケーションの一部として更新処理が行われた場合は、属性値の検査は一切行われません。これはシングルマスターレプリケーションには影響を与えませんが、プラグインはマルチマスターレプリケーションに対する属性の一意性を自動的に適用できません。

シングルマスターレプリケーションモデル

クライアントアプリケーションによる変更処理はすべてマスターレプリカ上で行われるので、UID 一意性検査プラグインをマスターサーバー上で有効にする必要があります。レプリケートされたサフィックスで一意性を適用するように、プラグインを設定する必要があります。マスターが該当の属性値が一意であることを確認するため、コンシューマサーバー上でプラグインを有効にする必要はありません。

1 つのマスターのコンシューマ上で UID 一意性検査プラグインを有効にしても、レプリケーションやサーバーの通常の操作には影響しません。しかし、パフォーマンスは若干低下する場合があります。

マルチマスターレプリケーションモデル

UID 一意性検査プラグインは、マルチマスターレプリケーションモデルでの使用を想定して設計されていません。マルチマスターレプリケーションは疎整合型のレプリケーションモデルを使用するので、両方のサーバーでプラグインが有効になっていても、同じ属性値が両方のサーバーに同時に追加された場合は検出されません。

ただし、一意性検査を実行している属性がネーミング属性であり、一意性検査プラグインがすべてのマスター上の同じサブツリーの同じ属性に対して有効になっている場合、UID 一意性検査プラグインを使用できます。

これらの条件を満たしている場合、一意性に関する競合は、レプリケーション時のネーミング競合として報告されます。ネーミング競合は手動で解決する必要があります。詳細については、290 ページの「よく発生するレプリケーションの競合の解決」を参照してください。

◆ ◆ ◆ 第 14 章

Directory Server のログ

この章では Directory Server ログの管理方法を説明します。

ログの方針の定義に役立つ情報が必要な場合は、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「ロギング方法の設計」のログポリシー情報を参照してください。

ログファイルとそれらの内容の説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 7 章「Directory Server Logging」を参照してください。

この章の内容は次のとおりです。

ログ解析ツール

Directory Server Resource Kit にはログ解析ツール `logconv` があり、Directory Server のアクセスログを解析できます。ログ解析ツールは使用状況の統計情報を抽出します。さらに、重要なイベントの発生もカウントします。このツールの詳細については、`logconv`(1) のマニュアルページを参照してください。

Directory Server ログの表示

デフォルトでは *instance-path*/logs ファイルにあるサーバーのログを直接表示できます。デフォルトのパスを変更した場合は、次のように dsconf コマンドを使用して、ログファイルの場所を検索できます。

```
$ dsconf get-log-prop -h host -p port log-type path
```

または Directory Service Control Center (DSCC) によってログファイルを表示できます。DSCC ではログエントリを表示し、ソートできます。

次の図に、DSCC の Directory Server のアクセスログの例を示します。

図 14–1 DSCC のアクセスログ

▼ Directory Server のログの末尾を表示する

dsadm コマンドを使用して、指定した行数の Directory Server のログを表示したり、指定した経過時間内に記録されたログエントリを表示したりできます。この例では、エラーログの末尾を表示します。アクセスログの末尾を表示する場合は、show-error-log の代わりに show-access-log を使用してください。

1 特定の経過時間内に記録されたエラーログエントリを表示します。

```
$ dsadm show-error-log -A duration instance-path
```

時間の単位を指定してください。たとえば、24 時間以内に記録されたエラーログエントリを表示するには、次のように入力します。

```
$ dsadm show-error-log -A 24h /local/ds
```

2 指定した行数 (末尾から) のエラーログを表示します。

```
$ dsadm show-error-log -L last-lines instance-path
```

行数は整数で表します。たとえば、最後の 100 行を表示するには、次のように入力します。

```
$ dsadm show-error-log -L 100 /local/ds
```

値を指定しない場合、デフォルトの表示行数は 20 行です。

Directory Server のログの設定

ログファイルのさまざまな側面を変更できます。次のような例があります。

- 監査ログの有効化

  アクセスログやエラーログとは異なり、監査ログはデフォルトでは無効にされています。詳細については、336 ページの「監査ログを有効にする」を参照してください。
- 一般設定
  - ログの有効化または無効化
  - ログのバッファリングの有効化または無効化
  - ログファイルの場所
  - 詳細ログ
  - ログレベル
- ログローテーション設定
  - 一定の時間間隔での新しいログの作成
  - 新しいログファイルが作成されるまでの最大ログファイルサイズ
- ログ削除設定
  - 削除されるまでの最大ファイル経過時間
  - 削除されるまでの最大ファイルサイズ
  - 削除されるまでの最小空きディスク容量

次に示す手順では、ログ設定を変更する方法と監査ログを有効にする方法を示します。

▼ ログ設定を変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 変更するログの設定を表示します。

```
$ dsconf get-log-prop -h host -p port log-type
```

たとえば、既存のエラーログ設定を表示するには、次のように入力します。

```
$ dsconf get-log-prop -h host1 -p 1389 error
Enter "cn=Directory Manager" password:
buffering-enabled         :  off
enabled                   :  on
level                     :  default
max-age                   :  1M
max-disk-space-size       :  100M
max-file-count            :  2
max-size                  :  100M
min-free-disk-space-size  :  5M
path                      :  /tmp/ds1/logs/errors
perm                      :  600
rotation-interval         :  1w
rotation-min-file-size    :  unlimited
rotation-time             :  undefined
verbose-enabled           :  off
```

2 新しい値を設定します。

プロパティーに目的の値を設定します。

```
$ dsconf set-log-prop -h host -p port log-type property:value
```

たとえば、エラーログのローテーション間隔を 2 日に設定するには、次のコマンドを使用します。

```
$ dsconf set-log-prop -h host1 -p 1389 error rotation-interval:2d
```

▼ 監査ログを有効にする

アクセスログやエラーログとは異なり、監査ログはデフォルトでは無効にされています。監査ログを表示するには、最初にログを有効にする必要があります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 監査ログを有効にします。

  ```
  $ dsconf set-log-prop -h host -p port audit enabled:on
  ```

Directory Server ログの手動でのローテーション

きわめて大きくなるログがある場合、任意の時間に手動でログをローテーションすることができます。ローテーションでは、既存のログファイルをバックアップし、新しいログファイルを作成します。

▼ ログファイルを手動でローテーションする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ログファイルをローテーションします。

  ```
  $ dsconf rotate-log-now -h host -p port log-type
  ```

  たとえば、アクセスログをローテーションするには、次のように入力します。

  ```
  $ dsconf rotate-log-now -h host1 -p 1389 access
  ```

◆ ◆ ◆ 第 15 章

Directory Server の監視

様々な方法で Directory Server を監視できます。これらの方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 3 章「Directory Server Monitoring」で説明しています。

この章では、Directory Server で管理上の監視を設定する方法を説明します。

この章の内容は次のとおりです。

Directory Server の SNMP の設定

この節では、SNMP によって監視するためにサーバーを設定する方法を説明します。

Directory Server の SNMP の実装については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Directory Server and SNMP」を参照してください。

▼ SNMP を設定する

この手順の一部として、DSCC を使用してこの作業を実行できます。詳細については、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。手順のその他の部分は、コマンド行を使用した場合にのみ実行できます。

1 **Java ES 管理フレームワークプラグインを有効にします。**

340 ページの「Java ES MF の監視の有効化」の手順を使用します。この手順では、Java ES MF に含まれる Common Agent Container も有効にします。

2 **MIB によって定義され、エージェントにより公開される SNMP 管理対象オブジェクトにアクセスします。**

この手順に必要な作業は、SNMP 管理システムによってまったく異なります。手順については、SNMP 管理システムのマニュアルを参照してください。

MIB の公開時に、この MIB に RFC テキスト ファイルを使用する必要がある場合があります。これらのファイルは、`http://www.ietf.org/rfc/rfc2605.txt` および `http://www.ietf.org/rfc/rfc2788.txt` から入手できます。

Java ES MF の監視の有効化

監視に Sun Java ES Management Framework (Java ES MF) を使用する場合、Java ES MF プラグインを有効にする必要があります。

Java ES MF の管理の詳細については、『Sun Java Enterprise System 5 監視ガイド (UNIX 版)』を参照してください。

▼ Java ES MF 監視を有効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **Java ES Monitoring Framework を初期化し、登録します。**

```
$ dsccsetup mfwk-reg
```

このコマンドの場所については、35 ページの「コマンドの場所」を参照してください。

2 **Java ES 管理フレームワークプラグインを有効にします。**

```
$ dsconf enable-plugin -h host -p port "Monitoring Plugin"
Enter "cn=Directory Manager" password:
Directory Server must be restarted for changes to take effect.
```

3 **Directory Server インスタンスを再起動します。**

```
$ dsadm restart instance-path
```

4 **Java ES Management Framework** プラグインが有効にされていることを確認します。

```
$ dsconf get-plugin-prop -h host -p port -v "Monitoring Plugin"
Enter "cn=Directory Manager" password:
Reading property values of the plugin "Monitoring Plugin"...
argument          :
depends-on-named  :
depends-on-type   :  database
desc              :  Monitoring plugin
enabled           :  on
feature           :  Monitoring
init-func         :  mf_init
lib-path          :  /opt/SUNWdsee/ds6/lib/mf-plugin.so
type              :  object
vendor            :  Sun Microsystems, Inc.
version           :  6.0
```

Java ES MF 監視のトラブルシューティング

Java ES MF 監視が機能しない場合は、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の第 3 章「Installing Directory Server Enterprise Edition 6.0」に説明するとおりに、Common Agent Container が正しくインストールされていることを確認します。

まだ問題が発生する場合は、『Sun Java Enterprise System 5 監視ガイド (UNIX 版)』を参照してください。

cn=monitor を使用したサーバーの監視

サーバーの状態、レプリケーションの状態、リソース使用状況、およびそのほかの監視情報を DSCC から入手できます。

または、次のエントリに対して、検索操作を実行して、LDAP クライアントから Directory Server の現在の動作を監視できます。

- `cn=monitor`
- `cn=monitor, cn=ldbm database, cn=plugins, cn=config`
- `cn=monitor,cn=`*dbName* `,cn=ldbm database,cn=plugins,cn=config`

  dbName は、監視するサフィックスのデータベース名です。匿名でバインドされているクライアントを含め、デフォルトではすべてのユーザーが各接続に関する情報を除き `cn=monitor` エントリを読み取れることに注意してください。

  次の例は、サーバーの一般的な統計情報を表示する方法を示しています。

```
$ ldapsearch -h host -p port -D cn=admin,cn=Administrators,cn=config -w - \
 -s base -b "cn=monitor" "(objectclass=*)"
```

これらのエントリで使用可能なすべての監視属性については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Directory ServerMonitoring Attributes」を参照してください。

監視できるパラメータの多くは、Directory Server のパフォーマンスを反映するので、設定や調整によって影響を受けます。設定可能な各属性の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Man Page Reference』の属性のマニュアル ページを参照してください。

パート II

Directory Proxy Server による管理

◆ ◆ ◆ 第 16 章

Directory Proxy Server のツール

Sun Java™ System Directory Proxy Server には、Directory Proxy Server のインスタンスの登録と管理を行うためのブラウザインタフェースとコマンド行ツールがあります。このブラウザインタフェースは、Directory Service Control Center (DSCC) と呼ばれています。この章では、DSCC やコマンド行による Directory Proxy Server の管理に必要な基本タスクについて説明します。

特殊なタスクの実行に DSCC を使用するか、コマンド行を使用するかを決定するには、44 ページの「DSCC を使用する場合とコマンド行を使用する場合の判断」を参照してください。

管理フレームワークの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「Directory Server Enterprise Edition の管理モデル」を参照してください。

この章の内容は次のとおりです。

Directory Proxy Server を管理するために DSCC を使用する

この節では、Directory Proxy Server の管理目的で、DSCC にアクセスする方法について説明します。

▼ Directory Proxy Server の管理目的で、DSCC にアクセスする

1 **Directory Server の場合と同じ方法で DSCC にアクセスします。**
46 ページの「DSCC にアクセスする」を参照してください。

2 「プロキシサーバー」タブをクリックして **Directory Proxy Server** を表示し管理します。

次の図に Directory Proxy Server の最初のウィンドウを示します。

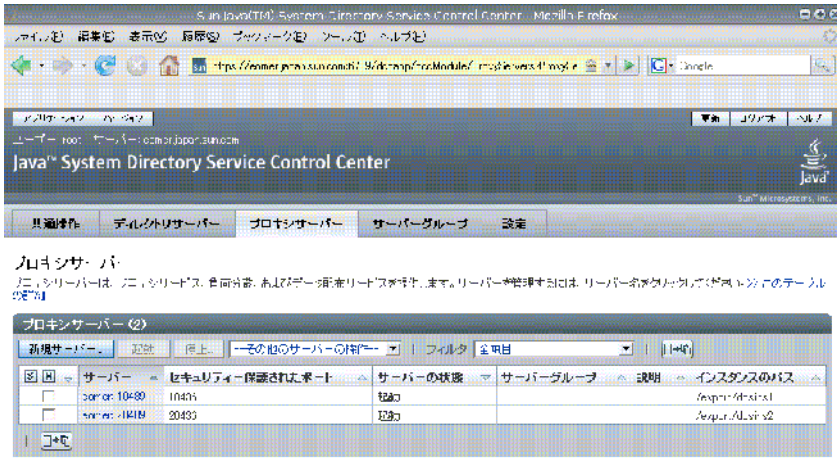


図 16–1 Directory Proxy Server の最初の DSCC ウィンドウ

3 **Directory Proxy Server** インスタンスをクリックして、その該当のサーバーを表示または管理します。

注 – DSCC の使い方の詳細は、オンラインヘルプを参照してください。

Directory Proxy Server のコマンド行ツール

Directory Proxy Server の操作に使用するコマンド行ツールは、`dpadm` および `dpconf` と呼ばれています。これらのコマンドの使用方法については、`dpadm`(1M) および `dpconf`(1M) のマニュアルページを参照してください。

この節では、`dpadm` コマンドと `dpconf` コマンドの場所について説明します。また、環境変数、これらのコマンドの比較、これらのコマンドを使用する際の参照情報の入手先についても記載します。

Directory Proxy Server コマンドの場所

Directory Proxy Server のコマンド行ツールは、デフォルトでは次の場所にあります。

install-path`/dps6/bin`

インストールパスはオペレーティングシステムによって異なります。すべてのオペレーティングシステムのインストールパスのリストは、33 ページの「デフォルトのパスとコマンドの場所」にあります。

dpconf に対する環境変数の設定

dpconf コマンドには、環境変数によってプリセットできるオプションが必要です。コマンドを使用する際にオプションが指定されていない場合や、環境変数が設定されていない場合は、デフォルト設定が使用されます。環境変数は次のオプションに対して設定できます。

-D *userDN*	ユーザーバインド DN。環境変数: LDAP_ADMIN_USER。デフォルト: cn=Proxy Manager。
-w *password-file*	ユーザーバインド DN のパスワードファイル。環境変数: LDAP_ADMIN_PWF。デフォルト: パスワードのプロンプト。
-h *host*	ホスト名または IP アドレス。環境変数: DIR_PROXY_HOST。デフォルト: localhost。
-p *LDAP-port*	LDAP ポート番号。環境変数: DIR_PROXY_HOST。デフォルト: サーバーインスタンスが root として実行中の場合は 389、サーバーインスタンスが通常のユーザーとして実行中の場合は 1389。
-e, --unsecured	dpconf がデフォルトでクリア接続を開くように指定します。環境変数: DIR_PROXY_UNSECURED。この変数が設定されていない場合、dpconf はデフォルトでセキュリティー保護された接続を開きます。

詳細は、dpconf(1M) のマニュアルページを参照してください。

dpadm と dpconf の比較

次の表に、dpadm コマンドと dpconf コマンドの比較を示します。

表 16–1　dpadm コマンドと dpconf コマンドの比較

	dpadm コマンド	dpconf コマンド
目的	Directory Proxy Server のローカルインスタンスのプロセスやファイルを管理すること	Directory Proxy Server のローカルやリモートのインスタンスを設定すること
ユーザー	オペレーティング システムのユーザー	LDAP ユーザー

表 16–1 dpadm コマンドと dpconf コマンドの比較 *(続き)*

	dpadm コマンド	dpconf コマンド
ローカルまたはリモート	コマンドはインスタンスに対してローカルでなければなりません。つまり、コマンドは、サーバーが実行中のホストで実行する必要があります。	コマンドはインスタンスに対してローカルにすることができますが、ネットワーク上のどの場所からも実行できます。
コマンドの使用例	Directory Proxy Server のインスタンスを作成します。 Directory Proxy Server のインスタンスを起動および停止します。 証明書データベースを管理します。	Directory Proxy Server のインスタンスの設定を変更します。 データビューを作成します。 データソースプールの負荷分散を設定します。
サーバーの状態	サーバーは稼動中でも停止していてもかまいません。	サーバーは動作している必要があります。
コマンドがサーバーインスタンスを識別する方法	インスタンスパスを指定することで識別します。インスタンスパスは相対でも絶対でもかまいません。	ホスト名か IP アドレスと、ポート番号を指定することで識別します。 コマンドは、LDAP ポート (-p) または LDAPS セキュアポート (-P) を使用します。コマンド行にポート番号が指定されていない場合は、PROXY_PORT 環境変数が使用されます。環境変数が設定されていない場合は、デフォルトポートが使用されます。

dpconf による複数の値を持つプロパティーの設定

Directory Proxy Server のプロパティーによっては、複数の値をとることがあります。複数の値を指定するには、次の構文を使用します。

```
$ dpconf set-container-prop -h host -p port \
 property:value [property:value]
```

たとえば、my-view という LDAP データビューに複数の書き込み可能属性を設定するには、次のコマンドを入力します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 \
 writable-attr:uid writable-attr:cn writable-attr:userPassword
```

すでに値が含まれている複数値プロパティーに値を追加するには、次のコマンドを入力します。

```
$ dpconf set-container-prop -h host -p port \
 property+:value
```

すでに値が含まれている複数値プロパティーから値を削除するには、次のコマンドを入力します。

```
$ dpconf set-container-prop -h host -p port\
 property-:value
```

たとえば、前述の例で、書き込み可能属性のリストに sn を追加するには、次のコマンドを入力します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 \
writable-attr+:sn
```

書き込み可能属性のリストから cn を削除するには、次のコマンドを入力します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 \
writable-attr-:cn
```

dpadm と dpconf に関するヘルプ情報を得るには

dpadm コマンドと dpconf コマンドの使用方法については、dpadm(1M) および dpconf(1M) のマニュアルページを参照してください。

- サブコマンドの一覧を表示するには、次の該当するコマンドを入力します。

  ```
  $ dpadm --help
  ```

  ```
  $ dpconf --help
  ```

- サブコマンドの使用方法についての説明を表示するには、次の該当するコマンドを入力します。

  ```
  $ dpadm subcommand --help
  ```

  ```
  $ dpconf subcommand --help
  ```

- dpconf コマンドで使用する設定プロパティーについての情報を得るには、次のように入力します。

  ```
  $ dpconf help-properties
  ```

- サブコマンドの設定プロパティーについての情報を得るには、次のコマンドを使用します。

```
$ dpconf help-properties subcommand-entity
```

たとえば、アクセスログプロパティーについての情報を調べるには、次のように入力します。

```
$ dpconf help-properties access-log
```

- サブコマンドで使用するプロパティーについての情報を得るには、次のコマンドを使用します。

```
$ dpconf help-properties subcommand-entity property
```

たとえば、`set-access-log-prop` サブコマンドの `log-search-filters` プロパティーについての情報を調べるには、次のように入力します。

```
$ dpconf help-properties access-log log-search-filters
```

- データビューや接続ハンドラなどのエンティティーのグループのキープロパティーを一覧表示するには、`list` サブコマンドで冗長オプション `-v` を指定してください。

  たとえば、接続ハンドラすべてのキープロパティーや相対プロパティーを表示するには、次のコマンドを使用します。

```
$ dpconf -h host -p port list-connection-handlers -v
Name                        is-enabled  priority  description
--------------------------  ----------  --------  --------------------------
anonymous                   false       99        unauthenticated connections
default connection handler  true        100       default connection handler
dscc administrators            true          1         Administrators connection
                                                      handler
```

個々のプロパティーの詳細は、該当のプロパティーのマニュアルページを参照してください。

◆ ◆ ◆ 第 17 章

Directory Proxy Server のインスタンス

この章では、Directory Proxy Server のインスタンスを管理する方法について説明します。この章の内容は次のとおりです。

Directory Proxy Server インスタンスの作成と削除

Directory Proxy Server のインスタンスを作成すると、インスタンスに必要なファイルとディレクトリが指定するパス内に作成されます。

▼ Directory Proxy Server インスタンスを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

DSCC を使用して新しいサーバーインスタンスを作成する場合は、既存のサーバーからサーバー設定の一部またはすべてをコピーするよう選択できます。

1 **Directory Proxy Server のインスタンスを作成します。**

```
$ dpadm create -p port instance-path
```

たとえば、ディレクトリ `/local/dps` 内に新しいインスタンスを作成するには、次のコマンドを使用します。

```
$ dpadm create -p 2389 /local/dps
```

インスタンスのほかのパラメータの指定については、dpadm(1M) のマニュアルページを参照してください。

2 必要に応じてパスワードを入力します。

3 インスタンスの状況を確認して、インスタンスが作成されていることを確認します。

```
$ dpadm info instance-path
```

4 **(省略可能) Directory Proxy Server が Sun Java™ Enterprise System インストーラまたはネイティブパッケージインストールを使用してインストールされていて、OS がサービス管理ソリューションを提供する場合は、次の表に示すサービスとしてサーバーを管理できます。**

オペレーティングシステム	コマンド
Solaris 10	`dpadm enable-service --type SMF` *instance-path*
Solaris 9	`dpadm autostart` *instance-path*
Linux、HP-UX	`dpadm autostart` *instance-path*
Windows	`dpadm enable-service --type WIN_SERVICE` *instance-path*

5 **(省略可能) 次のいずれかの方法で、サーバーインスタンスを登録します。**

- `https://localhost:6789` という URL で DSCC にアクセスし、ブラウザインタフェースにログインします。
- `dsccreg add-server` コマンドを使用します。

  詳細は、dsccreg(1M) のマニュアルページを参照してください。

▼ Directory Proxy Server インスタンスを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **(省略可能) Directory Proxy Server インスタンスを停止します。**

```
$ dpadm stop instance-path
```

インスタンスを停止していない場合は、削除コマンドによって自動的に停止します。ただし、サービス管理ソリューションでインスタンスが有効になっている場合は、手動で停止する必要があります。

2 (省略可能) 以前に **DSCC** を使用してサーバーを管理していた場合は、コマンド行を使用してサーバーを登録解除します。

```
$ dsccreg remove-server /local/dps
Enter DSCC administrator's password:
/local/dps is an instance of DPS
Enter password of "cn=Proxy Manager" for /local/dps:
Unregistering /local/dps from DSCC on localhost.
Connecting to /local/dps
Disabling DSCC access to /local/dps
```

詳細は、dsccreg(1M) のマニュアルページを参照してください。

3 (省略可能) 以前にサーバー管理ソリューションでサーバーインスタンスを有効にした場合は、サービスとしてのサーバーの管理を無効にします。

オペレーティングシステム	コマンド
Solaris 10	`dpadm disable-service --type SMF` *instance-path*
Solaris 9	`dpadm autostart --off` *instance-path*
Linux、HP-UX	`dpadm autostart --off` *instance-path*
Windows	`dpadm disable-service --type WIN_SERVICE` *instance-path*

4 インスタンスを削除します。

```
$ dpadm delete instance-path
```

Directory Proxy Server インスタンスの状況の確認

この手順では、Directory Proxy Server のインスタンスの状況を確認する方法について説明します。

▼ Directory Proxy Server インスタンスの状況を確認する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** のインスタンスの状況を確認します。

```
$ dpadm info instance-path
```

Directory Proxy Server インスタンスの起動、停止、再起動

この節では、コマンド行からの Directory Proxy Server の起動、停止、再起動について説明します。

▼ Directory Proxy Server を起動および停止する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** を起動または停止するには、次のいずれかを実行します。
  - Directory Proxy Server を起動するには、次のように入力します。

    ```
    $ dpadm start instance-path
    ```

    たとえば、インスタンスを `/local/dps` で起動するには、次のコマンドを入力します。

    ```
    $ dpadm start /local/dps
    ```
  - Directory Proxy Server を停止するには、次のように入力します。

    ```
    $ dpadm stop instance-path
    ```

    次に例を示します。

    ```
    $ dpadm stop /local/dps
    ```

▼ Directory Proxy Server インスタンスを再起動する必要があるかどうかを確認する

設定変更は、変更を有効にする前にサーバーの再起動が必要になる場合があります。設定変更後に Directory Proxy Server インスタンスを再起動する必要があるかどうかを確認するには、この手順に従います。

- サーバーを再起動する必要があるかどうかを確認します。

  ```
  $ dpconf get-server-prop -h host -p port is-restart-required
  ```

  - このコマンドが `true` を返す場合、Directory Proxy Server のインスタンスを再起動する必要があります。

- このコマンドが `false` を返す場合、Directory Proxy Server のインスタンスを再起動する必要はありません。

▼ Directory Proxy Server を再起動する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server を再起動します。**

  ```
  $ dpadm restart instance-path
  ```

  たとえば、インスタンスを `/local/dps` で再起動するには、次のコマンドを入力します。

  ```
  $ dpadm restart /local/dps
  ```

◆ ◆ ◆ 第 18 章

18 Directory Proxy Server の設定

この章では、Directory Proxy Server のインスタンスを設定する方法について説明します。この章で示す手順では、`dpadm` コマンドと `dpconf` コマンドを使用します。これらのコマンドについては、`dpadm`(1M) および `dpconf`(1M) のマニュアルページを参照してください。

この章の内容は次のとおりです。

設定例

この節では、Directory Proxy Server の 2 つの設定例を示します。1 つは負荷分散、もう 1 つはデータ配布のための設定例です。仮想ディレクトリの詳細は、438 ページの「仮想設定の例」を参照してください。

Directory Proxy Server で負荷分散を実行するための設定

単純な負荷分散の例として、検索操作と比較操作を 1 つのディレクトリセットに送信し、その他の操作をもう 1 つのディレクトリセットに送信します。Directory Proxy Server は、すべてのクライアント操作を受け取ります。Directory Proxy Server は、読み取りを取得するディレクトリセットとその他の操作を取得するディレクトリセットを判別する必要があります。

Directory Proxy Server でこの負荷分散を処理するための設定の主要な手順は次のとおりです。

1. Directory Proxy Server 用のデータソースとしてディレクトリを追加します。
2. それらのデータソースをデータソースプールに追加します。
3. それらのデータソースのいくつかで検索と比較の操作を受け入れるように設定し、残りのデータソースで追加、バインド、削除、変更、および DN 変更の操作を受け入れるように設定します。
4. データソースプールをデータビューに追加します。

次の例では、Directory Proxy Server はポート 9389 で待機しています。この例では、記述されているように、検索操作と比較操作を処理する Directory Server インスタンス (`ds1:1389`) と、その他の操作を処理する Directory Server インスタンス (`ds2:2389`) に負荷を分散するようにプロキシを設定します。

最初の手順では、データソースを作成し、そのデータソースを有効にします。この手順では、プロキシサーバーを再起動する必要があります。

```
$ dpconf create-ldap-data-source -p 9389 ds1 localhost:1389
$ dpconf create-ldap-data-source -p 9389 ds2 localhost:2389
$ dpconf set-ldap-data-source-prop -p 9389 ds1 is-enabled:true
$ dpconf set-ldap-data-source-prop -p 9389 ds2 is-enabled:true
$ dpadm restart /local/dps
```

2 番目の手順では、データソースをデータソースプールに追加します。

```
$ dpconf create-ldap-data-source-pool -p 9389 "Directory Pool"
$ dpconf attach-ldap-data-source -p 9389 "Directory Pool" ds1 ds2
```

3 番目の手順では、`ds1` が検索操作と比較操作を受け入れ、`ds2` がその他の操作を受け入れるように設定します。

```
$ dpconf set-attached-ldap-data-source-prop -p 9389 "Directory Pool" ds1 \
add-weight:disabled bind-weight:disabled compare-weight:1 delete-weight:disabled \
modify-dn-weight:disabled modify-weight:disabled search-weight:1
$ dpconf set-attached-ldap-data-source-prop -p 9389 "Directory Pool" ds2 \
add-weight:1 bind-weight:1 compare-weight:disabled delete-weight:1 \
modify-dn-weight:1 modify-weight:1 search-weight:disabled
```

4 番目の手順では、クライアントアプリケーション要求がデータソースプールに経路指定されるように、そのプールをデータビューに追加します。

```
$ dpconf create-ldap-data-view -p 9389 "Balanced View" "Directory Pool" \
dc=example,dc=com
```

サフィックスデータを配布するための Directory Proxy Server の設定

単純なデータ配布の例として、A から M までの文字で始まる UID を持つエントリを 1 つのディレクトリセットに格納し、N から Z までの文字で始まる UID を持つエントリをもう 1 つのディレクトリセットに格納します。Directory Proxy Server は、すべてのクライアント操作を受け取ります。Directory Proxy Server は、A から M を処理するディレクトリセットと N から Z を処理するディレクトリセットを判別する必要があります。

Directory Proxy Server でこのデータ配布を処理するための設定の主要な手順は次のとおりです。

1. Directory Proxy Server 用のデータソースとしてディレクトリを追加します。
2. それらのデータソースを、各データ配布を処理するデータソースプールに追加します。
3. クライアント要求を適切なデータプールに配布するためのデータビューを作成します。
4. 適切なデータソースに読み込まれるように LDIF を分割します。
5. 分割した LDIF を適切なデータソースにインポートします。
6. 適切なデータプールに接続されたデータソースごとに、操作ベースのウェイトを調整します。

次の例では、Directory Proxy Server はポート 9389 で待機しています。単純にするため、この例では、記述されているように 3 つの Directory Server インスタンスのみに配布するようにプロキシが設定されています。可用性と読み取りのスケーラビリティーを確保するため、レプリケートされたディレクトリトポロジを使用して LDAP データを格納します。一方の Directory Server インスタンス (`dsA-M:1389`) は、A から M までの文字で始まる UID を持つユーザーエントリを処理します。もう一方の Directory Server インスタンス (`dsN-Z:2389`) は、N から Z までの文字で始まる UID を持つユーザーエントリを処理します。最後のディレクトリインスタンス (`dsBase:3389`) は、サフィックスのベースエントリを処理します。

最初の手順では、データソースを作成して有効にします。ベースデータソースは、UID を持たない、サフィックスのルートに近いエントリを保持します。一般的な配備では、これらのエントリは、配布されるエントリよりもずっと少数です。

```
$ dpconf create-ldap-data-source -p 9389 dsA-M localhost:1389
$ dpconf set-ldap-data-source-prop -p 9389 dsA-M is-enabled:true

$ dpconf create-ldap-data-source -p 9389 dsN-Z localhost:2389
$ dpconf set-ldap-data-source-prop -p 9389 dsN-Z is-enabled:true

$ dpconf create-ldap-data-source -p 9389 dsBase localhost:3389
```

```
$ dpconf set-ldap-data-source-prop -p 9389 dsBase is-enabled:true
```

2 番目の手順では、データソースをデータソースプールに追加します。

```
$ dpconf create-ldap-data-source-pool -p 9389 "Base Pool"
$ dpconf attach-ldap-data-source -p 9389 "Base Pool" dsBase

$ dpconf create-ldap-data-source-pool -p 9389 "A-M Pool"
$ dpconf attach-ldap-data-source -p 9389 "A-M Pool" dsA-M

$ dpconf create-ldap-data-source-pool -p 9389 "N-Z Pool"
$ dpconf attach-ldap-data-source -p 9389 "N-Z Pool" dsN-Z
```

3 番目の手順では、クライアント要求を適切なデータプールに配布するためのデータビューを作成します。ベースプールでは `dc=example,dc=com` を処理するのに対して、UID 値に従って配布されたデータを保持するプールでは `ou=people,dc=example,dc=com` を処理します。この手順では、サーバーを再起動する必要があります。

```
$ dpconf create-ldap-data-view -p 9389 "Base View" "Base Pool" \
dc=example,dc=com

$ dpconf create-ldap-data-view -p 9389 "A-M View" "A-M Pool" \
ou=people,dc=example,dc=com
$ dpconf set-ldap-data-view-prop -p 9389 "A-M View" \
distribution-algorithm:lexicographic lexicographic-attrs:uid \
lexicographic-lower-bound:a lexicographic-upper-bound:m
The proxy server will need to be restarted in order for the changes to take effect

$ dpconf create-ldap-data-view -p 9389 "N-Z View" "N-Z Pool" \
ou=people,dc=example,dc=com
$ dpconf set-ldap-data-view-prop -p 9389 "N-Z View" \
distribution-algorithm:lexicographic lexicographic-attrs:uid \
lexicographic-lower-bound:n lexicographic-upper-bound:z
The proxy server will need to be restarted in order for the changes to take effect
$ dpadm restart /local/dps
```

4 番目の手順では、適切なデータソースに読み込まれるように LDIF を分割します。この例では、`dsadm split-ldif` コマンドを使用して最初の分割を実行し、さらに、ファイル編集を使用して、すべてのデータソースで最上位エントリを保持します。これにより、アクセス制御命令を指定する最上位エントリの保持と、各データソースに対する 1 つのインポートコマンドの使用が可能になります。

