インストールおよびチューニングガイド

Sun™ ONE Directory Server

Version 5.2

816-6847-10
2003 年 6 月

目 次

本書について

Sun™ ONE Directory Server 5.2 は、業界標準の LDAP (Lightweight Directory Access Protocol) に基づいたスケーラブルで強力な分散型ディレクトリサービスです。Sun ONE Directory Server は、Sun の標準に基づくソフトウェアのビジョン、アーキテクチャ、プラットフォーム、および専門技術の総称である Sun ONE (Sun Open Net Environment) の一部として、Services On Demand の構築や導入を目的としたソフトウェアです。

Sun ONE Directory Server は、社内イントラネットや、取引先とのエクストラネット、顧客との窓口となる公共のインターネット上で使用できる、集中・分散型のデータリポジトリを構築するための基盤となります。

このマニュアルの目的

このマニュアルでは、実際の運用環境で使用するための Directory Server のインストール方法について説明します。実際の運用環境において、Directory Server が高パフォーマンスを発揮するには、多くの設定とチューニングを行う必要があります。

Directory Server を評価目的のためだけにインストールし、実際の運用環境で使用しない場合は、第 1 章「Sun ONE Directory Server のインストール」だけをお読みください。

前提条件

導入目的を明確にしてから、実際の運用環境で使用するために Directory Server をインストールしてください。詳細は、『Sun ONE Directory Server Deployment Guide』を参照してください。

表記規則

ここでは、このマニュアルで使用する表記規則について説明します。

クーリエ (固定スペース) : このフォントは、属性名やオブジェクトクラス名を本文中で使用する場合などに、リテラル文字列で使用します。また、URL、ファイル名、および例にも使用します。

イタリック : このフォントは、強調、新出用語、および可変部分 (パス名など実際の値に置き換える必要がある文字列) で使用します。

大なり記号 (>) は、一連のメニューまたはサブメニューの項目を選択するときの区切り文字として使用します。たとえば、「オブジェクト」>「新規」>「ユーザー」は、「オブジェクト」メニューの「新規」サブメニューにある「ユーザー」項目を選択することを意味します。

注 「注」、「注意」、および「ヒント」は、重要な条件または制限を強調するためのものです。必ずこれらの注意事項を読んでから、作業を続けるようにしてください。

デフォルトパスおよびファイル名

Sun ONE Directory Server 製品マニュアルで使用されるパスおよびファイル名の例は、すべて次の 2 つの形式のいずれかです。

- *ServerRoot*/... : *ServerRoot* は Sun ONE Directory Server 製品のディレクトリです。このパスには、Directory Server と Sun ONE 管理サーバーの共有バイナリファイル、およびコマンド行ツールが含まれます。

  実際の *ServerRoot* パスは、使用するプラットフォーム、インストール内容、および設定によって異なります。デフォルトパスは、表 1 に示すように、製品のプラットフォームおよびパッケージによって異なります。

- *ServerRoot*/slapd-*serverID*/... : *serverID* は、インストールおよび設定時に定義した Directory Server インスタンスの名前です。このパスには、特定のインスタンスに固有のデータベースおよび設定ファイルが含まれます。

注 このマニュアルでは、パスを UNIX 形式のスラッシュ (/) で記述し、コマンドをファイル拡張子を付けずに記述します。Windows ベースの Sun ONE Directory Server を使用している場合は、同等の Windows 形式の円記号 (¥) に読み替えてください。通常、Windows システムの実行ファイルには、UNIX 形式の場合と同じ名前に `.exe` または `.bat` という拡張子が付いています。

表 1　　デフォルトの *ServerRoot* パス

製品のバージョン	*ServerRoot* パス
Solaris パッケージ[1]	/var/mps/serverroot : 設定後、このディレクトリには次の場所へのリンクが含まれる • /etc/ds/v5.2 (スタティック設定ファイル) • /usr/admserv/mps/admin (Sun ONE 管理サーバーバイナリ) • /usr/admserv/mps/console (サーバーコンソールバイナリ) • /usr/ds/v5.2 (Directory Server バイナリ)
Solaris およびその他の UNIX システム上の圧縮アーカイブインストール	/var/Sun/mps
Windows システム上の Zip インストール	C:¥Program Files¥Sun¥MPS

1. Solaris オペレーティング環境上で作業している時に、インストールされている Sun ONE Directory Server のバージョンが不確かな場合は、pkginfo コマンドを使用して SUNWdsvu などのキーパッケージの有無を確認してください。たとえば、次のようにします。pkginfo | grep SUNWdsvu

Directory Server インスタンスは、*ServerRoot*/slapd-*serverID*/ の下に配置されます。ここで、*serverID* は作成するインスタンスに指定されたサーバー識別子を表します。たとえば、Directory Server の名前を dirserv にした場合、実際のパスは表 2 に示されるようになります。Directory Server インスタンスを別の場所に作成した場合は、それに対応してパスを変更する必要があります。

表 2　　デフォルトの dirserv インスタンスの場所の例

製品のバージョン	インスタンスの場所
Solaris パッケージ	/var/mps/serverroot/slapd-dirserv
Solaris およびその他の UNIX システム上の圧縮アーカイブインストール	/usr/Sun/mps/slapd-dirserv
Windows システム上の Zip インストール	C:¥Program Files¥Sun¥MPS¥slapd-dirserv

Directory Server ツールのダウンロード

サポートされているプラットフォームの一部には、Directory Server にアクセスするためのネイティブツールが用意されています。LDAP ディレクトリサーバーをテストおよび保守するためのその他のツールを入手するには、Sun ONE Directory Server Resource Kit (DSRK) をダウンロードしてください。このソフトウェアは、次の Web サイトから入手できます。

```
http://wwws.sun.com/software/download/
```

DSRK ツールのインストール手順およびリファレンスマニュアルは、『Sun ONE Directory Server Resource Kit Tools Reference』を参照してください。

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場合には、iPlanet Directory SDK for C および iPlanet Directory SDK for Java を同じ Web サイトからダウンロードすることもできます。

さらに、JNDI (Java Naming and Directory Interface) テクノロジは、Java アプリケーションから LDAP および DSML v2 を使用して Directory Server へのアクセスをサポートします。JNDI に関する情報は次の Web サイトから入手できます。

```
http://java.sun.com/products/jndi/
```

JNDI チュートリアルには、JNDI の使用方法に関する詳細な説明と事例が記載されています。JNDI チュートリアルは次の Web サイトから入手できます。

```
http://java.sun.com/products/jndi/tutorial/
```

推奨参考文献

Sun ONE Directory Server 製品マニュアルには、HTML と PDF の両方の形式で提供される次のドキュメントが含まれています。

- 『Sun ONE Directory Server Getting Started Guide』: Sun ONE Directory Server 5.2 の多くの重要な機能を要約して説明します。
- 『Sun ONE Directory Server Deployment Guide』: ディレクトリトポロジ、データ構造、セキュリティ、および監視の設計方法について説明し、導入事例を検討します。
- 『Sun ONE Directory Server インストールおよびチューニングガイド』: インストールおよびアップグレード手順について説明し、Directory Server のパフォーマンスの最適化に関するヒントを提供します。
- 『Sun ONE Directory Server 管理ガイド』: コンソールおよびコマンド行を使用して、ディレクトリの内容を管理し、Directory Server のすべての機能を設定する手順について説明します。

- 『Sun ONE Directory Server Reference Manual』: Directory Server の設定パラメータ、コマンド、ファイル、エラーメッセージ、およびスキーマの詳細について説明します。
- 『Sun ONE Directory Server Plug-In API Programming Guide』: Directory Server プラグインの開発方法について説明します。
- 『Sun ONE Directory Server Plug-In API Reference』: Directory Server プラグイン API のデータ構造と機能の詳細について説明します。
- 『Sun ONE Server Console Server Management Guide』: Sun ONE 管理サーバーおよび Java ベースのコンソールを使用してサーバーを管理する方法について説明します。
- 『Sun ONE Directory Server Resource Kit Tools Reference』: 多くの有用なツールを含む Sun ONE Directory Server Resource Kit のインストールと機能について説明します。

その他の有用な情報は、次の Web サイトから入手できます。

- 製品オンラインマニュアル : `http://docs.sun.com/`
- Sun ソフトウェア : `http://wwws.sun.com/software/`
- Sun ONE サービス : `http://www.sun.com/service/sunps/sunone/`
- Sun サポートサービス : `http://www.sun.com/service/support/`
- Sun ONE 開発者向けサイト : `http://sunonedev.sun.com/`
- トレーニング : `http://suned.sun.com/`

第 1 部

インストール

第 1 章

Sun ONE Directory Server のインストール

この章では、Sun ONE Directory Server の初期インストールとアンインストールについての指針を示します。この章は、次の節で構成されています。

- 始める前に
- インストール
- アンインストール
- トラブルシューティング

始める前に

実際の運用環境で使用するにあたって Directory Server をインストールする前に、ディレクトリサービスを実行するためにシステムで最小限必要な要件が満たされ、設定されていることを確認します。少なくとも、『Sun ONE Directory Server Deployment Guide』で説明されている概念をよく理解しておいてください。

注	最高のパフォーマンスを得るには、このガイドで説明するチューニングと設定の指示にも従ってください。

ディレクトリサービスの導入計画

次の手順を実行します。ただし、基盤となるプラットフォームに関連する作業に関しては、オペレーティングシステムのマニュアルを参照してください。

1. ディレクトリサービスの導入を計画します。

   手順は、『Sun ONE Directory Server Deployment Guide』を参照してください。

2. ディレクトリサービスの導入により複数のディレクトリにインストールするためのサーバー設定、ユーザー、およびグループの集中管理を可能にするには、設定ディレクトリとユーザーディレクトリの位置を決めます。

   設定ディレクトリまたは Configuration Directory Server (CDS) には、Directory Server 自体の設定方法についての情報が格納されます。通常このディレクトリは最初にインストールされ、後続の各サーバーがこのディレクトリに登録されます。単一の設定ディレクトリを使用することによって、全サーバーを中央管理することができます。

   ユーザーディレクトリには、ディレクトリサービスにアクセスするユーザーやグループのエントリが格納されます。通常ユーザーディレクトリはネットワークドメインで一意であり、ほかのサーバーがユーザーやグループの情報を得るためにアクセスします。単一のユーザーディレクトリを使用することによって、ユーザーやグループを集中管理できます。

   小規模の導入では、設定、ユーザー、およびその他のディレクトリを同じディレクトリインスタンスにインストールすることが可能です。大規模の導入では、設定とユーザーのディレクトリを別々のサーバーに配置することを検討してください。

   設定、ユーザーおよびグループのデータを保存する適切な位置についての詳細は、『Sun ONE Server Console Server Management Guide』を参照してください。

3. 表 1-1 にまとめるように、サポート対象のアーキテクチャ上でサポート対象のプラットフォームがホストシステムで実行されていることを確認します。

表 1-1 サポート対象のプラットフォームとアーキテクチャ

プラットフォーム	アーキテクチャ
Sun Solaris™ 9 オペレーティング環境	SPARC プロセッサ 32 ビットまたは 64 ビットモード
	サポート対象の x86 プラットフォーム
Sun Solaris 8 オペレーティング環境	UltraSPARC プロセッサ 32 ビットまたは 64 ビットモード
Sun Linux 5.0	Sun LX50 サーバー
Hewlett Packard HP-UX 11i	PA-RISC 2.0 プロセッサ、32 ビットまたは 64 ビットモード
IBM AIX 5.1	PowerPC プロセッサ
Microsoft Windows 2000 Server SP 3	Pentium II 以降の IA-32 プロセッサ
Microsoft Windows 2000 Advanced Server SP 3	
Red Hat Linux 7.2	Pentium II 以降の IA-32 プロセッサ

4. ホストシステムが、少なくとも 82 ページの表 4-1 にまとめた最小のディスク空き容量とメモリの要件を満たしていることを確認します。
5. ホストシステムへの物理アクセスを制限します。
6. ホストシステムがスタティック IP アドレスを使用していることを確認します。
7. Directory Server インスタンス自体がネットワークのネームサービスを提供していない場合、または Directory Server の導入によりリモート管理を可能にするには、ネームサービスと、そのホストのドメイン名が適切に構成されていることを確認します。

Directory Server ソフトウェアの取得

17 ページの「ディレクトリサービスの導入計画」で示した手順を実行したら、次の手順を行います。

1. ソフトウェアを解凍するための `unzip` ユーティリティがインストールされていることを確認します。
2. ソフトウェアをダウンロードします。このマニュアルの作成時点では、次の場所からダウンロードできます。

   `http://wwws.sun.com/software/download/`
3. Directory Server のインストール先以外のディレクトリにソフトウェアを解凍します。

インストール

どの Directory Server インストール手順に従うかは、ユーザーの導入要件によって異なります。導入要件を念頭に置き、次の該当する節に従ってインストールを進めてください。

- インストールするソフトウェアの決定
- インストール方法の決定
- インストールに必要な情報の準備
- Solaris システム上でのインストール
- その他の UNIX システム上でのインストール
- Windows システム上でのインストール

インストールするソフトウェアの決定

インストールするソフトウェアを決定する前に、いくつかの候補を評価する必要があります。次の質問を検討してください。

- 大規模なキャッシュ機能を使って大規模な導入を行う必要があるか

  必要がある場合は、Directory Server を 64 ビットプロセスとして実行できるプラットフォームを使用して、64 ビットバージョンのソフトウェアのインストールを検討してください。

  Directory Server の導入規模が比較的小さい場合 (データベースサイズが 500M バイト未満である場合)、64 ビットバージョンをサポートするプラットフォームを使用するときでも、32 ビットバージョンのみをインストールすることを検討してください。

- Directory Server の管理をグラフィカルユーザーインタフェースを通じて行うか

  グラフィカルユーザーインタフェースを使用する場合は、Sun ONE サーバーコンソールと Sun ONE 管理サーバーをインストールしてください。

  Directory Server の管理をコマンド行インタフェースを通じてのみ行う予定であれば、コンソールと管理サーバーをインストールしなくてもかまいません。

  グラフィカルユーザーインタフェースによるリモート管理用としてシステムを使用する予定であれば、コンソールと管理サーバーのみをインストールすることも可能です。

- Directory Server を Sun Cluster ソフトウェア上に導入するか

  導入する場合、付録 C「Sun Cluster HA for Directory Server のインストール」の手順を参照してください。

インストール方法の決定

導入に最適なパッケージや、対話式にインストールするかどうかを決定する場合も、いくつかの候補を評価する必要があります。次の質問を検討してください。

- Solaris システム管理プロセスとより緊密に統合するか。同一システム上の複数の Sun ONE サーバー間でコンポーネントを共有するか

  統合および共有する場合、Solaris パッケージの使用を検討してください。

- 最初にスーパーユーザーにならずにインストールするか。同一システム上に Directory Server バイナリの複数の独立したセットをインストールするか

  上記のようにインストールする場合、たとえ Solaris システム上であっても、圧縮アーカイブを使用したインストールを検討してください。

- Directory Server を評価する目的で短時間でインストールするか。このバージョンの Directory Server をインストールするのはこれが最初か

  このような場合は、対話式のインストールを検討してください。

- インストールスクリプトを作成するか。似たような設定で数多くのシステムをインストールするか

  このような場合は、サイレントインストールプロセスの使用を検討してください。

インストールに必要な情報の準備

必要な情報を前もって準備しておくと、インストールプロセスをすばやく完了できます。対話式インストールを実行する前に、インストールに必要な情報が記載されたワークシートの作成を検討してください。通常のインストール時に必要となる情報を表 1-2 にまとめます。

表 1-2 通常のインストール時に必要となる基礎情報

内容	例	設定値
管理ドメイン	`example.com`	
管理サーバーのポート番号	`5201`	
設定ディレクトリの管理者 ID	`admin`	
設定ディレクトリの管理者パスワード	`$3kReT4wD`	
Directory Manager の DN[1] (ディレクトリのスーパーユーザー)	`cn=Directory Manager`	
Directory Manager のパスワード (8 文字以上)	`#$8Yk$-%&`	

表 1-2　通常のインストール時に必要となる基礎情報 (続き)

内容	例	設定値
Directory Server のポート番号 (1 ～ 65535)[2]	389 (デフォルト LDAP) 636 (デフォルト LDAP/SSL)	
完全修飾ホスト識別名	dirserv.example.com	
(省略可) 既存の設定ディレクトリを使用する場合は、設定ディレクトリのホスト、ポート、バインド ID、およびパスワード	config.example.com 389 admin $3kReT4wD	
(省略可) 既存のユーザーディレクトリを使用する場合は、ユーザーディレクトリのホスト、ポート、バインド DN、パスワード、およびサフィックス	usergroup.example.com 389 cn=Directory Manager #$8Yk$-%& dc=example, dc=com	
サーバー ID (ピリオドと空白は使用不可)	dirserv	
サーバーサフィックス (ディレクトリ内容を保持するサフィックスを少なくとも 1 つ)	dc=example,dc=com	
ServerRoot (ソフトウェアのインストールディレクトリ。詳細は 10 ページの「デフォルトパスおよびファイル名」を参照) 既存の旧バージョンに上書きインストールしないこと Directory Server と同じ *ServerRoot* に Sun ONE Web サーバーをインストールしないこと (UNIX プラットフォーム) 空白は使用不可	/var/mps/serverroot /var/Sun/mps C:¥Program Files¥Sun¥MPS	
(UNIX プラットフォーム) サーバーのグループ ID[3] グループ ID 番号ではなく名前を使用すること	noaccess	
(UNIX プラットフォーム) サーバーのユーザー ID ユーザー ID 番号ではなく名前を使用すること	diruser	
(Windows) 管理者パスワード (省略可、ほかのプラットフォーム) スーパーユーザーのパスワード	システム管理者に問い合わせのこと	

1. DN は、UTF-8 エンコーディングで入力する必要があります。RFC 2253 を参照してください。ISO-8859-1 のような以前のエンコーディングはサポートされていません。
2. Internet Assigned Numbers Authority は 1024 より小さいポート番号を割り当てます。1024 より小さいポート番号を使用するために、スーパーユーザーとしてインストールしてください。
3. インストール手順での説明に従って、UNIX に適切なユーザーとグループを作成します。

ディレクトリ管理者アカウントと Directory Manager アカウントの情報を入力するときに、Directory Manager のアクセス権限は Directory Server のアクセス制御メカニズムを使用して管理されます。また、Directory Server のアクセス制御には、Directory Manager アカウントは必要としません。

サイレントインストールの設定ファイルには、これと類似した情報が含まれます。

Solaris システム上でのインストール

Directory Server ソフトウェアのインストール方法は、どのパッケージを使用するかや、インストールプログラムと対話するかどうかによって異なります。次の該当する節に従ってインストールを進めてください。

- Solaris パッケージを使用したインストールの準備
- Solaris パッケージを使用した対話式インストールの実行
- Solaris パッケージを使用したサイレントインストールの実行

- 圧縮アーカイブを使用したインストールの準備
- 圧縮アーカイブを使用した対話式インストールの実行
- 圧縮アーカイブを使用したサイレントインストールの実行

- インストールプロセスの完了

Directory Server を Sun Cluster システムにインストールする場合、付録 C「Sun Cluster HA for Directory Server のインストール」の手順に従ってください。

Solaris パッケージを使用したインストールの準備

1. (省略可) Directory Server のユーザーとグループのアカウントを作成します。

   Directory Server は、インストール中に指定したユーザーとグループとして実行されます。ディレクトリやシステム上のその他のリソースへ不正にアクセスされないようにアクセス権を設定します。詳細は、101 ページの「ユーザーとグループ (UNIX プラットフォームの場合)」を参照してください。

2. (省略可) xhost(1) コマンドを使用したディスプレイへのアクセスを許可します。

   DISPLAY 環境変数を適切に設定し、ディスプレイにアクセスできるユーザーとしてインストールを実行すると、インストールプログラムではデフォルトでグラフィカルユーザーインタフェースを表示します。

   インストールプログラムでグラフィカルユーザーインタフェースが表示できない場合は、コマンド行モードでインストールを開始します。

3. 表 1-3 に記載した必須パッケージがインストールされていることを確認します。さらに、すべての Solaris パッケージが基本システムにデフォルトでインストールされていることを確認します。

表 1-3 前提条件の Solaris パッケージ

パッケージ	内容	32 ビット Directory Server で必要	64 ビット Directory Server で必要
SUNWj3rt[1]	J2SDK 1.4 実行環境	はい	はい
SUNWzlib	Zip 圧縮ライブラリ	はい	はい
SUNWzlibx	Zip 圧縮ライブラリ (64 ビット)	いいえ	はい

1. Java Runtime Environment はバージョン 1.4.1 以降を使用することを強くお勧めします。

Solaris パッケージを使用した対話式インストールの実行

次の手順を実行します。

Solaris パッケージのインストール

Solaris パッケージは pkgadd(1M) ユーティリティを使用してインストールします。たとえばアップグレードを実行する場合、pkginfo(1) を使用して、すでにインストールされているパッケージを判別します。パッケージを複数のホストにインストールするときは、admin(4) で説明するインストールデフォルトファイルで、デフォルトのインストール動作を定義することができます。いずれの場合でも、すべてのパッケージは、同一の *basedir* を共有する必要があります。

ソフトウェアパッケージの処理についての詳細は、Solaris オペレーティング環境のシステム管理マニュアルを参照してください。

1. 表 1-4 または 表 1-5 に記載されたパッケージ一覧のすべてに目を通します。

表 1-4 提供されている Solaris パッケージ (SPARC プラットフォーム)

パッケージ	内容
SUNWasha	Sun ONE Administration Server Component for Sun Cluster

表 1-4　提供されている Solaris パッケージ (SPARC プラットフォーム) (続き)

パッケージ	内容
SUNWasvc	Sun ONE Administration Console
SUNWasvcp	Sun ONE Administration Server Console Plug-In
SUNWasvr	Sun ONE Administration Server (Root)
SUNWasvu	Sun ONE Administration Server (Usr)
SUNWdsha	Sun ONE Directory Server Component for Sun Cluster
SUNWdsvcp	Sun ONE Directory Server Console Plug-In
SUNWdsvh	Sun ONE Directory Server Heap Allocator (Solaris 8 systems only)
SUNWdsvhx	Sun ONE Directory Server Heap Allocator (64-bit, Solaris 8 systems only)
SUNWdsvpl	Sun ONE Directory Server PerLDAP modules
SUNWdsvr	Sun ONE Directory Server (Root)
SUNWdsvu	Sun ONE Directory Server (Usr)
SUNWdsvx	Sun ONE Directory Server (64-bit)
SUNWicu	International Components for Unicode User Files
SUNWicux	International Components for Unicode User Files (64-bit)
SUNWjss	Network Security Services for Java (JSS)
SUNWldk	LDAP C SDK
SUNWldkx	LDAP C SDK (64-bit)
SUNWpr	Netscape Portable Runtime Interface
SUNWprx	Netscape Portable Runtime Interface (64-bit)
SUNWsasl	Simple Authentication and Security Layer
SUNWsaslx	Simple Authentication and Security Layer (64-bit)
SUNWtls	Network Security Services
SUNWtlsx	Network Security Services (64-bit)

表 1-5　提供されている Solaris パッケージ (x86 プラットフォーム)

パッケージ	内容
SUNWasvc	Sun ONE Administration Console
SUNWasvcp	Sun ONE Administration Server Console Plug-In

表 1-5　提供されている Solaris パッケージ (x86 プラットフォーム) (続き)

パッケージ	内容
SUNWasvr	Sun ONE Administration Server (Root)
SUNWasvu	Sun ONE Administration Server (Usr)
SUNWdsvcp	Sun ONE Directory Server Console Plug-In
SUNWdsvpl	Sun ONE Directory Server PerLDAP modules
SUNWdsvr	Sun ONE Directory Server (Root)
SUNWdsvu	Sun ONE Directory Server (Usr)
SUNWicu	International Components for Unicode User Files
SUNWjss	Network Security Services for Java (JSS)
SUNWldk	LDAP C SDK
SUNWpr	Netscape Portable Runtime Interface
SUNWsasl	Simple Authentication and Security Layer
SUNWtls	Network Security Services

パッケージをインストールする際には、/var のように書き込み可能な *basedir* を使用することをお勧めします。パッケージを再配置すると、SUNWasvr と SUNWdsvr によって起動スクリプトと停止スクリプトが *basedir*/etc に配置されます。

2. インストールするパッケージを決定する際は、表 1-6 のヒントを参考にしてください。

表 1-6　インストールするパッケージ

設定	インストールするパッケージのリスト[1]
32 ビット Directory Server、管理サーバー、およびコンソール	SUNWascv、SUNWasvcp、SUNWasvr、SUNWasvu、SUNWdsvcp、SUNWdsvh、SUNWdsvpl、SUNWdsvr、SUNWdsvu、SUNWicu、SUNWjss、SUNWldk、SUNWpr、SUNWsasl、SUNWtls
32 ビット Directory Server のみ (コンソールなし)	SUNWasvu、SUNWdsvh、SUNWdsvpl、SUNWdsvr、SUNWdsvu、SUNWicu、SUNWjss、SUNWldk、SUNWpr、SSUNWsasl、SUNWtls

表 1-6 インストールするパッケージ (続き)

設定	インストールするパッケージのリスト[1]
64 ビット Directory Server、32 ビット管理サーバー、およびコンソール	SUNWascv、SUNWasvcp、SUNWasvr、SUNWasvu、SUNWdsvcp、SUNWdsvh、SUNWdsvhx、SUNWdsvpl、SUNWdsvr、SUNWdsvu、SUNWdsvx、SUNWicu、SUNWicux、SUNWjss、SUNWldk、SUNWldkx、SUNWpr、SUNWprx、SUNWsasl、SUNWsaslx、SUNWtls、SUNWtlsx
64 ビット Directory Server のみ (コンソールなし)	SUNWasvu、SUNWdsvh、SUNWdsvhx、SUNWdsvpl、SUNWdsvr、SUNWdsvu、SUNWdsvx、SUNWicu、SUNWicux、SUNWjss、SUNWldk、SUNWldkx、SUNWpr、SUNWprx、SUNWsasl、SUNWsaslx、SUNWtls、SUNWtlsx
クラスタノード	SUNWasha と SUNWdsha を追加
Sun ONE サーバーコンソールと管理サーバーのみ (Directory Server なし、リモート管理のみ)	SUNWasvc、SUNWasvcp、SUNWasvr、SUNWasvu、SUNWdsvcp、SUNWjss、SUNWldk、SUNWpr、SUNWsasl、SUNWtls

1. パッケージ SUNWdsvh (32 ビット) と SUNWdsvhx (64 ビット) が必要なのは、Solaris 8 システム上の Directory Server だけです。

3. 目的のパッケージがまだインストールされていないことを確認します。

   システム上にすでにインストールされているパッケージを再インストールしないでください。

4. スーパーユーザーになります。

5. pkgadd(1M) ユーティリティを使用して、システムに製品パッケージを転送します。

   パッケージ SUNWicu と SUNWicux は、Directory Server をインストールする Solaris システムのバージョンに応じてインストールします。

   また、SUNWpr、SUNWprx、SUNWsasl、SUNWsaslx、SUNWtls、および SUNWtlsx コンポーネントパッケージをインストールする方法と、これらのパッケージに適用するパッチの詳細は、「必須パッチのインストール」を参照してください。

6. pkgadd を終了する前に、すべての必須製品パッケージがインストールされていることを確認してください。

IPLT* Solaris パッケージを使用してインストールされた iPlanet Directory Server 5.1 をアップグレードすると、iPlanet Directory Server 5.1 の /usr/sbin/directoryserver コマンドの名前が /usr/sbin/directoryserver.51bak に変更されます。この変更されたコマンド名を使用して iPlanet Directory Server 5.1 を管理することが可能です。