```
$ dpadm split-ldif /local/dps /local/ds6/ldif/Example.ldif /tmp/
[14/May/2007:21:14:13 +0200] - STARTUP    - INFO  - Java Version: 1.5.0_09
 (Java Home: /local/jre)
[14/May/2007:21:14:13 +0200] - STARTUP    - INFO  - Java Heap Space: Total Memory
 (-Xms) = 3MB,
```

```
 Max Memory (-Xmx) = 63MB
[14/May/2007:21:14:13 +0200] - STARTUP    - INFO  - Operating System: SunOS/sparc 5.10
[14/May/2007:21:14:15 +0200] - INTERNAL   - ERROR - Entry starting at line 0 does not
 start with a DN
[14/May/2007:21:14:15 +0200] - INTERNAL   - ERROR - Unable to parse line "# Kirsten is
 a Directory Administrator and therefore should not" of entry "uid=kvaughan, ou=People,
 dc=example,dc=com" starting at line 112 as an attribute/value pair -- no colon found.
[14/May/2007:21:14:15 +0200] - INTERNAL   - ERROR - Unable to parse line "# Robert is
 a Directory Administrator and therefore should not" of entry "uid=rdaugherty,
 ou=People, dc=example,dc=com" starting at line 298 as an attribute/value pair --
 no colon found.
[14/May/2007:21:14:16 +0200] - INTERNAL   - ERROR - Unable to parse line "# Harry is
 a Directory Administrator and therefore should not" of entry "uid=hmiller, ou=People,
 dc=example,dc=com" starting at line 556 as an attribute/value pair -- no colon found.
[14/May/2007:21:14:16 +0200] - INTERNAL   - INFO  - SplitLDIF processing complete.
  Processed 156 entries.
$ ls /tmp/*ldif
/tmp/a-m view.ldif    /tmp/base view.ldif   /tmp/n-z view.ldif
```

この手順では、インポートの前に LDIF に追加する最上位エントリも必要です。

```
$ cp /local/ds6/ldif/Example.ldif /tmp/top.ldif
$ vi /tmp/top.ldif
$ cat /tmp/top.ldif
dn: dc=example,dc=com
objectclass: top
objectclass: domain
dc: example
aci: (target ="ldap:///dc=example,dc=com")(targetattr !=
 "userPassword")(version 3.0;acl "Anonymous read-search access";
 allow (read, search, compare)(userdn = "ldap:///anyone");)
aci: (target="ldap:///dc=example,dc=com") (targetattr =
  "*")(version 3.0; acl "allow all Admin group"; allow(all) groupdn =
  "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)

$ cat /tmp/top.ldif /tmp/base\ view.ldif > /tmp/top\ and\ base\ view.ldif
$ cat /tmp/top.ldif /tmp/a-m\ view.ldif > /tmp/top\ and\ a-m\ view.ldif
$ cat /tmp/top.ldif /tmp/n-z\ view.ldif > /tmp/top\ and\ n-z\ view.ldif
```

5 番目の手順では、分割した LDIF を適切なデータソースにインポートします。ここでは、ベースエントリを処理するディレクトリはポート 3389 上にあります。A ～ M を処理するディレクトリはポート 1389 で待機します。N ～ Z を処理するディレクトリはポート 2389 で待機します。

```
$ dsconf import -p 1389 /tmp/top\ and\ a-m\ view.ldif dc=example,dc=com
...
Task completed (slapd exit code: 0).
```

```
$ dsconf import -p 2389 /tmp/top\ and\ n-z\ view.ldif dc=example,dc=com
...
Task completed (slapd exit code: 0).
$ dsconf import -p 3389 /tmp/top\ and\ base\ view.ldif dc=example,dc=com
...
Task completed (slapd exit code: 0).
```

6 番目の手順では、適切なデータプールに接続されたデータソースの操作ベースのウェイトを調整します。クライアントアプリケーションが検索以外の操作を実行する場合は、それらの操作のウェイトも設定してください。

```
$ dpconf set-attached-ldap-data-source-prop -p 9389 "Base Pool" dsBase search-weight:1
$ dpconf set-attached-ldap-data-source-prop -p 9389 "A-M Pool" dsA-M search-weight:1
$ dpconf set-attached-ldap-data-source-prop -p 9389 "N-Z Pool" dsN-Z search-weight:1
```

操作ベースのウェイトを設定すると、クライアントアプリケーションでは、データが物理的に配布されない場合と同じように Directory Proxy Server を検索できます。

次の検索では、UID が R で始まるユーザーを探します。

```
$ ldapsearch -p 9389 -b dc=example,dc=com uid=rfisher
version: 1
dn: uid=rfisher, ou=People, dc=example,dc=com
cn: Randy Fisher
sn: Fisher
givenName: Randy
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
ou: Human Resources
ou: People
l: Cupertino
uid: rfisher
mail: rfisher@example.com
telephoneNumber: +1 408 555 1506
facsimileTelephoneNumber: +1 408 555 1992
roomNumber: 1579
```

次の検索では、ベースエントリの 1 つを探します。

```
$ ldapsearch -p 9389 -b ou=groups,dc=example,dc=com cn=hr\ managers
version: 1
dn: cn=HR Managers,ou=groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
cn: HR Managers
ou: groups
```

```
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=cschmith, ou=People, dc=example,dc=com
description: People who can manage HR entries
```

Directory Proxy Server の設定の変更

この節では、Directory Proxy Server の設定を変更する方法について説明します。

▼ Directory Proxy Server の設定を変更する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 Directory Proxy Server の現在の設定を調べます。

```
$ dpconf get-server-prop -h host -p port
```

あるいは、1 つまたは複数のプロパティーの現在の設定を確認します。

```
$ dpconf get-server-prop -h host -p port property-name ...
```

たとえば、このコマンドを実行することで、未認証の操作が許可されているかどうかを調べます。

```
$ dpconf get-server-prop -h host -p port allow-unauthenticated-operations
allow-unauthenticated-operations  :  true
```

2 1 つまたは複数の設定パラメータを変更します。

```
$ dpconf set-server-prop -h host -p port property:value ...
```

たとえば、このコマンドを実行することで、未認証の操作を許可しないようにします。

```
$ dpconf set-server-prop -h host -p port allow-unauthenticated-operations:false
```

不正な変更を試みても、変更は行われません。たとえば、`allow-unauthenticated-operations` パラメータを `false` ではなく `f` に設定すると、次のようなエラーが発生します。

```
$ dpconf set-server-prop -h host -p port allow-unauthenticated-operations:f
The value "f" is not a valid value for the property "allow-unauthenticated-operations".
Allowed property values: BOOLEAN
The "set-server-prop" operation failed.
```

3 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

Directory Proxy Server インスタンスの設定情報の表示

Directory Proxy Server インスタンスの設定を表示するには、「`dpconf info`」と入力します。

```
$ dpconf info
インスタンスのパス              :  instance path
ホスト名                  :  host
セキュリティー保護された待機アドレス    :  IP address
ポート                    :  port
セキュリティー保護されたポート          :  secure port
SSL サーバー証明書  :  defaultServerCert

Directory Proxy Server を再起動する必要があります。
```

`dpconf info` では、「セキュリティー保護された待機アドレス」と「待機アドレス」は、これらのプロパティーがデフォルト以外の値に設定されている場合にのみ表示されます。この出力例では、「待機アドレス」のプロパティーがデフォルト値に設定されているため、この項目は表示されていません。

また、`dpconf info` では、必要な場合はインスタンスを再起動するようにユーザーに促します。

`dpadm info` でも Directory Proxy Server インスタンスの設定情報を表示できます。このコマンドは、停止されているインスタンスに対しても使用できます。

Directory Proxy Server インスタンスのバックアップと復元

`dpadm` を使って Directory Proxy Server をバックアップすると、設定ファイルとサーバー証明書がバックアップされます。Directory Proxy Server の仮想 ACI が実装されている場合は、ACI もバックアップされます。

Directory Proxy Server では、サーバーが正常に起動した場合は常に、`conf.ldif` ファイルが自動的にバックアップされます。

▼ Directory Proxy Server インスタンスをバックアップする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **Directory Proxy Server** のインスタンスを停止します。

```
$ dpadm stop instance-path
```

2 **Directory Proxy Server** のインスタンスをバックアップします。

```
$ dpadm backup instance-path archive-dir
```

archive-dir ディレクトリは `backup` コマンドによって作成され、このコマンドを実行する前から存在してはいけません。このディレクトリには、設定ファイルと証明書のそれぞれのバックアップが含まれます。

▼ Directory Proxy Server インスタンスを復元する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

復元操作を開始する前に、Directory Proxy Server インスタンスを作成してください。

1 **Directory Proxy Server** のインスタンスを停止します。

```
$ dpadm stop instance-path
```

2 **Directory Proxy Server** のインスタンスを復元します。

```
$ dpadm restore instance-path archive-dir
```

- インスタンスパスが存在する場合、復元操作はメッセージを表示せずに実行されます。*instance-path* ディレクトリ内の設定ファイルと証明書は、*archive-dir* ディレクトリ内のもので置き換えられます。
- インスタンスパスが存在しない場合、復元操作は失敗します。

Proxy Manager の設定

Proxy Manager とは、特権を持つ管理者のことで、UNIX® システムの root ユーザーにあたります。Proxy Manager のエントリは、Directory Proxy Server のインスタンスの作成時に定義されます。Proxy Manager のデフォルト DN は `cn=Proxy Manager` です。

Proxy Manager DN およびパスワードは、次の手順で示すように表示および変更できます。

▼ Proxy Manager を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **Proxy Manager の設定を調べます。**

```
$ dpconf get-server-prop -h host -p port configuration-manager-bind-dn configuration-manager-bind-pwd
configuration-manager-bind-dn   :  cn=proxy manager
configuration-manager-bind-pwd  :  {3DES}U77v39WX8MDpcWVrueetB0lfJlBc6/5n
```

Proxy Manager のデフォルト値は `cn=proxy manager` です。設定マネージャーのパスワードに対するハッシュ値が返されます。

2 **Proxy Manager の DN を変更します。**

```
$ dpconf set-server-prop -h host -p port configuration-manager-bind-dn:bindDN
```

3 **Proxy Manager に対するパスワードを含むファイルを作成し、そのファイルを指すプロパティーを設定します。**

```
$ dpconf set-server-prop -h host -p port configuration-manager-bind-pwd-file:filename
```

サーバーの再起動を必要とする設定変更

Directory Proxy Server とそのエントリに対するほとんどの設定変更は、オンラインで行うことができます。一部の変更は、変更を有効にするためにサーバーを再起動する必要があります。次のリストのプロパティーに対する設定変更を行う場合は、サーバーを再起動する必要があります。

```
aci-data-view
bind-dn
client-cred-mode
custom-distribution-algorithm
db-name
db-pwd
```

```
db-url
db-user
distribution-algorithm
ldap-address
ldap-port
ldaps-port
listen-address
listen-port
load-balancing-algorithm
num-bind-init
num-read-init
num-write-init
number-of-search-threads
number-of-threads
number-of-worker-threads
ssl-policy
use-external-schema
```

プロパティーの rws キーワードと rwd キーワードは、プロパティーを変更した場合にサーバーを再起動する必要があるかどうかを示します。

- プロパティーに rws (読み取り、書き込み、静的) キーワードが含まれている場合は、プロパティーを変更したときにサーバーを再起動する必要があります。
- プロパティーに rwd (読み取り、書き込み、動的) キーワードが含まれている場合、プロパティーに対する変更は、サーバーを再起動しなくても、動的に実装されます。

プロパティーに対する変更でサーバーを再起動する必要があるかどうかを確認するには、次のコマンドを実行します。

```
$ dpconf help-properties | grep property-name
```

たとえば、LDAP データのバインド DN の変更でサーバーを再起動する必要があるかどうかを確認するには、次のコマンドを実行します。

```
$ dpconf help-properties | grep bind-dn
connection-handler       bind-dn-filters        rwd  STRING | any
This property specifies a set of regular expressions. The bind DN
of a client must match at least one regular expression in order for
the connection to be accepted by the connection handler. (Default: any)
ldap-data-source      bind-dn               rws  DN | ""
This property specifies the DN to use when binding to the LDAP data
source. (Default: undefined)
```

設定変更のあとでサーバーを再起動する必要があるかどうかを確認するには、次のコマンドを実行します。

```
$ dpconf get-server-prop -h host -p port is-restart-required
```

Directory Proxy Server による Directory Server の設定エントリへのアクセス

Directory Proxy Server の設定エントリは cn=config 内にあります。Directory Proxy Server を使用して設定エントリにアクセスすると、デフォルトでは、Directory Proxy Server の設定エントリにアクセスします。

ディレクトリサーバーの設定エントリにアクセスするには、Directory Proxy Server ではなく Directory Server に直接接続することをお勧めします。Directory Server の設定方法の詳細は、第 3 章を参照してください。

注意 – ディレクトリサーバーの設定エントリにアクセスするよう Directory Proxy Server を設定し直すと、たいていは Directory Proxy Server の管理フレームワークが壊れます。

どうしても Directory Proxy Server からディレクトリサーバーの設定エントリにアクセスする必要がある場合は、Directory Proxy Server の管理フレームワークが壊れないよう、特別の手順をとります。この節では、Directory Proxy Server を使用してディレクトリサーバーの設定エントリにアクセスする方法について説明します。

▼ Directory Proxy Server を使用して Directory Server の設定エントリにアクセスする

1 1 つまたは複数のデータソースを作成します。これについては、381 ページの「LDAP データソースの作成と設定」を参照してください。

2 LDAP データソースプールを作成します。これについては、384 ページの「LDAP データソースプールの作成と設定」を参照してください。

3 1 つまたは複数のデータソースを、データソースプールに接続します。これについては、385 ページの「LDAP データソースのデータソースプールへの接続」を参照してください。

- 1 つの特定のデータソースの設定エントリを公開するには、1 つの LDAP データソースだけを LDAP データソースプールに接続します。

  ```
  $ dpconf attach-ldap-data-source -h host -p port pool-name data-source-name
  ```

  この手順を実行したあと、クライアントは、Directory Proxy Server に接続されているデータソースの設定エントリにアクセスできるようになります。

- 任意の特定のデータソースの設定エントリを公開するには、複数の **LDAP** データソースを **LDAP** データソースプールに接続します。

  ```
  $ dpconf attach-ldap-data-source -h host -p port pool-name data-source-name \
   data-source-name ...
  ```

  この手順を実行したあと、クライアントは、Directory Proxy Server に接続されているデータソースの 1 つの設定エントリにアクセスできるようになります。ただし、クライアントには、設定エントリがどのデータソースに属するかはわかりません。

4 **LDAP** データビューを作成して、`cn=config` を公開します。

```
$ dpconf create-ldap-data-view -h host -p port view-name pool-name cn=config
```

◆ ◆ ◆ 第 19 章

Directory Proxy Server の証明書

この章では、Directory Proxy Server で証明書を設定する方法について説明します。*Directory Server* での証明書の設定については、113 ページの「証明書を管理する」を参照してください。

この章で説明する手順では、`dpadm` コマンドと `dpconf` コマンドを使用します。これらのコマンドについては、`dpadm`(1M) および `dpconf`(1M) のマニュアルページを参照してください。

この章の内容は次のとおりです。

デフォルトの自己署名付き証明書

Directory Proxy Server インスタンスを作成すると、デフォルトの自己署名付き証明書が組み込まれます。自己署名付き証明書は公開鍵と非公開鍵のペアであり、公開鍵は Directory Proxy Server で自己署名が付けられます。

▼ デフォルトの自己署名付き証明書の表示

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- デフォルトの自己署名付き証明書を表示します。

```
$ dpadm show-cert instance-path defaultservercert
```

Directory Proxy Server 用の証明書の作成、要求、インストール

Directory Proxy Server で Secure Sockets Layer (SSL) を実行するには、自己署名付き証明書または公開鍵インフラストラクチャー (PKI) ソリューションのいずれかを使用する必要があります。

PKI ソリューションには外部認証局 (CA) が関与します。PKI ソリューションでは CA 署名付きサーバー証明書が必要であり、これには公開鍵と非公開鍵の両方が含まれます。この証明書は、1 つの Directory Proxy Server インスタンスに固有です。また、公開鍵が含まれる「信頼できる CA 証明書」も必要です。信頼できる CA 証明書は、CA からのサーバー証明書はすべて信頼できることを保証します。この証明書は、CA ルート鍵またはルート証明書と呼ばれることもあります。

デフォルト以外の自己署名付き証明書を作成する方法と、CA 署名付き証明書を要求しインストールする方法については、次の手順を参照してください。

▼ Directory Proxy Server 用のデフォルト以外の自己署名付き証明書を作成する

Directory Proxy Server インスタンスを作成すると、デフォルトの自己署名付き証明書が自動的に用意されます。デフォルト以外の設定で自己署名付き証明書を作成する場合は、次の手順を使用します。

この手順では、サーバー証明書用の公開鍵と非公開鍵のペアを作成し、公開鍵が Directory Proxy Server によって署名されます。自己署名付き証明書は、3 か月間有効です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** 用のデフォルト以外の自己署名付き証明書を作成するには、次のように入力します。

  ```
  $ dpadm add-selfsign-cert instance-path cert-alias
  ```

  ここで、*cert-alias* は自己署名付き証明書の名前です。

  たとえば、次のように入力して、 `my-self-signed-cert` という証明書を作成することもできます。

  ```
  $ dpadm add-selfsign-cert /local/dps my-self-signed-cert
  ```

  すべてのコマンドオプションの説明については、`dpadm`(1M) のマニュアルページを参照するか、コマンド行で `dpadm add-selfsign-cert --help` と入力してください。

▼ Directory Proxy Server 用の CA 署名付き証明書を要求する

自己署名付き証明書はテスト目的では便利です。ただし、稼働環境では、信頼できる認証局 (CA) 証明書を使用するほうがより安全です。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **CA 署名付きサーバー証明書を要求します。**

   ```
   $ dpadm request-cert instance-path cert-alias
   ```

   ここで、*cert-alias* は、要求する証明書の名前です。認証局は、サーバーを識別するためにコマンドのすべてのオプションを必要とすることがあります。すべてのコマンドオプションの説明については、`dpadm`(1M) マニュアルページを参照してください。

   CA 証明書を入手するプロセスは、使用する CA によって異なります。商用 CA のなかには、証明書をダウンロードできる Web サイトを備えているものもあります。また、証明書を電子メールで送信する CA もあります。

   たとえば、次のように入力して、 `my-CA-signed-cert` という証明書を要求することもできます。

   ```
   $ dpadm request-cert -S cn=my-request,o=test /local/dps my-CA-signed-cert
   -----BEGIN NEW CERTIFICATE REQUEST-----
   MIIBYDCBygIBADAhMQ0wCwYDVQQDEwRnZXJpMRAwDgYDVQQDEwdteWNlcnQ0MIGfMA0GCSqGSIb3
   DQEBAQUAA4GNADCBiQKBgQC3v9ubG468wnjBDAMbRrEkmFDTQzT+LO30D/ALLXOiElVsHrtRyWhJ
   PG9cURI9uwqs15crxCpJvho1kt3SB9+yMB8Ql+CKnCQDHlNAfnn30MjFHShv/sAuEygFsN+Ekci5
   W1jySYE2rzE0qKVxWLSILFo1UFRVRsUnORTX/Nas7QIDAQABoAAwDQYJKoZIhvcNAQEEBQADgYEA
   fcQMnZNLpPobiX1xy1ROefPOhksVz8didY8Q2fjjaHG5lajMsqOROzubsuQ9Xh4ohT8kIA6xcBNZ
   ```

```
g8FRNIRAHCtDXKOdOm3CpJ8da+YGI/ttSawIeNAKU1DApF9zMb7c2lS4yEfWmreoQdXIC9YeKtF6
zwbn2EmIpjHzETtS5Nk=
-----END NEW CERTIFICATE REQUEST-----
```

`dpadm request-cert` コマンドを使用して 証明書を要求するとき、証明書要求は PEM (Privacy Enhanced Mail) 形式の PKCS #10 証明書要求です。PEM は、RFC 1421 ～ 1424 で指定されている形式です。詳細は、`http://www.ietf.org/rfc/rfc1421.txt` を参照してください。PEM 形式は、base64 形式で符号化された ASCII 形式の証明書要求を表します。

CA 署名付き証明書を要求すると、一時的な自己署名付き証明書が作成されます。CA 署名付き証明書を CA から受信しインストールすると、新しい証明書が一時的な自己署名付き証明書に取って代わります。

2 その手順に従って、証明書要求を **CA** に送信します。

証明書要求を送信したら、証明書に関する CA からの回答を待つ必要があります。要求に対する回答が届くまでの時間は、状況によって異なります。たとえば、CA が社内にある場合は、短い時間で回答が届くこともあります。ただし、CA が社外にある場合は、数週間かかることもあります。

3 **CA** から受け取った証明書を保存します。

証明書をテキストファイルで保存し、安全な場所に証明書をバックアップします。

▼ Directory Proxy Server 用の CA 署名付きサーバー証明書をインストールする

CA 署名付きサーバー証明書を信頼するには、証明書を Directory Proxy Server インスタンスにインストールする必要があります。ここで示す手順では、CA 証明書の公開鍵を Directory Proxy Server 上の証明書データベースにインストールします。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 この **CA** に対する信頼できる **CA** 証明書がすでにインストール済みかどうかを確認します。

このためには、376 ページの「CA 証明書を一覧表示する」で示すように、インストールされているすべての CA 証明書をリストします。

2 信頼できる **CA** 証明書がインストールされていない場合は、それを**Directory Proxy Server** インスタンス上の証明書データベースに追加します。

```
$ dpadm add-cert instance-path cert-alias cert-file
```

ここで、*cert-alias* は信頼できる CA 証明書の名前で、*cert-file* は信頼できる CA 証明書が含まれるファイルの名前です。

3 **CA 署名付きサーバー証明書を証明書データベースにインストールします。**

```
$ dpadm add-cert instance-path cert-alias cert-file
```

ここで、*cert-alias* は CA 署名付きサーバー証明書の名前で、*cert-file* は CA 署名付きサーバー証明書が含まれるファイルの名前です。この *cert-alias* は、証明書要求で使用した *cert-alias* と同じでなければなりません。

たとえば、次のように入力して、`CA-cert` という CA 署名付きサーバー証明書を、`/local/dps` 上の証明書データベースに追加できます。

```
$ dpadm add-cert /local/dps CA-cert /local/safeplace/ca-cert-file.ascii
```

Directory Proxy Server 用の期限切れ CA 署名付き証明書の更新

この節では、期限切れ CA 署名付きサーバー証明書を更新する方法について説明します。

▼ Directory Proxy Server 用の期限切れ CA 署名付きサーバー証明書を更新する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **CA から更新された証明書を入手します。**

2 **証明書を Directory Proxy Server のインスタンスにインストールします。**

```
$ dpadm renew-cert instance-path cert-alias cert-file
```

ここで、*cert-alias* は新しい証明書の名前で、*cert-file* は証明書が含まれるファイルの名前です。すべてのコマンドオプションについては、`dpadm`(1M) のマニュアルページを参照してください。

証明書のリスト

サーバーと CA 証明書を一覧表示する方法については、次の手順を参照してください。

▼ サーバー証明書を一覧表示する

ここで示す手順では、Directory Proxy Server の 1 つのインスタンスにインストールされているすべての証明書を一覧表示します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** インスタンスにある証明書データベース内のサーバー証明書を一覧表示します。

  ```
  $ dpadm list-certs instance-path
  ```

  デフォルトでは、Directory Proxy Server のインスタンスには、`defaultservercert` というサーバー証明書が含まれます。`Same as issuer` は、デフォルト証明書が自己署名付きサーバー証明書であることを示します。

  次に例を示します。

```
$ dpadm list-certs /local/dps
Alias             Valid from       Expires on       Self-signed? Issued by          Issued to
----------------- ---------------- ---------------- ------------ ------------------ --------------
defaultservercert 2006/06/01 04:15 2008/05/31 04:15 y            CN=myserver:myport Same as issuer
1 certificate found.
```

▼ CA 証明書を一覧表示する

ここで示す手順では、Directory Proxy Server の 1 つのインスタンスにインストールされている CA 証明書を一覧表示します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** インスタンスにある証明書データベース内の **CA** 証明書を一覧表示します。

  ```
  $ dpadm list-certs -C instance-path
  ```

次に例を示します。

```
$ dpadm list-certs -C /local/dps
Alias   Valid from       Expires on        Built-in  Issued by    Issued to
------  ----------       ----------------  --------- ---------    ---------
CAcert1 1999/06/21 06:00 2020/06/21 06:00  y         CN=company1, O=company2
...
```

バックエンド LDAP サーバーから Directory Proxy Server 上の証明書データベースへの証明書の追加

この節では、証明書をバックエンド LDAP サーバーから Directory Proxy Server 上の証明書データベースに追加する方法について説明します。

▼ 証明書をバックエンド LDAP サーバーから Directory Proxy Server 上の証明書データベースに追加する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 次のコマンド構文を使用して、 **PEM** 形式のバックエンドディレクトリサーバーからの証明書を表示します。

```
dsadm show-cert -F ascii instance-path [cert-alias]
```

cert-alias を指定しないと、デフォルトのサーバー証明書が表示されます。すべてのコマンドオプションについては、dsadm(1M) のマニュアルページを参照してください。

たとえば、次のように入力して、デフォルトの自己署名付きサーバー証明書を表示します。

```
$ dsadm show-cert -F ascii /local/ds defaultCert
-----BEGIN CERTIFICATE-----
MIICJjCCAY+gAwIBAgIFAIKL36kwDQYJKoZIhvcNAQEEBQAwVzEZMBcGA1UEChMQ
U3VuIE1pY3Jvc3lzdGVtczEZMBcGA1UEAxMQRGlyZWN0b3J5IFNlcnZlcjENMAsG
A1UEAxMEMjAxMTEQMA4GA1UEAxMHY29uZHlsZTAeFw0wNjA1MjIxMTQxNTVaFw0w
NjA4MjIxMTQxNTVaMFcxGTAXBgNVBAoTEFN1biBNaWNyb3N5c3RlbXMxGTAXBgNV
BAMTEERpcmVjdG9yeSBTZXJ2ZXIxDTALBgNVBAMTBDIwMTExEDAOBgNVBAMTB2Nv
bmR5bGUwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAK9U3ry3sJmEzwQY8CGd
7S2MTZuBedo03Vea1lfDtD08WIsdDMzhHplTdeHAkWWNc8g2PDcEFXeWp9UXFMuD
```

```
Pcia7t8HtFkm73VmlriWhMd8nn3l2vkxhsPK2LHFEeOIUDR9LBBiMiEeLkjdoEhE
VLMSoYKqKI+Aa5grINdmtFzBAgMBAAEwDQYJKoZIhvcNAQEEBQADgYEAF4eDbSd7
qy2l10dIogT+rnXZ362gLTlQFCblhbGpmmptbegUdL1ITGv/62q1isPV2rW7CkjM
Cqb0fo3k5UkKKvW+JbMowpQeAPnlgpX612HuDr1tldnKV4eyU7gpG31t/cpACALQ
7OPi1A7oVb2Z8OJKfEJHkp3txBSsiI2gTkk=
-----END CERTIFICATE-----
```

2 証明書を保存します。

証明書をテキストファイルで保存し、安全な場所に証明書をバックアップします。

3 証明書をバックエンド **LDAP** サーバーから **Directory Proxy Server** のインスタンス上の証明書データベースに追加します。

```
$ dpadm add-cert instance-path cert-alias cert-file
```

ここで、*cert-alias* は証明書の名前で、*cert-file* は証明書に含まれるファイルの名前です。

たとえば、証明書 `defaultCert` は次のようにして追加できます。

```
$ dpadm add-cert /local/dps defaultCert /local/safeplace/defaultCert.ascii
```

バックエンド **LDAP** サーバーへの証明書のエクスポート

バックエンド LDAP サーバーは、Directory Proxy Server からの証明書を必要とすることがあります。この節では、証明書をバックエンド LDAP サーバーにエクスポートするように Directory Proxy Server を設定する方法について説明します。

▼ クライアント証明書をバックエンド **LDAP** サーバーにエクスポートするように **Directory Proxy Server** を設定する

1 バックエンド **LDAP** サーバーに送信する証明書を指定します。

```
$ dpconf set-server-prop -h host -p port ssl-client-cert-alias:cert-alias
```

ここで、*cert-alias* は証明書の名前です。すべてのコマンドオプションについては、`dpconf`(1M) のマニュアルページを参照してください。

2 証明書の内容をファイルにコピーします。

```
$ dpadm show-cert -F ascii -o filename instance-path cert-alias
```

3 116 ページの「CA 署名付きサーバー証明書と信頼できる CA 証明書を追加する」で示すように、証明書をバックエンド LDAP サーバーの証明書データベースに追加します。

次の手順 バックエンド LDAP サーバーをクライアント認証用に設定します。Directory Server 用にこれを行う方法については、124 ページの「資格レベルと認証方法の設定」を参照してください。

参照 クライアントと Directory Proxy Server 間の証明書ベースの認証の設定については、483 ページの「証明書ベースの認証を設定する」を参照してください。

Directory Proxy Server 用の証明書データベースのバックアップと復元

サーバー証明書は、`dpadm` を使って Directory Proxy Server をバックアップするときにバックアップされます。バックアップされた証明書は、 *archive-path*`/alias` ディレクトリに格納されます。

Directory Proxy Server のバックアップと復元の方法については、364 ページの「Directory Proxy Server インスタンスのバックアップと復元」を参照してください。

証明書データベースにアクセスするためのパスワードの入力要求

デフォルトでは、証明書データベース用のパスワードは内部的に管理されます。したがって、証明書パスワードを入力したりパスワードファイルを指定したりする必要はありません。証明書データベースが格納されているパスワードによって内部的に管理されている場合、パスワードは安全な環境に格納されます。

証明書のセキュリティーを高め、さらに管理するためには、コマンド行でパスワードの入力を要求するように Directory Proxy Server を設定します。それにより、すべての `dpadm` サブコマンドに対してパスワードを入力する要求されます。ただし、`autostart` 、`backup`、`disable-service`、 `enable-service`、`info`、`restore`、および `stop` は除きます。

パスワードの入力を要求する、または要求しないという Directory Proxy Server の設定の詳細は、次の手順を参照してください。

▼ 証明書データベースにアクセスするためのパスワードの入力を要求する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サーバーを停止します。

```
$ dpadm stop instance-path
Directory Proxy Server instance 'instance-path' stopped
```

2 パスワードプロンプトフラグを on に設定してから、証明書データベースのパスワードを入力し、確認します。

```
$ dpadm set-flags instance-path cert-pwd-prompt=on
Choose the certificate database password:
Confirm the certificate database password:
```

3 サーバーを起動してから、証明書データベースのパスワードを入力します。

```
$ dpadm start instance-path
Enter the certificate database password:
```

▼ 証明書データベースにアクセスするためのパスワードの入力要求を無効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サーバーを停止します。

```
$ dpadm stop instance-path
Directory Proxy Server instance 'instance-path' stopped
```

2 パスワードプロンプトフラグを off に設定してから、既存のパスワードを入力します。

```
$ dpadm set-flags instance-path cert-pwd-prompt=off
Enter the old password:
```

3 次のように入力して、サーバーを起動します。

```
$ dpadm start instance-path
```

◆ ◆ ◆ 第 20 章

LDAP データソースとデータソースプール

この章では、dpconf コマンドを使用して、LDAP データソースとデータソースプールを作成し、設定する方法を説明します。これらのトピックの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「LDAP Data Sources」を参照してください。

この章の内容は次のとおりです。

LDAP データソースの作成と設定

LDAP データソースの作成と設定の方法については、次の手順を参照してください。

▼ LDAP データソースを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースを作成します。

```
$ dpconf create-ldap-data-source -h host -p port source-name host:port
```

このコマンドで、*source-name* は新しいデータソースに割り当てる名前です。*host* と *port* は、LDAP サーバーが実行されているホストとポートを示します。データソースはデフォルトで SSL を使用しない点に注意してください。

ホストが IP V6 アドレスで指定されている場合、データソースの作成時に IP V6 参照を使用する必要があります。たとえば、Directory Proxy Server がポート 2389 で IP V6

アドレス fe80::209:3dff:fe00:8c93 を持つホストにバインドされる場合、次のコマンドを使用してデータソースを作成します。

```
$ dpconf create-ldap-data-source -h host1 -p 1389 ipv6-host \
 [fe80::209:3dff:fe00:8c93]:2389
```

コンソールを使用してデータソースを作成する場合は、実際の IP V6 アドレスを角括弧なしで指定する必要があります。

LDAP データソースのプロパティーの変更方法については、382 ページの「LDAP データソースを設定する」を参照してください。

2 (省略可能) データソースの一覧を表示します。