必須パッチのインストール

Directory Server は、最近の修正プログラムを含めてアップデートされたパッケージ SUNWpr、SUNWprx、SUNWsasl、SUNWsaslx、SUNWtls、SUNWtlsx、および推奨システムパッチに依存します。

1. pkginfo(1) を -x オプションを指定して実行し、これらのパッケージのどれがシステムにインストールされているのかを判断します。表 1-7 に基づいて、使用するシステムに適したバージョンのパッケージがインストールされていることを具体的に確認します。

表 1-7 各コンポーネントの適切なバージョンとパッチ

システムのバージョンとアーキテクチャ	SUNWpr(x) のバージョン	SUNWsasl(x) のバージョン	SUNWtls(x) のバージョン	パッチ
Solaris 9 (SPARC プラットフォーム)	4.1.2 以降	2.01 以降	3.3.2 以降	114049、115342
Solaris 9 (x86 プラットフォーム)	4.1.3 以降	2.01 以降	3.3.3 以降	114050、115343
Solaris 8 (SPARC プラットフォーム)	4.1.2 以降	2.01 以降	3.3.2 以降	114045、115328

2. showrev(1M) を -p オプションを指定して実行し、表 1-7 に示した適切なパッチがプラットフォームにすでに適用されているかどうかを判断します。
3. コンポーネントのパッチを適用するかどうかを決定する際は、表 1-8 のヒントを参考にしてください。

表 1-8 コンポーネントのパッチを適用するかどうかのヒント

システムの状態	実行内容
パッケージがすでにインストールされていて、パッチもすでに適用されている	手順 4 に進む
パッケージはすでにインストールされているが、パッチがまだ適用されていない	Directory Server で提供されている、プラットフォームに適切なパッチを適用する
パッケージがまだインストールされていない	Directory Server で提供されているパッケージと適切なパッチをインストールする

4. スーパーユーザーとして次のコマンドを実行します。

   ```
   root# /usr/sbin/directoryserver idsktune -q > idsktune.out
   ```

   idsktune は、システムに対する変更案を提供します。このサブコマンド自体はシステムを変更しません。

5. 少なくとも示される ERROR 条件はすべて修正する必要があります。

   ERROR 条件を修正しない場合、インストールが失敗することがあります。idsktune サブコマンドは、リリース時点で推奨されていて、システムにインストールされていないパッチはすべて「見つからない」というメッセージが表示されます。システムにインストールされていないパッケージのパッチも同様です。

   パッチは http://sunsolve.sun.com/ からダウンロードできます。

   詳細は、第 5 章「オペレーティングシステムのチューニング」を参照してください。

Directory Server の設定

1. 設定プログラムを起動します。

   グラフィカルユーザーインタフェースを使用するには、次のコマンドを入力します。

   ```
   root# /usr/sbin/directoryserver configure
   ```

   コマンド行インタフェースを使用するには、次のコマンドを入力します。

   ```
   root# /usr/sbin/directoryserver configure -nodisplay
   ```

   最初のインストール画面が表示されます。

2. 21 ページの「インストールに必要な情報の準備」で作成したワークシートを参照しながら、各画面の指示に従ってください。

管理サーバーの設定

1. 設定プログラムを起動します。

   グラフィカルユーザーインタフェースを使用するには、次のコマンドを入力します。

   ```
   root# /usr/sbin/mpsadmserver configure
   ```

   コマンド行インタフェースを使用するには、次のコマンドを入力します。

   ```
   root# /usr/sbin/mpsadmserver configure -nodisplay
   ```

   最初のインストール画面が表示されます。

2. 21 ページの「インストールに必要な情報の準備」で作成したワークシートを参照しながら、各画面の指示に従ってください。

34 ページの「インストールプロセスの完了」に進んでください。

Solaris パッケージを使用したサイレントインストールの実行

次の手順を実行します。

Solaris パッケージのインストール

24 ページの「Solaris パッケージのインストール」の手順に従います。

必須パッチのインストール

28 ページの「必須パッチのインストール」の手順に従います。

仕様ファイルの作成

完全なサイレントインストールを実行するには、まず、インストール仕様を含む 2 つのファイルを作成する必要があります。1 つは Directory Server 用、もう 1 つは管理サーバー用です。Directory Server のインストール仕様ファイルのテンプレートについては、`/usr/ds/v5.2/setup/typical.ins` を参照してください。管理サーバーについては、`/usr/sadm/mps/admin/v5.2/setup/admin/typicalInstall.ins` を参照してください。

注　仕様ファイルにはパスワードがクリアテキストで含まれていることがあります。適切なファイルアクセス権限を使用して仕様ファイルを保護してください。

サイレントインストール仕様ファイルを作成するには、テンプレートファイルのコピーを手動で編集するか、Directory Server と管理サーバーの設定プログラムを使用して対話式の設定を実行します。

Directory Server と管理サーバー用のサイレントインストール仕様ファイルを対話式に作成するには、次の手順を実行します。

1. `-saveState` オプションを使用して、Directory Server を設定します。

   ```
   root# /usr/sbin/directoryserver configure -saveState dirserv-file
   ```

   (仕様ファイル *dirserv-file* を作成する場合)

2. `-saveState` オプションを使用して、管理サーバーを設定します。

   ```
   root# /usr/sbin/mpsadmserver configure -saveState admserv-file
   ```

   (仕様ファイル *admserv-file* を作成する場合)

3. 仕様ファイル *dirserv-file* と *admserv-file* を使用してほかのシステム上でインストールを実行する前に、両ファイルを調整します。

   `FullMachineName` などといった、サイレントインストール仕様ファイルの指令の一部は、基盤となるホストシステムに直接依存するため、汎用的な方法では生成できません。

サイレントインストール仕様ファイルには、インストールプログラムのビルドバージョンに対応したチェックサム文字列が含まれています。サイレントインストール仕様ファイルをインストールプログラムの別のビルドやリリースで再使用するには、[STATE_BEGIN と [STATE_DONE で始まる行のチェックサム文字列を更新します。Directory Server の更新されたチェックサムは /usr/ds/v5.2/setup/typical.ins 内に、管理サーバーの更新されたチェックサムは /usr/sadm/mps/admin/v5.2/setup/admin/typicalInstall.ins 内に、それぞれ存在します。コード例 1-1 は、チェックサムの例です。

コード例 1-1 サイレントインストールのチェックサム行

```
[STATE_BEGIN Sun ONE Directory Distribution
a7cc64b2f71a0452899e1c3b853ecead72027b3b]
```

仕様ファイルを使用したインストール

Directory Server と管理サーバーを対話式に設定するには、次の手順を実行します。

1. サイレントインストール仕様ファイルに加えた変更を検証します。
2. Directory Server の設定をサイレントモードで実行します。

   ```
   root# /usr/sbin/directoryserver configure -f dirserv-file
   ```

   ここで *dirserv-file* はサイレントインストール設定ファイルです。
3. 管理サーバーの設定をサイレントモードで実行します。

   ```
   root# /usr/sbin/mpsadmserver configure -f admserv-file
   ```

   ここで *admserv-file* はサイレントインストール設定ファイルです。

34 ページの「インストールプロセスの完了」に進んでください。

圧縮アーカイブを使用したインストールの準備

1. 解凍したソフトウェアが含まれるディレクトリ内にある idsktune ユーティリティを実行します。詳細は、19 ページの「Directory Server ソフトウェアの取得」を参照してください。idsktune では、適切なパッチをチェックし、システムがディレクトリサービスの高パフォーマンスをサポートするようにチューニングされているかどうかを確認します。

   スーパーユーザーとして、次のコマンドを入力します。

   ```
   root# ./idsktune -q > idsktune.out
   ```

   提案される変更をシステムに手動で適用します。idsktune 自体はシステムに変更を適用しません。

2. 少なくとも、idsktune によって示される ERROR 条件はすべて修正します。ERROR 条件を修正しない場合、インストールが失敗することがあります。idsktune は、リリース時点で推奨されていて、システムにインストールされていないパッチはすべて「見つからない」というメッセージを表示します。システムにインストールされていないパッケージのパッチも同様です。

   パッチは http://jp.sunsolve.sun.com/ からダウンロードできます。

   詳細は、第 5 章「オペレーティングシステムのチューニング」を参照してください。

3. (省略可) Directory Server のユーザーとグループのアカウントを作成します。

   Directory Server は、インストール中に指定したユーザーとグループとして実行されます。ディレクトリやシステム上のその他のリソースへ不正にアクセスされないようにアクセス権を設定します。詳細は、101 ページの「ユーザーとグループ (UNIX プラットフォームの場合)」を参照してください。

4. (省略可) 別のユーザーとして対話式にインストールする場合、xhost(1) コマンドを使ってディスプレイへのアクセスを許可します。

   DISPLAY 環境変数を適切に設定し、ディスプレイにアクセスできるユーザーとしてインストールを実行すると、インストールプログラムではデフォルトでグラフィカルユーザーインタフェースを表示します。

   インストールプログラムでグラフィカルユーザーインタフェースが表示できない場合は、コマンド行モードでインストールを開始します。

圧縮アーカイブを使用した対話式インストールの実行

1. 解凍したソフトウェアを含むディレクトリにあるインストールプログラムを起動します。

   グラフィカルユーザーインタフェースを使用する場合

   ```
   root# ./setup
   ```

   コマンド行インタフェースを使用する場合

   ```
   root# ./setup -nodisplay
   ```

   最初のインストール画面が表示されます。

2. 21 ページの「インストールに必要な情報の準備」で作成したワークシートを参照しながら、各画面の指示に従ってください。

注 32 ビット Directory Server をインストールする場合、「コンポーネントの選択」ウィザード画面の「Sun ONE Directory Suite」>「Sun ONE Directory Server (64-bit support)」の横にあるチェックボックスの選択を必ず解除してください。

このバージョンの Directory Server を以前のバージョンと同じディレクトリにインストールしないでください。同じディレクトリ位置をもう一度使用する場合は、先に以前のバージョンをアンインストールします。詳細は、第 2 章「旧バージョンからのアップグレード」を参照してください。

圧縮アーカイブを使用したサイレントインストールの実行

次の手順を実行します。

仕様ファイルの作成

サイレントインストールを実行するには、まず、インストール仕様を含むファイルを作成する必要があります。サイレントインストール仕様ファイルのテンプレートについては、ソフトウェアの解凍先ディレクトリにある `setup_data/typical.ins` を参照してください。

注 仕様ファイルにはパスワードがクリアテキストで含まれていることがあります。適切なファイルアクセス権限を使用して仕様ファイルを保護してください。

サイレントインストール仕様ファイルを作成するには、テンプレートファイルのコピーを手動で編集するか、インストールプログラムを使って対話式な設定を実行します。

1. スーパーユーザーになります。
2. インストールプログラムを `-saveState` オプションを指定して実行します。

   `root# ./setup -saveState` *filename*

   (仕様ファイル *filename* を作成する場合)
3. 対話式インストールを実行します。
4. 仕様ファイル *filename* を使用してほかのシステム上でインストールを実行する前に、そのファイルを調整します。

   `FullMachineName` などといった、サイレントインストール仕様ファイルの指令の一部は、基盤となるホストシステムに直接依存するため、汎用的な方法では生成できません。

サイレントインストール仕様ファイルには、インストールプログラムのビルドバージョンに対応したチェックサム文字列が含まれています。サイレントインストール仕様ファイルをインストールプログラムの別のビルドやリリースで再使用するには、[STATE_BEGIN と [STATE_DONE で始まる行のチェックサム文字列を更新します。更新されたチェックサムは typical.ins 内にあります。31 ページのコード例 1-1 は、チェックサムの例です。

仕様ファイルを使用したインストール

1. インストール仕様ファイルに加えた変更を検証します。
2. インストールプログラムをサイレントモードで起動します。

   ```
   root# ./setup -noconsole -nodisplay -state filename
   ```

   ここで *filename* はサイレントインストール仕様ファイルです。

インストールプロセスの完了

1. *ServerRoot*/alias の下にあるファイルへのアクセス権が、*ServerRoot* の下にインストールするサーバー以外のユーザーからのアクセスをすべて禁止するように設定されていることを確認します。
2. (省略可) 圧縮アーカイブを使用してインストールした場合、システムのリブート時に Directory Server を起動するサポートを追加します。このサポートは、Solaris パッケージに含まれています。

   詳細は、Solaris システム管理マニュアルを参照してください。
3. (省略可) core ファイルの生成を有効にします。

   Directory Server をスーパーユーザーとしてインストールしたにもかかわらずユーザー ID やグループ ID に別のアカウントの ID を設定した場合、Directory Server では障害時に core ファイルを生成することができないことがあります。core ファイル用に十分な空き容量を確保して、障害時に Directory Server がファイルを生成できるようにすることを強くお勧めします。

   coreadm(1M) を使用して core ファイルの生成を管理することができるため、たとえば次のようにして、Directory Server で core ファイルを生成することができます。

   ```
   root# coreadm -e proc-setid
   ```

   詳細は、91 ページの「(UNIX プラットフォーム) core ファイル」を参照してください。
4. (省略可) Perl で書かれたほとんどのコマンド行スクリプトで、バインドパスワードの対話式読み取りが可能になりました (-w - オプションを使用)。この機能を有効にするには、次の手順を実行します。

   a. Term::ReadKey Perl モジュールをインストールします。このモジュールは CPAN とは独立して利用できます。

b. バインドパスワードを対話式に読み取れるように、各 Perl スクリプトを編集します。それには、対応する行のコメントを外します。

Term::ReadKey モジュールがなくても、Perl スクリプトのその他の機能はすべて利用可能なままです。

これで Directory Server の起動に必要な要件が設定されました。

その他の UNIX システム上でのインストール

次の該当する節に従ってインストールを進めてください。

- インストールの準備
- 対話式インストールの実行
- サイレントインストールの実行
- インストールプロセスの完了

インストールの準備

次の該当する節に従ってインストールを進めてください。

- すべての UNIX プラットフォームに共通する手順
- AIX システムの追加手順
- HP-UX システムの追加手順

すべての UNIX プラットフォームに共通する手順

1. 解凍したソフトウェアが含まれるディレクトリ内にある idsktune ユーティリティを実行します。idsktune では、適切なパッチが適用されているか確認し、システムがディレクトリサービスの高パフォーマンスをサポートするようにチューニングされているかどうかを確認します。