```
$ dpconf list-ldap-data-sources -h host -p port
```

▼ LDAP データソースを設定する

この手順では、Directory Proxy Server と LDAP データソース間の認証を設定します。また、Directory Proxy Server が LDAP データソースを監視する方法も設定します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 次のコマンド構文を使用して、データソースのプロパティーを表示します。

```
dpconf get-ldap-data-source-prop -h host -p port [-M unit] [-Z unit] source-name [property...]
```

このコマンドで -M と -Z は、データを表示する単位を示します。M オプションは時間の単位を指定します。-M の値は、月、週、日、時間、分、秒、ミリ秒を示すために、M、 w、d、h、m、 s、または ms にできます。-Z オプションはデータサイズの単位を指定します。-Z の値は、T バイト、G バイト、M バイト、K バイト、バイトを示すために、T、 G、M、k、または b にできます。

プロパティーを指定しないと、すべてのプロパティーが表示されます。LDAP データソースのデフォルトプロパティーは次のとおりです。

```
bind-dn                       :  -
bind-pwd                      :  -
client-cred-mode              :  use-client-identity
connect-timeout               :  10s
description                   :  -
is-enabled                    :  false
is-read-only                  :  true
ldap-address                  :  host
ldap-port                     :  port
ldaps-port                    :  ldaps
```

```
monitoring-bind-timeout        :  5s
monitoring-entry-dn            :  ""
monitoring-entry-timeout       :  5s
monitoring-inactivity-timeout  :  2m
monitoring-interval            :  30s
monitoring-mode                :  proactive
monitoring-search-filter       :  (|(objectClass=*)(objectClass=ldapSubEntry))
num-bind-incr                  :  10
num-bind-init                  :  10
num-bind-limit                 :  1024
num-read-incr                  :  10
num-read-init                  :  10
num-read-limit                 :  1024
num-write-incr                 :  10
num-write-init                 :  10
num-write-limit                :  1024
proxied-auth-check-timeout     :  1.8s
proxied-auth-use-v1            :  false
ssl-policy                     :  never
use-tcp-no-delay               :  true
```

2 データソースを有効にします。

```
$ dpconf set-ldap-data-source-prop -h host -p port source-name is-enabled:true
```

3 デフォルト設定を変更する場合は、手順 **1** に一覧表示されているプロパティーをすべて設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port source-name property:value
```

たとえば、データソース上のエントリを変更する場合、書き込み操作を許可するようにデータソースを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port source-name is-read-only:false
```

サブコマンドで使用するプロパティーについての情報を見つけるには、次のコマンドを実行します。

```
$ dpconf help-properties ldap-data-source property
```

データソースの主要なプロパティーを一覧表示するには、`list` サブコマンドとともに冗長オプション `-v` を使用します。

```
$ dpconf list-ldap-data-sources -v
Name         is-enabled  ldap-address  ldap-port  ldaps-port  description
-----------  ----------  ------------  ---------  ----------  -----------
datasource0  true        myHost        myPort     ldaps       -
datasource1  true        myHost        myPort     ldaps       -
```

4 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。サーバーの再起動が必要な設定の変更の一覧は、366 ページの「サーバーの再起動を必要とする設定変更」を参照してください。

LDAP データソースプールの作成と設定

データソースプールの作成と設定の方法については、次の手順を参照してください。

▼ LDAP データソースプールを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 1 つまたは複数のデータソースプールを作成します。

```
$ dpconf create-ldap-data-source-pool -h host -p port pool-name
```

最初の *pool-name* のあとに、追加のデータソースプールを指定できます。データソースプールのプロパティーの変更方法については、384 ページの「LDAP データソースプールを設定する」を参照してください。

2 (省略可能) データソースプールの一覧を表示します。

```
$ dpconf list-ldap-data-source-pools -h host -p port
```

▼ LDAP データソースプールを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 次のコマンド構文を使用して、データソースプールのプロパティーを表示します。

```
dpconf get-ldap-data-source-pool-prop -h host -p port [-M unit] [-Z unit] \
 pool-name [property...]
```

このコマンドで `-M` と `-Z` は、データを表示する単位を示します。`M` オプションは時間の単位を指定します。`-M` の値は、月、週、日、時間、分、秒、ミリ秒を示すために、`M`、 `w`、`d`、`h`、`m`、 `s`、または `ms` にできます。`-Z` オプションはデータサイズの単位を指定します。`-Z` の値は、T バイト、G バイト、M バイト、K バイト、バイトを示すために、`T`、 `G`、`M`、`k`、または `b` にできます。

プロパティーを指定しないと、すべてのプロパティーが表示されます。LDAP データソースプールのデフォルトプロパティーは次のとおりです。

```
client-affinity-policy    :  write-affinity-after-write
client-affinity-timeout   :  20s
description               :  -
enable-client-affinity    :  false
load-balancing-algorithm  :  proportional
```

2 手順 1に一覧表示されているプロパティーを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 property:value
```

負荷分散とクライアントアフィニティーのためにデータソースプールのプロパティーを設定する方法については、第 21 章を参照してください。

LDAP データソースのデータソースプールへの接続

データソースプールに接続されたデータソースは、接続済みデータソースと呼ばれます。接続済みデータソースのプロパティーによって、データソースプールの負荷分散設定が決まります。接続済みデータソースのウェイトを設定する場合は、データソースプールのすべての接続済みデータソースのウェイトを考慮します。ウェイトの設定どおりに負荷分散が機能することを確認します。負荷分散のためにウェイトを設定する方法については、388 ページの「負荷分散のウェイトを設定する」を参照してください。

▼ LDAP データソースをデータソースプールに接続する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 1 つまたは複数のデータソースをデータソースプールに接続します。

```
$ dpconf attach-ldap-data-source -h host -p port pool-name \
 source-name [source-name ...]
```

2 (省略可能) 特定のデータソースプールの接続済みデータソースをすべて一覧表示します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -E pool-name
```

このコマンドで、 -E はオプションであり、1 行に 1 つずつプロパティー値を表示するように表示を変更します。

3 (省略可能) 特定のデータソースプールの接続済みデータソースの主要なプロパティーを表示します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
```

このコマンドで、-v は冗長出力を指定します。たとえば、データソースプールの例のプロパティーを表示します。

```
$ dpconf list-attached-ldap-data-sources -h host1 -p 1389 -v My-pool
Name         add-weight  bind-weight  compare-weight
-----------  ----------  -----------  --------------
datasource0  disabled    disabled     disabled
datasource1  disabled    disabled     disabled

delete-weight  modify-dn-weight  modify-weight  search-weight
-------------  ----------------  -------------  -------------
disabled       disabled          disabled       disabled
disabled       disabled          disabled       disabled
```

4 (省略可能) 次のコマンド構文を使用して、接続済みデータソースのプロパティーを表示します。

```
$ dpconf get-attached-ldap-data-source-prop -h host -p port [-M unit] [-Z unit] \
 pool-name source-name [property...]
```

このコマンドで -M と -Z は、データを表示する単位を示します。M オプションは時間の単位を指定します。-M の値は、月、週、日、時間、分、秒、ミリ秒を示すために、M、 w、d、h、m、 s、または ms にできます。-Z オプションはデータサイズの単位を指定します。-Z の値は、T バイト、G バイト、M バイト、K バイト、バイトを示すために、T、 G、M、k、または b にできます。

プロパティーを指定しないと、すべてのプロパティーが表示されます。

接続済みデータソースのプロパティーは、負荷分散で各種の操作のウェイトを定義します。接続済みデータソースのデフォルトウェイトは次のとおりです。

```
add-weight        :  disabled
bind-weight       :  disabled
compare-weight    :  disabled
delete-weight     :  disabled
modify-dn-weight  :  disabled
modify-weight     :  disabled
search-weight     :  disabled
```

負荷分散のために接続済みデータソースのウェイトを設定する方法については、388 ページの「負荷分散のウェイトを設定する」を参照してください。

◆ ◆ ◆ 第 21 章

Directory Proxy Server による負荷分散とクライアントアフィニティー

負荷分散とクライアントアフィニティーについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 16 章「Directory Proxy Server Load Balancing and Client Affinity」を参照してください。この章の内容は次のとおりです。

負荷分散の設定

負荷分散の詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Load Balancing」を参照してください。この節では、負荷分散を設定する方法について説明し、設定例を示します。

▼ 負荷分散アルゴリズムを選択する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **LDAP** データソースプールのプロパティーを表示することで、現在の負荷分散アルゴリズムを取得します。

```
$ dpconf get-ldap-data-source-pool-prop -h host -p port pool-name
```

LDAP データソースプールのデフォルトプロパティーは次のとおりです。

```
client-affinity-policy    :  write-affinity-after-write
client-affinity-timeout   :  20s
description               :  -
enable-client-affinity    :  false
load-balancing-algorithm  :  proportional
```

デフォルトでは、負荷分散アルゴリズムは `proportional` です。

2 アルゴリズムを使用するように **LDAP** データソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
  load-balancing-algorithm:selected-algorithm
```

ここで、*selected-algorithm* は次のいずれかです。

- `failover`
- `operational-affinity`
- `proportional`
- `saturation`

アルゴリズムの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Introduction to Load Balancing」を参照してください。

3 **Directory Proxy Server** のインスタンスを再起動します。

```
$ dpadm restart instance-path
```

▼ 負荷分散のウェイトを設定する

データソースのウェイトは、データソースプールに接続されているほかのすべてのデータソースのウェイトを考慮して設定する必要があります。データソースに操作のタイプに対して `disabled` のウェイトがある場合、そのタイプの要求がそのデータソースに送信されることはありません。データソースに ウェイト `0` (zero) がある場合、他のすべてのデータソースが使用不可でないかぎり、そのデータソースには要求は配信されません。このため、ウェイトが `0` に設定されたデータソースは、他のすべてのデータソースが停止している場合だけ使用されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースプールに接続されているデータソースのリストを表示します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port pool-name
```

2 接続済みデータソースのいずれかのプロパティーを表示します。

```
$ dpconf get-attached-ldap-data-source-prop pool-name \
 attached-data-source-name
```

接続済みデータソースのプロパティーは、各タイプの操作に対するウェイトを定義します。接続済みデータソースのデフォルトウェイトは次のとおりです。

```
add-weight        :  disabled
bind-weight       :  disabled
compare-weight    :  disabled
```

```
delete-weight     :  disabled
modify-dn-weight  :  disabled
modify-weight     :  disabled
search-weight     :  disabled
```

3 接続済みデータソースのいずれかのウェイトを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name \
 attached-data-source-name add-weight:value \
 bind-weight:value compare-weight:value delete-weight:value \
 modify-dn-weight:value modify-weight:value search-weight:value
```

4 接続されているほかのデータソースに対して、手順 2 と 手順 3 を繰り返します。

5 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
```

たとえば、データソースプールには、次のようなウェイトを持つデータソースを含めることができます。

```
$ dpconf list-attached-ldap-data-sources -h host1 -p 1389 -v myPool
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
---- ---------- ----------- -------------- ------------- ---------------- ------------- -------------
DS-1 disabled   3               disabled       disabled      disabled         disabled      disabled
DS-2 2          2           2              2             2                2             2
DS-3 1          1           1              1             1                1             1
```

負荷分散の設定例

この節では、各負荷分散アルゴリズムの設定手順の例を示します。

▼ 比例アルゴリズムを用いて負荷分散を設定する

比例アルゴリズムについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Proportional Algorithm for Load Balancing」を参照してください。

この例では、データソース *ds–1* が、他の 2 つのデータソースのウェイトの 2 倍に設定されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に データソースプールに少なくとも 3 つの接続済みデータソースが含まれていることを確認します。データソースとデータソースプールを作成する方法については、第 20 章を参照してください。

1. 負荷分散に対して比例アルゴリズムを使用するようにデータソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
  load-balancing-algorithm:proportional
```

2. 最初のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-1 \
 add-weight:2 bind-weight:2 compare-weight:2 delete-weight:2 modify-dn-weight:2 \
 modify-weight:2 search-weight:2
```

3. 2番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-2 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

4. 3番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-3 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

5. 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
---  ---------- ----------- -------------- ------------- ---------------- ------------- -------------
ds-1 2          2           2              2             2                2             2
ds-2 1          1           1              1             1                1             1
ds-3 1          1           1              1             1                1             1
```

6. **Directory Proxy Server** のインスタンスを再起動します。

```
$ dpadm restart instance-path
```

▼ 飽和アルゴリズムを用いて負荷分散を設定する

飽和アルゴリズムについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Saturation Algorithm for Load Balancing」を参照してください。

この例では、データソース `ds-1` で大半のバインド操作が実行されますが、ほかのタイプの操作は実行されません。3つのデータソースが次のウェイトによって設定されます。

- `ds-1` は、バインド操作に対してウェイト3を持つよう設定され、他のすべてのタイプの操作に対しては無効です。
- `ds-2` は、すべての操作に対してウェイト2を持つよう設定されます。
- `ds-3` は、すべての操作に対してウェイト1を持つよう設定されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　データソースプールに少なくとも 3 つの接続済みデータソースが含まれていることを確認します。データソースとデータソースプールを作成する方法については、第 20 章を参照してください。

1. 負荷分散に対して飽和アルゴリズムを使用するようにデータソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 load-balancing-algorithm:saturation
```

2. 最初のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-1 \
 add-weight:disabled bind-weight:3 compare-weight:disabled delete-weight:disabled \
 modify-dn-weight:disabled modify-weight:disabled search-weight:disabled
```

3. 2 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-2 \
 add-weight:2 bind-weight:2 compare-weight:2 delete-weight:2 modify-dn-weight:2 \
 modify-weight:2 search-weight:2
```

4. 3 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-3 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

5. 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
---  ---------- ----------- -------------- ------------- ---------------- ------------- -------------
ds-1 disabled   3               disabled       disabled      disabled         disabled      disabled
ds-2 2          2           2              2             2                2             2
ds-3 1          1           1              1             1                1             1
```

6. Directory Proxy Server のインスタンスを再起動します。

```
$ dpadm restart instance-path
```

▼ グローバルアカウントロックアウトに対してアフィニティーアルゴリズムを設定する

このアルゴリズムについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Operational Affinity Algorithm for Global Account Lockout」を参照してください。

この例には、3 つのデータソースがあります。データソース `ds-1` は、すべてのバインド要求を受信するように設定されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　データソースプールに少なくとも 3 つの接続済みデータソースが含まれていることを確認します。データソースとデータソースプールを作成する方法については、第 20 章を参照してください。

1. アフィニティーアルゴリズムを使用するようにデータソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 load-balancing-algorithm:operational-affinity
```

2. 最初のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-1 \
 add-weight:1 bind-weight:100 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

3. 2 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-2 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

4. 3 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-3 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

5. 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
--   ---------- ----------- -------------- ------------- ---------------- ------------- -------------
ds-1 1          1           1              1             1                1             1
ds-2 1          100         1              1             1                1             1
ds-3 1          1           1              1             1                1             1
```

6 **Directory Proxy Server** のインスタンスを再起動します。

```
$ dpadm restart instance-path
```

▼ キャッシュの最適化のためにアフィニティーアルゴリズムを設定する

このアルゴリズムについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Operational Affinity Algorithm for Cache Optimization」を参照してください。

この例には、3 つのデータソースがあります。すべての検索操作と比較操作は、データソース `ds-1` で処理されます。`ds-1` が要求に応答すると、ターゲットエントリがキャッシュ内に格納されます。`ds-1` が同じ要求に繰り返し応答する場合、データソースはキャッシュに入れられたデータを使用できます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　データソースプールに少なくとも 3 つの接続済みデータソースが含まれていることを確認します。データソースとデータソースプールを作成する方法については、第 20 章を参照してください。

1 アフィニティーアルゴリズムを使用するようにデータソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 load-balancing-algorithm:operational-affinity
```

2 最初のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-1 \
 add-weight:1 bind-weight:1 compare-weight:100 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:100
```

3 2 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-2 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

4 3 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-3 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

5 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
---  ---------- ----------- -------------- ------------- ---------------- ------------- -------------
ds-1 1          1           100            1             1                1             100
ds-2 1          1           1              1             1                1             1
ds-3 1          1           1              1             1                1             1
```

6 **Directory Proxy Server** のインスタンスを再起動します。

```
$ dpadm restart instance-path
```

▼ フェイルオーバーアルゴリズムを用いて負荷分散を設定する

フェイルオーバーアルゴリズムについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Failover Algorithm for Load Balancing」を参照してください。

この例には、3 つのデータソースがあります。データソース `ds-1` はすべての要求を受信します。`ds-1` が失敗した場合は、`ds-1` が回復するまで `ds-2` がすべての要求を受信します。`ds-1` が回復する前に `ds-2` が失敗した場合は、`ds-3` がすべての要求を受信します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に データソースプールに少なくとも 3 つの接続済みデータソースが含まれていることを確認します。データソースとデータソースプールを作成する方法については、第 20 章を参照してください。

1 負荷分散に対してフェイルオーバーアルゴリズムを使用するように、データソースプールを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 load-balancing-algorithm:failover
```

2 最初のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-1 \
 add-weight:3 bind-weight:3 compare-weight:3 delete-weight:3 modify-dn-weight:3 \
 modify-weight:3 search-weight:3
```

3 2 番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-2 \
 add-weight:2 bind-weight:2 compare-weight:2 delete-weight:2 modify-dn-weight:2 \
 modify-weight:2 search-weight:2
```

4 3番目のデータソースのプロパティーを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h host -p port pool-name ds-3 \
 add-weight:1 bind-weight:1 compare-weight:1 delete-weight:1 modify-dn-weight:1 \
 modify-weight:1 search-weight:1
```

5 接続済みデータソースのキーパラメータを比較します。

```
$ dpconf list-attached-ldap-data-sources -h host -p port -v pool-name
Name add-weight bind-weight compare-weight delete-weight modify-dn-weight modify-weight search-weight
---  ---------- ----------- -------------- ------------- ---------------- ------------- -------------
ds-1 3          3           3              3             3                3             3
ds-2 2          2           2              2             2                2             2
ds-3 1          1           1              1             1                1             1
```

6 Directory Proxy Serverのインスタンスを再起動します。

```
$ dpadm restart instance-path
```

クライアントアフィニティーの設定

クライアントアフィニティーにより、負荷分散された配備での伝播遅延のリスクが削減されます。クライアントアフィニティーについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Client Affinity」を参照してください。この節では、クライアント接続とデータソース間のアフィニティーを設定する方法を説明し、設定例を示します。

▼ クライアントアフィニティーを設定する

この手順では、クライアント接続とデータソース間のアフィニティーを設定する方法について説明します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースプールのプロパティーを表示することで、現在の負荷分散アルゴリズムを表示します。

```
$ dpconf get-ldap-data-source-pool-prop -h host -p port pool-name
```

データソースプールのデフォルトプロパティーは、次のとおりです。

```
client-affinity-policy    :  write-affinity-after-write
client-affinity-timeout   :  20s
description               :  -
enable-client-affinity    :  false
load-balancing-algorithm  :  proportional
```

次のパラメータは、クライアントアフィニティーを設定します。
`client-affinity-policy` 、`client-affinity-timeout`、`enable-client-affinity` 。プロパティーの詳細とそれらの有効な値のリストについては、次のように入力します。

```
dpconf help-properties ldap-data-source-pool client-affinity-policy \
 client-affinity-timeout enable-client-affinity
```

プロパティーの詳細は、次のマニュアルページを参照してください。`client-affinity-policy`(5dpconf)、`client-affinity-timeout`(5dpconf)、および`enable-client-affinity`(5dpconf)。

2 クライアントアフィニティーを有効にします。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 enable-client-affinity:true
```

3 クライアントアフィニティーに対するポリシーを選択します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 client-affinity-policy:selected-policy
```

ここで、*selected-policy* は次のいずれかです。

`write-affinity-after-write`
　最初の書き込み要求のあとの書き込み要求に対するアフィニティー

`read-write-affinity-after-write`
　最初の書き込み要求のあとのすべての要求に対するアフィニティー

`read-write-affinity-after-any`
　最初の読み取り要求または書き込み要求のあとのすべての要求に対するアフィニティー

`read-affinity-after-write`
　書き込み要求のあとの最初の読み取り要求に対するアフィニティー

4 クライアントアフィニティーの期間を設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 client-affinity-timeout:time-out[unit]
```

タイムアウトのデフォルトの *unit* はミリ秒です。

クライアントアフィニティーの設定例

この節では、クライアントアフィニティーに関連する設定の例を示し、レプリケーションの遅延、書き込み操作の検証、接続ベースのルーティングの例を示します。

▼ データソースプールにマスターとコンシューマが含まれている場合に、レプリケーションの遅延に対するクライアントアフィニティーを設定する

この手順では、最初の書き込み操作後、3 秒までの間に行われるすべての読み取り操作と書き込み操作に対するクライアントアフィニティーを設定します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- データソースプールのアフィニティーパラメータを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 client-affinity-policy:read-write-affinity-after-write client-affinity-timeout:3000 \
 enable-client-affinity:true
```

▼ 書き込み操作のそれぞれを読み取り操作で検証するようにクライアントアフィニティーを設定する

この手順では、それぞれの書き込み操作のあとの最初の読み取り操作に対するクライアントアフィニティーを設定します。例は、読み取り操作を行うことで指定した DN がそれぞれの書き込み操作の妥当性検査を行うアプリケーションの場合もあります。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- データソースプールのアフィニティーパラメータを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 client-affinity-policy:read-affinity-after-write enable-client-affinity:true
```

▼ 接続ベースのルーティングに対するクライアントアフィニティーを設定する

Directory Proxy Server 6.0 より前のバージョンでは、クライアントと LDAP サーバー間で確立される接続数は 1 つです。この接続が閉じられるまで、クライアントからのすべての要求に使用されました。このタイプのルーティングを、*connection-based routing* といいます。この手順では、接続ベースのルーティングに対してクライアントアフィニティーを設定する方法について説明します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

始める前に　すべてのデータソースがデータソースプールに接続されていて、`clientCredentialsForwarding` が `useBind` に設定されていることを確認します。

- データソースプールのアフィニティーパラメータを設定します。

```
$ dpconf set-ldap-data-source-pool-prop -h host -p port pool-name \
 client-affinity-policy:read-write-affinity-after-any enable-client-affinity:true
```

◆ ◆ ◆ 第 22 章

22 Directory Proxy Server によるデータ配布

Directory Proxy Server によるデータ配布の概要とユースケースの例の説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 17 章「Directory Proxy Server Distribution」を参照してください。

この章の内容は次のとおりです。

LDAP データビューの作成と設定

LDAP データビューの作成と設定の方法については、次の手順を参照してください。

▼ LDAP データビューを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **LDAP データビューを作成します。**

```
$ dpconf create-ldap-data-view -h host -p port view-name pool-name suffix-DN
```

LDAP データビューのプロパティーの変更方法については、400 ページの「LDAP データビューを設定する」を参照してください。

2 **LDAP データビューの一覧を表示します。**

```
$ dpconf list-ldap-data-views -h host -p port
```

▼ LDAP データビューを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 LDAP データビューのプロパティーを表示します。

```
$ dpconf get-ldap-data-view-prop -h host -p port view-name
```

プロパティーを設定せずにデータビューを作成すると、データビューは次のような設定になります。

```
alternate-search-base-dn              :  ""
attr-name-mappings                       :  none
base-dn                                  :  suffix-DN
contains-shared-entries                  :  false
custom-distribution-algorithm-class      :  none
description                              :  -
distribution-algorithm                   :  none
dn-join-rule                             :  none
dn-mapping-attrs                         :  none
dn-mapping-source-base-dn                :  none
excluded-subtrees                        :  -
filter-join-rule                         :  none
is-enabled                               :  true
is-read-only                             :  false
is-routable                              :  true
ldap-data-source-pool                    :  pool-name
lexicographic-attrs                      :  all
lexicographic-lower-bound                :  none
lexicographic-upper-bound                :  none
non-viewable-attr                        :  none
non-writable-attr                        :  none
numeric-attrs                            :  all
numeric-default-data-view                :  false
numeric-lower-bound                      :  none
numeric-upper-bound                      :  none
pattern-matching-base-object-search-filter  :  all
pattern-matching-dn-regular-expression      :  all
pattern-matching-one-level-search-filter    :  all
pattern-matching-subtree-search-filter      :  all
process-bind                             :  -
replication-role                         :  master
viewable-attr                            :  all except non-viewable-attr
writable-attr                            :  all except non-writable-attr
```

注 – プロキシマネージャー以外のすべてのユーザーには、バックエンドサーバーの cn=config および cn=monitor サフィックスが表示されます。プロキシマネージャーは、デフォルトでバックエンドサーバーからのデータを使用できません。プロキシマネージャーが使用できる cn=config および cn=monitor サブツリーは、プロキシ自体のサブツリーです。

Directory Proxy Server インスタンスを作成すると、プロキシマネージャーの接続ハンドラが空のデータビューポリシーで作成されます。プロキシマネージャーがバックエンドデータへのアクセスを必要とする場合、プロキシマネージャーの接続ハンドラのデータビューポリシーにデータビューを追加する必要があります。このようなデータビューでは、cn=config および cn=monitor サブツリーは、デフォルトで除外されます。

2 手順 1で一覧表示されるプロパティーの 1 つまたは複数を変更します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
 property:value [property:value ... ]
```

たとえば、データソースの dc=example,dc=com サブツリーにアクセスするには、データビューで base-dn と指定します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 myDataView base-dn:dc=example,dc=com
```

複数値プロパティーに値を追加するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name property+:value
```

複数値プロパティーから値を削除するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name property-:value
```

3 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

▼ カスタム配布アルゴリズムを設定する

全種類のデータビュー (ldap-data-view、jdbc-data-view、ldif-data-view、および join-data-view) でカスタム配布アルゴリズムを設定できます。次の手順では、このアルゴリズムを ldap-data-view だけで設定します。

1 配布アルゴリズムクラスが格納された **Java Archive (JAR)** ファイルのパスが含まれるように extension-jar-file-url プロパティーを設定します。

```
$ dpconf set-server-prop -h host -p port extension-jar-file-url:jar file path
```

jar file path を、`file:/expt/dps/custom_plugin/myjar.jar` のような有効な JAR ファイルパスに置き換えることができます。

2. `custom-distribution-algorithm` を設定する前に、`distribution-algorithm` を `none` に設定します。

```
$ dpconf set-ldap-data-view-prop view name distribution-algorithm:none
```

3. `custom-distribution-algorithm` プロパティーをカスタム配布アルゴリズムクラスに設定します。

```
$ dpconf set-ldap-data-view-prop view name custom-distribution-algorithm:PackageName.AlgoClassName
```

属性と DN の名前の変更

ディレクトリ内の各エントリは、DN および一連の属性とその値によって識別されます。しばしば、クライアント側で定義された DN と属性は、サーバー側で定義された DN と属性にマップされません。DN と属性の名前を変更するためにデータビューを定義できます。クライアントが要求を行うと、DN と属性の名前がサーバー側と一致するように変更されます。結果がクライアントに返されると、DN と属性はクライアント側と一致するよう元に戻されます。

属性と DN の名前の変更については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Attribute Renaming and DN Renaming」を参照してください。属性と DN の名前の変更方法については、次の手順を参照してください。

▼ 属性の名前の変更を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 属性のマッピングを設定するデータビューで 1 つまたは複数の `attr-name-mappings` プロパティーを設定します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
 attr-name-mappings:client-side-attribute-name#server-side-attribute-name
  [attr-name-mappings:client-side-attribute-name#server-side-attribute-name ...]
```

たとえば、クライアント側の `surname` をサーバー側では `sn` と名前変更します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 myDataView \
 attr-name-mappings:surname#sn
```

既存のマッピングリストに属性マッピングを追加するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
 attr-name-mappings+:client-side-attribute-name#server-side-attribute-name
```

既存のマッピングリストから属性マッピングを削除するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
 attr-name-mappings-:client-side-attribute-name#server-side-attribute-name
```

▼ DN の名前の変更を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 **DN** の名前を変更するデータビューの `base-dn` プロパティーと **DN** マッピングプロパティーを表示します。

```
$ dpconf get-ldap-data-view-prop -h host -p port view-name base-dn \
 dn-mapping-source-base-dn dn-mapping-attrs
```

プロパティーの意味は次のとおりです。

- `base-dn` は、クライアント側のサブツリーの DN で、データビューのベース DN と同じです。
- `dn-mapping-source-base-dn` はサーバー側のサブツリーの DN です。
- `dn-mapping-attrs` はエントリの DN を含む属性の一覧を定義します。

たとえば、DN の名前の変更が定義されていない場合、クライアント側の `dc=example,dc=com` データベースのデータビューは次の値になります。

```
$ dpconf get-ldap-data-view-prop myDataView base-dn \
 dn-mapping-source-base-dn dn-mapping-attrs
 base-dn                   :  dc=example,dc=com
 dn-mapping-attrs          :  none
 dn-mapping-source-base-dn  :  none
```

2 クライアント側の **DN** をサーバー側の **DN** にマップします。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
  dn-mapping-source-base-dn:server-side-dn
```

たとえば、クライアント側の dc=example,dc=com データベースをサーバー側の dc=example,dc=org にマップします。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 myDataView \
 dn-mapping-source-base-dn:dc=example,dc=org
```

3 手順 2 の影響を受ける DIT の一部で属性に DN が含まれている場合、これらの属性の名前を変更します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
  dn-mapping-attrs:attribute-name [dn-mapping-attrs:attribute-name ...]
```

たとえば、手順 2の名前変更操作に影響を受ける名前空間で group 属性に DN が含まれている場合、次のように属性の名前を変更します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 myDataView dn-mapping-attrs:group
```

既存のマッピングリストに DN マッピングを追加するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name dn-mapping-attrs+:attribute-name
```

既存のマッピングリストから DN マッピングを削除するには、次のコマンドを使用します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name dn-mapping-attrs-:attribute-name
```

4 DN の名前を変更したデータビューの base-dn プロパティーと DN マッピングプロパティーを表示します。

```
$ dpconf get-ldap-data-view-prop -h host -p port view-name base-dn \
 dn-mapping-source-base-dn dn-mapping-attrs
```

たとえば、DN の名前の変更後、クライアント側の dc=example,dc=com データベースのデータビューは次の値になります。

```
$ dpconf get-ldap-data-view-prop -h host1 -p 1389 myDataView base-dn \
 dn-mapping-source-base-dn dn-mapping-attrs
 base-dn                   :  dc=example,dc=com
 dn-mapping-attrs          :  group
 dn-mapping-source-base-dn  :  dc=example,dc=org
```

excluded-subtrees と alternate-search-base-dn の設定

下位のデータビューが作成されると、Directory Proxy Server は上位のデータビューから下位のデータビューを自動的に除外します。要求のターゲットが下位のデータビューの場合、要求は上位のデータビューではなく、下位のデータビューに送られます。

下位のデータビューで代替検索ベースが指定されている場合、上位のデータビューをターゲットとした検索操作も下位のデータビューで実行されます。

デフォルトで、Directory Proxy Server は `excluded-subtrees` プロパティーと `alternate-search-base-dn` プロパティーを自動的に設定します。次の手順では、これらのプロパティーの手動での設定方法を説明します。

▼ excluded-subtrees プロパティーと alternate-search-base-dn プロパティーを手動で設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 手動で要求を経路指定するよう **Directory Proxy Server** を設定します。

```
$ dpconf set-server-prop -h host -p port data-view-automatic-routing-mode:manual
```

`data-view-automatic-routing-mode` が `manual` の場合、Directory Proxy Server は `excluded-subtrees` プロパティーと `alternate-search-base-dn` プロパティーを生成しません。これらのプロパティーの値は手動で設定する必要があります。ここで設定した値は、Directory Proxy Server で確認されません。これらの値を間違って設定すると、管理パスが壊れる可能性があることに注意してください。

または、要求を部分的に手動で経路指定するように Directory Proxy Server を設定します。

```
$ dpconf set-server-prop -h host -p port data-view-automatic-routing-mode:limited
```

`data-view-automatic-routing-mode` が `limited` の場合、Directory Proxy Server は `excluded-subtrees` プロパティーと `alternate-search-base-dn` プロパティーを生成しません。ただし、ここで設定した値が管理パスと競合しないか、Directory Proxy Server が確認します。

2 ビュー除外ベースを設定します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name excluded-subtrees:suffix-DN
```

ビュー除外ベースによって、データビューでエントリが公開されない DIT のエントリが決まります。

3 代替検索ベースを設定します。

```
$ dpconf set-ldap-data-view-prop -h host -p port view-name \
 alternate-search-base-dn:search-base-DN
```

代替検索ベースによって、このデータに属しているエントリが配置される DIT のほかのエントリが決まります。ベース DN は、すべてのデータビューで代替検索ベースとしてデフォルトで定義されます。

データビューの作成と設定に関するユースケースの例

この節では、データビューの次の情報とその作成および設定方法について説明します。

- 406 ページの「デフォルトデータビュー」
- 407 ページの「要求のターゲット DN にかかわりなく、要求をすべて経路指定するデータビュー」
- 408 ページの「サブツリーの一覧がデータとして等価な複数のデータソースに保存されている場合に要求を経路指定するデータビュー」
- 410 ページの「異なるサブツリーが異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビュー」
- 412 ページの「サブツリーの異なる部分が異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビュー」
- 414 ページの「上位サブツリーと下位サブツリーが異なるデータソースに保存される場合に単一のアクセスポイントを提供するデータビュー」
- 416 ページの「階層と配布アルゴリズムを持つデータビュー」

ここでの例は、接続ハンドラが Directory Proxy Server によって処理されるすべてのクライアント接続を許可していることを前提としています。

デフォルトデータビュー

プロパティーを設定せずにデータビューを作成すると、データビューは次のような設定になります。

```
alternate-search-base-dn                    :  ""
alternate-search-base-dn                    :  base-DN
attr-name-mappings                          :  none
base-dn                                     :  suffix-DN
contains-shared-entries                     :  -
description                                 :  -
distribution-algorithm                      :  -
```

```
dn-join-rule                                 :  -
dn-mapping-attrs                             :  none
dn-mapping-source-base-dn                    :  none
excluded-subtrees                            :  -
filter-join-rule                             :  -
is-enabled                                   :  true
is-read-only                                 :  false
is-routable                                  :  true
ldap-data-source-pool                        :  pool-name
lexicographic-attrs                          :  all
lexicographic-lower-bound                    :  none
lexicographic-upper-bound                    :  none
non-viewable-attr                            :  -
non-writable-attr                            :  -
numeric-attrs                                :  all
numeric-default-data-view                    :  false
numeric-lower-bound                          :  none
numeric-upper-bound                          :  none
pattern-matching-base-object-search-filter   :  all
pattern-matching-dn-regular-expression       :  all
pattern-matching-one-level-search-filter     :  all
pattern-matching-subtree-search-filter       :  all
process-bind                                 :  -
replication-role                             :  master
viewable-attr                                :  all except non-viewable-attr
writable-attr                                :  all except non-writable-attr
```

要求のターゲット DN にかかわりなく、要求をすべて経路指定するデータビュー

この節では、要求のターゲット DN にかかわりなく、要求をすべてデータソースプールに経路指定するデータビューの設定について説明します。このデータビューは、*root* データビューと呼ばれます。root データビューは、Directory Proxy Server のインスタンスが作成されると、デフォルトで作成されます。root データビューについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views to Route All Requests, Irrespective of the Target DN of the Request」を参照してください。

root データビューには次の設定があります。

```
alternate-search-base-dn                  :  -
attr-name-mappings                        :  none
base-dn                                   :  ""
contains-shared-entries                   :  -
description                               :  Automatically-generated data view
                                             able to route client operations
```

```
                                                independently of the operation base dn
distribution-algorithm                        :  -
dn-join-rule                                  :  -
dn-mapping-attrs                              :  none
dn-mapping-source-base-dn                     :  none
excluded-subtrees                             :  ""
excluded-subtrees                             :  cn=config
excluded-subtrees                             :  cn=monitor
excluded-subtrees                             :  cn=proxy manager
excluded-subtrees                             :  cn=virtual access controls
excluded-subtrees                             :  dc=example,dc=com
filter-join-rule                              :  -
is-enabled                                    :  true
is-read-only                                  :  false
is-routable                                   :  true
ldap-data-source-pool                         :  defaultDataSourcePool
lexicographic-attrs                           :  all
lexicographic-lower-bound                     :  none
lexicographic-upper-bound                     :  none
non-viewable-attr                             :  -
non-writable-attr                             :  -
numeric-attrs                                 :  all
numeric-default-data-view                     :  false
numeric-lower-bound                           :  none
numeric-upper-bound                           :  none
pattern-matching-base-object-search-filter :  all
pattern-matching-dn-regular-expression     :  all
pattern-matching-one-level-search-filter   :  all
pattern-matching-subtree-search-filter     :  all
process-bind                                  :  -
replication-role                              :  master
viewable-attr                                 :  all except non-viewable-attr
writable-attr                                 :  all except non-writable-attr
```

サブツリーの一覧がデータとして等価な複数のデータソースに保存されている場合に要求を経路指定するデータビュー

この節では、サブツリーの一覧をターゲットとする要求をデータ同等のデータソースセットに経路指定するデータビューの設定方法を説明します。このような配備については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views to Route Requests When a List of Subtrees Are Stored on Multiple, Data-Equivalent Data Sources」を参照してください。

ここでの例には、同じサブツリーのセットを含む複数のデータソースが含まれています。データソースはデータと同等で、負荷分散のために 1 つのデータソースプールにプールされます。データビューは、サブツリーをクライアント要求に公開するために、各サブツリーに対して設定されます。次の図は、配備の例を示しています。

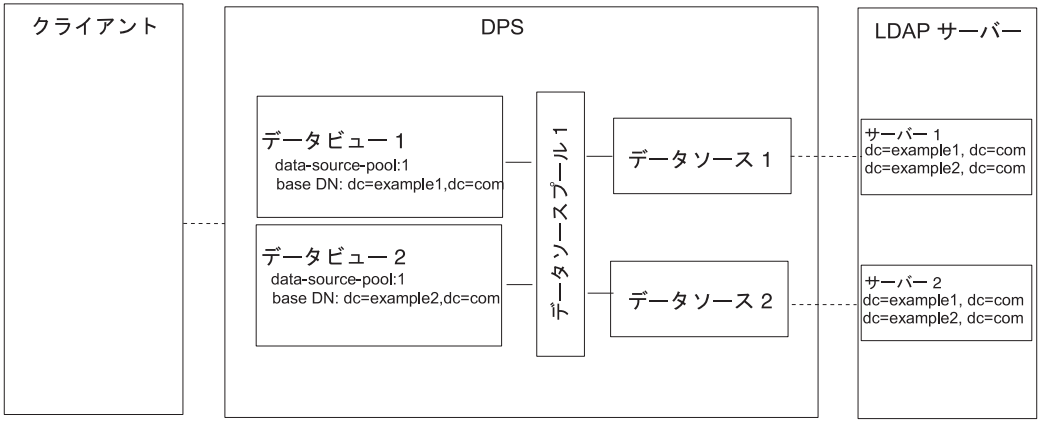


図 22–1　サブツリーの一覧が複数のデータ同等のデータソースに保存されると要求を経路指定する配備の例

▼ サブツリーの一覧がデータとして等価な複数のデータソースに保存されている場合に要求を経路指定するデータビューを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **381 ページの「LDAP データソースの作成と設定」**で説明しているように、各 **LDAP** サーバーにデータソースを作成します。
2. **384 ページの「LDAP データソースプールの作成と設定」**で説明しているように、データソースプールを作成します。
3. **385 ページの「LDAP データソースのデータソースプールへの接続」**で説明しているように、データソースをデータソースプールに接続します。
4. **(省略可能) 負荷分散を設定します。**
   詳細は、387 ページの「負荷分散の設定」を参照してください。

5 dc=example1,dc=com でデータソースプールを参照するベース DN を持つデータビューを作成します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-1 \
  base-dn:dc=example1,dc=com ldap-data-source-pool:data-source-pool-1
```

6 dc=example2,dc=com でデータソースプールを参照するベース DN を持つデータビューをもう 1 つ作成します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-2 \
  base-dn:dc=example2,dc=com ldap-data-source-pool:data-source-pool-1
```

データビューのもう 1 つのプロパティーは、406 ページの「デフォルトデータビュー」のデフォルトデータビューと同じです。

7 必要に応じて、変更を有効にするために Directory Proxy Server のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

異なるサブツリーが異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビュー

この節では、異なるサブツリーが異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビューを設定する方法を説明します。このような配備については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views to Provide a Single Point of Access When Different Subtrees Are Stored on Different Data Sources」を参照してください。

この節の例では、各サブツリーのデータビューが含まれます。データソースプールは、データ同等のデータソースのセットごとに設定されます。次の図は、配備の例を示しています。

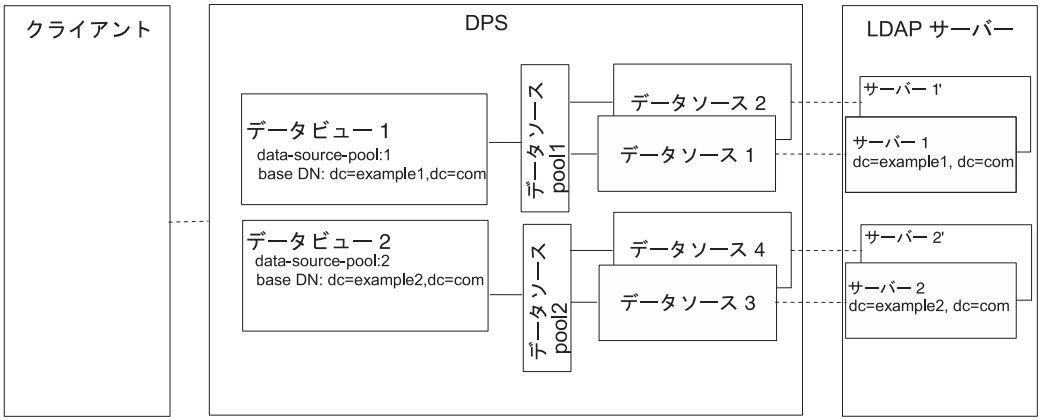


図 22–2　異なるサブツリーが異なるデータソースに保存されている場合に単一のアクセスポイントを提供する配備の例

▼ 異なるサブツリーが異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビューを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 381 ページの「LDAP データソースの作成と設定」で説明しているように、各 **LDAP** サーバーにデータソースを作成します。

2. 384 ページの「LDAP データソースプールの作成と設定」で説明しているように、**2** つのデータソースプールを作成します。

3. 385 ページの「LDAP データソースのデータソースプールへの接続」で説明しているように、`dc=example1,dc=com` を含むデータソースを `data-source-pool-1` に、`dc=example2,dc=com` を含むデータソースを `data-source-pool-2` に接続します。

4. (省略可能) 負荷分散を設定します。

   詳細は、387 ページの「負荷分散の設定」を参照してください。

5. `dc=example1,dc=com` で `data-source-pool-1` を参照するベース **DN** を持つデータビューを作成します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-1 \
  base-dn:dc=example1,dc=com ldap-data-source-pool:data-source-pool-1
```

6 dc=example2,dc=com で data-source-pool-2 を参照するベース DN を持つデータビューをもう 1 つ作成します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-2 \
  base-dn:dc=example2,dc=com ldap-data-source-pool:data-source-pool-2
```

データビューのもう 1 つのプロパティーは、406 ページの「デフォルトデータビュー」のデフォルトデータビューと同じです。

7 必要に応じて、変更を有効にするために Directory Proxy Server のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

サブツリーの異なる部分が異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビュー

この節では、サブツリーのさまざまな部分への単一のアクセスポイントを提供するデータビューを設定する方法を説明します。この例では、同じベース DN の 2 つのデータビューが含まれています。数値配布アルゴリズムは、エントリを様々なデータビューに分割するために使用されます。データソースプールは、データ同等のデータソースのセットごとに設定されます。次の図は、配備の例を示しています。

このような配備については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views to Route Requests When Different Parts of a Subtree Are Stored in Different Data Sources」を参照してください。

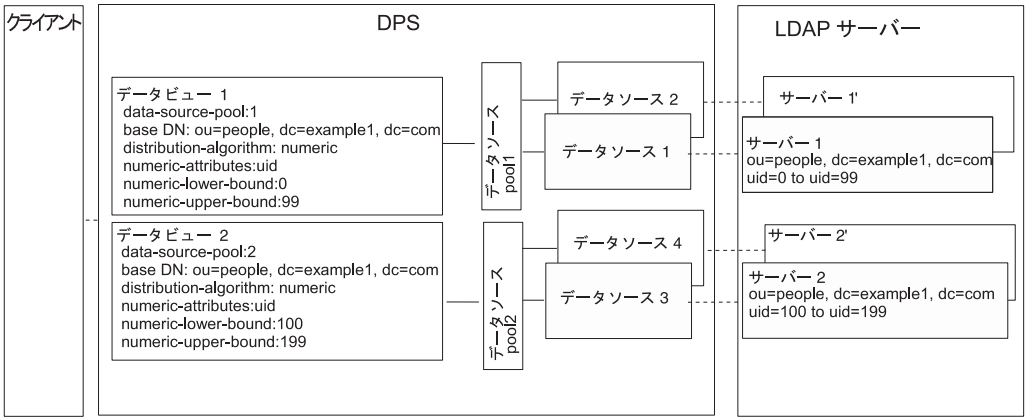


図 22–3 サブツリーの異なる部分が異なるデータソースに保存されている場合に単一のアクセスポイントを提供する配備の例

▼ サブツリーの異なる部分が異なるデータソースに保存されている場合に単一のアクセスポイントを提供するデータビューを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 381 ページの「LDAP データソースの作成と設定」で説明しているように、各 **LDAP** サーバーにデータソースを作成します。
2. 384 ページの「LDAP データソースプールの作成と設定」で説明しているように、**2** つのデータソースプールを作成します。
3. 385 ページの「LDAP データソースのデータソースプールへの接続」で説明しているように、サブツリーのある部分を含むデータソースを `data-source-pool-1` に、サブツリーのもう **1** つの部分を含むデータソースを `data-source-pool-2` に接続します。
4. (省略可能) 負荷分散を設定します。

   詳細は、387 ページの「負荷分散の設定」を参照してください。
5. `ou=people,dc=example,dc=com` で `uid` が `0` から `99` までのエントリが選択されるように、配布アルゴリズムを使ってデータビューを作成し、要求を `data-source-pool-1` に送信するようデータビューを設定します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-1 \
 ldap-data-source-pool:data-source-pool-1 base-dn:ou=people,dc=example,dc=com \
 distribution-algorithm    :numeric numeric-attrs:uid numeric-lower-bound    :0 \
 numeric-upper-bound    :99
```

6 ou=people,dc=example,dc=com で uid が 100 から 199 までのエントリが選択されるように、配布アルゴリズムを使ってもう 1 つのデータビューを作成し、要求を data-source-pool-2 に送信するようデータビューを設定します。

```
$ dpconf set-ldap-data-view-prop -h host1 -p 1389 dataview-2 \
 ldap-data-source-pool:data-source-pool-2 base-dn:ou=people,dc=example,dc=com \
 distribution-algorithm:numeric numeric-attrs:uid numeric-lower-bound:100
 numeric-upper-bound    :199
```

データビューのもう 1 つのプロパティーは、406 ページの「デフォルトデータビュー」のデフォルトデータビューと同じです。

7 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

上位サブツリーと下位サブツリーが異なるデータソースに保存される場合に単一のアクセスポイントを提供するデータビュー

この節では、上位サブツリーと下位サブツリーが異なるデータソースに保存される場合に、単一のアクセスポイントを提供するデータビューを設定する方法を説明します。このような配備については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views to Route Requests When Superior and Subordinate Subtrees Are Stored in Different Data Sources」を参照してください。

ここでの例には、3 つのデータビューが含まれます。データビュー 1 のベース DN は、データビュー 2 のベース DN とデータビュー 3 のベース DN より上位です。つまり、データソースプール 2 とデータソースプール 3 にはデータソースプール 1 のサブツリーの下位であるサブツリーが含まれます。次の図は、配備の例を示しています。

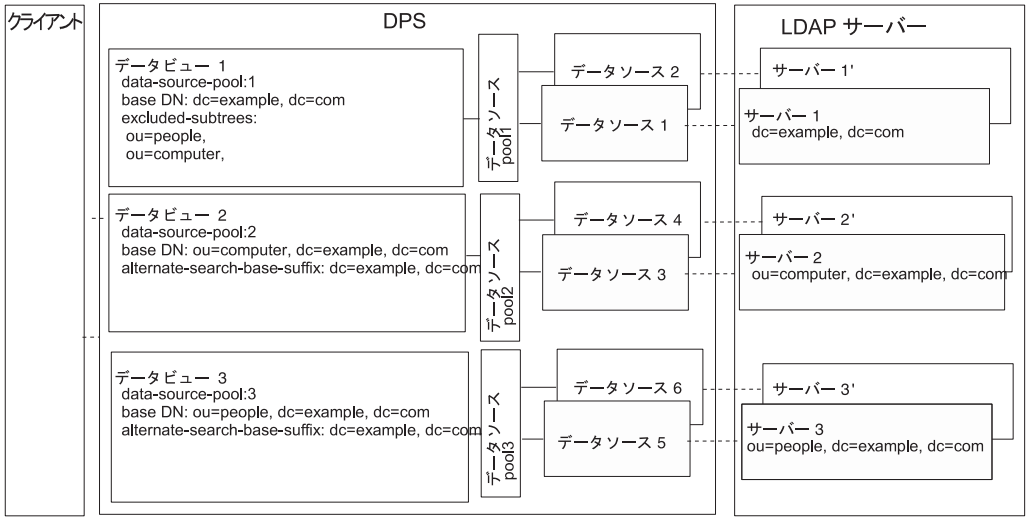


図 22–4　上位サブツリーと下位サブツリーが異なるデータソースに保存される場合に要求を経路指定する配備の例

下位エントリが別のデータビューのベース DN として設定されると、Directory Proxy Server は、サブツリーの下位エントリをデータビューから自動的に除外します。

▼ 上位サブツリーと下位サブツリーが異なるデータソースに保存される場合に単一のアクセスポイントを提供するデータビューを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 381 ページの「LDAP データソースの作成と設定」で説明しているように、各 **LDAP** サーバーにデータソースを作成します。
2. 384 ページの「LDAP データソースプールの作成と設定」で説明しているように、**3** つのデータソースプールを作成します。
3. 385 ページの「LDAP データソースのデータソースプールへの接続」の指示に従って、データソースをデータソースプールに接続します。
   - `dc=example,dc=com` を含むデータソースを `data-source-pool-1` に接続します。
   - `ou=computer,dc=example,dc=com` を含むデータソースを `data-source-pool-2` に接続します。

- ou=people,dc=example,dc=com を含むデータソースを data-source-pool-3 に接続します。

4 (省略可能) 負荷分散を設定します。
詳細は、387 ページの「負荷分散の設定」を参照してください。

5 ベース DN を dc=example,dc=com、データソースプールを data-source-pool-1 と指定したデータビューを作成します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-1 \
 data-source-pool-1 dc=example,dc=com
```

6 ベース DN を ou=computer,dc=example,dc=com、データソースプールを data-source-pool-2 と指定したデータビューを作成します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-2 \
 data-source-pool-2 ou=computer,dc=example,dc=com
```

7 ベース DN を ou=people,dc=example,dc=com、データソースプールを data-source-pool-3 と指定したデータビューを作成します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-3 \
 data-source-pool-3 ou=people,dc=example,dc=com
```

8 excluded-subtrees パラメータをチェックして、サブツリー ou=computer,dc=example,dc=com と ou=people,dc=example, dc=com が dataview-1 から除外されていることを確認します。

```
$ dpconf get-ldap-data-view-prop -h host1 -p 1389 dataview-1 excluded-subtrees
```

除外されたサブツリーの一覧が返されます。

9 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。
Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

階層と配布アルゴリズムを持つデータビュー

この節では、階層と配布アルゴリズムを組み合わせるデータビューを設定する方法を説明します。このような配備については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Data Views With Hierarchy and a Distribution Algorithm」を参照してください。

ここでの例には、4 つのデータビューが含まれます。データビュー 1 のベース DN は、その他のデータビューのベース DN より上位です。データビュー 3 とデータビュー 4 のベース DN は同じですが、数値配布アルゴリズムによって、エントリがデータビュー 3 か 4 に割り振られます。

下位エントリが別のデータビューのベース DN として設定されると、Directory Proxy Server は、サブツリーの下位エントリをデータビューから自動的に除外します。数値配布アルゴリズムは同じサブツリーのエントリが異なるデータビューに分割します。データソースプールは、データ同等のデータソースのセットごとに設定されます。

次の図は、配備の例を示しています。

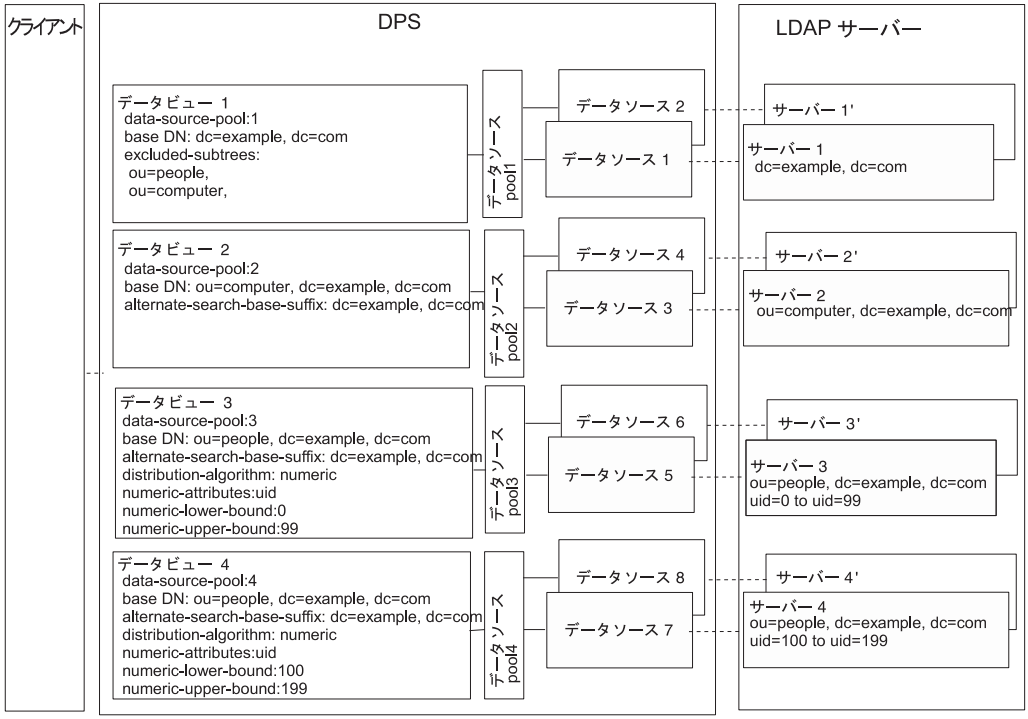


図 22–5　階層と配布アルゴリズムを持つデータビューの例

▼ 階層と配布アルゴリズムを持つデータビューを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. 381 ページの「LDAP データソースの作成と設定」で説明しているように、各 LDAP サーバーにデータソースを作成します。
2. 384 ページの「LDAP データソースプールの作成と設定」で説明しているように、4 つのデータソースプールを作成します。

3 385 ページの「LDAP データソースのデータソースプールへの接続」の指示に従って、データソースをデータソースプールに接続します。

- dc=example,dc=com を含むデータソースを data-source-pool-1 に接続します。
- ou=computer,dc=example,dc=com を含むデータソースを data-source-pool-2 に接続します。
- ou=people,dc=example,dc=com で uid が 0 と 99 の間のエントリを持つデータソースを data-source-pool-3 に接続します。
- ou=people,dc=example,dc=com で uid が 100 と 199 の間のエントリを持つデータソースを data-source-pool-4 に接続します。

4 (省略可能) 負荷分散を設定します。

詳細は、387 ページの「負荷分散の設定」を参照してください。

5 ベース DN が dc=example,dc=com で data-source-pool-1 を参照するデータビューを作成します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-1 \
 data-source-pool-1 dc=example,dc=com
```

6 ベース DN が ou=computer,dc=example,dc=com で data-source-pool-2 を参照するデータビューを作成します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-2 \
 data-source-pool-2 ou=computer,dc=example,dc=com
```

7 ベース DN が ou=people,dc=example,dc=com で data-source-pool-3 を参照するデータビューを作成します。uid が 0 から 99 までのエントリを選択するようにデータビューで配布アルゴリズムを設定します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-3 \
 data-source-pool-3 ou=people,dc=example,dc=com
$ dpconf set-ldap-data-view-prop dataview-3 distribution-algorithm:numeric \
 numeric-attrs:uid numeric-lower-bound:0 numeric-upper-bound:99
```

8 ベース DN が ou=people,dc=example,dc=com で data-source-pool-4 を参照するデータビューを作成し、uid が 100 から 199 までのエントリを選択するようデータビューで配布アルゴリズムを設定します。

```
$ dpconf create-ldap-data-view -h host1 -p 1389 dataview-4 \
 data-source-pool-4 ou=people,dc=example,dc=com
$ dpconf set-ldap-data-view-prop dataview-4 distribution-algorithm:numeric \
 numeric-attrs:uid numeric-lower-bound:100 numeric-upper-bound:199
```

9 excluded-subtrees パラメータをチェックして、サブツリー ou=computer,dc=example,dc=com と ou=people,dc=example, dc=com が dataview-1 から除外されていることを確認します。

```
$ dpconf get-ldap-data-view-prop -h host1 -p 1389 dataview-1 excluded-subtrees
```

除外されたサブツリーの一覧が返されます。

10 変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

◆ ◆ ◆ 第 23 章

Directory Proxy Server による仮想化

この章では、仮想データビューの作成方法を説明します。仮想データビューは、ソースデータを変換し、そのデータをクライアントアプリケーションに異なるビューで表示します。仮想データビューには、変換された LDAP データビュー、LDIF データビュー、結合データビュー、および JDBC ™データビューが含まれます。仮想データビューの機能の概要とユースケースの例の説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 18 章「Directory Proxy Server Virtualization」を参照してください。

Directory Service Control Center (DSCC) を使用してこの章の手順を実行することはできません。コマンド行を使用する必要があります。

この章の内容は次のとおりです。

LDIF データビューの作成と設定

LDIF データビューは、LDIF ファイルを LDAP データソースのように見せかける、単純な仮想データビューです。LDAP データビューとは異なり、LDIF データビューを設定する場合にデータソースやデータソースプールは作成しません。代わりに、データビューを作成する場合に LDIF ファイルを指定します。デフォルトで、LDIF データビューに書き込むことはできません。詳細については、435 ページの「仮想データビューでのアクセス制御の定義」を参照してください。

LDIF データビューの作成と設定については、次の手順を参照してください。

▼ LDIF データビューを作成する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **LDIF データビューを作成します。**

```
$ dpconf create-ldif-data-view -h host -p port view-name path-to-ldif-file suffix-dn
```

2 **(省略可能) LDIF データビューの一覧を表示します。**

```
$ dpconf list-ldif-data-views -h host -p port
```

デフォルトで設定されている LDIF データビューは、`virtual access controls` データビューのみです。このデータビューは、サーバーによって生成され、要求を仮想アクセス制御命令 (ACI) に経路指定できるようにします。

▼ LDIF データビューを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **LDIF データビューのプロパティーを表示します。**

```
$ dpconf get-ldif-data-view-prop -h host -p port view-name
```

LDIF データビューには、次のデフォルトプロパティーがあります。

```
alternate-search-base-dn                 :  ""
alternate-search-base-dn                 :  dc=com
attr-name-mappings                       :  none
base-dn                                  :  suffixDN
bind-pwd-attr                            :  userPassword
contains-shared-entries                  :  -
db-pwd-encryption                        :  clear-text
description                              :  -
distribution-algorithm                   :  -
dn-join-rule                             :  -
dn-mapping-attrs                         :  none
dn-mapping-source-base-dn                :  none
excluded-subtrees                        :  -
filter-join-rule                         :  -
is-enabled                               :  true
is-read-only                             :  false
is-routable                              :  true
ldif-data-source                         :  /path/to/filename.ldif
lexicographic-attrs                      :  all
lexicographic-lower-bound                :  none
```

```
lexicographic-upper-bound                   :  none
non-viewable-attr                           :  -
non-writable-attr                           :  -
numeric-attrs                               :  all
numeric-default-data-view                   :  false
numeric-lower-bound                         :  none
numeric-upper-bound                         :  none
pattern-matching-base-object-search-filter  :  all
pattern-matching-dn-regular-expression      :  all
pattern-matching-one-level-search-filter    :  all
pattern-matching-subtree-search-filter      :  all
process-bind                                :  -
replication-role                            :  master
viewable-attr                               :  all except non-viewable-attr
writable-attr                               :  all except non-writable-attr
```

2 手順 **1**で一覧表示されるプロパティーの **1** つまたは複数を変更します。

```
$ dpconf set-ldif-data-view-prop -h host -p port view-name property:value \
 [property:value ... ]
```

たとえば、データビューのソース LDIF ファイルを変更するには、 `ldif-data-source` プロパティーを設定します。

```
$ dpconf set-ldif-data-view-prop -h host1 -p 1389 -D cn="Proxy Manager" \
myLDIFDataView ldif-data-source:/local/files/example.ldif
```

仮想データ変換の設定

仮想データ変換は、既存のデータビュー上で定義され、物理データビューから仮想データビューを作成します。機能方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Virtual Data Transformations」を参照してください。

仮想データ変換は次のどの種類のデータビューにも追加できます。LDAP データビュー、LDIF データビュー、結合データビュー、または JDBC データビュー。

▼ 仮想変換を追加する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 変換をデータビューに追加します。

```
$ dpconf add-virtual-transformation -h host -p port view-name \
 transformation-model transformation-action attribute-name [parameters...]
```

transformation-model と *transformation-action* によっては、*parameters* は必須である場合があります。変換のモデル、変換アクション、変換パラメータについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Virtual Data Transformations」を参照してください。

2 (省略可能) データビューで定義された仮想変換の一覧を表示します。

```
$ dpconf list-virtual-transformations -h host -p port view-name
```

結合データビューの作成と設定

結合データビューは複数のデータビューを 1 つにまとめたものです。結合データビューの機能方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Join Data Views」を参照してください。

結合データビューの作成と設定の方法については、次の手順を参照してください。

▼ 結合データビューを作成する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 結合ビューを形成するためにまとめる一次データビューと二次データビューを特定します。

結合ビューを作成する前に、一次データビューと二次データビューを準備しておきます。一次ビューと二次ビューは、LDAP データビュー、LDIF データビュー、JDBC データビュー、またはその他の結合データビューを含むどんな種類のデータビューでも構いません。特定のプロパティーを二次ビュー上で設定し、結合ビューのソースとして機能できるようにします。詳細については、427 ページの「結合ビューの二次ビューを設定する」を参照してください。

2 結合データビューを作成します。

```
$ dpconf create-join-data-view -h host -p port view-name primary-view secondary-view \
 suffix-dn
```

3 (省略可能) データビューが正常に作成されたことを確認するために、結合ビューの一覧を表示します。

```
$ dpconf list-join-data-views -h host -p port
```

▼ 結合データビューを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 結合データビューのプロパティーを表示します。

```
$ dpconf get-join-data-view-prop -h host -p port view-name
```

結合データビューのデフォルトプロパティーは次のとおりです。

```
alternate-search-base-dn                    :  ""
alternate-search-base-dn                    :  dc=com
attr-name-mappings                          :  none
base-dn                                     :  suffixDN
contains-shared-entries                     :  -
description                                 :  -
distribution-algorithm                      :  -
dn-join-rule                                :  -
dn-mapping-attrs                            :  none
dn-mapping-source-base-dn                   :  none
excluded-subtrees                           :  -
filter-join-rule                            :  -
is-enabled                                  :  true
is-read-only                                :  false
is-routable                                 :  true
join-rule-control-enabled                   :  false
lexicographic-attrs                         :  all
lexicographic-lower-bound                   :  none
lexicographic-upper-bound                   :  none
non-viewable-attr                           :  -
non-writable-attr                           :  -
numeric-attrs                               :  all
numeric-default-data-view                   :  false
numeric-lower-bound                         :  none
numeric-upper-bound                         :  none
pattern-matching-base-object-search-filter  :  all
pattern-matching-dn-regular-expression      :  all
pattern-matching-one-level-search-filter    :  all
pattern-matching-subtree-search-filter      :  all
primary-view                                :  primary-view
process-bind                                :  -
replication-role                            :  master
secondary-view                              :  secondary-view
viewable-attr                               :  all except non-viewable-attr
writable-attr                               :  all except non-writable-attr
```

2 手順 1で一覧表示されるプロパティーの 1 つまたは複数を変更します。

```
$ dpconf set-join-data-view-prop -h host -p port view-name property:value \
 [property:value ... ]
```

たとえば、データソースの一次データビューを myLDAPDataView に変更するには、次のコマンドを使用します。

```
$ dpconf set-join-data-view-prop -h host1 -p 1389 -D cn="Proxy Manager" \
 myJoinDataView primary-view:myLDAPDataView
```

3 結合データビューを設定する場合は、一次データビューと二次データビューで viewable-attr および writable-attr プロパティーを設定します。

これらのプロパティーを設定すると、一次データビューと二次データビューで検索フィルタを適切に分割できます。これらのプロパティーを設定しなければ、二次データビューからの属性が検索フィルタに含まれている場合、検索結果に矛盾が生じる可能性があります。

4 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

▼ 複数の結合データビューが 1 つのデータビューを参照できるよう結合データビューを設定する

結合データビューで結合ルール設定情報を設定することにより、そのデータビューが複数の結合データビューから参照されるようにします。これを行うには、次の手順を実行します。

1 結合データビューで join-rule-control-enabled を true に設定します。