   スーパーユーザーとして、次のコマンドを入力します。

   ```
   root# ./idsktune -q > idsktune.out
   ```

   提案される変更をシステムに手動で適用します。idsktune 自体はシステムに変更を適用しません。

2. 少なくとも、idsktune によって示される ERROR 条件はすべて修正します。ERROR 条件を修正しない場合、インストールが失敗することがあります。

   表 1-9 に、システムにインストールされていない正式なパッチの入手先を示します。

表 1-9　パッチの入手先 (プラットフォームごと)

プラットフォーム	参照
Hewlett Packard HP-UX	`http://www.hp.com/support/`
IBM AIX	`http://www.ibm.com/support/`
Red Hat Linux	`http://www.redhat.com/`

詳細は、99 ページ以降の第 5 章「オペレーティングシステムのチューニング」を参照してください。

3. (省略可) Directory Server のユーザーとグループのアカウントを作成します。

   Directory Server は、インストール中に指定したユーザーとグループとして実行されます。ディレクトリやシステム上のその他のリソースへ不正にアクセスされないようにアクセス権を設定します。詳細は、101 ページの「ユーザーとグループ (UNIX プラットフォームの場合)」を参照してください。

4. (省略可) 別のユーザーとして対話式にインストールする場合、`xhost`(1) コマンドを使ってディスプレイへのアクセスを許可します。

   `DISPLAY` 環境変数を適切に設定し、ディスプレイにアクセスできるユーザーとしてインストールを実行すると、インストールプログラムではデフォルトでグラフィカルユーザーインタフェースを表示します。

   インストールプログラムでグラフィカルユーザーインタフェースが表示できない場合は、コマンド行モードでインストールを開始します。

AIX システムの追加手順

- コンソールを使用する場合は、`X11.adt` パッケージをインストールします。

  このパッケージは標準バンドルに含まれていません。IBM 社から入手してください。

HP-UX システムの追加手順

1. Directory Server で IPv6 インタフェースを使用する予定がない場合でも、IPv6 のサポートがインストールされていることを確認します。
2. US English でサポートされていないフォントを持つロケールを使用してインストールする前に、リモートセッションでフォントエイリアスを利用できることを確認してください。

   手順については、オペレーティングシステムのマニュアルを参照してください。

対話式インストールの実行

1. 解凍したソフトウェアを含むディレクトリにあるインストールプログラムを起動します。

   グラフィカルユーザーインタフェースを使用する場合

   ```
   root# ./setup
   ```

   コマンド行インタフェースを使用する場合

   ```
   root# ./setup -nodisplay
   ```

   最初のインストール画面が表示されます。

2. 21 ページの「インストールに必要な情報の準備」で作成したワークシートを参照しながら、各画面の指示に従ってください。

注 64 ビットバージョンのサーバーをサポートするプラットフォーム上に 32 ビット Directory Server をインストールする場合、「コンポーネントの選択」ウィザード画面の「Sun ONE Directory Suite」>「Sun ONE Directory Server (64-bit support)」の横にあるチェックボックスの選択を必ず解除してください。

このバージョンの Directory Server を以前のバージョンと同じディレクトリにインストールしないでください。同じディレクトリ位置をもう一度使用する場合は、先に以前のバージョンをアンインストールします。詳細は、第 2 章「旧バージョンからのアップグレード」を参照してください。

38 ページの「インストールプロセスの完了」に進んでください。

サイレントインストールの実行

次の手順を実行します。

仕様ファイルの作成

サイレントインストールを実行するには、まず、インストール仕様を含むファイルを作成する必要があります。サイレントインストール仕様ファイルのテンプレートについては、ソフトウェアの解凍先ディレクトリにある `setup_data/typical.ins` を参照してください。

注 仕様ファイルにはパスワードがクリアテキストで含まれていることがあります。適切なファイルアクセス権限を使用して仕様ファイルを保護してください。

サイレントインストール仕様ファイルを作成するには、テンプレートファイルのコピーを手動で編集するか、インストールプログラムを使って対話式の設定を実行します。

1. スーパーユーザーになります。
2. インストールプログラムを `-saveState` オプションを指定して実行します。

   ```
   root# ./setup -saveState filename
   ```

   (仕様ファイル *filename* を作成する場合)
3. 対話式インストールを実行します。
4. 仕様ファイル *filename* を使用してほかのシステム上でインストールを実行する前に、そのファイルを調整します。

   `FullMachineName` などといった、サイレントインストール仕様ファイルの指令の一部は、基盤となるホストシステムに直接依存するため、汎用的な方法では生成できません。

サイレントインストール仕様ファイルには、インストールプログラムのビルドバージョンに対応したチェックサム文字列が含まれています。サイレントインストール仕様ファイルをインストールプログラムの別のビルドやリリースで再使用するには、`[STATE_BEGIN` と `[STATE_DONE` で始まる行のチェックサム文字列を更新します。更新されたチェックサムは `typical.ins` 内にあります。31 ページのコード例 1-1 は、チェックサムの例です。

仕様ファイルを使用したインストール

1. インストール仕様ファイルに加えた変更を検証します。
2. インストールプログラムをサイレントモードで起動します。

   ```
   root# ./setup -noconsole -nodisplay -state filename
   ```

   ここで *filename* はサイレントインストール仕様ファイルです。

インストールプロセスの完了

1. *ServerRoot*`/alias` の下にあるファイルへのアクセス権が、*ServerRoot* の下にインストールするサーバー以外のユーザーからのアクセスをすべて禁止するように設定されていることを確認します。
2. (省略可) システムのリブート時に Directory Server を起動するサポートを追加します。

   詳細は、オペレーティングシステムのマニュアルを参照してください。

3. (省略可) core ファイルの生成を有効にします。

   Directory Server をスーパーユーザーとしてインストールしたにもかかわらずユーザー ID やグループ ID に別のアカウントの ID を設定した場合、Directory Server では障害時に core ファイルを生成することができないおそれがあります。core ファイル用に十分な空き容量を確保して、障害時に Directory Server がファイルを生成できるようにすることを強くお勧めします。

   詳細は、91 ページの「(UNIX プラットフォーム) core ファイル」を参照してください。

4. (省略可) Perl で書かれたほとんどのコマンド行スクリプトで、バインドパスワードの対話式読み取りが可能になりました (-w - オプションを使用)。この機能を有効にするには、次の手順を実行します。

   a. Term::ReadKey Perl モジュールをインストールします。このモジュールは CPAN とは独立して利用できます。

   a. バインドパスワードを対話式に読み取れるように、各 Perl スクリプトを編集します。それには、対応する行のコメントを外します。

   Term::ReadKey モジュールがなくても、Perl スクリプトのその他の機能はすべて利用可能なままです。

これで Directory Server の起動に必要な要件が設定されました。

Windows システム上でのインストール

次の該当する節に従ってインストールを進めてください。

- インストールの準備
- 対話式インストールの実行
- サイレントインストールの実行
- インストールプロセスの完了

インストールの準備

1. Windows 2000 システムをインストールする場合は、ネットワークセキュリティサービスの依存度を下げるために、既存のドメインまたはワークグループのメンバーではない、スタンドアロンサーバーのコンピュータを指定します。
2. サービスパック 3 を適用します。
3. ディスプレイドライバが少なくとも 256 色をサポートしていることを確認します。
4. Administrator の権限を持つユーザーとしてログオンします。
5. TEMP 環境変数を、一時ファイル用の有効なフォルダに設定します。

対話式インストールの実行

1. 解凍したソフトウェアを含むフォルダ内の `setup.exe` をダブルクリックします。

   最初のインストール画面が表示されます。

2. 21 ページの「インストールに必要な情報の準備」で作成したワークシートを参照しながら、各画面の指示に従ってください。

   このバージョンの Directory Server を以前のバージョンと同じフォルダにインストールしないでください。同じフォルダをもう一度使用する場合は、先に以前のバージョンをアンインストールします。詳細は、第 2 章「旧バージョンからのアップグレード」を参照してください。

41 ページの「インストールプロセスの完了」に進んでください。

サイレントインストールの実行

次の手順を実行します。

仕様ファイルの作成

サイレントインストールを実行するには、まず、インストール仕様を含むファイルを作成する必要があります。サイレントインストール仕様ファイルのテンプレートについては、ソフトウェアの解凍先フォルダにある `setup_data¥typical.ins` を参照してください。

注　仕様ファイルにはパスワードがクリアテキストで含まれていることがあります。適切なファイルアクセス権限を使用して仕様ファイルを保護してください。

サイレントインストール仕様ファイルを作成するには、テンプレートファイルのコピーを手動で編集するか、インストールプログラムを使って対話式の設定を実行します。

1. 管理者の権限を持つユーザーとしてログオンします。
2. インストールプログラムを `-saveState` オプションを指定して実行します。

   製品を解凍したフォルダから、次のコマンドを入力します。

   Prompt`>setup -saveState` *filename*

   (仕様ファイル *filename* を作成する場合)
3. 対話式インストールを実行します。

4. 仕様ファイル *filename* を使用してほかのシステム上でインストールを実行する前に、そのファイルを調整します。

   `FullMachineName` などといった、サイレントインストール仕様ファイルの指令の一部は、基盤となるホストシステムに直接依存するため、汎用的な方法では生成できません。

サイレントインストール仕様ファイルには、インストールプログラムのビルドバージョンに対応したチェックサム文字列が含まれています。サイレントインストール仕様ファイルをインストールプログラムの別のビルドやリリースで再使用するには、`[STATE_BEGIN` と `[STATE_DONE` で始まる行のチェックサム文字列を更新します。更新されたチェックサムは `typical.ins` 内にあります。31 ページのコード例 1-1 は、チェックサムの例です。

仕様ファイルを使用したインストール

1. インストール仕様ファイルに加えた変更を検証します。
2. インストールプログラムをサイレントモードで起動します。

   製品を解凍したフォルダから、次のコマンドを入力します。

   Prompt`>setup -noconsole -nodisplay -state` *filename*

   ここで *filename* はサイレントインストール仕様ファイルです。

インストールプロセスの完了

1. *ServerRoot*`¥alias` の下にあるファイルへのアクセス権が、*ServerRoot* の下にインストールするサーバー以外のユーザーからのアクセスをすべて禁止するように設定されていることを確認します。
2. インストールしたら、次のファイルに対して、特殊なアクセス権限を手動で設定します。管理サーバーを実行するユーザーとグループだけが読み書きアクセス権を持ち、その他の全ユーザーにはアクセス権なしとするようにします。

   - *ServerRoot*`¥admin-serv¥config¥adm.conf`
   - *ServerRoot*`¥admin-serv¥config¥admpw`
   - *ServerRoot*`¥admin-serv¥config¥magnus.conf`
   - *ServerRoot*`¥admin-serv¥config¥obj.conf`
   - *ServerRoot*`¥admin-serv¥config¥secmod.db`
   - *ServerRoot*`¥admin-serv¥config¥server.xml`

   ファイルに対して特殊なアクセス権限を設定する手順については、Windows ヘルプを参照してください。この変更によって、権限のないユーザーが管理サーバーの設定データを変更できないようにします。

3. (省略可) Perl で書かれたほとんどのコマンド行スクリプトで、バインドパスワードの対話式の読み取りが可能となりました (`-w -` オプションを使用)。この機能を有効にするには、次の手順を実行します。

   a. `Term::ReadKey` Perl モジュールをインストールします。このモジュールは CPAN とは独立して利用できます。

   b. バインドパスワードを対話式に読み取れるように、各 Perl スクリプトを編集します。それには、対応する行のコメントを外します。

   `Term::ReadKey` モジュールがなくても、Perl スクリプトのその他の機能はすべて利用可能なままです。

これで Directory Server の起動に必要な要件が設定されました。

アンインストール

アンインストールでは、ソフトウェアとそれに関連するデータがコンピュータから削除されます。Directory Server は利用できなくなり、すべての設定とデータが失われます。

アンインストールでは、サーバーソフトウェアだけでなくシステムに格納されているレジストリデータも削除します。アンインストールプログラムを使用する前に手動でファイルを削除すると、レジストリを破損することがあります。レジストリの破損を避けるには、手動で製品を削除する前にアンインストールプログラムを使用します。

注

`o=NetscapeRoot` サフィックスにある設定情報を含む設定ディレクトリのアンインストールを行う前は、警告を受信しません。

ほかのディレクトリが設定情報で利用している一元管理されている設定ディレクトリをアンインストールすると、そのディレクトリは管理できなくなります。

次の該当する節に従ってアンインストールを進めてください。

- Solaris システム上でのアンインストール
- その他の UNIX システム上でのアンインストール
- Windows システム上でのアンインストール

Solaris システム上でのアンインストール

Directory Server ソフトウェアの削除方法は、インストールプロセス時にどのパッケージが使用されたか、アンインストールプログラムと対話するかどうか、によって異なります。次の該当する節に従ってアンインストールを進めてください。

- Solaris パッケージを使用したインストール後の対話式アンインストールの実行
- 圧縮アーカイブを使用したインストール後の対話式アンインストールの実行
- Solaris パッケージを使用したインストール後のサイレントアンインストールの実行
- 圧縮アーカイブを使用したインストール後のサイレントアンインストールの実行

Solaris パッケージを使用したインストール後の対話式アンインストールの実行

次の該当する節に従ってインストールを進めてください。

- 以前のバージョンの Directory Server のアンインストール
- 管理サーバー設定の削除
- Directory Server 設定の削除
- パッケージの削除

以前のバージョンの Directory Server のアンインストール

- 重要 : Solaris システム上の Directory Server 5.1 から Directory Server 5.2 へのアップグレードを行う場合、Directory Server 5.1 は `IPLT*` Solaris パッケージからインストールされているため、次のようにしてアンインストールプログラムを起動します。

  ```
  root# /usr/sbin/directoryserver.51bak uninstall
  ```