```
$ dpconf set-join-data-view-prop view-name join-rule-control-enabled:true
```

join-rule-control-enabled を true に設定すると、その結合データビューに格納された結合ルール設定情報がサーバーで使用されます。ある結合データビューに対して、結合ルール設定情報が二次データビューに格納されている場合、この情報はサーバーで使用されません。この情報をサーバーに使用させるには、結合データビューレベルで設定情報を手動で追加してください。

2 二次ビューが一次ビューに関連付けられる方法を決定する結合ルールを定義します。

次のどちらかの結合ルールを使用できます。

- DN 結合ルール

```
$ dpconf set-join-data-view-prop view-name \
dn-join-rule:uid=\${primary-view-name.uid},ou=People,dc=example
```

- フィルタ結合ルール

```
$ dpconf set-join-data-view-prop view-name \
filter-join-rule:uid=\${primary-view-name.uid}
```

▼ 結合ビューの二次ビューを設定する

特定のプロパティーを二次データビュー上で設定し、結合ビューのソースとして機能できるようにします。二次ビューはどんな種類のデータビューでも構わないため、使用するコマンドは、データビューの種類によって異なります。次のサンプルコマンドは、二次ビューが LDAP データビューであることを前提としています。ここで説明するプロパティーの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Additional Secondary Data View Properties」を参照してください。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 二次ビューが一次ビューに関連付けられる方法を決定する結合ルールを定義します。

次のどちらかの結合ルールを使用できます。

- DN 結合ルール

```
$ dpconf set-ldap-data-view-prop -h host -p port secondary-view-name \
dn-join-rule:uid=\${primary-view-name.uid},ou=People,dc=example
```

- フィルタ結合ルール

```
$ dpconf set-ldap-data-view-prop -h host -p port secondary-view-name \
filter-join-rule:uid=\${primary-view-name.uid}
```

`dn-join-rule` および `filter-join-rule` プロパティーの設定がサーバーで使用されるのは、結合データビューで `join-rule-control-enabled` プロパティーが `false` に設定されている場合だけです。結合データビューで `join-rule-control-enabled` プロパティーが `true` に設定されていると、二次ビューで設定された情報は無視されます。

2 結合データビューでフィルタ結合ルールが設定されている場合は、二次データビューで仮想変換ルールを設定して、その結合データビューでエントリを追加できるようにする必要があります。

```
dpconf add-virtual-transformation secondary-view-name \
write add-attr-value dn uid=\${uid}
```

注 – このルールを設定しなければ、結合データビューにエントリを追加できません。

3 (省略可能) 二次ビューでバインドを許可するかどうかを指定します。

デフォルトでは、すべてのデータビューでバインドが可能です。二次データビューのバインドを禁止する場合は、次のコマンドを実行します。

```
$ dpconf set-ldap-data-view-prop -h host -p port secondary-view-name process-bind:false
```

このプロパティーの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Handling of Binds」を参照してください。

4 (省略可能) 二次データビューに共有エントリが含まれるかどうかを指定します。

```
$ dpconf set-ldap-data-view-prop -h host -p port secondary-view-name \
contains-shared-entries:true
```

このプロパティーの詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Handling of Shared Entries」を参照してください。

JDBC データビューの作成と設定

JDBC データビューを使用すると、LDAP クライアントアプリケーションがリレーショナルデータベースにアクセスできるようになります。JDBC データビューの機能方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「JDBC Data Views」を参照してください。

JDBC データビューの作成と設定の方法については、次の手順を参照してください。

▼ JDBC データビューを作成する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 リレーショナルデータベース用の **JDBC** データソースを作成します。

```
$ dpconf create-jdbc-data-source -h host -p port -b db-name -B db-url -J driver-url \
[-J driver-url]... -S driver-class source-name
```

現在、各 JDBC データビューでサポートされる JDBC データソースは 1 つだけです。つまり、JDBC データソースで負荷分散することはできません。複数の JDBC データソースにアクセスするには、各データソース用にデータビューを作成し、それらをすべて結合データビューに結合します。

JDBC データソースを作成する場合は、次のプロパティーを設定します。

`db-name` リレーショナルデータベースの名前。たとえば、`payrolldb` などです。

`db-url` データベースへの URL。形式は、`jdbc:` *vendor*`:` *driver*`://` *dbhost*`:` *dbport* です。

`db-url` には、データベース名が含まれていないため、JDBC データベース URL としては完全ではありません。(データベース名は、`db-name` プロパティーで指定されます。)

`db-url` の末尾には、MySQL、DB2、および Derby データベースの場合は `/`、Oracle データベースの場合は `:` を付けてください。

`driver-class` JDBC ドライバクラス。たとえば、`org.hsqldb.jdbcDriver` などです。

`driver-url` JDBC ドライバへのパス。たとえば、`file:///` *path/to/hsqldb*`/lib/hsqldb.jar` などです。

`driver-url` プロパティーは複数値プロパティーです。そのため、`driver-url` で JDBC ドライバ用の複数の JAR ファイルを設定することにより、各種のプラットフォームで JDBC ソースに接続できます。

2 **JDBC** データソースプールを作成します。

```
$ dpconf create-jdbc-data-source-pool -h host -p port pool-name
```

3 **JDBC** データソースを **JDBC** データソースプールに接続します。

```
$ dpconf attach-jdbc-data-source -h host -p port pool-name source-name
```

4 **JDBC** データビューを作成します。

```
$ dpconf create-jdbc-data-view -h host -p port view-name pool-name suffix-DN
```

5 (省略可能) データビューが正常に作成されたことを確認するために、**JDBC** データビューの一覧を表示します。

```
$ dpconf list-jdbc-data-views -h host -p port
```

▼ JDBC データビューを設定する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **JDBC データビューのプロパティーを表示します。**

```
$ dpconf get-jdbc-data-view-prop -h host -p port view-name
```

JDBC データビューのデフォルトプロパティーは次のとおりです。

```
alternate-search-base-dn                    :  -
attr-name-mappings                          :  none
base-dn                                     :  o=sql1
contains-shared-entries                     :  -
description                                 :  -
distribution-algorithm                      :  -
dn-join-rule                                :  -
dn-mapping-attrs                            :  none
dn-mapping-source-base-dn                   :  none
excluded-subtrees                           :  -
filter-join-rule                            :  -
is-enabled                                  :  true
is-read-only                                :  false
is-routable                                 :  true
jdbc-data-source-pool                       :  pool-name
lexicographic-attrs                         :  all
lexicographic-lower-bound                   :  none
lexicographic-upper-bound                   :  none
non-viewable-attr                           :  -
non-writable-attr                           :  -
numeric-attrs                               :  all
numeric-default-data-view                   :  false
numeric-lower-bound                         :  none
numeric-upper-bound                         :  none
pattern-matching-base-object-search-filter  :  all
pattern-matching-dn-regular-expression      :  all
pattern-matching-one-level-search-filter    :  all
pattern-matching-subtree-search-filter      :  all
process-bind                                :  -
replication-role                            :  master
viewable-attr                               :  all except non-viewable-attr
writable-attr                               :  all except non-writable-attr
```

2 手順 1で一覧表示されるプロパティーの 1 つまたは複数を変更します。

```
$ dpconf set-jdbc-data-view-prop -h host -p port view-name property:value \
 [property:value ... ]
```

▼ JDBC テーブル、属性、オブジェクトクラスを設定する

JDBC データビューを設定する場合、次のオブジェクトも設定する必要があります。

- **JDBC** オブジェクトクラス。1 つまたは複数の JDBC テーブルを LDAP オブジェクトクラスにマップします。
- **JDBC** テーブル。各リレーショナルデータベーステーブルに対して定義します。
- **JDBC** 属性。JDBC テーブル内の指定された列の LDAP 属性を定義します。

1 リレーショナルデータベースの各テーブルの **JDBC** テーブルを作成します。

```
% dpconf create-jdbc-table jdbc-table-name db-table
```

db-table の名前は大文字と小文字を区別します。リレーショナルデータベースで使用したのと同じ文字 (大文字または小文字) を使用してください。異なる文字を使用すると、そのテーブルをターゲットとする操作は失敗する場合があります。

2 各リレーショナルデータベーステーブル内で各列の **JDBC** 属性を作成します。

```
% dpconf add-jdbc-attr table-name attr-name sql-column
```

JDBC 属性を作成すると、テーブル列が LDAP 属性にマップされます。

3 **(省略可能) リレーショナルデータベース内の列が大文字と小文字を区別する場合、JDBC 属性の LDAP 構文を変更します。**

```
% dpconf set-jdbc-attr-prop table-name attr-name ldap-syntax:ces
```

`ldap-syntax` の値は、デフォルトで `cis` です。これは、`jdbc-attr` が大文字と小文字を区別しないことを意味します。リレーショナルデータベースが大文字と小文字を区別する場合、値を `ces` に変更します。

Oracle や DB2 など、特定のリレーショナルデータベースはデフォルトで大文字と小文字を区別します。LDAP はデフォルトで大文字と小文字を区別しません。Directory Proxy Server はリレーショナルデータベーステーブルの列が大文字と小文字を区別することを検出すると、フィルタ内の対応する属性の `ldapsearch` クエリーが `UPPER` 関数を使用して SQL クエリーに変換されます。

たとえば、クエリー `ldapsearch -b "dc=mysuffix" "(attr=abc)"` は次の SQL クエリーに変換されます。

```
SELECT * FROM mytable WHERE (UPPER(attr)='ABC')
```

デフォルトで、この種類のクエリーはインデックスが生成されません。このため、このようなクエリーは、パフォーマンスに大きな影響を与える可能性があります。

次の 2 つの方法でパフォーマンスへの影響を緩和できます。

- jdbc-attr の `ldap-syntax` プロパティーを `cis` に設定する。
- LDAP フィルタで使用されている可能性のある各 `jdbc-attr` の `UPPER` 関数でインデックスを作成する。

注 – リレーショナルデータベースが大文字と小文字を区別しない場合は、`ldap-syntax` をデフォルト値の `cis` に設定します。`ldap-syntax:ces` は、大文字と小文字を区別しないデータベースではサポートされません。

---

4 **LDAP リレーショナルデータベーステーブルの JDBC オブジェクトクラスを作成します。**

```
% dpconf create-jdbc-object-class view-name objectclass primary-table \
  [secondary-table... ] DN-pattern
```

JDBC オブジェクトクラスを作成すると、原則的にこれらのテーブルが関連付けられる LDAP オブジェクトクラスが指定されます。JDBC オブジェクトクラスは、一次テーブルと二次テーブルが存在する場合、これらも指定します。

JDBC オブジェクトクラスを作成する場合、DN パターンを指定します。DN パターンは、エントリの DN の構築方法を示します。

5 **二次テーブルが存在する場合、一次テーブルと二次テーブル間の結合ルールを定義します。**

```
% dpconf set-jdbc-table-prop secondary-table-name filter-join-rule:join-rule
```

結合ルールは二次テーブルで定義され、そのテーブルからのデータが一次テーブルのデータにリンクされる方法を決定します。オブジェクトクラスの一次テーブルと二次テーブル間の関係の定義方法は重要です。詳細については、432 ページの「JDBC テーブル間の関係の定義」を参照してください。

6 **JDBC オブジェクトクラスのスーパークラスを指定します。**

```
% dpconf set-jdbc-object-class-prop view-name objectclass super-class:value
```

スーパークラスは、JDBC オブジェクトクラスが継承した LDAP オブジェクトクラスを示します。

JDBC テーブル間の関係の定義

もっとも単純な場合、JDBC オブジェクトクラスにはテーブルが 1 つ (一次) しか含まれません。二次テーブルはなく、このため、テーブル間の関係を定義する必要はありません。

オブジェクトクラスに複数のテーブルが含まれる場合、これらのテーブル間の関係を明確に定義します。テーブル間の関係は、常に二次テーブル上で定義されます。二次テーブルの次のプロパティーによって、これらの関係を定義できます。

- `is-single-row-table` は、テーブルで LDAP エントリに一致する行が 1 つだけであると指定します。
- `contains-shared-entries` は、二次テーブルの行が一次テーブルの複数の行に使用されることを指定します。
- `filter-join-rule` は、エントリが一次テーブルに基づいて二次テーブルから取得される方法を示します。

次の例は、最初の 2 つのプロパティーの値に基づいて、フィルタ結合ルールがどのように定義されるかを示しています。以降の例では、オブジェクトクラスに一次テーブルと二次テーブルが 1 つずつあることを前提としています。

例 **23–1** `is-single-row-table:true` と `contains-shared-entries:true`

それぞれプロパティーのデフォルト値が指定されています。この場合、一次テーブルと二次テーブルの関係は、*n->1* です。つまり、一次テーブルの *n* 行は二次テーブルの共有行の 1 つを参照します。

リレーショナルデータベースで、外部キー (FK) が一次テーブルで定義され、二次テーブルの列をポイントします。

たとえば、複数の従業員が同じマネージャーを共有できる組織の場合を考えてみます。2 つのリレーショナルデータベーステーブルは、次の構造で定義されます。

```
primary table : EMPLOYEE [ID, NAME, FK_MANAGER_ID]
secondary table : MANAGER  [ID, NAME]
```

次のオブジェクトクラスと属性が定義されます。

```
object-class : employee
attr : name (from primary EMPLOYEE.NAME)
attr : manager (from secondary MANAGER.NAME)
```

次のフィルタ結合ルールが、二次テーブルで定義されます。

```
"${ID}=${EMPLOYEE.FK_MANAGER_ID}"
```

この設定の場合、LDAP 操作で次の動作が行われます。

- 従業員エントリの追加。従業員エントリのマネージャーがテーブルに存在しない場合、新しい行が作成されます。マネージャーが存在する場合は、既存の行が使用されます。

例 23–1 is-single-row-table:true と contains-shared-entries:true (続き)

- エントリ内の「**manager**」属性の値の置き換え。MANAGER.NAME 行の値が変更されます。
- 従業員エントリの削除。マネージャーエントリが共有されているため、二次テーブルの行は削除されません。
- エントリから「**manager**」 属性の削除。二次テーブルの行が削除され、外部キー (EMPLOYEE.FK_MANAGER_ID) が NULL に設定されます。

例 23–2 is-single-row-table:true と contains-shared-entries:false

この場合、一次テーブルと二次テーブルの関係は、*1->1* または *1<-1* です。つまり、一次テーブルの 1 行が二次テーブルの 1 行で参照されます。

リレーショナルデータベースで、外部キー (FK) が一次テーブルまたは二次テーブルで定義されることがあります。

たとえば、従業員の UID が 1 つのテーブルに保存され、従業員の姓が二次テーブルに保存されている組織の場合を考えてみます。2 つのリレーショナルデータベーステーブルは、次の構造で定義されます。

```
primary table : UID [ID, VALUE, FK_SN_ID]
secondary table : SN [ID, VALUE]
```

次のオブジェクトクラスと属性が定義されます。

```
object-class : employee
attr : uid (from primary UID.VALUE)
attr : sn (from secondary ID.VALUE)
```

次のフィルタ結合ルールが、二次テーブルで定義されます。

```
"${ID}=${UID.FK_SN_ID}"
```

別の方法として、二次テーブルに保存されている外部キー FK_UID_ID を UID.ID にポイントさせることでも同等の設定が可能です。

例 23–3 is-single-row-table:false と contains-shared-entries:false

この場合、一次テーブルと二次テーブルの関係は、*1->n* です。つまり、一次テーブルの 1 行が二次テーブルの *n* 行で参照されます。この例は、複数値属性の場合を示しています。複数値属性は、属性値ごとに 1 行で表され、それぞれ二次テーブル内の行のセットと対応します。

例 23–3 is-single-row-table:false と contains-shared-entries:false (続き)

リレーショナルデータベースで、外部キーが二次テーブルで定義され、一次テーブルの列をポイントします。

たとえば、従業員が複数の電話番号を持っている可能性のある組織の場合を考えてみます。2 つのリレーショナルデータベーステーブルは、次の構造で定義されます。

```
primary table : EMPLOYEE [ID, NAME]
secondary table : PHONE [ID, VALUE, USER_ID]
```

次のオブジェクトクラスと属性が定義されます。

```
object-class : employee
attr : cn (from primary EMPLOYEE.NAME)
attr : telephoneNumber (from secondary PHONE.VALUE)
```

次のフィルタ結合ルールが、二次テーブルで定義されます。

```
"${USER_ID}=${EMPLOYEE.ID}"
```

例 23–4 is-single-row-table:false と contains-shared-entries:true

これは、現在 Directory Proxy Server でサポートされていません。

仮想データビューでのアクセス制御の定義

仮想データビューの ACI は、LDAP ディレクトリまたは LDIF ファイルに保存できます。仮想 ACI の機能方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Access Control On Virtual Data Views」を参照してください。

Directory Proxy Server インスタンスを作成すると、仮想アクセス制御の次のデフォルト設定が定義されます。

- ACI がデフォルトで保存される LDIF ファイル (*instance-path*/config/access_controls.ldif)
- virtual access controls という名前の LDIF データビュー

  このデータビューは、Directory Proxy Server が LDIF ファイルに保存された ACI にアクセスできるようにします。

▼ 新しい ACI ストレージリポジトリを定義する

前述のデフォルト ACI 設定を使用しない場合は、別のストレージリポジトリを定義できます。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 仮想 ACI が保存されるリポジトリのデータビューを作成します。

- ACI が LDAP ディレクトリに保存される場合は、399 ページの「LDAP データビューの作成と設定」の説明に従って、LDAP データソースと LDAP データビューを作成します。
- ACI が LDIF ファイルに保存される場合は、421 ページの「LDIF データビューの作成と設定」の説明に従って、LDIF データビューを作成します。

2 前の手順で作成したデータビューに ACI データビューとして名前を指定します。

```
$ dpconf set-virtual-aci-prop -h host -p port aci-data-view:data-view-name
```

3 ACI リポジトリが LDAP ディレクトリの場合は、ACI データビューへのアクセスに必要な資格を指定します。

```
$ dpconf set-virtual-aci-prop -h host -p port aci-manager-bind-dn:bind-dn
$ dpconf set-virtual-aci-prop -h host -p port aci-manager-bind-pwd-file:filename
```

▼ 仮想アクセス制御を設定する

使用する ACI リポジトリに関係なく、仮想アクセス制御を設定する必要があります。

注 – ACI のプールを作成し、ACI データビューによって直接 ACI を管理できるのはプロキシマネージャーだけです。ACI リポジトリが LDAP ディレクトリの場合、`aciSource` オブジェクトクラスと `dpsaci` 属性が含まれるようにそのディレクトリのスキーマを変更します。スキーマのカスタマイズの詳細については、309 ページの「Directory Server スキーマの拡張」を参照してください。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 **ACI** リポジトリに **ACI** のプールを作成し、グローバル **ACI** を設定します。

グローバル ACI の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Global ACIs」を参照してください。グローバル ACI を設定するには、ACI データビューのビューベースの下に `aciSource` エントリを追加します。次に例を示します。

```
% ldapmodify -p port -D "cn=proxy manager" -w -
dn: cn=data-source-name,cn=virtual access controls
changetype: add
objectclass: aciSource
dpsaci: (targetattr="*") (target = "ldap:///ou=people,o=virtual") (version 3.0; \
 acl "perm1"; allow(all) groupdn="ldap:///cn=virtualGroup1,o=groups,o=virtual";)
cn: data-source-name
```

2 この **ACI** のプールを使用するよう **1** つまたは複数の接続ハンドラを設定します。

```
% dpconf set-connection-handler-prop -h host -p port connection-handler \
aci-source:data-source-name
```

3 必要な **ACI** をデータに追加します。

これを行うには、ACI を含む仮想エントリを作成します。次に例を示します。

```
% ldapmodify -p port -D "cn=virtual application,ou=application users,dc=com" -w -
dn: ou=people,o=virtual
changetype: modify
add: dpsaci
dpsaci: (targetattr="*")(version 3.0; acl "perm1"; allow(all) userdn ="ldap:///self";)
dpsaci: (targetattr="*")(version 3.0; acl "perm1"; allow(search, read, compare) \
 userdn ="ldap:///anyone";)
```

注 – 適切なアクセス権限をもつユーザーなら誰でも、データビューを使用して仮想 ACI を追加、取得できます。

仮想データビューでのスキーマチェックの定義

一般に、LDAP データビューの場合、スキーマチェックはバックエンドディレクトリによって、バックエンドディレクトリのスキーマを使用して実行されます。Directory Proxy Server でスキーマチェックを実行する場合は、次の手順に従います。

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

要求を正規化するには、特に DN の場合、サーバーの `use-external-schema` プロパティーを次のように設定します。

▼ スキーマチェックを定義する

1. サーバーインスタンスが外部スキーマを使用するように設定します。

   ```
   $ dpconf set-server-prop -h host -p port use-external-schema:true
   ```

2. 接続ハンドラでスキーマチェックを有効にします。

   ```
   $ dpconf set-connection-handler-prop -h host -p port connection-handler \
    schema-check-enabled:true
   ```

3. cn=schema を公開するデータビューを作成します。

   外部スキーマが LDAP ディレクトリで定義される場合、399 ページの「LDAP データビューの作成と設定」での説明に従って、ビューベースを cn=schema にして LDAP データビューを作成します。

   外部スキーマが LDIF ファイルで定義される場合、421 ページの「LDIF データビューの作成と設定」での説明に従って、ビューベースを cn=schema にして LDIF データビューを作成します。

4. 接続ハンドラによって公開されるデータビューの一覧にこのデータビューを追加します。

   デフォルトで、データビューはすべて接続ハンドラによって公開されます。接続ハンドラによって公開されたデータビューのカスタムリストを定義している場合、このデータビューをリストに追加します。

   ```
   $ dpconf set-connection-handler-prop -h host -p port connection-handler \
    data-view-routing-custom-list+:data-view-name
   ```

仮想設定の例

次の節では、2 つの設定例について説明します。これらの設定は、仮想ディレクトリの主な機能と、これらの機能の設定方法を示しています。

LDAP ディレクトリと MySQL データベースの結合

ここでの手順は、LDAP ディレクトリと MySQL データベースを結合する仮想設定の例について説明しています。LDAP ディレクトリは、一次データソースとして、ユーザー情報のほとんどが含まれています。mySQL データベースには、ユーザーについての追加情報が含まれています。結果として得られる設定を次の図に示します。

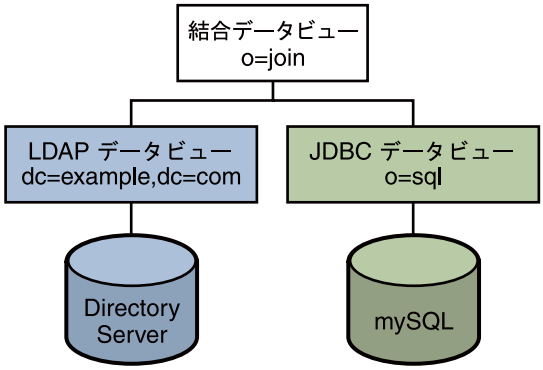


図 23–1 仮想設定の例

install-path /ds6/ldif/Example.ldif のサンプルデータを使用して、この例を複製したり、サンプルデータを独自のデータに置き換えられます。

この設定は、3 つの部分に分割できます。

- LDAP データビューの設定とテスト
- JDBC データビューの設定とテスト
- 結合データビューの設定とテスト

わかりやすいように、ここでのコマンドはすべて Directory Proxy Server が /local/dps のローカルホストで実行されていることを前提としています。コマンドは、次の環境変数が設定されていることも前提としています。

DIR_PROXY_PORT	1389
LDAP_ADMIN_PWF	pwd.txt、管理者パスワードを含むファイル
DIRSERV_PORT	4389
LDAP_ADMIN_USER	cn=Directory Manager

LDAP データビューの設定とテスト

▼ LDAP データビューを設定する

始める前に ここでの作業は次の情報を前提としています。

- Directory Server インスタンスが host1 上のポート 4389 で実行されている。
- Directory Server のデータがサフィックス dc=example,dc=com の下に保存されている。この例と同じ環境を構築するには、Directory Server インスタンスとサフィックス dc=example,dc=com を作成し、*install-path*/ds6/ldif/Example.ldif のサンプルデータをインポートしてください。

1. **Directory Server** インスタンスに対して、myds1 という名前の **LDAP** データソースを作成します。

```
% dpconf create-ldap-data-source myds1 host1:4389
```

2. データソースを有効にして、データソースへの書き込み操作を許可します。

```
% dpconf set-ldap-data-source-prop myds1 is-enabled:true is-read-only:false
```

3. myds1-pool という名前の **LDAP** データソースプールを作成します。

```
% dpconf create-ldap-data-source-pool myds1-pool
```

4. **LDAP** データソースを **LDAP** データソースプールに接続します。

```
% dpconf attach-ldap-data-source myds1-pool myds1
```

5. データソースがそのデータソースプールからのバインド、追加、検索、および変更操作の **100%** を受け取るように指定します。

```
% dpconf set-attached-ldap-data-source-prop myds1-pool myds1 add-weight:100 \
 bind-weight:100 modify-weight:100 search-weight:100
```

6. ベース **DN** が dc=example,dc=com で myds1–view という名前のデータソースプールの **LDAP** データビューを作成します。

```
% dpconf create-ldap-data-view myds1-view myds1-pool dc=example,dc=com
```

▼ LDAP データビューをテストする

1. dc=example,dc=com のユーザーとして、**LDAP** データソース内のエントリをすべて検索して、データビューから読み取りができることを確認します。

```
% ldapsearch -p 1389 -D "uid=kvaughan,ou=people,dc=example,dc=com" -w bribery \
 -b dc=example,dc=com "objectclass=*"
```

注 – dc=example,dc=com のユーザーの資格を使用します。cn=Directory Manager を使用する場合は、この DN を処理するようにデータビューを定義する必要があります。

---

2. dc=example,dc=com のユーザーとして、userPassword 属性を変更して、データビューに書き込みができることを確認します。

```
% ldapmodify -p 1389 -D "uid=kvaughan,ou=people,dc=example,dc=com" -w bribery
dn: uid=kvaughan,ou=people,dc=example,dc=com
changetype: modify
replace: userPassword
userPassword: myNewPassword
```

注 – Directory Server 内のデフォルト ACI はユーザーが自分自身のパスワードを変更することを許可しています。

JDBC データビューの設定とテスト

次の作業は、mySQL データベースがインストールされて実行中であり、データベースにデータが存在し、mySQL データベースが次のように設定されていることが前提となっています。

- データベース名 : `sample_sql`
- データベース URL : `host2.example.com:3306/`
- JDBC ドライバ URL : `file:/net/host2.example/local/mysql/lib/jdbc.jar`
- ドライバクラス : `com.mysql.jdbc.Driver`
- データベースユーザー : `root`
- データベースパスワードファイル : `mysqlpwd.txt`

次の表は、データベース内のテーブルと複合フィールドを説明しています。JDBC データビューを設定するためにこの情報が必要です。

mySQL テーブル	フィールド
EMPLOYEE	ID、 SURNAME、 PASSWORD、 TITLE、 COUNTRY_ID
COUNTRY	ID、 NAME
PHONE	USER_ID、 NUMBER

▼ JDBC データビューを設定する

1 **SQL** データベース用の **mysql1** という名前の **JDBC** データソースを作成します。

```
% dpconf create-jdbc-data-source -b sample_sql -B jdbc:mysql://host2.example.com:3306 \
 -J file:/net/host2.example/local/mysql/lib/jdbc.jar -S com.mysql.jdbc.Driver mysql1
```

2 **SQL** データベースのユーザー名とパスワードファイルを指定します。

```
% dpconf set-jdbc-data-source-prop mysql1 db-pwd-file:sqlpwd.txt db-user:root
```

3 プロキシサーバーを再起動します。

```
% dpadm restart /local/dps
```

4 データソースを有効にして、データソースへの書き込み操作を許可します。

```
% dpconf set-jdbc-data-source-prop mysql1 is-enabled:true is-read-only:false
```

5 mysql1–pool という名前の JDBC データソースプールを作成します。

```
% dpconf create-jdbc-data-source-pool mysql1-pool
```

6 JDBC データソースをデータソースプールに接続します。

```
% dpconf attach-jdbc-data-source mysql1-pool mysql1
```

7 ベース DN が o=sql で myjdbc1–view という名前のデータソースプールの JDBC データビューを作成します。

```
% dpconf create-jdbc-data-view mysql1-view mysql1-pool o=sql
```

8 MySQL データベースの各テーブルの JDBC テーブルを作成します。

```
% dpconf create-jdbc-table employee1 EMPLOYEE
% dpconf create-jdbc-table country1 COUNTRY
% dpconf create-jdbc-table phone1 PHONE
```

SQL データベースのテーブル名は大文字と小文字を区別します。SQL データベースの場合と同じ文字 (大文字または小文字) を使用していることを確認してください。

9 各テーブルの各列の JDBC 属性を作成します。

JDBC 属性を作成すると、MySQL 列が LDAP 属性にマップされます。

```
% dpconf add-jdbc-attr employee1 uid ID
% dpconf add-jdbc-attr employee1 sn SURNAME
% dpconf add-jdbc-attr employee1 userPassword PASSWORD
% dpconf add-jdbc-attr employee1 room ROOM
% dpconf add-jdbc-attr phone1 tel NUMBER
% dpconf add-jdbc-attr country1 country NAME
```

phone1 user_id 列と country1 id 列は MySQL データベースのコンテキストでのみ使用されるため、これらの列の JDBC 属性を必ずしも作成する必要はありません。これらには、対応する LDAP 属性がありません。

10 LDAP person オブジェクトクラスに対して JDBC オブジェクトクラスを作成します。

ここでは、employee1 テーブルを一次テーブル、country1 テーブルと phone1 テーブルを二次テーブルとします。JDBC オブジェクトクラスの作成にも DN が必要です。この例では、DN は uid 属性とデータビューのベース DN から構築されます。

```
% dpconf create-jdbc-object-class mysql1-view person employee1 country1 phone1 uid
```

11 一次テーブルと二次テーブル間の結合ルールを定義します。

結合ルールは二次テーブルで定義され、そのテーブルからのデータが一次テーブルのデータにリンクされる方法を決定します。

```
% dpconf set-jdbc-table-prop country1 filter-join-rule:'ID=${EMPLOYEE.COUNTRY_ID}'
% dpconf set-jdbc-table-prop phone1 filter-join-rule:'USER_ID=${EMPLOYEE.ID}'
```

12 **JDBC** オブジェクトクラスのスーパークラスを指定します。

スーパークラスは、JDBC オブジェクトクラスが属性を継承した LDAP オブジェクトクラスを示します。

```
% dpconf set-jdbc-object-class-prop mysql1-view person super-class:top
```

▼ 必要な **ACI** を作成する

JDBC データビューをテストする前に、ACI を設定してデータビューへの書き込みアクセスを有効にします。デフォルトで、LDAP データビュー以外への書き込みアクセスは拒否されます。この例では、ユーザーに自分のパスワードの変更を許可するグローバル ACI を 1 つ追加するだけで十分です。

1 プロキシマネージャーとして、**ACI** のプールを **JDBC** データソースに追加し、ユーザーに自分のエントリの変更を許可するグローバル **ACI** を追加します。

```
% ldapmodify -p 1389 -D "cn=proxy manager" -w password
dn: cn=mysql1,cn=virtual access controls
changetype: add
objectclass: acisource
dpsaci: (targetattr="*") (target = "ldap:///o=sql") \
 (version 3.0; acl "enable all access for all users "; allow(all) \
userdn="ldap:///uid=kvaughan,o=sql";)
cn: mysql1
```

2 `o=sql` ドメインへの接続を処理するために接続ハンドラを作成します。

```
% dpconf create-connection-handler mysql1-handler
```

3 接続ハンドラを有効にして、`o=sql` ドメイン内のユーザーからのバインドをすべて処理するように設定します。

```
% dpconf set-connection-handler-prop mysql1-handler is-enabled:true \
 bind-dn-filters:"uid=.*,o=sql"
```

4 前の手順で追加した **ACI** のプールを使用するように接続ハンドラを設定します。

```
% dpconf set-connection-handler-prop mysql1-handler aci-source:mysql1
```

▼ **JDBC** データビューをテストする

1 `o=sql` のユーザーとして **JDBC** データソースを検索し、データビューから読み取りができることを確認します。

```
% ldapsearch -p 1389 -D "uid=kvaughan,o=sql" -w mypwd -b o=sql "objectclass=*"
```

注 – `o=sql` のユーザーまたは匿名バインドの資格を使用します。

---

2 o=sql のユーザーとして、 userPassword 属性を変更して、データビューに書き込みができることを確認します。

```
% ldapmodify -p 1389 -D "uid=kvaughan,o=sql" -w mypwd
dn: uid=kvaughan,o=sql
changetype: modify
replace: userPassword
userPassword: myNewpwd
```

結合データビューの作成とテスト

▼ 結合データビューを作成する

1 myjoin1–view という名前の結合データビューを作成します。

LDAP データビューを一次データビュー、JDBC データビューを二次データビューに指定します。

```
% dpconf create-join-data-view myjoin1-view myds1-view mysql1-view o=join
```

2 二次データビューで結合ルールを定義します。

次の結合ルールは、二次データビューのエントリの uid 属性が一次データビューのエントリの uid 属性に一致することを指定します。