管理サーバー設定の削除

- 管理サーバー設定を削除します。

  ```
  root# /usr/sbin/mpsadmserver unconfigure
  ```

  最初のアンインストール画面が表示されます。各画面の指示に従います。

Directory Server 設定の削除

- Directory Server 設定を削除します。

  ```
  root# /usr/sbin/directoryserver unconfigure
  ```

  最初のアンインストール画面が表示されます。各画面の指示に従います。

パッケージの削除

- pkgrm(1M) ユーティリティを使用して、24 ページの「Solaris パッケージを使用した対話式インストールの実行」でインストールしたパッケージを削除します。

圧縮アーカイブを使用したインストール後の対話式アンインストールの実行

1. *ServerRoot* ディレクトリで、アンインストールプログラムを起動します。

   ```
   root# ./uninstall_dirserver
   ```

   最初のアンインストール画面が表示されます。

2. 各画面の指示に従います。

選択したソフトウェアが削除されます。アンインストールプログラムで *ServerRoot* ディレクトリ内のすべてのファイルを削除できない場合は、メッセージが表示されます。*ServerRoot* 内にファイルが残った場合は、手動で削除します。

Solaris パッケージを使用したインストール後のサイレントアンインストールの実行

1. アンインストール仕様ファイル *ServerRoot*/setup/uninstall.ins を編集して、適切な管理者識別子とパスワードを含めるようにします。

コード例 1-2 アンインストール仕様ファイルの例

```
[STATE_BEGIN Sun ONE Directory Distribution checksum]

ConfigDirectoryAdminID = admin-user
ConfigDirectoryAdminPwd = admin-password

[STATE_DONE Sun ONE Directory Distribution checksum]
```

2. Solaris システム上の Directory Server 5.1 から Directory Server 5.2 へのアップグレードを行う際に、Directory Server 5.1 は IPLT* Solaris パッケージからインストールされている場合は、次のようにしてアンインストールプログラムを起動します。

```
root# /usr/sbin/directoryserver.51bak uninstall -f 51-uninstaller-file
```

3. 管理サーバーの設定を、unconfigure サブコマンドを使って削除します。

```
root# /usr/sbin/mpsadmserver unconfigure -f ServerRoot/setup/uninstall.ins
```

4. Directory Server の設定を、unconfigure サブコマンドを使って削除します。

```
root# /usr/sbin/directoryserver unconfigure -f ServerRoot/setup/uninstall.ins
```

5. pkgrm(1M) ユーティリティを使用して、30 ページの「Solaris パッケージを使用したサイレントインストールの実行」でインストールしたパッケージを削除します。

アンインストールの完了後、残ったファイルを手動で削除します。

圧縮アーカイブを使用したインストール後のサイレントアンインストールの実行

1. アンインストール仕様ファイル *ServerRoot*/setup/uninstall.ins を 44 ページのコード例 1-2 で示したように編集して、適切な管理者識別子とパスワードを含めるようにします。
2. アンインストールプログラムをサイレントモードで実行します。

```
root# cd ServerRoot
root# ./uninstall_dirserver -noconsole -nodisplay -state setup/uninstall.ins
```

アンインストールの完了後、残ったファイルを手動で削除します。

その他の UNIX システム上でのアンインストール

該当する節に従ってアンインストールを進めてください。

対話式アンインストールの実行

1. *ServerRoot* ディレクトリで、アンインストールプログラムを起動します。

   ```
   root# ./uninstall_dirserver
   ```

   最初のアンインストール画面が表示されます。
2. 各画面の指示に従います。

選択したソフトウェアが削除されます。アンインストールプログラムで *ServerRoot* ディレクトリ内のすべてのファイルを削除できない場合は、メッセージが表示されます。*ServerRoot* 内にファイルが残った場合は、手動で削除します。

サイレントアンインストールの実行

1. アンインストール仕様ファイル *ServerRoot*/setup/uninstall.ins を 44 ページのコード例 1-2 で示したように編集して、適切な管理者識別子とパスワードを含めるようにします。
2. アンインストールプログラムをサイレントモードで実行します。

```
root# cd ServerRoot
root# ./uninstall_dirserver -noconsole -nodisplay -state setup/uninstall.ins
```

アンインストールの完了後、残ったファイルを手動で削除します。

Windows システム上でのアンインストール

該当する節に従ってアンインストールを進めてください。

対話式アンインストールの実行

1. 「スタート」をクリックし、「設定」> 「コントロールパネル」を選択します。
2. 「アプリケーションの追加と削除」をダブルクリックします。
3. 「アプリケーションの追加と削除」ウィンドウで「Directory Server」を選択し、「削除」をクリックします。
4. 「Sun ONE アンインストール」ウィンドウの指示に従います。

   Directory Server をアップグレードしている場合はカスタムアンインストールモードを使用し、Basic System Libraries を削除しないようにします。このライブラリにはアップグレード後の Directory Server インスタンスで共有される .dll ファイルがあります。

サイレントアンインストールの実行

1. アンインストール仕様ファイル *ServerRoot*¥setup¥uninstall.ins を 44 ページのコード例 1-2 で示したように編集して、適切な管理者識別子とパスワードを含めるようにします。
2. アンインストールプログラムをサイレントモードで実行します。

```
Prompt>cd ServerRoot
Prompt>uninstall_dirserver -noconsole -nodisplay -state setup¥uninstall.ins
```

アンインストールの完了後、ファイルが残った場合は手動で削除します。

アンインストールの完了後、Windows システムをリブートすることを強くお勧めします。

トラブルシューティング

表 1-10 一般的なインストール上の問題とその解決策

問題	考えられる解決策
ライブラリが見つからないというメッセージが表示される	idsktune を実行し、少なくとも ERROR 条件はすべて修正する。また推奨されるパッチはすべてインストールする
インストールしたが動作せず、アンインストールもできない。どうしたら良いか	製品のレジストリファイルを削除する。削除しない場合、ほかの製品に悪影響が及ぶ • Solaris システム : /var/sadm/install/productregistry (スーパーユーザーとしてインストールした場合) • その他の UNIX システム : /var/tmp/productregistry • Windows : Windows システムフォルダの下の system32 フォルダにある productregistry (例 : C:¥WINNT¥system32¥productregistry) 続いて、部分的にインストールされたファイルを手動で削除した後、再インストールする
インストールに失敗した理由がわからない。インストールログがどこかに存在するか	ログは次の場所に存在している • Solaris システム : /var/sadm/install/logs (スーパーユーザーとしてインストールした場合) または /var/tmp (通常のユーザーとしてインストールした場合) • その他の UNIX システム : /var/tmp • Windows システム : %TEMP% フォルダ
クライアントがサーバーを検出できない	dirserv というようにホスト名を使用して試す 動作しない場合は、DNS など、使用しているネームサービスのリストにサーバーが含まれていることを確認する。次に dirserv.example.com などの完全修飾ドメイン名を試す 動作しない場合は、192.168.0.30 などのホストの IP アドレスを使用して試す

表 1-10 一般的なインストール上の問題とその解決策 (続き)

問題	考えられる解決策
ポートが使用中である	アップグレード中である場合は、アップグレードの前に Directory Server を停止しなかった可能性がある。古いサーバーを停止し、アップグレードしたサーバーを手動で起動する そうでない場合は、インストールされているほかのサーバーがそのポートを使用している可能性がある。ポートが利用可能なままであるかどうかを調べるために、UNIX システムでは `-a` オプションを指定して `netstat`(1M) ユーティリティを使用するなど、適切なツールを使用して使用中のポートを調べる
LDAP 認証エラーになりインストールに失敗する	`dirserv.example.com` ではなく `dirserv.nisDomain.Example.COM` などの正しくない完全修飾ドメイン名をインストール中に指定した可能性がある
Directory Manager の DN とパスワードを忘れてしまった場合	Directory Manager の DN は *ServerRoot*`/slapd-`*serverID*`/config/dse.ldif` に `nsslapd-rootdn` の値として記録されている Directory Manager のパスワードは `dse.ldif` に `nsslapd-rootpw` の値として記録されている。パスワードが暗号化されていない場合、`dse.ldif` ではクリアテキストになり、`{SSHA}` などの暗号化スキーム識別子が先頭に付かない。暗号化することを強く推奨する パスワードが暗号化されている場合、問題は手動で解決する必要がある 1. Directory Server を停止する 2. `dse.ldif` 内の `nsslapd-rootpw` の値を変更する。末尾にスペースを追加しないように注意する 3. `dse.ldif` を保存して閉じる 4. サーバーを再起動する 5. `nsslapd-rootpw` に割り当てた値を使用して Directory Manager としてログインする 6. 『Sun ONE Directory Server 管理ガイド』で説明するように Directory Manager パスワードの暗号化スキームを設定し、もう一度パスワードを変更する

表 1-10 一般的なインストール上の問題とその解決策 (続き)

問題	考えられる解決策
誤って 32 ビットバージョンの Directory Server をインストールしてしまった 代わりに 64 ビットバージョンを実行する方法について	1. 『Sun ONE Directory Server 管理ガイド』で説明しているように、すべてのサフィックスを LDIF にエクスポートする 2. すべてのデータベースファイルを削除する。データベースファイルはインスタンスの `cn=config,cn=ldbm database,cn=plugins,cn=config` の `nsslapd-directory` で示されるパスにある 3. 64 ビットコンポーネントをまだインストールしていない場合はインストールする 4. *ServerRoot*`/bin/slapd/server/64/ns-slapd` を実行可能にする 5. オペレーティングシステムが 32 ビットモードで実行中の場合は、64 ビットモードでリブートする 6. 必要に応じて、キャッシュサイズの設定を 32 ビットモードで動作するように変更する。詳細は、第 6 章「キャッシュサイズのチューニング」を参照 7. 『Sun ONE Directory Server 管理ガイド』で説明しているように、エクスポートした LDIF ですべてのサフィックスを初期化する 8. サーバーを再起動する
誤って 64 ビットバージョンの Directory Server をインストールしてしまった 代わりに 32 ビットバージョンを実行する方法について	1. 『Sun ONE Directory Server 管理ガイド』で説明しているように、すべてのサフィックスを LDIF にエクスポートする 2. すべてのデータベースファイルを削除する。データベースファイルはインスタンスの `cn=config,cn=ldbm database,cn=plugins,cn=config` の `nsslapd-directory` で示されるパスにある 3. *ServerRoot*`/bin/slapd/server/64/ns-slapd` のモードを実行できないようにする 4. 『Sun ONE Directory Server 管理ガイド』で説明しているように、エクスポートした LDIF ですべてのサフィックスを初期化する 5. サーバーを再起動する

表 1-10 一般的なインストール上の問題とその解決策 (続き)

問題	考えられる解決策
インストールを処理したスクリプトを記述した。作成したスクリプトでインストールを試みたが、インストールプログラムは 0 ではなく 73 を返した。 何が起きたのか	インストールプログラムでは、次のようにコードを返す 0 - SUCCESS 1 - WARNING_REBOOT_REQUIRED 2 - WARNING_PLATFORM_SUPPORT_LIMITED 3 - WARNING_RESOURCE_NOT_FOUND 4 - WARNING_CANNOT_WRITE_LOG 5 - WARNING_LOCALE_NOT_SUPPORTED 50 - ERROR_FATAL 51 - ERROR_ACCESS 52 - ERROR_PLATFORM_NOT_SUPPORTED 53 - ERROR_NO_WINDOWING_SYSTEM_AVAILABLE 54 - ERROR_RESOURCE_NOT_FOUND 55 - ERROR_TASK_FAILURE 56 - ERROR_USER_EXIT 57 - ERROR_CANNOT_UPGRADE 58 - ERROR_NOTHING_TO_DO 59 - ERROR_IN_SERIALIZATION 60 - ERROR_ABNORMAL_EXIT 61 - ERROR_INCOMPATIBLE_STATEFILE 62 - ERROR_UNKNOWN_COMMANDLINE_OPTION 70 - ERROR_NOT_INSTALLED 71 - PARTIALLY_UNINSTALLED 72 - FULLY_UNINSTALLED 73 - INSTALLED 74 - ERROR_FAILED 75 - ERROR_STOPPED 76 - ERROR_STOPPED_ON_ERROR 77 - PARTIALLY_INSTALLED つまり、73 はインストールに成功したことを示している

第 2 章

旧バージョンからのアップグレード

この章では、Netscape Directory Server 4.x および iPlanet Directory Server 5.x から Sun ONE Directory Server 5.2 へのアップグレードについて説明します。

注	Innosoft Distributed Directory Server 4.5.1 からのアップグレード方法については、この章では取り扱っていません。

この章では、主に古いサーバーから新しいサーバーへのディレクトリデータの移行方法について詳しく説明します。古いサーバーから新しいサーバーに移行する設定属性の詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

アップグレードの前に

アップグレードを行う前に、Sun ONE Directory Server 5.2 で使用する新しい機能を理解しておいてください。詳細は、このマニュアルの 12 ページの「推奨参考文献」を参照してください。この機会に、既存のディレクトリサービスの実装時に行われた設計上の決定事項を検討してください。

単一サーバーインスタンスをアップグレードする場合

サーバーインスタンスのアップグレードでは、異なる *ServerRoot* に既存のサーバーとともに新しいサーバーをインストールします。それぞれ異なる *serverID*、管理サーバー、および Directory Server のポート番号を使用し、古いサーバーを停止してから、設定およびディレクトリデータを移行し、クライアントで新しいサーバーへの要求を行います。

注 既存のサーバーを実行するホスト上に十分なディスクの空き容量があることを確認します。アップグレードプロセスでは、少なくとも新しいサーバーと古いサーバーの両方のバイナリファイルとデータベースを格納できる十分なローカルディスク容量が必要です。さらに、既存のすべてのサフィックスのエントリを含む LDIF ファイルを保持するための追加スペースも必要になります。必要なローカルディスク容量は、次の計算式で得られる結果よりも多めに見積もる必要があります。