```
% dpconf set-jdbc-data-view-prop mysql1-view filter-join-rule:uid='${myds1-view.uid}'
```

3 結合データビューでフィルタ結合ルールが設定されている場合、二次データビューで仮想変換ルールを設定して、その結合データビューでエントリを追加できるようにする必要があります。

```
dpconf add-virtual-transformation secondary-view-name \
write add-attr-value dn uid=\${uid}
```

注 – このルールを設定しなければ、結合データビューにエントリを追加できません。

---

4 結合データビューを使用して、一次データビューに対して読み書きができる属性のセットを定義します。

```
% dpconf set-ldap-data-view-prop myds1-view viewable-attr:dn viewable-attr:cn \
 viewable-attr:sn viewable-attr:givenName viewable-attr:objectClass viewable-attr:ou \
 viewable-attr:l viewable-attr:uid viewable-attr:mail viewable-attr:telephoneNumber \
 viewable-attr:facsimileTelephoneNumber viewable-attr:roomNumber viewable-attr:userPassword
% dpconf set-ldap-data-view-prop myds1-view writable-attr:dn writable-attr:cn \
 writable-attr:sn writable-attr:givenName writable-attr:objectClass writable-attr:ou \
 writable-attr:l writable-attr:uid writable-attr:mail writable-attr:telephoneNumber \
 writable-attr:facsimileTelephoneNumber writable-attr:roomNumber writable-attr:userPassword
```

これらの定義は、結合ビューのコンテキストにのみ適用されます。LDAP データビューに直接アクセスした場合、デフォルトでは、すべての属性が読み書きできます。

5 結合データビューを使用して、二次データビューに対して読み書きができる属性のセットを定義します。

```
% dpconf set-jdbc-data-view-prop mysql1-view viewable-attr:dn viewable-attr:objectclass \
 viewable-attr:sn viewable-attr:room viewable-attr:userpassword viewable-attr:jobtitle \
 viewable-attr:country viewable-attr:tel
% dpconf set-jdbc-data-view-prop mysql1-view writable-attr:dn writable-attr:objectclass \
 writable-attr:sn writable-attr:room writable-attr:userpassword writable-attr:jobtitle \
 writable-attr:country writable-attr:tel
```

これらの定義は、結合ビューのコンテキストにのみ適用されます。JDBC データビューに直接アクセスした場合、デフォルトでは、すべての属性が読み書きできます。

▼ 必要な ACI を作成する

1 プロキシマネージャーとして、結合データビューへの匿名アクセスを許可するグローバル **ACI** を追加します。

```
% ldapmodify -p 1389 -D "cn=proxy manager" -w password
dn: cn=myjoin1,cn=virtual access controls
changetype: add
objectclass: acisource
dpsaci: (targetattr="*") (target = "ldap:///o=join") \
 (version 3.0; acl "anonymous_access"; allow(all) userdn="ldap:///anyone";)
cn: myjoin1
```

2 `o=join` ドメインへの接続を処理するために接続ハンドラを作成します。

```
% dpconf create-connection-handler myjoin1-handler
```

3 接続ハンドラを有効にして、`o=join` のユーザーからのバインドをすべて処理するように設定します。

```
% dpconf set-connection-handler-prop myjoin1-handler is-enabled:true \
 bind-dn-filters:"uid=.*,ou=people,o=join"
```

4 前の手順で追加した **ACI** のプールを使用するように接続ハンドラを設定します。

```
% dpconf set-connection-handler-prop myjoin1-handler aci-source:myjoin1
```

▼ 結合データビューをテストする

1 匿名ユーザーとして、結合データビューを検索します。

ここでは、Kirsten Vaughan のエントリを検索し、両方の結合ビューからのデータが取得されるかどうかを確認します。

```
% ldapsearch -p 1389 -b o=join "uid=kvaughan"
```

返されるエントリには、LDAP データビューと JDBC データビューの両方からの属性が含まれていることに注意してください。

2 `o=join` のユーザーとして、`userPassword` 属性を変更して、結合データビューに書き込みができることを確認します。

```
% ldapmodify -p 1389 -D "uid=kvaughan,ou=people,o=join" -w myNewPassword
dn: uid=kvaughan,ou=people,o=join
changetype: modify
replace: userPassword
userPassword: myPassword
```

複数の異種データソースの結合

この設定は、仮想ディレクトリの機能のいくつかが特定のディレクトリサービス要件を満たしている Example.com という組織で説明します。

データストレージシナリオ

Example.com は複数の異種データソースに組織データを保存しています。過去の経緯により、ユーザーデータは LDAP ディレクトリ、フラット LDIF ファイル、SQL データベースに分散されています。HR 部門は `o=example.com` のベース DN でユーザーデータを LDAP ディレクトリに保存しています。給与部門はデータを SQL データベースに保存しています。部門やビルの番号などの管理データは`dc=example,dc=com` のベース DN で管理部門の LDIF ファイルに保存されています。

さらに、Example.com は Company22 という名前の企業を買収しました。Company 22 もユーザーデータを `dc=company22,dc=com` のベース DN で LDAP ディレクトリに保存しています。

次の図は、Example.com のユーザーデータがどのように保存されているかをまとめたものです。

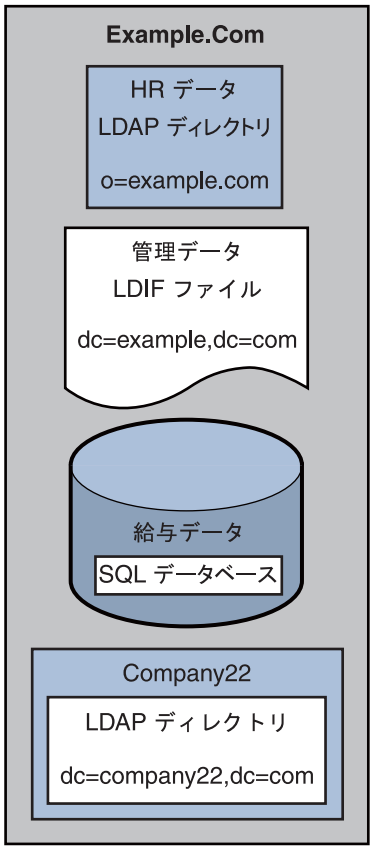


図 23–2 異種ソースのデータストレージ

クライアントアプリケーション要件

Example.com には、異種データソースに保存されたデータにアクセスする必要のある LDAP クライアントアプリケーションがいくつかあります。クライアントアプリケーションの要件はすべて同じではありません。異なるデータビューは必要です。場合によっては、クライアントはデータを集約する必要があります。さらに、Example.com の新しい従業員を以前からの従業員とともに管理できるように、一部のクライアントアプリケーションは Company22 のユーザーデータにアクセスする必要があります。

次の図は、Example.com のクライアントアプリケーション要件をまとめたものです。

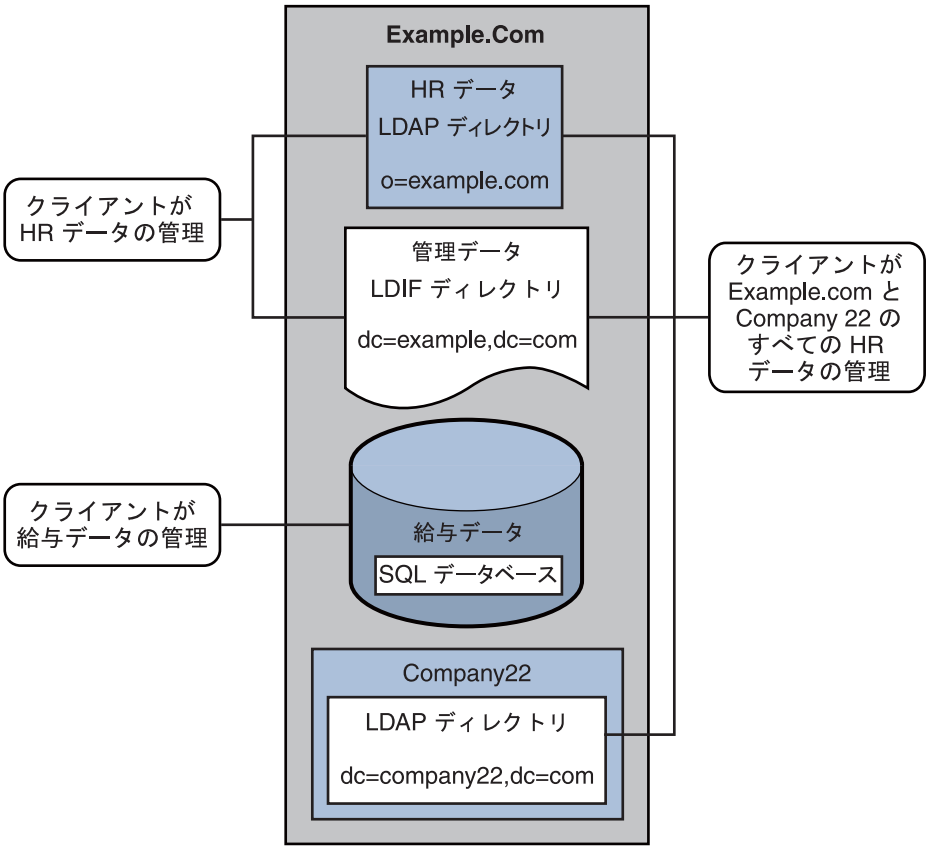


図 23–3 クライアントアプリケーション要件

次の節では、Directory Proxy Server データビューがこのサンプルシナリオで説明したクライアントアプリケーション要件を十分満たすことができる設定について見ていきます。データビューの機能方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 17 章「Directory Proxy Server Distribution」および『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 18 章「Directory Proxy Server Virtualization」を参照してください。

サンプルシナリオの設定は次のセクションで構成されています。

- 449 ページの「HR LDAP ディレクトリと管理 LDIF ファイルのデータの集約」
- 451 ページの「DN の名前を変更して Company 22 からのデータを Example.Com の DIT に追加」
- 453 ページの「Company 22 のデータの HR データへの追加」
- 454 ページの「LDAP クライアントによる SQL データベース内の給与データへのアクセスの有効化」
- 457 ページの「仮想アクセス制御の追加」

HR LDAP ディレクトリと管理 LDIF ファイルのデータの集約

HR 部門は従業員名、職務開始データ、職務レベルなどの情報を保存しています。管理部門は、建築基準法や会社の電話番号などの追加データを保存しています。HR データを処理するクライアントアプリケーションは、両方のソースからの複合データにアクセスする必要があります。各エントリ内の属性 `employeeNumber` は両方のデータソースに共通です。

次の図は、クライアントアプリケーションの要件を示しています。

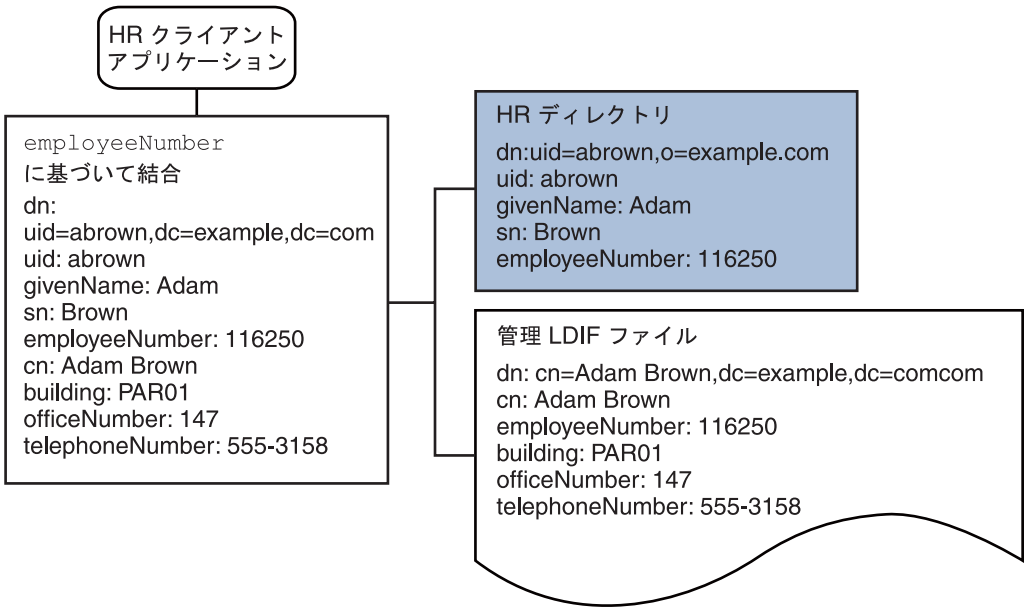


図 23–4 LDAP ディレクトリと LDIF ファイルのデータの集約

このアプリケーション要件を満たすために、給与ディレクトリと管理 LDIF ファイル用にデータビューが作成されます。これらの 2 つのデータビューは、その後、集約されたデータにアクセスできるよう結合されます。この共通属性によって、Directory Proxy Server は各ユーザーのデータを集約できます。

わかりやすいように、ここで使用するコマンドは次の情報を前提としています。

- Directory Proxy Server インスタンスはローカルホスト上のデフォルト LDAP ポート (389) で実行されている。
- Directory Proxy Server インスタンスは、`/local/myDPS` にある。
- プロキシマネージャーのパスワードを含むファイルへのパスは、変数 `LDAP_ADMIN_PWF` として設定されている。Directory Proxy Server 環境変数の設定の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「Environment Variables」を参照してください。

- 給与 LDAP ディレクトリは payrollHost という名前のホストのポート 2389 で実行されている。
- 管理データを保存している LDIF ファイルの名前は、 example.ldif である。

各コマンドの完全な構文を取得するために、オプションを使用せずにコマンドを実行します。次に例を示します。

```
$ dpconf create-ldap-data-view
Operands are missing
Usage: dpcfg create-ldap-data-view VIEW_NAME POOL_NAME SUFFIX_DN
```

▼ 給与ディレクトリの LDAP データビューの作成と有効化

1 給与ディレクトリの LDAP データソースを作成します。

```
$ dpconf create-ldap-data-source payroll-directory payrollHost:2389
```

2 給与ディレクトリの LDAP データソースプールを作成します。

```
$ dpconf create-ldap-data-source-pool payroll-pool
```

3 給与データソースをデータソースプールに接続します。

```
$ dpconf attach-ldap-data-source payroll-pool payroll-directory
```

4 接続済みデータソースのウェイトを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h payrollHost -p 2389 \
payroll-pool payroll-directory add-weight:2 \
bind-weight:2 compare-weight:2 delete-weight:2 \
modify-dn-weight:2 modify-weight:2 search-weight:2
```

5 給与ディレクトリの LDAP データビューを作成します。

```
$ dpconf create-ldap-data-view payroll-view payroll-pool o=example.com
```

6 クライアント要求がこのデータビューに経路指定できるように LDAP データビューを有効にします。

```
$ dpconf set-ldap-data-view-prop payroll-view is-enabled:true
```

7 変更を有効にするために、 Directory Proxy Server を再起動します。

```
$ dpadm restart /local/myDPS
```

▼ 管理データの LDIF データビューの作成と有効化

1 管理データの LDIF データビューを作成します。

```
$ dpconf create-ldif-data-view admin-view example.ldif dc=example,dc=com
```

2 管理データの **LDIF** データビューを有効にします。

```
$ dpconf set-ldif-data-view-prop admin-view is-enabled:true
```

3 管理ビューに給与ビューの複数のエントリで使用されるエントリが含まれるように指定します。

```
$ dpconf set-ldif-data-view-prop admin-view contains-shared-entries:true
```

このプロパティーが TRUE に設定されている場合、給与データビューのエントリを削除しても、管理データビューの共有エントリは削除されません。給与データビューにエントリを追加すると、それがすでに存在していなければ、二次データビューのエントリのみが追加されます。

4 変更を有効にするために、**Directory Proxy Server** を再起動します。

```
$ dpadm restart /local/myDPS
```

▼ 給与データビューと管理データビューの結合

1 データの集約方法を指定するフィルタ結合ルールを管理データビューで作成します。

次の結合ルールは、ユーザーエントリの `employeeNumber` 属性に基づいてデータが結合されるよう指定します。

```
$ dpconf set-ldif-data-view-prop admin-view \
filter-join-rule:'employeeNumber=\${payroll-view.employeeNumber}'
```

2 **2** つのデータビューを集約する結合データビューを作成します。

結合データビューに対して、組織はサフィックス DN `dc=example,dc=com` を使用します。

```
$ dpconf create-join-data-view example-join-view payroll-view admin-view \
dc=example,dc=com
```

DN の名前を変更して Company 22 からのデータを Example.Com の DIT に追加

Company 22 のユーザーデータは、DN `dc=company22,dc=com` で保存されています。Example.com はこのユーザーデータをほとんどの場合に区別しておきたいと考えていますが、あるクライアントアプリケーションは Company 22 の従業員を Example.com の残りの従業員とともに管理する必要があります。このクライアントアプリケーションは Company 22 のユーザーデータが Example.com のデータのように見えることを必要としています。

次の図は、クライアントアプリケーションの要件を示しています。

図 23–5　DN の名前の変更

このアプリケーション要件を満たすために、仮想 DN の `dc=example,dc=com` のデータビューが Company 22 ディレクトリに対して作成されます。

わかりやすいように、ここで使用するコマンドは次の情報を前提としています。

- Directory Proxy Server インスタンスはローカルホスト上のデフォルト LDAP ポート (389) で実行されている。
- Directory Proxy Server インスタンスは、`/local/myDPS` にある。
- プロキシマネージャーのパスワードを含むファイルへのパスは、変数 `LDAP_ADMIN_PWF` として設定されている。Directory Proxy Server 環境変数の設定の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「Environment Variables」を参照してください。
- Company 22 LDAP ディレクトリは `company22Host` という名前のホストのポート 2389 で実行されている。

▼ 仮想 DN を持つ Company 22 のディレクトリに対するデータビューの作成

1. **Company 22 のディレクトリの LDAP データソースを作成します。**

```
$ dpconf create-ldap-data-source company22-directory company22Host:2389
```

2. **Company 22 のディレクトリの LDAP データソースプールを作成します。**

```
$ dpconf create-ldap-data-source-pool company22-pool
```

3. **Company 22 のデータソースをデータソースプールに接続します。**

```
$ dpconf attach-ldap-data-source company22-pool company22-directory
```

4. 接続済みデータソースのウェイトを設定します。

```
$ dpconf set-attached-ldap-data-source-prop -h company22Host -p 2389 \
company22-pool company22-directory add-weight:2 \
bind-weight:2 compare-weight:2 delete-weight:2 \
modify-dn-weight:2 modify-weight:2 search-weight:2
```

5 仮想 **DN** の `dc=example,dc=com` で **Company 22** のディレクトリの **LDAP** データビューを作成します。

```
$ dpconf create-ldap-data-view company22-view company22-pool dc=example,dc=com
```

6 この仮想 **DN** を **Company 22** のディレクトリの実際の **DN** にマップするよう **Directory Proxy Server** に命令します。

```
$ dpconf set-ldap-data-view-prop company22-view \
dn-mapping-source-base-dn:dc=company22,dc=com
```

7 クライアント要求がこのデータビューに経路指定できるように、**Company 22** のディレクトリの **LDAP** データビューを有効にします。

```
$ dpconf set-ldap-data-view-prop company22-view is-enabled:true
```

8 変更を有効にするために、**Directory Proxy Server** を再起動します。

```
$ dpadm restart /local/myDPS
```

Company 22 のデータの HR データへの追加

HR 部門は Example.com と新しく買収した Company 22 の HR データの集約されたビューを必要としています。次の図は、グローバル HR アプリケーションの要件を示しています。

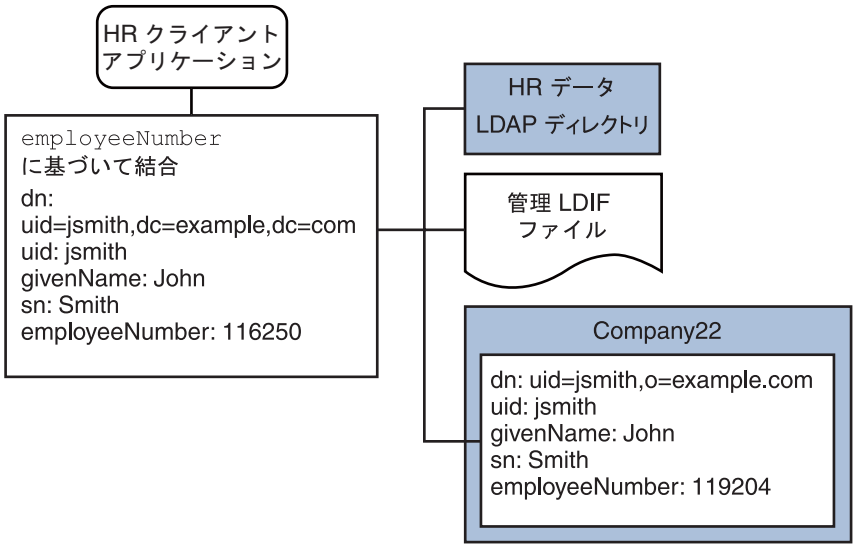


図 **23–6** 結合データビューと LDAP データビューからのデータの集結

▼ 結合データビューの例と Company 22 データビューの結合

1 データの集約方法を指定するフィルタ結合ルールを **Company 22** データビューで作成します。

次の結合ルールは、ユーザーエントリの `employeeNumber` 属性に基づいてデータが結合されるよう指定します。

```
$ dpconf set-ldif-data-view-prop company22-view \
filter-join-rule:'employeeNumber=\${example-join-view.employeeNumber}'
```

2 **Company 22** のデータビューと **Example.com** の結合データビューを集約する結合データビューを作成します。

```
$ dpconf create-join-data-view global-join-view example-join-view \
company22-view dc=example,dc=com
```

LDAP クライアントによる SQL データベース内の給与データへのアクセスの有効化

Example.com の給与部門は給与データを SQL データベースに保存しています。データベースには、`employee` テーブルと `salary` テーブルの 2 つのテーブルがあります。Example.com には、このデータへのアクセスを必要とする LDAP クライアントアプリケーションがあります。クライアントアプリケーションは SQL データが LDAP データのように見えることを必要としています。

次の図は、クライアントアプリケーションの要件を示しています。

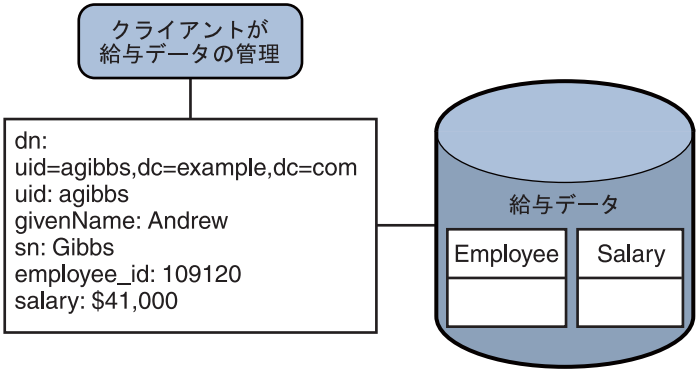


図 23–7 SQL データベースへのアクセスを提供する JDBC データビュー

このアプリケーション要件を満たすために、SQL テーブル内の列を LDAP 属性にマップする JDBC データビューが作成されます。

わかりやすいように、ここで使用するコマンドは次の情報を前提としています。

- Directory Proxy Server インスタンスはローカルホスト上のデフォルト LDAP ポート (389) で実行されている。
- Directory Proxy Server インスタンスは、`/local/myDPS` にある。
- プロキシマネージャーのパスワードを含むファイルへのパスは、変数 `LDAP_ADMIN_PWF` として設定されている。Directory Proxy Server 環境変数の設定の詳細については、『Sun Java System Directory Server Enterprise Edition 6.1 Installation Guide』の「Environment Variables」を参照してください。
- SQL データベースが起動して実行中である。
- `JAVA_HOME` 変数が正しい Java パスに設定されている。
- SQL データベースへのパスワードは、`myPasswordFile` ファイルに格納された `myPassword` である。

▼ Example.com の給与データベースに対する JDBC データビューの作成

1 給与データベース用の **JDBC** データソースを作成します。

```
$ dpconf create-jdbc-data-source -b payrollsqldb \
  -B jdbc:payrollsqldb:payrollsql://localhost/ \
  -J file://payrollsqldb.jar \
  -S org.payrollsqldb.jdbcDriver payroll-src
```

2 **SQL** データベースのプロパティーで **JDBC** データソースを設定します。

```
$ dpconf set-jdbc-data-source-prop payroll-src \
  db-user:proxy
  db-pwd-file:password-file-location/myPasswordFile
```

3 **JDBC** データソースを有効にします。

```
$ dpconf set-jdbc-data-source-prop payroll-src is-enabled:true
```

4 給与データベース用の **JDBC** データソースプールを作成します。

```
$ dpconf create-jdbc-data-source-pool payroll-pool
```

5 給与データソースをデータソースプールに接続します。

```
$ dpconf attach-jdbc-data-source payroll-pool payroll-src
```

6 仮想 **DN** の `o=payroll` で給与データベースの **JDBC** データビューを作成します。

```
$ dpconf create-jdbc-data-view payroll-view payroll-pool o=payroll
```

7 **SQL** データベースの各テーブルの **JDBC** テーブルを作成します。

```
$ dpconf create-jdbc-table jdbc-employee employee
$ dpconf create-jdbc-table jdbc-salary salary
```

8 各 **SQL** テーブルの各列の **JDBC** 属性を作成します。

```
$ dpconf add-jdbc-attr jdbc-employee eid employee_id
$ dpconf add-jdbc-attr jdbc-employee first firstname
$ dpconf add-jdbc-attr jdbc-employee last lastname
$ dpconf add-jdbc-attr jdbc-employee description description
$ dpconf add-jdbc-attr jdbc-employee spouse spousename
$ dpconf add-jdbc-attr jdbc-salary salary salary
$ dpconf add-jdbc-attr jdbc-salary social ssn
```

9 **JDBC** データビューで表示できる属性と書き込める属性を指定します。

```
$ dpconf set-jdbc-data-view-prop payroll-view \
 viewable-attr:eid \
 viewable-attr:first \
 viewable-attr:last \
 viewable-attr:desc \
 viewable-attr:spouse \
 viewable-attr:salary \
 viewable-attr:social
 $ dpconf set-jdbc-data-view-prop payroll-view \
 writable-attr:eid \
 writable-attr:first \
 writable-attr:last \
 writable-attr:description \
 writable-attr:spouse \
 writable-attr:salary \
writable-attr:social
```

10 **LDAP** オブジェクトクラスにマップされる **JDBC** オブジェクトクラスを作成します。

次のコマンドは、LDAP `person` オブジェクトクラスにマップされるオブジェクトクラスを作成します。オブジェクトクラスは、従業員テーブルは一次テーブルとして使用され、給与テーブルが二次テーブルとして使用されるよう指定します。`eid` 属性は、DN を構築するために使用されます。

```
$ dpcfg create-jdbc-object-class payroll-view \
 person jdbc-employee jdbc-salary eid
```

11 二次テーブルからのデータが一次テーブルからのデータにリンクされる方法を指定したフィルタ結合ルールを二次テーブル上で作成します。

次の結合ルールは、`employee_id` 属性に基づいてデータが結合されるよう指定します。

```
$ dpconf set-jdbc-table-prop jdbc-salary \
filter-join-rule:'employee_id=\${employee.employee_id}'
```

12 **JDBC** オブジェクトクラス上のスーパークラスを作成します。

```
$ set-jdbc-object-class-prop payroll-view person super-class:extensibleObject
```

仮想アクセス制御の追加

LDAP ディレクトリ上のアクセス制御は、ディレクトリ自体で ACI を定義することで処理されます。仮想データビューによってデータソースにアクセスされる場合、これらのデータビューで表示されるデータのみに適用される ACI を定義する必要があります。

Directory Proxy Server を経由するアクセスはすべて、接続ハンドラによって制御されます。接続ハンドラについては、第 25 章を参照してください。

▼ 匿名アクセスを許可する **ACI** の追加

1 **ACI** を追加します。

```
$ ldapadd -v -D "cn=proxy manager" -w password -p 389
dn: cn=ldifonly-acis,cn=virtual access controls
objectclass: top
objectclass: aciSource
cn: ldifonly-acis
dpsaci: (targetattr="*")(version 3.0; acl "anonymous_access"; allow(all) \
(userdn="ldap:///anyone");)
```

2 接続ハンドラで仮想 **ACI** をポイントします。

```
$ dpconf set-connection-handler-prop anonymous aci-source:ldifonly-acis
```

3 接続ハンドラを有効にします。

```
$ dpconf set-connection-handler-prop anonymous is-enabled:true
```

◆ ◆ ◆ 第 24 章

Directory Proxy Server とバックエンド LDAP サーバーの接続

この章では、Directory Proxy Server とバックエンド LDAP サーバーの接続の設定方法について説明します。この章の内容は次のとおりです。

Directory Proxy Server とバックエンド LDAP サーバーの接続の設定

LDAP データソースを作成した場合、その LDAP データソースに対して開かれるデフォルトの接続数は 6 で、読み取り操作、バインド操作、および書き込み操作にそれぞれ 2 つずつ使用されます。デフォルトの接続数を確認するには、次のコマンドを入力します。

```
dpconf get-ldap-data-source-prop src-name num-read-init num-write-init num-bind-init
num-bind-init   :  2
num-read-init   :  2
num-write-init  :  2
```

接続数は、トラフィックが増加すると自動的に増やされます。

Directory Proxy Server とバックエンド LDAP サーバーの接続の設定方法については、次の手順を参照してください。

▼ Directory Proxy Server とバックエンド LDAP サーバーの接続の数を設定する

注 – この手順では、バインド操作の接続数を設定します。読み取り操作または書き込み操作の接続数を設定するには、同じ手順を実行しますが `bind` を `read` または `write` に置き換えます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. バインド操作のための **Directory Proxy Server** とバックエンド **LDAP** サーバーの接続数の初期値を設定します。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    num-bind-init:new-value
   ```

2. バインド操作のための接続の増分を設定します。

   増分は、現在の数より多い接続が要求されるたびに追加される接続数です。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    num-bind-incr:new-value
   ```

3. バインド操作のための接続の最大数を設定します。

   この接続の最大数に達すると、それ以上接続を追加できなくなります。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    num-bind-limit:new-value
   ```

▼ 接続のタイムアウトを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** がデータソースに接続し続けることのできる最長時間を設定します。

  ```
  $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
   connect-timeout:new-value
  ```

  たとえば、接続のタイムアウトを 10 ミリ秒に設定します。

  ```
  $ dpconf set-ldap-data-source-prop -h host1 -p 1389 data-source-name connect-timeout:10
  ```

▼ 接続プール待機タイムアウトを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **Directory Proxy Server** が接続プール内の確立された接続を使用できるようになるまで待機できる最長時間を設定します。

```
$ dpconf set-server-prop -h host -p port data-source-name \
 connection-pool-wait-timeout:value
```

たとえば、タイムアウトを 20 秒に設定します。

```
$ dpconf set-ldap-data-source-prop -h host1 -p 1389 data-source-name \
 connection-pool-wait-timeout:20000
```

Directory Proxy Server とバックエンド LDAP サーバーとの間の SSL の設定

次の手順では、Directory Proxy Server とバックエンド LDAP サーバーとの間の SSL の設定方法を説明しています。

▼ Directory Proxy Server とバックエンド LDAP サーバーとの間の SSL を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **Directory Proxy Server** とバックエンド **LDAP** サーバー間のセキュアポートを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 ldaps-port:port-number
```

2. **Directory Proxy Server** とバックエンド **LDAP** サーバーの接続にいつ **SSL** が使用されるかを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name ssl-policy:value
```

   - *value* が `always` の場合、接続に常に SSL が使用されます。
   - *value* が `client` の場合、クライアントが SSL を使用している場合に SSL が使用されます。

接続が SSL を使用していない場合、`startTLS` コマンドを使用して SSL への接続をプロモートできます。

3 462 ページの「**Directory Proxy Server の SSL 暗号化方式と SSL プロトコルの選択**」で説明されているように、**SSL** のプロトコルと暗号化方式を選択します。

4 バックエンド **LDAP** サーバーからの **SSL** サーバー証明書を検証するよう **Directory Proxy Server** を設定します。

詳細は、377 ページの「証明書をバックエンド LDAP サーバーから Directory Proxy Server 上の証明書データベースに追加する」を参照してください。

5 バックエンド **LDAP** サーバーが **Directory Proxy Server** からの証明書を要求する場合は、**SSL** クライアント証明書を送信するよう **Directory Proxy Server** を設定します。

詳細については、378 ページの「バックエンド LDAP サーバーへの証明書のエクスポート」を参照してください。

6 変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

Directory Proxy Server の SSL 暗号化方式と SSL プロトコルの選択

Directory Proxy Server で使用できる暗号化方式とプロトコルは、使用している Java™ 仮想マシン (JVM™) によって異なります。デフォルトで、Directory Proxy Server は JVM マシンで有効なデフォルトの暗号化方式とプロトコルを使用します。

▼ 暗号化方式とプロトコルの一覧を選択する

この手順に従って、サポートされている暗号化方式とプロトコルおよび有効な暗号化方式とプロトコルを取得します。暗号化方式またはプロトコルがサポートされている場合は、それを有効または無効にできます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 サポートされている暗号化方式とプロトコルの一覧を表示します。