2 * (既存のサーバー用スペース) + (LDIF ファイル用スペース)

データの移行はネットワークドライブを介して行うことができないため、アップグレードプロセスは、同じホスト上の両方のサーバーで実行する必要があります。

Sun ONE Directory Server 5.2 には、サーバーインスタンスのデータの移行に役立つスクリプトが用意されています。移行スクリプトは、次のタスクを順番に実行します。

1. 既存のサーバーを停止して、現在の設定をバックアップする
2. スキーマ設定ファイルを確認し、既存のサーバーで使用されているスキーマ設定ファイルと標準スキーマ設定ファイルとの相違をレポートする

   (4.x から 5.2 への移行のみ) 既存の 4.x サーバーが、デフォルト (*ServerRoot*/slapd-*serverID*/config) 以外の場所にインストールされたカスタムスキーマを使用している場合は、ディレクトリデータを移行する前に手動で設定を調整する必要があります。
3. 古いサーバーに格納されているサフィックスのデータベースを作成する

   (4.x から 5.2 への移行のみ) 4.x サーバーは、データベースごとに複数のサフィックスをサポートしていました。移行スクリプトは、新しいサーバー上に各サフィックスのデータベースを作成します。
4. サーバーとデータベースの設定パラメータを移行する

   4.x サーバーは、これらのパラメータを slapd.conf ファイルに格納します。5.x サーバーは、これらのパラメータをエントリとして dse.ldif ファイルに格納します。

注 このスクリプトは、o=NetscapeRoot の下にあるデータを移行しません。

このサフィックスのデータを利用する Sun ONE Messaging Server などのサーバーを導入する場合は、o=NetscapeRoot のデータを手動で移行するか、または対象となるサーバーに付属のツールを使用して移行します。

(4.x から 5.2 への移行のみ) この移行スクリプトは、4.x サーバーのすべてのパラメータを移行しません。場合によっては、手動で 4.x の属性値を移行する必要があります。詳細は、現在のバージョンの『Sun ONE Directory Server Reference Manual』を参照してください。

5. ユーザー定義のスキーマオブジェクトを移行する
6. インデックスを移行する
7. 標準サーバープラグインを移行する

   手動でカスタムプラグインを移行する必要があります。少なくとも、すべてのカスタムプラグインを再コンパイルする必要があります。プラグイン API 更新の詳細リストについては、『Sun ONE Directory Server Plug-In API Programming Guide』を参照してください。
8. (5.x から 5.2 への移行のみ) レプリケーションアグリーメントを移行する

注 Directory Server 5.2 から 5.1 サーバーにレプリケートする前に、cn=config の nsslapd-schema-repl-useronly を on に設定します。この設定を先に行わない場合、5.2 スキーマが 5.1 サーバーにレプリケートされ、重複オブジェクトが作成されるため、5.1 サーバーが再起動できなくなります。

9. 証明書データベースおよび SSL パラメータを移行する
10. (5.x から 5.2 への移行のみ) データベースリンクを移行する
11. (5.x から 5.2 への移行のみ) レプリケーションエントリを移行する
12. SNMP 設定を移行する

移行スクリプトが完了すると、クライアントは新しいサーバーに要求を送信することができます。

複数のレプリケーションサーバーをアップグレードする場合

当然のことながら、複数のサーバーのアップグレードは、各サーバーを個別にアップグレードすることによって行われます。ただし、サーバーをアップグレードする「順序」は、既存のサーバーソフトウェアのバージョンとレプリケーショントポロジによって決まります。

5.x から 5.2 にアップグレードする場合、標準プロセスはボトムアップです。最初にコンシューマを移行します。次にハブをアップグレードします。最後にマスターをアップグレードします。特定のインスタンスでこのプロセスを実行する方法については、69 ページの「5.x アップグレードの例」を参照してください。

4.x から 5.2 にアップグレードするには、4.x マスターをアップグレードしてから、次にマスターからレプリケートされたコンシューマの各分岐のアップグレードに進みます。この場合、レプリケーションを基準にして最も近いコンシューマからアップグレードを開始します。特定のインスタンスでこのプロセスを実行する方法については、61 ページの「4.x アップグレードの例」を参照してください。

既存の環境に複数のレプリケーションサーバーが含まれている場合は、アップグレードを続行する前に、この章のすべての関連する項目を入念に読んでください。不必要なダウンタイムを回避するために、アップグレード方法を十分に計画する必要があります。

アップグレードのヘルプについて

Sun プロフェッショナルサービスは、重要なディレクトリサービスのアップグレードを支援します。

問い合わせ先情報については、Web サイト `http://jp.sun.com/service/sunps/sunone/` を参照してください。

単一サーバーのアップグレード

この節では、単一の既存のサーバーから単一の 5.2 サーバーへのアップグレードプロセスについて説明します。

注 既存の 4.x サーバーがカスタムスキーマを使用する場合、データを移行する前に移行スクリプトがカスタムスキーマを検出できることを確認します。詳細は、56 ページの「カスタムスキーマの処理 (4.x から 5.2 にアップグレードする場合)」をお読みください。

移行スクリプトがカスタムスキーマを認識しない場合、スキーマは移行されないので、新しいサーバーにデータを移行した後に標準スキーマファイルを適用します。カスタムスキーマに適合するエントリに標準スキーマを適用すると変更ができなくなり、アップグレードディレクトリは読み取り専用になります。

新しいサーバーのインストール

既存のサーバーと同じホストに新しいサーバーをインストールするには、第 1 章「Sun ONE Directory Server のインストール」で説明する手順に従います。

注 新しいサーバーをインストールする前に、既存のサーバーの現在の内容がバックアップされていることを確認します。

新しいサーバーは、既存のサーバーとは別の *ServerRoot* ディレクトリに配置する必要があります。また、*serverID* も別にする必要があります。

元のインストールに適用したほとんどの設定情報を再利用できますが、既存のポート番号は再利用しないでください。既存のデータの移行後、新しいサーバーのポート番号を変更できます。

カスタムスキーマの処理 (4.x から 5.2 にアップグレードする場合)

付属のデータ移行スクリプトは、標準の slapd.user_oc.conf および slapd.user_at.conf ファイルに配置されたカスタムスキーマ、またはほかのファイルに配置され、useroc および userat 指令を使用して slapd.conf に組み込まれたカスタムスキーマだけを認識します。たとえば、カスタムスキーマが slapd.at.conf または slapd.oc.conf ファイルに直接組み込まれている場合、移行スクリプトはこれらのカスタムスキーマを認識しません。

アップグレードを進める前に、次の手順を実行します。

1. slapd.at.conf または slapd.oc.conf ファイルを、新しいサーバーの *ServerRoot*/bin/slapd/install/version4/ の下にある標準ファイルと比較して、カスタムスキーマを slapd.user_oc.conf と slapd.user_at.conf ファイルに転記します。

   カスタマイズしたオブジェクトクラスに継承関係がある場合は、スキーマ設定ファイルで、上位オブジェクトクラスがその他のクラスよりも先行していることを確認します。

2. slapd.oc.conf の標準オブジェクトクラスにカスタム属性が追加された場合は、slapd.user_oc.conf の属性を含む新しいオブジェクトクラスを作成して、カスタム属性を使用する既存のディレクトリのエントリすべてにこの新しいオブジェクトクラスを追加します。

3. useroc と userat 指令を使用して、既存のサーバーの slapd.conf ファイルに slapd.user_oc.conf と slapd.user_at.conf ファイルを組み込み、ほかのファイルの文を組み込むための新しい指令を隣接して配置します。

この時点で、既存のサーバーが使用するすべてのカスタムスキーマが slapd.user_oc.conf または slapd.user_at.conf ファイルに配置され、これらのファイルが useroc と userat 指令を使用して、slapd.conf に組み込まれます。

既存のデータの移行

カスタムスキーマを処理したら、次の手順を実行して既存のデータを新しいサーバーに移行します。

1. 新しい Directory Server のレプリケーションを、ファイルからオフラインで初期化する場合は、初期化を進める前にファイルを取得します。

   Directory Server のデータをエクスポートする手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。

2. 新しい Directory Server が動作していることを確認します。

3. 新しいサーバーと古いサーバーの両方でデータベースのエクスポートとインポートを起動、停止、および実行するために、それらの権利を持つユーザーとして作業します。

   たとえば、スーパーユーザーになるか、または Administrator としてログオンします。

4. 表 2-1 に示すように、環境変数を設定します。

表 2-1 移行の環境変数

変数	値
PATH	(UNIX の場合) *ServerRoot*/bin/slapd/admin/bin:$PATH (Windows の場合) *ServerRoot*/bin/slapd/admin/bin;%PATH%
PERL5LIB	*ServerRoot*/bin/slapd/admin/bin

5. 新しいサーバーインスタンスで、次のように移行スクリプトを実行します。

```
# cd ServerRoot/bin/slapd/admin/bin
# perl migrateInstance5 -p port52 -D "cn=directory manager" -w password -o oldServ -n ¥
newServ
```

   ここで、*oldServ* は、/usr/iplanet/servers/slapd-ldap または /usr/iplanet/ds5/slapd-ldap など古いサーバーインスタンスへの絶対パスを表し、*newServ* は、/var/ds/v5.2/slapd-dirserv など新しいサーバーインスタンスへの絶対パスを表します。

スクリプトは実行に伴って出力を生成します。移行完了後のレビューのために、この出力をファイルに転送することができます。

既存のデータを新しいサーバーに移行したら、古いサーバーだけを解除します。

レプリケーションアグリーメントの作成 (4.x から 5.2 にアップグレードする場合)

既存の 4.x サーバーがレプリケーションに含まれている場合、アップグレードによってデータの移行後にレプリケーションアグリーメントが再作成されます。アップグレードプロセスを進める前に、58 ページの「レプリケーションサーバーのアップグレード (4.x から 5.2 にアップグレードする場合)」をお読みください。

5.2 サーバーのレプリケーションを設定する手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。

既存のポート番号の再利用 (必要に応じて)

古いサーバーから新しいサーバーにデータを移行後、古いサーバーを解除して、新しいサーバーを古いサーバーと同じポートで待機させることができます。古いサーバーと同じポートを使用すると、設定を変更せずにクライアントアプリケーションを続けて使用できます。

サーバーポートを変更する手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。新しいサーバーが古いポートで待機し始める前に、必ず古いサーバーを停止してください。

レプリケーションサーバーのアップグレード (4.x から 5.2 にアップグレードする場合)

4.x レプリケーションサーバーをアップグレードする場合は、新しいマスターにレプリケートしてから、レプリケーショントポロジを介して分岐ごとにアップグレードを進めます。この方法は、サーバー同期トラフィックの量を制限します。

注 レプリケーションの設定と初期化に関する詳細手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。

新しいマスターの準備

アップグレードの間に、5.2 サーバーはマスターとして設定されますが、4.x トポロジの旧バージョンのコンシューマとして機能します。アップグレード完了後、4.x コンシューマ機能が無効になり、新しいサーバーは 5.2 トポロジのマスターとして機能します。

この手順では、手動で新しいマスターサーバーを設定する必要があります。このため、新しいマスターを既存のマスターと別のホストにインストールすることができます。

1. 第 1 章「Sun ONE Directory Server のインストール」の手順に従って、新しいサーバーをインストールします。
2. 新しいサーバーに、4.x マスターの設定を手動でもう一度作成します。
3. 新しいサーバーをマスターにします (5.2 トポロジの場合)。

   手順は、『Sun ONE Directory Server 管理ガイド』を参照してください。

4. 新しいサーバーを 4.x マスターの旧バージョンのコンシューマとします (4.x トポロジの場合)。

   手順は、『Sun ONE Directory Server 管理ガイド』を参照してください。

5. 4.x マスターから新しいサーバーへのレプリケーションを初期化します。

   このプロセスについては、『Netscape Directory Server 管理者ガイド』の第 12 章「レプリケーションの管理」で説明しています。「Consumer の手動作成」というタイトルの節を参照してください。

これで、コンシューマをアップグレードできるようになりました。

コンシューマのアップグレード

次の手順は、コンシューマのアップグレード方法の概要について説明します。詳細は、後続の手順を参照してください。

1. 4.x トポロジのすべての分岐をアップグレードします。
2. 必要に応じて、5.2 トポロジにさらにサーバーを追加します。
3. 新しいマスターの旧バージョンのコンシューマアグリーメントを無効にして、新しいトポロジと古いトポロジの関係を解除します。

この手順が完了すると、更新プロセスが完了します。

分岐のアップグレード

既存の 4.x レプリケーショントポロジを、ルート要素としてマスターを持つツリーとみなします。ここでは、*branch* は、レプリケーションフローがルートノードのサプライヤから始まり、ツリー内のコンシューマを経由して、最終的に最下位のノードのコンシューマサーバーに到達する一連のレプリケーションサーバーを示します。

分岐のアップグレードは、トップダウン方式で分岐内のすべての古いサーバーを新しいサーバーに置き換えるプロセスで構成されます。

注 サーバーをアップグレードする間に、レプリケーションフローは分岐内のすべてのダウンストリームサーバーで停止します。アップグレード中にクライアントの要求を別の分岐に転送することを検討してください。

1. 55 ページの「単一サーバーのアップグレード」で説明する手順に従って、分岐の最上位のサーバーをアップグレードします。

   これにより、分岐へのレプリケーションフローが切断され、分岐内のダウンストリームサーバーにあるレプリケーションの更新が一時的に中断されます

2. 5.2 分岐内の新しいサーバーのレプリケーションアグリーメントを設定して、レプリケーショントポロジの近くにある 5.2 サーバーから新しいマスターへの更新を受信します。

   たとえば、新しい分岐の最上位サーバーを設定して、5.2 マスターからの更新を受信します。

3. 5.2 サプライヤから新しい 5.2 サーバーへのレプリケーションを初期化します。

   ネットワーク容量および更新時と比較したディレクトリデータの量によっては、オフラインによる初期化の方がオンラインによる初期化よりも速くなる場合があります。

4. 最下位のすべてのコンシューマに対してこの手順が完了するまで、分岐に沿って手順 1、手順 2、および 手順 3 を繰り返し実行します。

レプリケーションアグリーメントの設定とレプリケーションの初期化に関する手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。

この時点で、分岐の更新プロセスが完了します。残りの 4.x 分岐に対して、この手順を繰り返します。

さらにサーバーを追加

4.x トポロジから 5.2 トポロジへのアップグレードの完了後、新しいトポロジの必要に応じてさらにマスター、ハブ、およびコンシューマを追加することができます。

各追加サーバーに対して次の手順を実行します。

1. 第 1 章「Sun ONE Directory Server のインストール」で説明する手順に従って、新しいサーバーのインストールを実行します。

2. 計画したトポロジに適合するように、新しいサーバーのレプリケーションアグリーメントを調整します。

   手順は、『Sun ONE Directory Server 管理ガイド』を参照してください。

3. 新しいサーバーのレプリケーションを初期化します。

   手順は、『Sun ONE Directory Server 管理ガイド』を参照してください。

4.x アップグレードの例

4.x マスターのアップグレードを、1 つは単一のコンシューマ、もう 1 つは 2 つのコンシューマに供給するハブを持つ 2 つの分岐にレプリケートする場合について検討します。この節では、新しいマルチマスタートポロジにアップグレードするための手順を示します。

図 2-1 は、アップグレードする前の 4.x トポロジを示します。

図 2-1　既存の 4.x トポロジの例

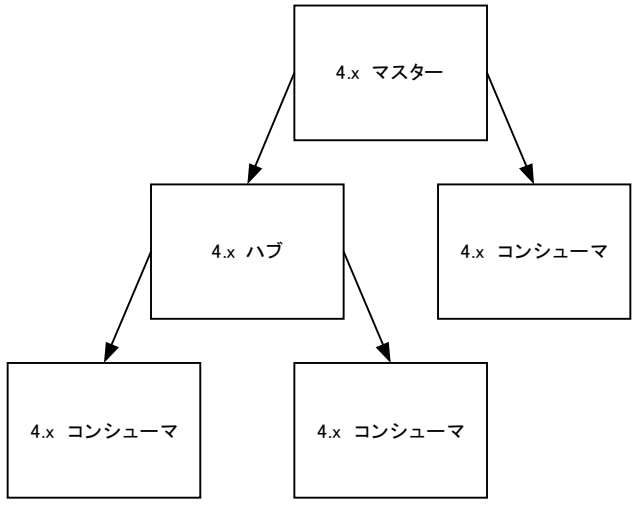


図 2-2 は、4.x マスターの旧バージョンのコンシューマとしても機能する 5.2 マスターの追加を示します。

図 2-2 新しいサーバーが追加された 4.x トポロジの例

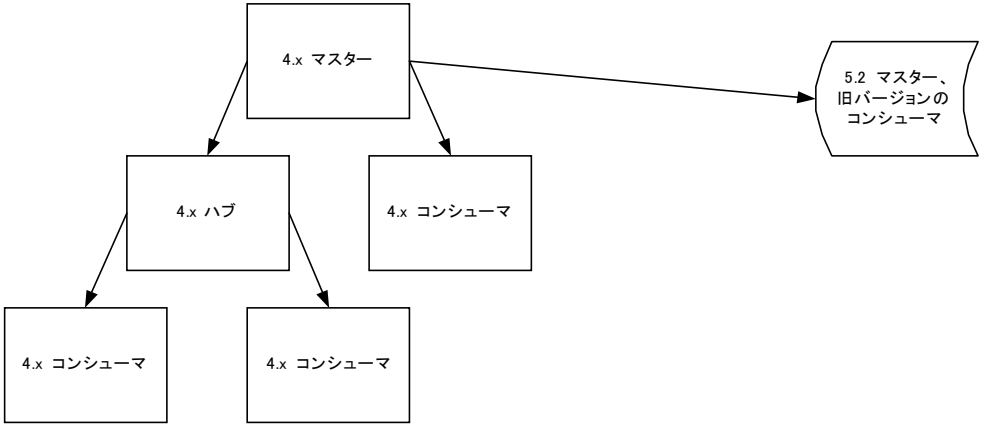


図 2-3 は、4.x 分岐を置き換える最初の手順を示します。

アップグレードの間、すべての分岐がレプリケーションの更新の受信を停止することに注意してください。この中断は、アップグレードのために、アップストリームの 4.x コンシューマを停止したときに始まり、4.x コンシューマを再起動したときに終了します。

手順で説明されているように、クライアントが利用可能な最新の更新を要求する場合は、クライアントの要求を別の分岐のコンシューマに転送することもできます。

図 2-3 アップグレード中の 4.x 分岐の例 − 手順 1

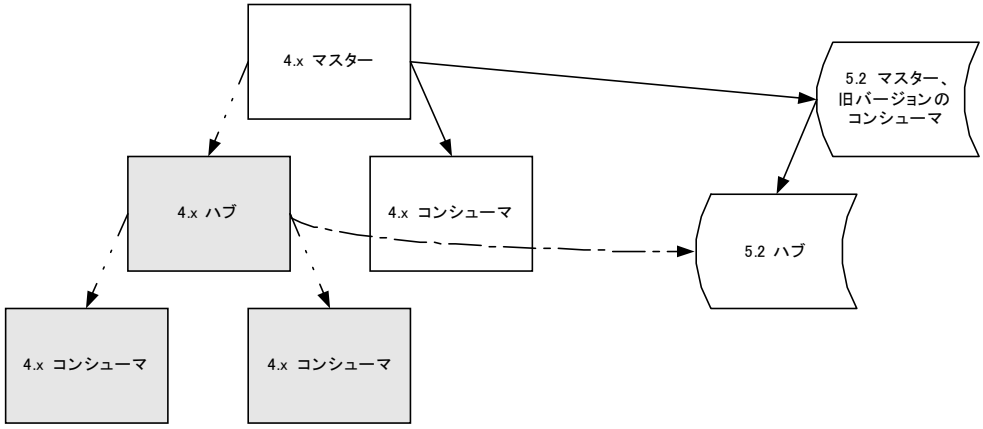


図 2-4 は、4.x 分岐を置き換える次の手順を示します。

図 2-4 アップグレード中の 4.x 分岐の例 − 手順 2

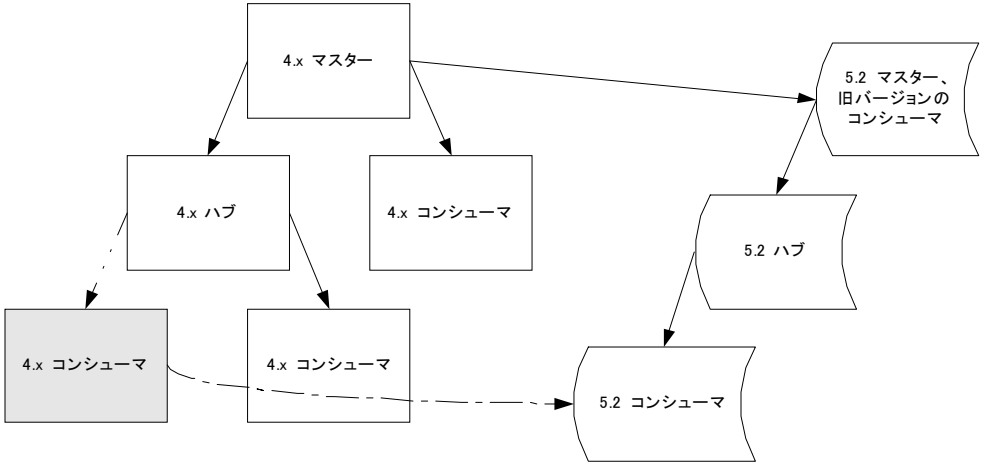


図 2-5 は、4.x 分岐を置き換える次の手順を示します。

図 2-5 アップグレード中の 4.x 分岐の例 − 手順 3

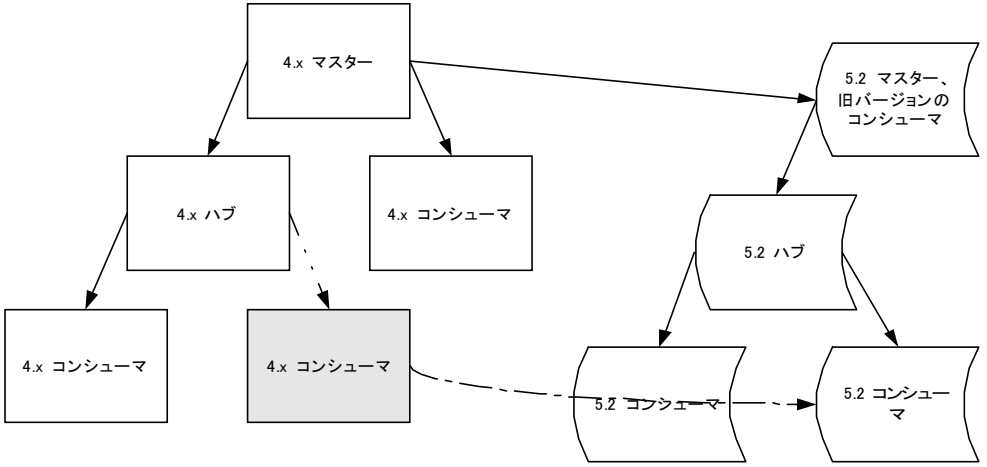


図 2-6 は、その他の 4.x 分岐の置き換えを示します。

図 2-6 アップグレード中の 4.x 分岐の例 − 次の分岐

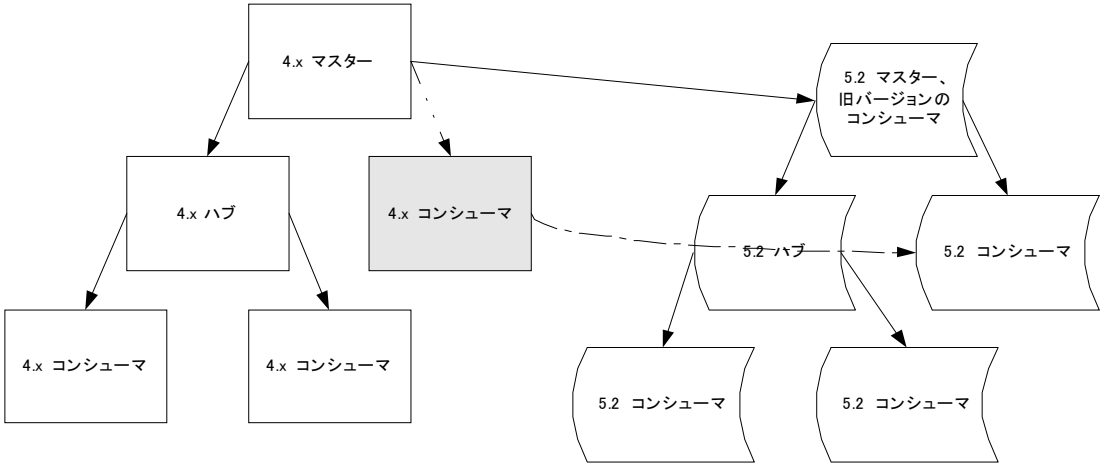


図 2-7 は、並行する 2 つのトポロジを示します。

図 2-7 アップグレード中の 4.x および 5.2 トポロジの例

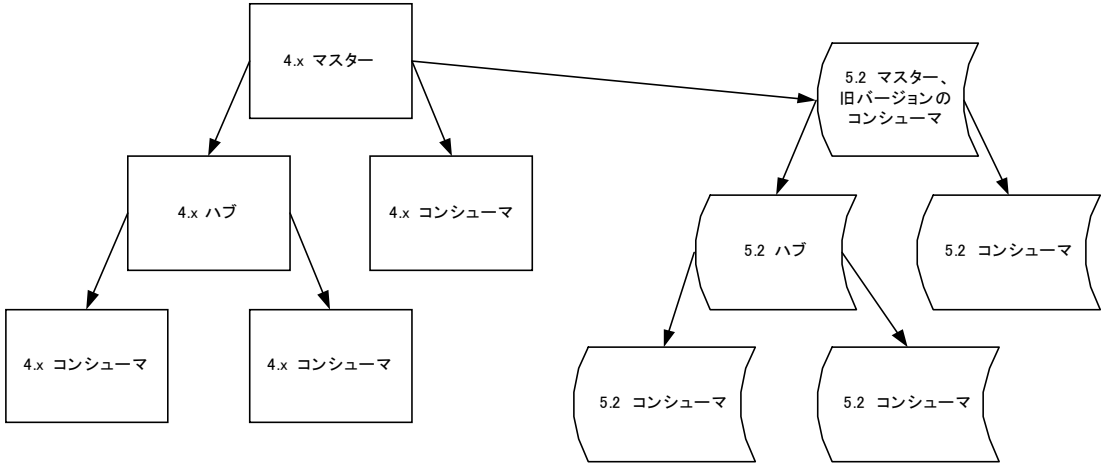
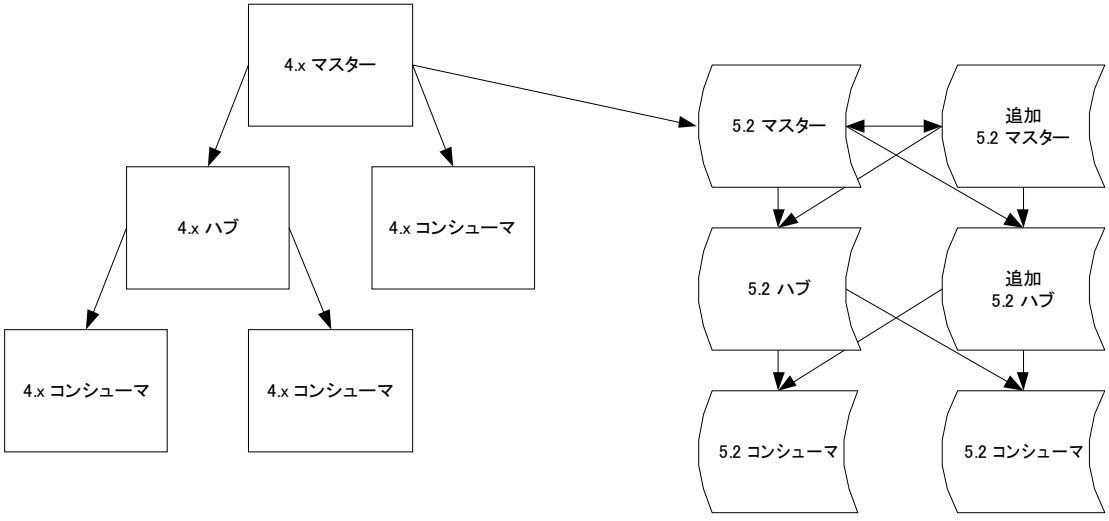