```
$ dpconf get-server-prop -h host -p port supported-ssl-cipher-suites \
 supported-ssl-protocols
```

2 有効な暗号化方式とプロトコルの一覧を表示します。

```
$ dpconf get-server-prop -h host -p port enabled-ssl-cipher-suites \
 enabled-ssl-protocols
```

3 1 つまたは複数のサポートされている暗号化方式またはプロトコルを有効にします。

a. 1 つまたは複数のサポートされている暗号化方式を有効にします。

```
$ dpconf set-server-prop -h host -p port \
 enabled-ssl-cipher-suites:supported-ssl-cipher-suite \
 [enabled-ssl-cipher-suites:supported-ssl-cipher-suite ...]
```

サポートされている暗号化方式の既存リストに暗号化方式を追加するには、次のコマンドを使用します。

```
$ dpconf set-server-prop -h host -p port \
 enabled-ssl-cipher-suites+:supported-ssl-cipher-suite
```

b. 1 つまたは複数のサポートされているプロトコルを有効にします。

```
$ dpconf set-server-prop -h host -p port \
 enabled-ssl-cipher-protocols:supported-ssl-cipher-protocol \
 [enabled-ssl-cipher-protocols:supported-ssl-cipher-protocol ...]
```

サポートされているプロトコルの既存リストにプロトコルを追加するには、次のコマンドを使用します。

```
$ dpconf set-server-prop -h host -p port \
 enabled-ssl-cipher-protocols+:supported-ssl-cipher-protocol
```

4 (省略可能) サポートされている暗号化方式またはプロトコルを無効にします。

```
$ dpconf set-server-prop -h host -p port \
 enabled-ssl-cipher-protocols-:supported-ssl-cipher-protocol
```

バックエンド LDAP サーバーへの要求の転送

この節では、Directory Proxy Server からの要求をバックエンド LDAP サーバーに転送するためのさまざまな方法について説明します。

バインド再実行での要求の転送

Directory Proxy Server のクライアント資格のバインド再実行については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Directory Proxy Server Configured for BIND Replay」を参照してください。次の手順は、Directory Proxy Server からの要求をバックエンド LDAP サーバーにバインド再実行を使用して転送する方法について説明しています。

▼ バインド再実行で要求を転送する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- クライアントによって提供された資格を使用してバックエンド **LDAP** サーバーへの接続を認証するデータソースクライアント資格を設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 client-cred-mode:use-client-identity
```

プロキシ承認での要求の転送

Directory Proxy Server 内のプロキシ承認については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Directory Proxy Server Configured for Proxy Authorization」を参照してください。

この節では、プロキシ承認とプロキシ承認制御を使用して要求を転送する手順について説明します。

▼ プロキシ承認を使用して要求を転送する

1. **version 1** または **version 2** のプロキシ承認制御を受け入れるようデータソースを設定します。

   たとえば、version 1 のプロキシ承認制御を受け入れるようデータソースを設定します。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    proxied-auth-use-v1:true
   ```

   または、version 2 のプロキシ承認制御を受け入れるようデータソースを設定します。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    proxied-auth-use-v1:false
   ```

2. プロキシ承認を使用してバックエンド **LDAP** サーバーへの接続を認証するようデータソースを設定します。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    client-cred-mode:use-proxy-auth
   ```

   書き込み操作のみのため、プロキシ承認を使用してバックエンド LDAP サーバーへの接続を認証するようデータソースを設定するには、次のコマンドを実行します。

   ```
   $ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
    client-cred-mode:use-proxy-auth-for-write
   ```

書き込み操作のみがプロキシ承認制御で実行される場合、クライアントのアイデンティティーは読み取り要求のために LDAP サーバーに転送されません。クライアントアイデンティティーのない要求の転送の詳細については、465 ページの「クライアントアイデンティティーなしでの要求の転送」を参照してください。

3 **Directory Proxy Server** のバインド資格でデータソースを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 bind-dn:DPS-bind-dn bind-pwd-file:filename
```

4 タイムアウトでデータソースを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 proxied-auth-check-timeout:value
```

Directory Proxy Server は、`getEffectiveRights` コマンドを使用して、プロキシ承認に適した ACI がクライアント DN にあることを確認します。その結果は Directory Proxy Server のキャッシュに格納され、`proxied-auth-check-timeout` の期限が終了すると更新されます。

5 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

▼ 要求にプロキシ承認制御が含まれている場合に、プロキシ承認を使用して要求を転送する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- **version 1**、**version 2**、またはその両方のプロキシ承認制御を受け入れるよう **Directory Proxy Server** を設定します。

```
$ dpconf set-server-prop -h host -p port allowed-ldap-controls:proxy-auth-v1 \
 allowed-ldap-controls:proxy-auth-v2
```

クライアントアイデンティティーなしでの要求の転送

次の手順は、クライアントアイデンティティーを転送せずに Directory Proxy Server からの要求をバックエンド LDAP サーバーに転送する方法について説明しています。

▼ クライアントアイデンティティーなしでの要求を転送する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. **Directory Proxy Server** の資格を使用してバックエンド **LDAP** サーバーへの接続を認証するようデータソースを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 client-cred-mode:use-specific-identity
```

2. **Directory Proxy Server** のバインド資格でデータソースを設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port data-source-name \
 bind-dn:bind-dn-of-DPS bind-pwd-file:filename
```

3. 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

   Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

代替ユーザーとしての要求の転送

ここでは、代替ユーザーとして要求を転送する方法について説明します。

▼ リモートユーザーマッピングを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 代替ユーザーで転送が行えるように設定を有効にします。

```
$ dpconf set-server-prop -h host -p port enable-user-mapping:true
```

2. リモートマッピング用の **ID** を含む属性の名前を指定します。

```
$ dpconf set-server-prop -h host -p port \
 remote-user-mapping-bind-dn-attr:attribute-name
```

3. **Directory Proxy Server** がクライアント **ID** をリモートでマップできるようにします。

```
$ dpconf set-server-prop -h host -p port enable-remote-user-mapping:true
```

4. デフォルトマッピングを設定します。

```
$ dpconf set-server-prop -h host -p port \
 user-mapping-default-bind-dn:default-mapping-bind-dn \
 user-mapping-default-bind-pwd-file:filename
```

マップしたアイデンティティーがリモート LDAP サーバー上に見つからない場合、クライアントアイデンティティーはデフォルトアイデンティティーにマップされます。

5 リモート **LDAP** サーバー上のクライアントのエントリでユーザーマッピングを設定します。

Directory Server でのユーザーマッピングの設定については、162 ページの「プロキシ承認」を参照してください。

▼ ローカルユーザーマッピングを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 代替ユーザーで転送が行えるように設定を有効にします。

```
$ dpconf set-server-prop -h host -p port enable-user-mapping:true
```

2 **Directory Proxy Server** がクライアント **ID** をリモートでマップする設定がされていないことを確認します。

```
$ dpconf set-server-prop -h host -p port enable-remote-user-mapping:false
```

3 デフォルトマッピングを設定します。

```
$ dpconf set-server-prop -h host -p port \
 user-mapping-default-bind-dn:default-mapping-bind-dn \
 user-mapping-default-bind-pwd-file:filename
```

リモート LDAP サーバー上のマッピングに失敗すると、クライアント ID がこの DN にマップされます。

4 認証されていないユーザーに操作の実行を許可する場合は、認証されていないクライアントに対するマッピングを設定します。

```
$ dpconf set-server-prop -h host -p port \
 user-mapping-anonymous-bind-dn:anonymous-mapping-bind-dn \
 user-mapping-anonymous-bind-pwd-file:filename
```

認証されていないユーザーに操作の実行を許可する方法については、483 ページの「匿名アクセスを設定する」を参照してください。

5 クライアントの **ID** を設定します。

```
$ dpconf set-user-mapping-prop -h host -p port \
 user-bind-dn:client-bind-dn user-bind-pwd-file:filename
```

6 代替ユーザーの ID を設定します。

```
$ dpconf set-user-mapping-prop -h host -p port \
 mapped-bind-dn:alt-user-bind-dn mapped-bind-pwd-file:filename
```

▼ 匿名クライアントのユーザーマッピングを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 認証されていないクライアントのマッピングを設定します。

```
$ dpconf set-server-prop -h host -p port \
 user-mapping-anonymous-bind-dn:anonymous-mapping-bind-dn \
 user-mapping-anonymous-bind-pwd-file:filename
```

リモート LDAP サーバーには匿名クライアントのエントリが含まれていないため、匿名クライアントのマッピングは、Directory Proxy Server で設定されます。

認証されていないユーザーへの操作の実行の許可については、483 ページの「匿名アクセスを設定する」を参照してください。

◆ ◆ ◆ 第 25 章

クライアントと Directory Proxy Server の接続

クライアントと Directory Proxy Server の接続の概要、接続ハンドラ、および接続ハンドラで使用される条件とポリシーの説明については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 20 章「Connections Between Clients and Directory Proxy Server」を参照してください。

この章の内容は次のとおりです。

接続ハンドラの作成、設定、削除

接続ハンドラの作成、設定、削除の方法と、データビューに対するアフィニティーの設定方法については、次の手順を参照してください。

▼ 接続ハンドラを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 接続ハンドラを作成します。

```
$ dpconf create-connection-handler -h host -p port connection-handler-name
```

2 (省略可能) 接続ハンドラのリストを表示します。

```
$ dpconf list-connection-handlers -h host -p port
```

▼ 接続ハンドラを設定する

始める前に　接続ハンドラのプロパティーは、Directory Proxy Server インスタンスに定義されているその他の接続ハンドラのプロパティーに関連して定義する必要があります。さまざまな基準のセットを確実に指定し、正しい優先順位を設定するため、すべての接続ハンドラのプロパティーを考慮してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 接続ハンドラの詳細リストを表示して、それらのキープロパティーと対応する優先順位を確認します。

```
$ dpconf list-connection-handlers -h host -p port -v
Name                        is-enabled  priority  description
--------------------------  ----------  --------  --------------------------
anonymous                   false       99        unauthenticated connections
default connection handler  true        100       default connection handler
```

接続ハンドラ `anonymous` および `default connection handler` は、Directory Proxy Server のインスタンスの作成時に作成されます。

2 1 つの接続ハンドラのすべてのプロパティーを表示します。

```
$ dpconf get-connection-handler-prop -h host -p port connection-handler-name
```

新しい接続ハンドラのデフォルトプロパティーは、次のようになります。

```
aci-source                    :  -
allowed-auth-methods          :  anonymous
allowed-auth-methods          :  sasl
allowed-auth-methods          :  simple
allowed-ldap-ports            :  ldap
allowed-ldap-ports            :  ldaps
bind-dn-filters               :  any
data-view-routing-custom-list :  -
data-view-routing-policy      :  all-routable
description                   :  -
domain-name-filters           :  any
enable-data-view-affinity     :  false
ip-address-filters            :  any
is-enabled                    :  false
is-ssl-mandatory              :  false
priority                      :  99
request-filtering-policy      :  no-filtering
resource-limits-policy        :  no-limits
schema-check-enabled          :  false
user-filter                   :  any
```

3 接続ハンドラの優先順位を設定します。

```
$ dpconf set-connaection-handler-prop -h host -p port connection-handler-name priority:value
```

優先順位は 1 ～ 100 の任意の数にすることができ、もっとも高い優先順位は 1 です。Directory Proxy Server のインスタンスでは、接続ハンドラは優先順位の順に評価されます。

4 (省略可能) 接続ハンドラの DN フィルタリングプロパティーを指定します。

このプロパティーにより、バインド DN の一部またはすべてに基づいてアクセスを制御できます。プロパティーの値は正規表現です。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 bind-dn-filters:regular-expression
```

バインド DN フィルタは、Java™ 正規表現の形式をとります。Java 正規表現の作成については、http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html を参照してください。

たとえば、すべてのバインドを `ou=people,dc=example,dc=com` の下のユーザーから `secure-handler` という接続ハンドラに送信するには、次のように `bind-dn-filters` プロパティーを設定します。

```
$ dpconf set-connection-handler-prop -h host1 -p 1389 secure-handler \
 bind-dn-filters:"uid=.*,ou=people,dc=example,dc=com"
```

5 (省略可能) この接続ハンドラで使用する要求フィルタリングポリシーの名前を指定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 request-filtering-policy:policy-name
```

ここで、*policy-name* は既存の要求フィルタリングポリシーの名前です。要求フィルタリングポリシーの作成と設定の方法については、473 ページの「要求フィルタリングポリシーと検索データの非表示ルールの作成と設定」を参照してください。

6 (省略可能) この接続ハンドラで使用するリソース制限ポリシーの名前を指定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 resource-limits-policy:policy-name
```

ここで、*policy-name* は既存のリソース制限ポリシーの名前です。リソース制限ポリシーの作成と設定の方法については、477 ページの「リソース制限ポリシーの作成と設定」を参照してください。

7 手順 2 で一覧表示されるほかのプロパティーを設定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 property:value [property:value ...]
```

たとえば、SSL 接続のみを受け付けるように接続ハンドラを設定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 is-ssl-mandatory:true
```

プロパティーとその有効な値のリストについては、次のコマンドを実行します。

```
$ dpconf help-properties connection-handler
```

8 接続ハンドラを有効にします。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name is-enabled:true
```

9 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

▼ 接続ハンドラを削除する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 (省略可能) 接続ハンドラのリストを表示します。

```
$ dpconf list-connection-handlers -h host -p port
```

2 1 つまたは複数の接続ハンドラを削除します。

```
$ dpconf delete-connection-handler -h host -p port connection-handler-name [connection-handler-name ... ]
```

▼ データビューに対するアフィニティーを設定する

接続ハンドラに接続を割り当てると、その接続にある要求は、その接続ハンドラに設定されているデータビューのリスト、または設定されている データビューのすべてに公開されます。その接続のそれ以降の要求は、最初の要求に対して使われるデータビューに排他的に公開されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データビューに対するアフィニティーを有効にします。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 enable-data-view-affinity:true
```

2 (省略可能) 要求をデータビューのカスタムリストに経路指定するように接続ハンドラを設定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name data-view-routing-policy:custom
```

3 (省略可能) データビューのリストを設定します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 data-view-routing-custom-list:view-name [data-view-routing-custom-list:view-name ...]
```

データビューの既存リストにデータビューを追加するには、次のコマンドを使用します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 data-view-routing-custom-list+:view-name
```

データビューの既存リストからデータビューを削除するには、次のコマンドを使用します。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 data-view-routing-custom-list-:view-name
```

要求フィルタリングポリシーと検索データの非表示ルールの作成と設定

要求フィルタリングポリシーの概要は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Request Filtering Policies for Connection Handlers」を参照してください。検索データの非表示ルールの概要は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Search Data Hiding Rules in the Request Filtering Policy」を参照してください。

要求フィルタリングポリシーと検索データの非表示ルールの作成と設定の方法については、次の手順を参照してください。

▼ 要求フィルタリングポリシーを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 要求フィルタリングポリシーを作成します。

```
$ dpconf create-request-filtering-policy policy-name
```

2 要求フィルタリングポリシーを接続ハンドラと関連付けます。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 request-filtering-policy:policy-name
```

▼ 要求フィルタリングポリシーを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 要求フィルタリングポリシーのプロパティーを表示します。

```
$ dpconf get-request-filtering-policy-prop -h host -p port policy-name
```

要求フィルタリングポリシーのデフォルトプロパティーは次のとおりです。

```
allow-add-operations                :  true
allow-bind-operations               :  true
allow-compare-operations            :  true
allow-delete-operations             :  true
allow-extended-operations           :  true
allow-inequality-search-operations  :  true
allow-modify-operations             :  true
allow-rename-operations             :  true
allow-search-operations             :  true
allowed-comparable-attrs            :  all
allowed-search-scopes               :  base
allowed-search-scopes               :  one-level
allowed-search-scopes               :  subtree
allowed-subtrees                    :  ""
description                         :  -
prohibited-comparable-attrs         :  none
prohibited-subtrees                 :  none
```

2 手順 1 で一覧表示されるプロパティーの 1 つまたは複数を設定することで、要求フィルタリングポリシーを設定します。

```
$ dpconf set-request-filtering-policy-prop -h host -p port policy-name \
  property:value [property:value ...]
```

手順 1 で一覧表示されるプロパティーを設定することで、次のような要求フィルタリングポリシーの機能を設定します。

- クライアントに実行が許可されている操作のタイプ
- クライアントに公開されるサブツリー、またはクライアントから非表示にされるサブツリー
- 検索操作の範囲

- 検索フィルタのタイプ
- 検索操作と比較操作で比較できる属性と比較できない属性のタイプ

▼ 検索データ非表示ルールを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 要求フィルタリングポリシーに対して、1 つまたは複数の検索データ非表示ルールを作成します。

```
$ dpconf create-search-data-hiding-rule -h host -p port policy-name rule-name \
 [rule-name ...]
```

2 検索データ非表示ルールのプロパティーを表示します。

```
$ dpconf get-search-data-hiding-rule-prop policy-name rule-name
```

検索データ非表示ルールのデフォルトプロパティーは、次のとおりです。

```
attrs                           :  -
rule-action                     :  hide-entry
target-attr-value-assertions    :  -
target-dn-regular-expressions   :  -
target-dns                      :  -
```

3 手順 2 で一覧表示されるプロパティーの 1 つまたは複数を設定することで、検索データ非表示ルールを設定します。

```
$ dpconf set-search-data-hiding-rule-prop -h host -p port policy-name rule-name \
  property:value [property:value ...]
```

次のいずれかのルールアクションを使用できます。

`hide-entry`	ターゲットエントリは返されません。
`hide-attributes`	ターゲットエントリは返されますが、指定した属性は表示されません。
`show-attributes`	ターゲットエントリは返されますが、指定した属性以外は表示されません。

ルールは次のエントリに適用できます。

`target-dns`	指定した DN を持つエントリ
`target-dn-regular-expressions`	指定した DN パターンを持つエントリ
`target-attr-value-assertions`	指定した属性名と属性値のペアを持つエントリ (*attrName#attrValue*)

次の設定では、タイプ inetorgperson のエントリを非表示にする検索データ非表示ルールを定義します。

```
$ dpconf set-search-data-hiding-rule-prop -h host1 -p port my-policy my-rule \
  target-attr-value-assertions:objectclass#inetorgperson
```

要求フィルタリングポリシーと検索データ非表示ルールの例

次の例には、要求フィルタリングポリシーと検索データ非表示ルールが含まれます。要求フィルタリングポリシーを検索データ非表示ルールと組み合わせると、次のようにデータへのアクセスが制限されます。

- 次のタイプの操作は許可されません。追加、削除、拡張、変更、名前変更。
- アクセスできるのは、ou=people,dc=sun,dc=com サブツリーだけです。
- 検索操作では inetorgperson タイプ以外のエントリが返されます。

例 25–1 要求フィルタリングポリシーの例

```
allow-add-operations                :  false
allow-bind-operations               :  true
allow-compare-operations            :  true
allow-delete-operations             :  false
allow-extended-operations           :  false
allow-inequality-search-operations  :  true
allow-modify-operations             :  false
allow-rename-operations             :  false
allow-search-operations             :  true
allowed-comparable-attrs            :  all
allowed-search-scopes               :  base
allowed-search-scopes               :  one-level
allowed-search-scopes               :  subtree
allowed-subtrees                    :  ou=people,dc=sun,dc=com
description                         :  myRequestFilteringPolicy
prohibited-comparable-attrs         :  none
prohibited-subtrees                 :  none
```

例 25–2 検索データ非表示ルールの例

```
attrs                           :  -
rule-action                     :  hide-entry
target-attr-value-assertions    :  objectclass:inetorgperson
target-dn-regular-expressions   :  -
target-dns                      :  -
```

リソース制限ポリシーの作成と設定

リソース制限ポリシーの概要については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Resource Limits Policies for Connection Handlers」を参照してください。リソース制限ポリシーを作成する方法と検索制限をカスタマイズする方法については、次の手順を参照してください。

▼ リソース制限ポリシーを作成する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 リソース制限ポリシーを作成します。

```
$ dpconf create-resource-limits-policy -h host -p port policy-name
```

リソース制限ポリシーのプロパティーを変更する方法については、477 ページの「リソース制限ポリシーを設定する」を参照してください。

2 リソース制限ポリシーを接続ハンドラに関連付けます。

```
$ dpconf set-connection-handler-prop -h host -p port connection-handler-name \
 resource-limits-policy:policy-name
```

▼ リソース制限ポリシーを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 リソース制限ポリシーのプロパティーを表示します。

```
$ dpconf get-resource-limits-policy-prop -h host -p port policy-name
```

リソース制限ポリシーのデフォルトプロパティーは次のとおりです。

```
description                                 :  -
max-client-connections                      :  unlimited
max-connections                             :  unlimited
max-simultaneous-operations-per-connection  :  unlimited
max-total-operations-per-connection         :  unlimited
minimum-search-filter-substring-length      :  unlimited
referral-bind-policy                        :  default
referral-hop-limit                          :  default
```

```
referral-policy                         :  default
search-size-limit                       :  unlimited
search-time-limit                       :  unlimited
```

2 手順 1 で一覧表示されるプロパティーの 1 つまたは複数を設定することで、リソース制限ポリシーを設定します。

```
$ dpconf set-resource-limits-policy-prop -h host -p port policy-name \
  property:value [property:value ...]
```

▼ 検索制限をカスタマイズする

検索ベースと検索範囲に従って行われる検索操作に対して、カスタマイズした検索制限を定義することができます。検索操作のターゲット DN と範囲が指定した条件と一致すると、検索結果で返されるエントリの数が制限されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 1 つまたは複数のカスタムの検索制限を作成します。

```
$ dpconf create-custom-search-size-limit -h host -p port policy-name \
  custom-search-limit-name [custom-search-limit-name ...]
```

2 カスタムの検索制限に対する条件を設定します。

```
$ dpconf set-custom-search-size-limit-prop -h host -p port policy-name \
  custom-search-limit-name one-level-search-base-dn:value subtree-search-base-dn:value
```

3 手順 2 で検索が条件の 1 つを満たす場合に返される結果の数の制限を設定します。

```
$ dpconf set-custom-search-size-limit-prop -h host -p port policy-name \
 custom-search-limit-name search-size-limit:value
```

4 カスタムの検索制限のプロパティーを表示します。

```
$ dpconf get-custom-search-size-limit-prop -h host -p port policy-name \
  custom-search-limit-name
```

カスタムの検索制限のデフォルトプロパティーは、次のとおりです。

```
one-level-search-base-dn  :  -
search-size-limit         :  unlimited
subtree-search-base-dn    :  -
```

接続ベースのルーターとしての Directory Proxy Server の設定

Directory Proxy Server 5.2 は接続ベースのルーターです。Directory Proxy Server 5.2 では、クライアント接続は特定のディレクトリサーバーに経路指定されます。そのクライアント接続のすべての要求は、接続が切断されるかクライアントがアンバインドされるまでは、同じディレクトリサーバーに送信されます。

Directory Proxy Server 6.1 は、操作ベースのルーターです。ただし、互換性のため、このバージョンの Directory Proxy Server は、次の手順で説明するように接続ベースのルーターとして設定できます。

▼ Directory Proxy Server を接続ベースのルーターとして設定する

1 469 ページの「接続ハンドラの作成、設定、削除」で説明するように、1 つまたは複数の接続ハンドラを作成し、設定します。

デフォルトの接続ハンドラを使用することもできます。

2 要求を `root data view` のみに経路指定するように、すべての接続ハンドラを設定します。

次に例を示します。

```
$ dpconf set-connection-handler-prop -h host1 -p 1389 myConnectionHandler \
 data-view-routing-policy:custom data-view-routing-custom-list:"root data view"
```

3 381 ページの「LDAP データソースの作成と設定」で説明するように、各バックエンド LDAP サーバーのデータソースを作成し設定します。

次に例を示します。

```
$ dpconf create-ldap-data-source -h host1 -p 1389 myDataSource host2:2389
```

4 384 ページの「LDAP データソースプールの作成と設定」で説明するように、データソースプールを作成し設定します。

次に例を示します。

```
$ dpconf create-ldap-data-source-pool -h host1 -p 1389 myDataSourcePool
```

5 385 ページの「LDAP データソースのデータソースプールへの接続」で説明するように、すべてのデータソースをデータソースプールに接続します。

次に例を示します。

```
$ dpconf attach-ldap-data-source -h host1 -p 1389 myDataSourcePool myDataSource
```

6 463 ページの「バインド再実行での要求の転送」で説明するように、**BIND** 再実行を使用してクライアントを認証するように各データソースを設定します。

次に例を示します。

```
$ dpconf set-ldap-data-source-prop -h host1 -p 1389 myDataSource \
 client-cred-mode:use-client-identity
```

7 395 ページの「クライアントアフィニティーの設定」で説明するように、クライアント接続とデータソースプール間のアフィニティーを設定します。

次に例を示します。

```
$ dpconf set-ldap-data-source-pool-prop -h host1 -p 1389 myDataSourcePool \
 enable-client-affinity:true client-affinity-policy:read-write-affinity-after-write
```

◆ ◆ ◆ 第 26 章

Directory Proxy Server のクライアント認証

Directory Proxy Server のクライアント認証の概要については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 21 章「Directory Proxy Server Client Authentication」を参照してください。

この章の内容は次のとおりです。

クライアントと Directory Proxy Server 間のリスナーの設定

Directory Proxy Server には、クライアントとの通信のために、セキュリティー保護されたリスナーとセキュリティー保護されていないリスナーがあります。Directory Proxy Server のリスナーについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Directory Proxy Server Client Listeners」を参照してください。ここでは、リスナーの設定方法について説明します。

▼ クライアントと Directory Proxy Server 間のリスナーを設定する

注 – この手順では、クライアントと Directory Proxy Server 間のセキュリティー保護されていないリスナーを設定します。セキュリティーが確保されているリスナーを設定するには、手順は同じですが、`ldap` を `ldaps` に置き換えます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。 DSCC では、このプロパティーを「パフォーマンス」タブで設定できます。

1 セキュリティー保護されていないリスナーのプロパティーを表示します。

```
$ dpconf get-ldap-listener-prop -h host -p port
```

セキュリティー保護されていないリスナーのデフォルトプロパティーは、次のとおりです。

```
connection-idle-timeout          :  1h
connection-read-data-timeout     :  2s
connection-write-data-timeout    :  1h
is-enabled                       :  true
listen-address                   :  0.0.0.0
listen-port                      :  port-number
max-connection-queue-size        :  128
max-ldap-message-size            :  unlimited
number-of-threads                :  2
use-tcp-no-delay                 :  true
```

2 手順 1 に一覧表示されているプロパティーの 1 つまたは複数を要件に応じて変更します。

```
$ dpconf set-ldap-listener-prop -h host -p port property:new-value
```

たとえば、host1 で実行されている Directory Proxy Server のインスタンスのセキュリティー保護されていないポートを無効にするには、次のコマンドを実行します。

```
$ dpconf set-ldap-listener-prop -h host1 -p 1389 is-enabled:false
```

注意 – 非特権ポート番号を使用する場合は、Directory Proxy Server を root として実行する必要があります。

セキュリティー保護されていないポート番号を変更するには、次のコマンドを実行します。

```
$ dpconf set-ldap-listener-prop -h host -p port listen-port:new-port-number
```

3 必要に応じて、変更を有効にするために **Directory Proxy Server** のインスタンスを再起動します。

特定のリスナープロパティーの変更には、サーバーの再起動が必要です。サーバーの再起動が必要な場合は、`dpconf` アラートが表示されます。Directory Proxy Server の再起動については、355 ページの「Directory Proxy Server を再起動する」を参照してください。

Directory Proxy Server に対するクライアントの認証

デフォルトでは、Directory Proxy Server は単純バインド認証用に設定されています。単純バインド認証では、追加の設定は必要ありません。

クライアントと Directory Proxy Server 間の認証については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Client Authentication Overview」を参照してください。認証の設定方法については、次の手順を参照してください。

▼ 証明書ベースの認証を設定する

クライアントの証明書ベースの認証については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Configuring Certificates in Directory Proxy Server」を参照してください。この節では、証明書ベースの認証の設定方法について説明します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

注 – 証明書ベースの認証は、SSL 接続でのみ実行できます。

- クライアントが **SSL** 接続を確立する場合に証明書の提示を必要とするように **Directory Proxy Server** を設定します。

```
$ dpconf set-server-prop -h host -p port allow-cert-based-auth:require
```

▼ 匿名アクセスを設定する

匿名アクセスについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Anonymous Access」を参照してください。匿名クライアントのアイデンティティーを別のアイデンティティーにマップする方法については、466 ページの「代替ユーザーとしての要求の転送」を参照してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 非認証ユーザーに操作の実行を許可します。

```
$ dpconf set-server-prop -h host -p port allow-unauthenticated-operations:true
```

▼ SASL 外部バインド用に Directory Proxy Server を設定する

SASL 外部バインドについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Using SASL External Bind」を参照してください。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 非認証操作を禁止します。

   ```
   $ dpconf set-server-prop -h host -p port allow-unauthenticated-operations:false
   ```

2. 接続の確立時に証明書を提示するようクライアントに求めます。

   ```
   $ dpconf set-server-prop -h host -p port allow-cert-based-auth:require
   ```

   クライアントが DN が含まれた証明書を提供します。

3. **SASL** 外部バインドによってクライアントの認証を有効にします。

   ```
   $ dpconf set-server-prop -h host -p port allow-sasl-external-authentication:true
   ```

4. バックエンド **LDAP** サーバーでクライアント証明書をマップするために **Directory Proxy Server** に使用されアイデンティティーを設定します。

   ```
   $ dpconf set-server-prop -h host -p port cert-search-bind-dn:bind-DN \
    cert-search-bind-pwd-file:filename
   ```

5. **Directory Proxy Server** が検索するサブツリーのベース **DN** を設定します。

   Directory Proxy Server がサブツリーを検索し、クライアント証明書にマップされたユーザーエントリを見つけます。

   ```
   $ dpconf set-server-prop -h host -p port cert-search-base-dn:base-DN
   ```

6. クライアント証明書の情報を **LDAP** サーバー上の証明書にマップします。

   a. 証明書を含む **LDAP** サーバー上の属性に名前を付けます。

      ```
      $ dpconf set-server-prop cert-search-user-attribute:attribute
      ```

   b. クライアント証明書上の属性を、証明書のある **LDAP** サーバー上のエントリの **DN** にマップします。

      ```
      $ dpconf set-server-prop -h host -p port \
       cert-search-attr-mappings:client-side-attribute-name:server-side-attribute-name
      ```

たとえば、DN が cn=user1,o=sun,c=us のクライアント証明書を DN が uid=user1,o=sun の LDAP エントリにマップするには、次のコマンドを実行します。

```
$ dpconf set-server-prop -h host1 -p 1389 cert-search-attr-mappings:cn:uid \
 cert-search-attr-mappings:o:o
```

7 **(省略可能) SASL** 外部バインド操作の要求をすべてのデータビューまたはデータビューのカスタムリストに経路指定します。

- すべてのデータビューに要求を経路指定するには、次のコマンドを実行します。

```
$ dpconf set-server-prop -h host -p port cert-data-view-routing-policy:all-routable
```

- データビューのリストに要求を経路指定するには、次のコマンドを実行します。

```
$ dpconf set-server-prop -h host -p port cert-data-view-routing-policy:custom \
 cert-data-view-routing-custom-list:view-name [view-name...]
```

◆ ◆ ◆ 第 27 章

27 Directory Proxy Server のログ

Directory Proxy Server では、アクセスログおよびエラーログに情報が記録されます。Directory Server とは異なり、Directory Proxy Server には監査ログはありません。Directory Proxy Server のログについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の第 23 章「Directory Proxy Server Logging」を参照してください。

この章の内容は次のとおりです。

Directory Proxy Server ログの表示

Directory Proxy Server のログは、ログファイルを直接表示するか、Directory Service Control Center (DSCC) を通して参照できます。

デフォルトでは、ログは次のディレクトリ内に格納されます。

instance-path`/logs`

次の図は、DSCC 上の Directory Proxy Server のエラーログの画面キャプチャーです。

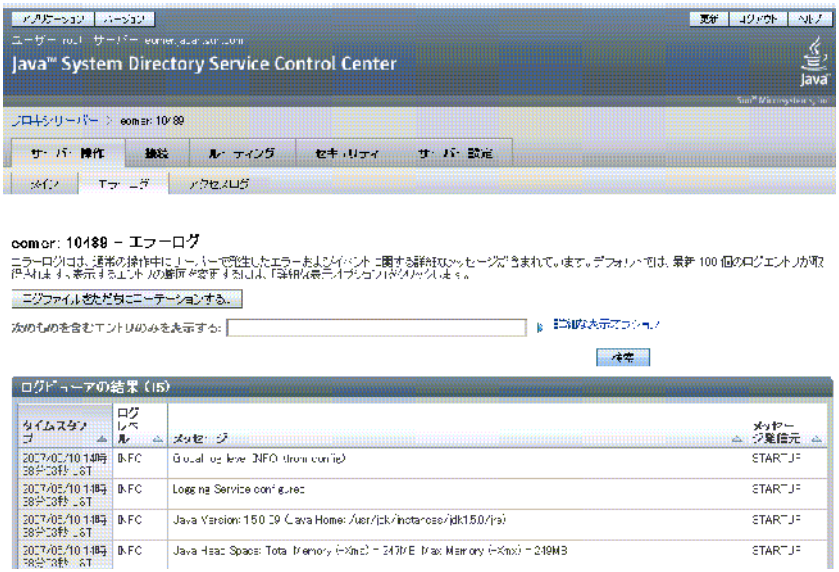


図 27–1　Directory Proxy Server のエラーログウィンドウ

Directory Proxy Server のログの設定

Directory Proxy Server のエラーログとアクセスログは、`dpconf` コマンドまたは DSCC を使用することで設定できます。DSCC によるログの設定方法については、Directory Proxy Server のオンラインヘルプを参照してください。この節では、`dpconf` コマンドを使用して Directory Proxy Server のログを設定する方法について説明します。

次のコマンドを実行することで、設定オプションのリストを、個々のオプションで設定可能な値およびデフォルトの設定値の情報とともに取得できます。

```
$ dpconf help-properties error-log

$ dpconf help-properties access-log
```

▼ Directory Proxy Server のアクセスログとエラーログを設定する

この手順では、Directory Proxy Server のアクセスログを設定します。Directory Proxy Server のエラーログを設定する場合には、以下の手順に出てくる「`access`」を「`error`」に置き換えて、同じ手順を実行します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 アクセスログのプロパティーを表示します。

```
$ dpconf get-access-log-prop -h host -p port
```

アクセスログのデフォルトプロパティーは、次のとおりです。

```
default-log-level                :  info
enable-log-rotation                 :  true
log-buffer-size                  :  9.8k
log-file-name                    :  logs/access
log-file-perm                    :  600
log-level-client-connections     :  -
log-level-client-disconnections  :  -
log-level-client-operations      :  -
log-level-connection-handlers    :  -
log-level-data-sources           :  -
log-level-data-sources-detailed  :  -
log-min-size                        :  100M
log-rotation-frequency           :  1h
log-rotation-policy              :  size
log-rotation-size                :  100M
log-rotation-start-day           :  1
log-rotation-start-time          :  0000
log-search-filters               :  false
max-age                          :  unlimited
max-log-files                    :  10
max-size                         :  unlimited
min-free-disk-space-size         :  1M
```

2 手順 1 で一覧表示されるプロパティーのうち、1 つまたは複数を変更します。

```
$ dpconf set-access-log-prop -h host -p port property:value \
 [property:value ...]
```

たとえば、すべてのメッセージカテゴリのデフォルトのログレベルを warning に設定するには、`default-log-level` プロパティーの値を `warning` に設定します。

```
$ dpconf set-access-log-prop -h host1 -p 1389 default-log-level:warning
```

すべてのログを無効にするには、各メッセージカテゴリのログレベルに関係なく、`default-log-level` プロパティーの値を none に設定します。

```
$ dpconf set-access-log-prop -h host1 -p 1389 default-log-level:none
```

特定のログレベルをデフォルトのログレベルにリセットするには、該当のログレベルのプロパティーを inherited に設定します。たとえば、クライアント接続のログレベルをリセットするには、次のコマンドを実行します。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-level-client-connections:inherited
```

set-access-log-prop サブコマンドで設定できるプロパティーを調べるには、次のように入力します。

```
$ dpconf help-properties access-log
```

Directory Proxy Server ログのローテーションの設定

デフォルトでは、ログファイルのサイズが 100M バイトに達すると、ログファイルのローテーションが行われます。デフォルトでは 10 個のログファイルが保存され、それ以降はローテーション手順によってもっとも古いログファイルの上書きが始められます。この節では、Directory Proxy Server ログを予定されたローテーションに対して設定する方法、ログのローテーションを手動で行う方法、ログのローテーションを無効にする方法について説明します。設定例は、492 ページの「ログローテーションの設定例」を参照してください。

▼ アクセスログとエラーログの定期ローテーションを設定する

この手順では、Directory Proxy Server のアクセスログを設定します。Directory Proxy Server のエラーログを設定する場合には、以下の手順に出てくる「access」を「error」に置き換えて、同じ手順を実行します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. (省略可能) アクセスログのプロパティーを表示します。

   ```
   $ dpconf get-access-log-prop -h host -p port
   ```

2. (省略可能) アクセスログのプロパティーに対して有効な値を表示します。

   ```
   $ dpconf help-properties access-log
   ```

3. 一定のサイズに達するとログのローテーションが行われるようにするには、次のプロパティーを設定します。

   ```
   $ dpconf set-access-log-prop -h host -p port \
    log-rotation-policy:size log-rotation-size:maximum file size
   ```

最大ファイルサイズの単位を指定しない場合、デフォルトの単位である「バイト」が使用されます。ログファイルが定義されたサイズに達すると、ログのローテーションが行われます。ファイルサイズは、少なくとも 1M バイトにし、2G バイトを超えないようにする必要があります。

サイズによるログのローテーションの例については、492 ページの「ログサイズに基づくログのローテーション」を参照してください。

4 ログのローテーションを定期的に行うには、ログサイズに関係なく、次のプロパティーを設定します。

```
$ dpconf set-access-log-prop -h host -p port \
 log-rotation-frequency:interval in months, weeks, hours, or minutes \
 log-rotation-policy:periodic \
 log-rotation-start-day:day in week (1-7) or day in the month (1-31) \
 log-rotation-start-time:time of day (hhmm)
```

ログのローテーションが月の 31 日に行われるように設定すると、その月の日数が 31 日より短い場合、ログのローテーションは次の月の最初の日に行われます。

定期的なログのローテーションの例については、493 ページの「時間に基づくログのローテーション」を参照してください。

5 ログファイルが十分に大きくなったときに定期的なログのローテーションを行うには、`log-rotation-frequency` プロパティーと `log-min-size` プロパティーを設定します。

```
$ dpconf set-access-log-prop -h host -p port \
 log-rotation-frequency:interval in months, weeks, hours, or minutes \
 log-rotation-policy:periodic log-min-size:minimum file size
 log-rotation-start-day:day in week (1-7) or day in the month (1-31) \
 log-rotation-start-time:time of day (hhmm)
```

`log-min-size` プロパティーは、ログの最小サイズを示します。ローテーションは、ログファイルが指定したサイズより大きくなった場合のみ、予定された時間に実行されます。

ログのローテーションが月の 31 日に行われるように設定すると、その月の日数が 31 日より短い場合、ログのローテーションは次の月の最初の日に行われます。

ファイルサイズが十分に大きくなったときに定期的にログのローテーションを行う方法の例については、494 ページの「時間とログサイズに基づくログのローテーション」を参照してください。

▼ アクセスログとエラーログのファイルのローテーションを手動で行う

この手順では、Directory Proxy Server のアクセスログのローテーションを行います。Directory Proxy Server のエラーログのローテーションを行う場合には、以下の手順に出てくる「access」を「error」に置き換えて、同じ手順を実行します。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- アクセスログのローテーションを行います。

```
$ dpconf rotate-log-now -h host -p port access
```

▼ アクセスログとエラーログのローテーションを無効にする

この手順では、Directory Proxy Server のアクセスログのローテーションを無効にします。Directory Proxy Server のエラーログのローテーションを無効にする場合には、以下の手順に出てくる「access」を「error」に置き換えて、同じ手順を実行します。

- ログファイルのローテーションを無効にします。

```
$ dpconf set-access-log-prop -h host -p port enable-log-rotation:false
```

ログローテーションの設定例

ログサイズ、時間、またはその両方によってログのローテーションを設定する方法の例は、次のとおりです。

ログサイズに基づくログのローテーション

この節では、ログサイズだけに従ってログのローテーションを設定する方法の例を示します。次の設定では、最後にログのローテーションが行われてから経過した時間に関係なく、ログが 10M バイトに達したときにローテーションが行われます。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-policy:size \
  log-rotation-size:10M
```

時間に基づくログのローテーション

この節で示す例では、ログサイズに関係なく、最後のローテーションからの経過時間に従ってログローテーションを設定する方法を示します。

- 次の設定では、ログのローテーションは今日の 3:00 とそれ以降の 8 時間ごとに行われ、ログファイルのサイズとは無関係です。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:8h \
 log-rotation-policy:periodic log-rotation-start-time:0300
```

- 次の設定では、ログファイルのサイズとは無関係に、毎日 3:00、13:00、23:00 にログのローテーションが行われます。`log-rotation-start-time` パラメータは `log-rotation-frequency` パラメータに優先するため、ログのローテーションは 23:00 に行われると、次回は 10 時間後ではなく、4 時間後の 3:00 に行われます。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:10h \
 log-rotation-policy:periodic log-rotation-start-time:0300
```

- 次の設定では、ログのローテーションは月曜日の正午に行われ、そのあと毎週同じ時間に行われ、ログファイルのサイズとは無関係です。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:1w \
 log-rotation-policy:periodic log-rotation-start-day:2 log-rotation-start-time:1200
```

- 次の設定では、ログファイルのサイズとは無関係に、月曜日の正午に最初のログのローテーションが行われ、それ以降は、3 日おきに正午にローテーションが行われます。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:3d \
 log-rotation-policy:periodic log-rotation-start-day:2 log-rotation-start-time:1200
```

  ログのローテーションが行われる曜日は、月曜日、木曜日、日曜日、水曜日というように続きます。`log-rotation-start-day` パラメータは最初の週だけに適用されます。2 週目の月曜日にはログのローテーションは行われません。

- 次の設定では、ログサイズとは無関係に、毎月 22 日の正午にログのローテーションが行われます。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:1m \
 log-rotation-policy:periodic log-rotation-start-day:22 \
 log-rotation-start-time:1200
```

`log-rotation-start-day` を 31 に設定したときに、30 日しかない月の場合、ログのローテーションは次の月の最初の日に行われます。`log-rotation-start-day` を 31 に設定したときに、28 日しかない月 (2 月) の場合、ログのローテーションは 3 日に行われます。

時間とログサイズに基づくログのローテーション

この例では、ファイルサイズが十分に大きくなった場合の、指定した間隔でのログのローテーションの設定方法を示します。

次の設定では、ログファイルのサイズが 1M バイトを超えた場合に、毎日 3:00、11:00、19:00 にログのローテーションが行われます。ログファイルのサイズが 1M バイトを超えなければ、ログファイルのローテーションは行われません。

```
$ dpconf set-access-log-prop -h host1 -p 1389 log-rotation-frequency:8h \
 log-rotation-policy:periodic log-min-size:1M log-rotation-start-time:0300
```

Directory Proxy Server ログの削除

Directory Proxy Server では、時間、サイズ、またはディスクの空き容量 (デフォルト) に基づいたログの削除を設定できます。これらの削除ポリシーの詳細は、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Log File Deletion」を参照してください。

次の手順では、アクセスログのログの削除を設定します。エラーログのログの削除を設定するときも同じコマンドを使います。ただし、 `access` を `error` に置き換えます。

▼ 時間に基づいたアクセスログとエラーログの削除を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ログファイルの最長有効期間を指定します。

  ```
  $ dpconf set-access-log-prop -h host -p port max-age:duration
  ```

  ここで、*duration* には、日 (`d`)、週 (`w`)、または月 (`M`) の単位を指定します。たとえば、5 日前より古いバックアップログファイルを削除するには、次のコマンドを使用します。

  ```
  $ dpconf set-access-log-prop -h host1 -p 1389 max-age:5d
  ```

▼ ファイルサイズに基づいたアクセスログとエラーログの削除を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- ログファイルの最大サイズを指定します。

  ```
  $ dpconf set-access-log-prop -h host -p port max-size:memory-size
  ```

  たとえば、1M バイトより大きいバックアップログファイルを削除するには、次のコマンドを使用します。

  ```
  $ dpconf set-access-log-prop -h host1 -p 1389 max-size:1M
  ```

▼ ディスクの空き容量に基づいたアクセスログとエラーログの削除を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

- 使用可能な最小ディスク容量を指定します。

  ```
  $ dpconf set-access-log-prop -h host -p port min-free-disk-space-size:memory-size
  ```

  たとえば、使用可能なディスク容量が 2M バイトより少なくなったときにバックアップログファイルを削除するには、次のコマンドを使用します。

  ```
  $ dpconf set-access-log-prop -h host1 -p 1389 min-free-disk-space-size:2M
  ```

syslogd デーモンに対するアラートのログ

この節では、syslogd デーモンに対するアラートメッセージのログを設定する方法と、syslog アラートを受け付けるようにオペレーティングシステムを設定する方法について説明します。

▼ syslogd デーモンに対するアラートをログに記録するように Directory Proxy Server を設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. (省略可能) システムログアラートのプロパティーの現在の値を表示します。

```
$ dpconf get-server-prop -h host -p port syslog-alerts-enabled \
 syslog-alerts-facility syslog-alerts-host
```

システムログアラートのデフォルトプロパティーは、次のとおりです。

```
syslog-alerts-enabled   :  false
syslog-alerts-facility  :  USER
syslog-alerts-host      :  localhost
```

`syslog-alerts-host` プロパティーは、メッセージの送信先である `syslogd` デーモンのホスト名を定義します。`syslog-alerts-facility` プロパティーは読み取り専用であり、メッセージをシステムログ内の `user` カテゴリに送信します。

2. `syslogd` デーモンにアラートメッセージのログを記録できるようにします。

```
$ dpconf set-server-prop -h host -p port syslog-alerts-enabled:true
```

3. (省略可能) アラートメッセージを別のホスト上の `syslogd` デーモンに送信します。

```
$ dpconf set-server-prop -h host -p port syslog-alerts-host:hostname
```

syslog アラートを受け付けるオペレーティング システムの設定

この節では、`syslog` アラートを受け付けるように Solaris™、Linux、および HP-UX オペレーティング システムを設定する手順を示します。

▼ syslog アラートを受け付けるように Solaris OS を設定する

1. 適切な機能を `syslog` 設定ファイルに追加します。

たとえば、すべてのアラートを `USER` 機能を使用して格納するには、`/etc/syslog.conf` に次の行を追加します。

```
user.info       /var/adm/info
```

ここでは、例として、`/var/adm/info` をメッセージの格納先になるローカルディレクトリとします。続行する前に、`/var/adm/info` が存在することを確認します。

2 syslogd デーモンを再起動します。

a. **Solaris 8** および **9** の場合は、次のように入力して syslogd を再起動します。

```
$ /etc/init.d/syslog stop | start
```

b. **Solaris 10** の場合は、次のように入力して syslogd を再起動します。

```
$ svcadm restart system/system-log
```

3 メッセージが syslog に記録されることを確認します。

```
$ logger -p user.info "Test message"
$ cat /var/adm/info
 Jun 19 17:18:38 host user: [ID 12345 user.info] Test message
```

▼ syslog アラートを受け付けるように **Linux** を設定する

1 適切な機能を syslog 設定ファイルに追加します。

たとえば、すべてのアラートを USER 機能を使用して格納するには、/etc/syslog.conf に次の行を追加します。

```
user.info       /var/adm/info
```

ここでは、例として、/var/adm/info をメッセージの格納先になるローカルディレクトリとします。続行する前に、/var/adm/info が存在することを確認します。

2 - r オプションを指定して実行するように、syslogd デーモンを設定します。

このオプションを使えば、syslogd でネットワークからの接続を受け付け可能になります。デフォルトでは、-r オプションは設定されていません。

-r オプションを設定するには、/etc/sysconfig/syslog に次の行を追加します。

```
SYSLOGD_OPTIONS="-m 0 -r"
```

/etc/sysconfig/syslog が存在しない場合は、同じ行を /etc/init.d/syslog に追加します。

3 syslogd デーモンを再起動します。

```
$ /etc/init.d/syslog stop | start
```

4 メッセージが syslog に記録されることを確認します。

```
$ logger -p user.info "Test message"
$ cat /var/adm/info
 Jun 19 17:18:38 host user: [ID 12345 user.info] Test message
```

▼ syslog アラートを受け付けるように HP-UX を設定する

1 適切な機能を syslog 設定ファイルに追加します。

たとえば、すべてのアラートを USER 機能を使用して格納するには、/etc/syslog.conf に次の行を追加します。

```
user.info       /var/adm/info
```

ここでは、例として、/var/adm/info をメッセージの格納先になるローカルディレクトリとします。続行する前に、/var/adm/info が存在することを確認します。

2 syslogd デーモンを再起動します。

```
$ /sbin/init.d/syslogd stop | start
```

3 メッセージが syslog に記録されることを確認します。

```
$ logger -p user.info "Test message"
$ cat /var/adm/info
 Jun 19 17:18:38 host user: [ID 12345 user.info] Test message
```

Directory Proxy Server および Directory Server のアクセスログによるクライアント要求の追跡

クライアント要求のパスを追跡するには、要求が Directory Proxy Server のアクセスログと Directory Server のアクセスログに記録される方法を理解する必要があります。この節の内容を理解するには、最初に『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Tracking Client Requests Through Directory Proxy Server and Directory Server Access Logs」をお読みください。

▼ Directory Proxy Server から Directory Server を経由したクライアントアプリケーションへの操作を追跡する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1 Directory Server のアクセスログ内で追跡する操作の接続番号を見つけます。

たとえば、アクセスログ内の次の行は、接続番号が conn=12839 である op=2 という操作を示します。

```
[20/Jul/2006:18:01:49 -0500] conn=12839 op=2 msgId=4 - SRCH base="dc=example,dc=com"
 scope=2 filter="(objectClass=organizationalunit)" attrs=ALL
```

2 その接続についての **Directory Proxy Server** の接続情報を入手します。

この情報を入手するには、Directory Server のアクセスログを検索して、対応する接続番号を持つすべての操作を見つけます。たとえば、UNIX システムの場合、次のような `grep` コマンドを実行して、Directory Server のアクセスログ内の接続 `conn=12839` に対応するすべての行を見つけます。

```
$ grep conn=12839 access
```

最初の LDAP 接続を示す行が探している行であり、次のようになります。

```
[19/Jul/2006:16:32:51 -0500] conn=12839 op=-1 msgId=-1 - fd=27 slot=27
 LDAP connection from 129.153.160.175:57153 to 129.153.160.175
```

前述の行は、**129.153.160.175:57153** から Directory Server への LDAP 接続があることを示しています。ポート番号 (**57153**) は、Directory Proxy Server のアクセスログで接続情報を調べる際に必要となります。このポート番号を使って、Directory Proxy Server のログ内で対応する接続を見つけることができ、この接続からクライアント情報を見つけることができます。

接続が最初に確立されてからログファイルのローテーションが行われている場合は、アーカイブされたログファイルと現在のアクセスログファイルを検索する必要があります。

3 **Directory Proxy Server** のアクセスログ内で対応する接続を見つけます。

この情報を入手するには、Directory Proxy Server のアクセスログを検索して、対応するポート番号を持つすべての操作を見つけます。

ログファイル内で、同じポート番号を持つ複数のエントリが見つかることがあります。正しいエントリを確実に見つけるには、Directory Server のログを検索したときに見つかったエントリに記載されていたタイムスタンプも条件に含めて検索します。

たとえば、UNIX システムでは、次のように `grep` コマンドを実行して、Directory Server のログで見つけたエントリのタイムスタンプおよびポート番号に対応する接続エントリを見つけます。

```
$ grep 19/Jul/2006:16:32 access | grep 57153
```

サーバー間でのわずかな時間のずれを考慮に入れて、検索条件にタイムスタンプを含めるときには、秒の値は除きます。

Directory Proxy Server のログ内の対応する行は、次のようになります。

```
[19/Jul/2006:16:32:51 -0500] - SERVER_OP  - INFO  - Created BIND  LDAP connection
 s_conn=sunds-d1m1-9389:34 client=0.0.0.0:57153
 server=idm160.central.sun.com:9389 main
```

この行は、Directory Proxy Server で `s_conn=sunds-d1m1-9389:34` への `BIND` 接続が作成されたことを示します。Directory Proxy Server は、自分自身を TCP ポート `57153` 上のクライアント `client=0.0.0.0` として識別します。

ログのこの行から得られる重要な情報は、サーバー ID とポート番号 (**s_conn=sunds-d1m1-9389:34**) です。

4 前述の手順で識別されたサーバー **ID** とポート番号に対応するすべての操作を見つけます。

この情報を入手するには、Directory Proxy Server のアクセスログで、対応するサーバー ID とポート番号を持つすべての操作をを検索します。

たとえば、UNIX システムでは、次のような `grep` コマンドを実行して、前述の手順で見つけたサーバー ID に対応する操作を見つけます。

```
$ grep s_conn=sunds-d1m1-9389:34 access
```

この例では、関連する操作が数日にまたがっている可能性があるため、タイムスタンプを用いた検索は有効ではないかもしれません。ただし、検索によって返される操作が正しいものであることを確認する必要があります。複数の `Create` 接続ステートメントがある場合は、必ず、Directory Server のログで見つけた Create 接続ステートメントと合致するものを見つけてください。そのためには、手順 1 で見つかったエントリのタイムスタンプと同じタイムスタンプのエントリを見つけます。

Directory Proxy Server のアクセスログの次の抜粋は、`s_conn=sunds-d1m1-9389:34` に対して返されたすべての操作を示します。

```
[19/Jul/2006:16:32:51 -0500] - SERVER_OP  - INFO - Created BIND LDAP connection
 s_conn=sunds-d1m1-9389:34 client=0.0.0.0:57153 server=idm160.central.sun.com:9389 main
[20/Jul/2006:18:01:49 -0500] - SERVER_OP  - INFO  - conn=31 op=0 BIND dn="cn=directory manager"
 method="SIMPLE" s_msgid=3 s_conn=sunds-d1m1-9389:34
[20/Jul/2006:18:01:49 -0500] - SERVER_OP  - INFO  - conn=31 op=0 BIND RESPONSE err=0 msg=""
 s_conn=sunds-d1m1-9389:34
[20/Jul/2006:18:01:49 -0500] - SERVER_OP  - INFO  - conn=31 op=1 SEARCH base="dc=example,dc=com"
 scope=2 s_msgid=4 s_conn=sunds-d1m1-9389:34
[20/Jul/2006:18:01:49 -0500] - SERVER_OP  - INFO  - conn=31 op=1 SEARCH RESPONSE err=0 msg=""
 nentries=1 s_conn=sunds-d1m1-9389:34
```

この情報により、Directory Proxy Server のこの検索操作に対する接続 ID が `31` (`conn=31`) であることがわかります。

5 前述の手順で見つかった接続 **ID** に対応する、クライアント接続 **IP** アドレスを見つけます。

この情報を入手するには、Directory Proxy Server のアクセスログで、正しい接続 ID とタイムスタンプを持つすべての操作を検索します。検索時には、手順 1 で検索したステートメントのタイムスタンプを使用します。

たとえば、UNIX システムでは、次のような `grep` コマンドを実行して、クライアント接続 IP アドレスを見つけます。

```
$ grep "20/Jul/2006:18:01" access | grep conn=31
```

目的の行は次のようになります。

```
[20/Jul/2006:18:01:49 -0500] - CONNECT - INFO  - conn=31 client=129.150.64.156:2031
server=0.0.0.0:11389 protocol=LDAP
```

6 前述の手順で見つかった IP アドレスの所有者を確認します。

この情報により、Directory Server 上で行われた操作をだれが担当したかを、正確に証明できます。

◆ ◆ ◆ 第 28 章

Directory Proxy Server の監視とアラート

監視により、Directory Proxy Server とデータソースの障害が検出されます。

Directory Proxy Server の監視フレームワークの詳細と、cn=monitor エントリの詳細なレイアウトについては、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Monitoring Directory Proxy Server」を参照してください。この章の内容は次のとおりです。

Directory Proxy Server に関する監視データの取得

Directory Proxy Server に関する監視データを取得するには、cn=monitor エントリを使用します。このエントリは、Directory Proxy Server によって、ローカルのメモリー内データベース内で管理されます。cn=monitor エントリで LDAP 検索を実行することで、cn=monitor の下にある属性を取得できます。このエントリを検索するには、Proxy Manager としてバインドする必要があります。

監視データを取得するための JVM の使用については、508 ページの「JVM による Directory Proxy Server についての監視データの取得」を参照してください。

データソースに関する監視データの取得

Directory Proxy Server でデータソースの健全性を監視する方法については、『Sun Java System Directory Server Enterprise Edition 6.1 Reference』の「Monitoring Data Sources」を参照してください。この節では、データソースの監視を設定する方法について説明します。

▼ エラーを待機することでデータソースを監視する

このタイプの監視では、Directory Proxy Server は、Directory Proxy Server とデータソース間のトラフィックのエラーを待機します。このタイプの監視をリアクティブな監視といいます。これは、Directory Proxy Server が、エラーが検出されると反応を示しますが、データソースを積極的にはテストしないためです。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースの監視モードを `reactive` に設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource monitoring-mode:reactive
```

2 **506 ページの「Directory Proxy Server に対する管理アラートの設定」**で説明するように、エラーが検出された場合、またはデータソースがオフラインまたはオンラインになる場合に送信されるアラートを設定します。

▼ 専用接続を定期的に確立することでデータソースを監視する

Directory Proxy Server では、指定した時間内にデータソースへの要求やデータソースからの応答がなかった場合、データソースへの専用接続が作成されます。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースの監視モードを `proactive` に設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource monitoring-mode:proactive
```

2 **Directory Proxy Server** が、データソースからの活動がないことを検出してから専用接続を確立するまでの、最長時間を設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource \
 monitoring-inactivity-timeout:time
```

デフォルトでは、非活動タイムアウトは 120 秒です。

3 506 ページの「Directory Proxy Server に対する管理アラートの設定」で説明するように、データベースがオフラインまたはオンラインとして検出された場合に送信されるアラートを設定します。

▼ 確立された接続をテストすることでデータソースを監視する

このタイプの監視では、Directory Proxy Server は、各データソースへの各接続で、定期的な間隔で検索を行います。このようにして Directory Proxy Server では、閉じた接続が検出され、停止しているために接続がドロップすることがないようにします。

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 データソースの監視モードを `proactive` に設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource monitoring-mode:proactive
```

2 **Directory Proxy Server** で実行する、監視する検索要求を設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource \
  monitoring-bind-timeout:timeout monitoring-entry-dn:dn \
  monitoring-search-filter:filter monitoring-entry-timeout:timeout
```

検索要求では次のプロパティーが使用されます。

`monitoring-bind-timeout`	Directory Proxy Server がデータソースへの接続の確立を待機する時間の長さ。デフォルトでは、このプロパティーの値は 5 秒です。
`monitoring-entry-dn`	検索要求にあるターゲットエントリの DN。デフォルトでは、このプロパティーはルート DSE エントリ (`""`) です。
`monitoring-search-filter`	検索フィルタ。
`monitoring-entry-timeout`	Directory Proxy Server が検索応答を待機する時間の長さ。デフォルトでは、このプロパティーの値は 5 秒です。

3 ポーリング間隔を設定します。

```
$ dpconf set-ldap-data-source-prop -h host -p port datasource monitoring-interval:interval
```

接続がダウンすると、Directory Proxy Server が、その復旧を検出するために、この間隔で接続をポーリングします。デフォルトでは、監視間隔は 30 秒です。

4 506 ページの「Directory Proxy Server に対する管理アラートの設定」で説明するように、データベースがオフラインまたはオンラインとして検出された場合に送信されるアラートを設定します。

Directory Proxy Server に対する管理アラートの設定

管理アラートの設定方法については、次の手順を参照してください。

▼ 管理アラートを有効にする

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 有効になっているアラートを表示します。

```
% dpconf get-server-prop -h host -p port enabled-admin-alerts
```

2 1 つまたは複数の管理アラートを有効にします。

```
% dpconf set-server-prop -h host -p port enabled-admin-alerts:alert1 \
  [enabled-admin-alerts:alert2 ...]
```

たとえば、使用可能なすべてのアラートを有効にするには、次のコマンドを実行します。

```
% dpconf set-server-prop -h host -p port \
 enabled-admin-alerts:error-configuration-reload-failure-with-impact \
 enabled-admin-alerts:error-server-shutdown-abrupt \
 enabled-admin-alerts:info-configuration-reload \
 enabled-admin-alerts:info-data-source-available \
 enabled-admin-alerts:info-server-shutdown-clean \
 enabled-admin-alerts:info-server-startup \
 enabled-admin-alerts:warning-configuration-reload-failure-no-impact \
 enabled-admin-alerts:warning-data-source-unavailable \
 enabled-admin-alerts:warning-data-sources-inconsistent \
 enabled-admin-alerts:warning-listener-unavailable
```

すべてのアラートを無効にするには、次のコマンドを実行します。

```
% dpconf set-server-prop -h host -p port enabled-admin-alerts:none
```

有効なアラートの既存リストにアラートを追加するには、次のコマンドを実行します。

```
% dpconf set-server-prop -h host -p port enabled-admin-alerts+:alert-name
```

有効なアラートの既存リストからアラートを削除するには、次のコマンドを実行します。

```
% dpconf set-server-prop -h host -p port enabled-admin-alerts-:alert-name
```

デフォルトでは、有効なアラートはありません。

参照　詳細は、`enabled-admin-alerts`(5dpconf) を参照してください。

▼ Syslog に送信する管理アラートを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 506 ページの「管理アラートを有効にする」で説明するように、`syslog` デーモンに送信されるアラートを選択します。
2. `syslog` デーモンに送信されるアラートを有効にします。

   ```
   $ dpconf set-server-prop -h host -p port syslog-alerts-enabled:true
   ```

   すべてのアラートは、`USER` の機能によって `syslog` に送信されます。
3. アラートの送信先である、`syslog` デーモンのホスト名を設定します。

   ```
   $ dpconf set-server-prop -h host -p port syslog_hostname:hostname
   ```

▼ 電子メールに送信する管理アラートを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1. 506 ページの「管理アラートを有効にする」で説明するように、`syslog` に送信されるアラートを選択します。

2 電子メールのアドレスと特性を設定します。

```
$ dpconf set-server-prop -h host -p port email-alerts-smtp-host:host-name \
  email-alerts-smtp-port:port-number \
  email-alerts-message-from-address:sender-email-address \
  email-alerts-message-to-address:receiver-email-address \
  [email-alerts-message-to-address:receiver-email-address ...] \
  email-alerts-message-subject:email-subject
```

3 電子メールに送信されるアラートを有効にします。

```
$ dpconf set-server-prop -h host -p port email-alerts-enabled:true
```

4 (省略可能) 電子メールのなかにアラートコードを組み込むフラグを設定します。

```
$ dpconf set-server-prop -h host -p port \
 email-alerts-message-subject-includes-alert-code:true
```

▼ スクリプトを実行するように管理アラートを設定する

DSCC を使用してこの作業を実行できます。詳細は、45 ページの「Directory Service Control Center のインタフェース」と DSCC のオンラインヘルプを参照してください。

1 506 ページの「管理アラートを有効にする」で説明するように、syslog に送信されるアラートを選択します。

2 アラートでスクリプトを実行できるようにします。

```
$ dpconf set-server-prop -h host -p port scriptable-alerts-enabled:true
```

3 実行するスクリプトの名前を設定します。

```
$ dpconf set-server-prop -h host -p port scriptable-alerts-command:script-name
```

JVM による Directory Proxy Server についての監視データの取得

Directory Proxy Server は Java 仮想マシン (JVM) の内部で実行され、JVM マシンのメモリーに依存します。Directory Proxy Server が確実に正しく実行されるようにするには、JVM マシンのメモリー消費量を監視する必要があります。

JVM マシン用にパラメータを調整する方法については、『Sun Java System Directory Server Enterprise Edition 6.1 配備計画ガイド』の「Directory Proxy Server のハードウェアサイジング」を参照してください。

デフォルトでは、JVM マシンのヒープサイズは 250M バイトです。Directory Proxy Server に十分な物理メモリーがないと、ヒープサイズが 250M バイトより少なくなることがあります。

Directory Proxy Server が実行中の場合は、JVM マシンのヒープサイズを監視して、メモリー不足にならないようにすることができます。このためには、Java Development Kit (JDK) とともに配布される標準ツールを使用します。これらのツールは、次のディレクトリ内にあります。`$JAVA_HOME/bin/jps` および `$JAVA_HOME/bin/jstat`。

▼ JVM のヒープサイズを表示する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

- **JVM のヒープサイズを表示します。**

```
$ dpadm get-flags instance-path jvm-args
jvm-args: -Xms250M  -Xmx250M
```

▼ Directory Proxy Server の実行時に JVM のヒープサイズを監視する

DSCC を使用してこの作業を実行することはできません。この手順で説明しているように、コマンド行を使用してください。

1. **Directory Proxy Server のインスタンスの PID を表示します。**

```
$ jps
```

2. **JVM マシンで使われるメモリーを表示します。**

```
$ jstat -gcutil PID
```

   - ゼロ列が 100% に近い場合、JVM マシンに十分なメモリーはありません。
   - FGC は、フルガベージコレクション (GC) のイベントの数です。ガベージコレクションには膨張性があります。
   - GCT (ガベージコレクション時間) は、GC が費やす時間の量です。

索引

E

G

I

J

K

L

M

N

R

S

T

U

V

あ

い