図 2-8 は、新しいトポロジに追加されたマスター、ハブ、およびさらに追加されたレプリケーションアグリーメントを示します。

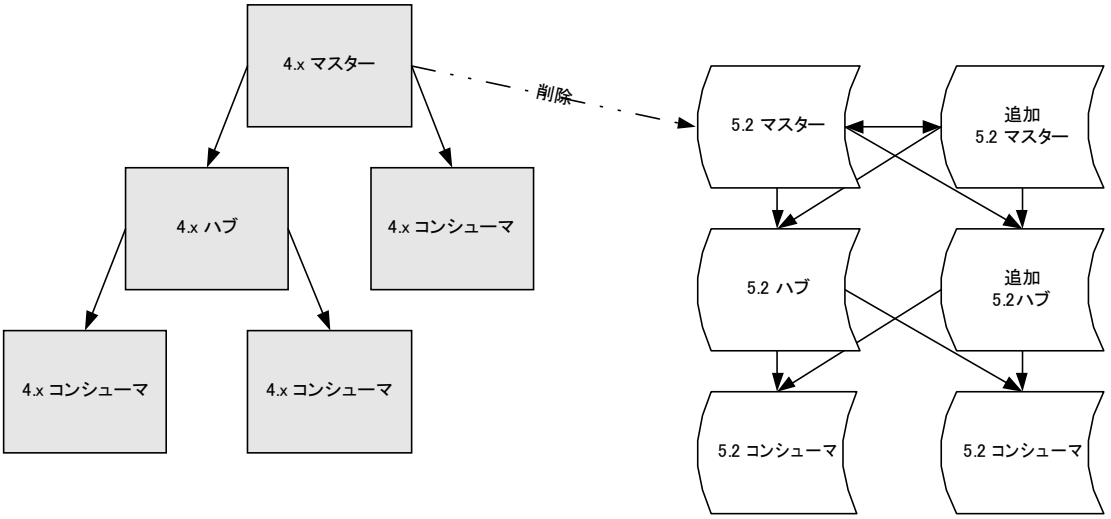
図 2-8 5.2 トポロジへのサーバーの追加



アップグレードプロセスの完了後、さらにサーバーを追加することもできます。

図 2-9 は、古い 4.x マスターから新しい 5.2 マスターへのレプリケーションアグリーメントの削除を示します。

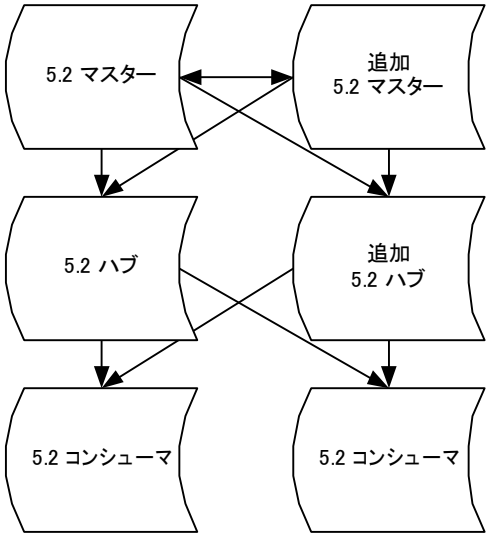
図 2-9 レプリケーションアグリーメントの削除



クライアントの要求を転送し、レプリケーションアグリーメントを削除した後に、4.x サーバーを無効にすることができます。

図 2-10 は、結果として作成された 5.2 トポロジを示します。

図 2-10　結果として作成された 5.2 トポロジ



ここで、クライアントの要求が 5.2 トポロジに転送されています。

レプリケーションサーバーのアップグレード (5.x から 5.2 にアップグレードする場合)

5.x サーバーをアップグレードする場合、通常はコンシューマから開始し、次にハブを更新し、最後にマスターを更新して終了します。このボトムアップ方法では、一度に中断されるサーバーは 1 台だけです。レプリケーショントポロジの特定の分岐全体が中断されることはありません。また、この方法を使用すると、マスターとコンシューマ間でカスタムスキーマの同期問題が発生する可能性が低下します。

注 ここで説明する手順は、5.x トポロジをアップグレードする標準的な方法を使用します。

ただし、このボトムアップ方法が特定の要件に対応できない場合は、別の方法を計画してください。

5.x サーバーのアップグレード

1. 既存のトポロジの各コンシューマの場合、55 ページの「単一サーバーのアップグレード」で説明する手順に従って、コンシューマのアップグレードを進めます。
2. 既存のトポロジのそれぞれのハブについては、同じ手順に従って、ハブを更新します。
3. 既存のトポロジのそれぞれのマスターについては、同じ手順に従って、ハブを更新します。

さらにサーバーを追加

5.x トポロジから 5.2 トポロジへのアップグレードの完了後、新しいトポロジの必要に応じてさらにマスター、ハブ、およびコンシューマを追加することができます。

各追加サーバーに対して次の手順を実行します。

1. 第 1 章「Sun ONE Directory Server のインストール」の手順に従って、新しいサーバーをインストールします。
2. 計画したトポロジに適合するように、新しいサーバーのレプリケーションアグリーメントを調整します。
3. 新しいサーバーのレプリケーションを初期化します。

レプリケーションアグリーメントの設定とレプリケーションの初期化に関する手順については、『Sun ONE Directory Server 管理ガイド』を参照してください。

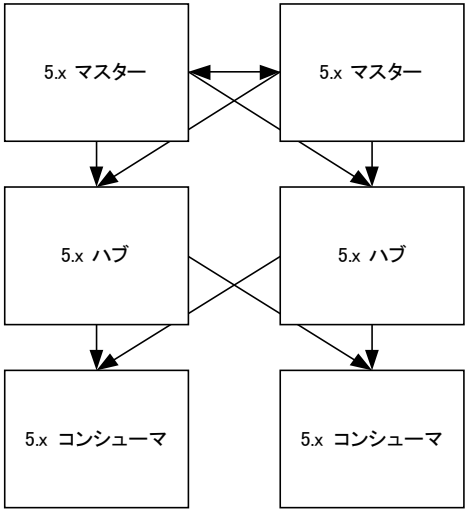
この手順が完了すると、更新プロセスが完了します。クライアントは、アップグレードされたレプリケーショントポロジでサーバーの使用を開始することができます。

5.x アップグレードの例

5.x デュアルマスターのアップグレードを 2 つのコンシューマに供給する 2 つのハブにレプリケートする場合について検討します。この節では、5.2 サーバーを使用するトポロジへアップグレードするための手順を示します。

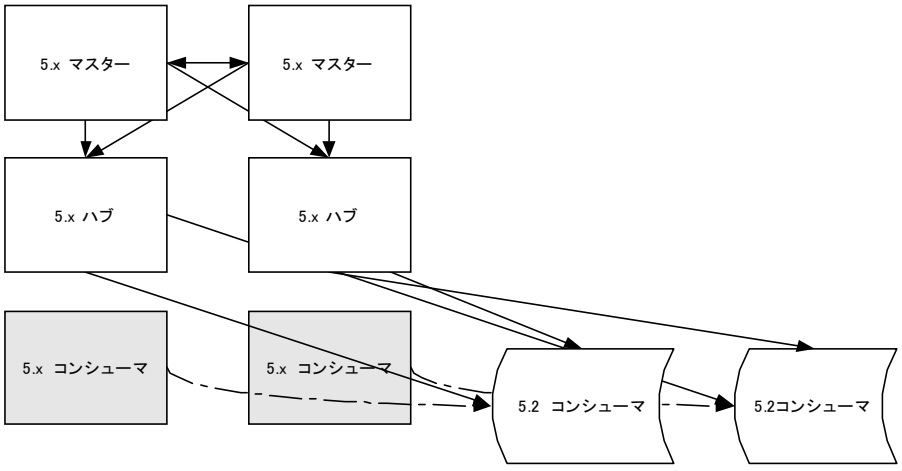
図 2-11 は、アップグレードする前の 5.x トポロジを示します。

図 2-11　既存の 5.x トポロジの例



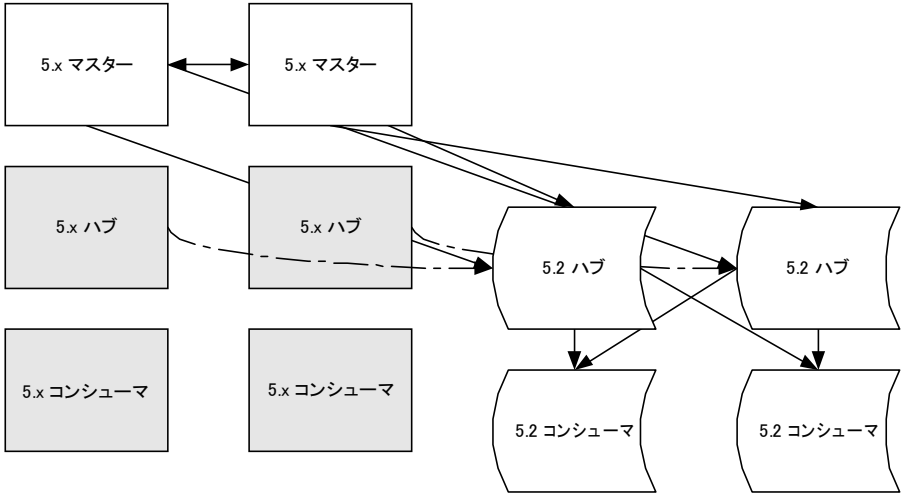
最初の手順では、コンシューマのアップグレードが行われます。図 2-12 は、結果として作成されたトポロジを示します。

図 2-12　5.x コンシューマのアップグレード手順の例



次の手順では、ハブのアップグレードが行われます。図 2-13 は、その結果を示します。

図 2-13　5.x ハブのアップグレード手順の例



次の手順では、マスターのアップグレードが行われます。図 2-14 は、その結果を示します。

図 2-14　5.x マスターのアップグレードの例 ― 手順 3

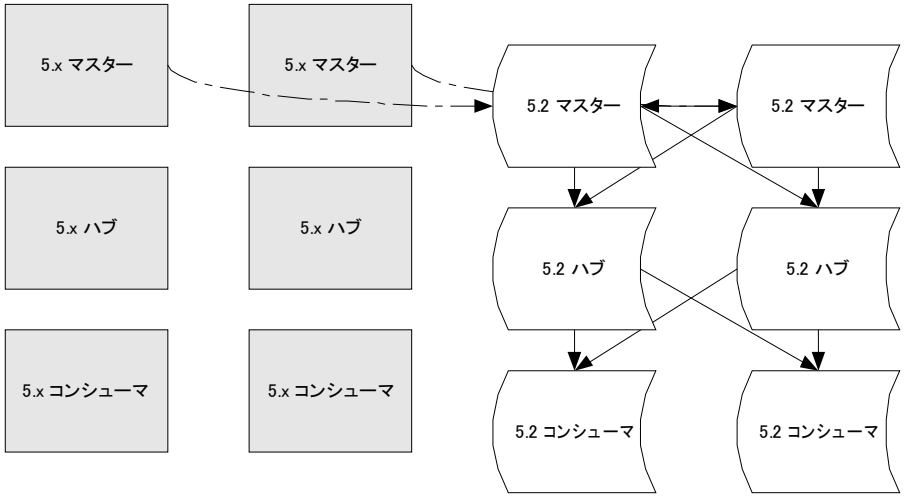
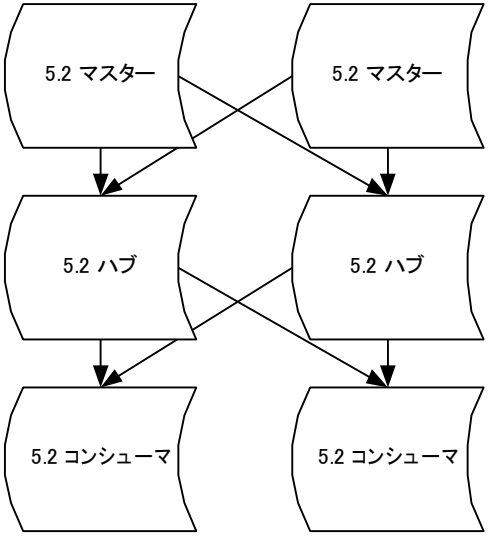


図 2-15 は、アップグレード後の 5.2 トポロジを示します。この時点では、古いトポロジのサーバーは解除され、新しいサーバーが 5.2 トポロジに追加されています。

図 2-15　アップグレード後の 5.2 トポロジの例



ここで、クライアントの要求が 5.2 トポロジに転送されています。

第 2 部

チューニング

第 3 章「チューニングのヒント」

第 4 章「ハードウェアのサイジング」

第 5 章「オペレーティングシステムのチューニング」

第 6 章「キャッシュサイズのチューニング」

第 7 章「インデックスのチューニング」

第 8 章「ログのチューニング」

第 9 章「その他のリソースの使用の管理」

第 3 章

チューニングのヒント

パフォーマンスをチューニングすることは、特定の導入要件を反映するためにデフォルトの設定を変更することを意味します。

このマニュアルでは、単一の Directory Server インスタンスをチューニングする方法について説明します。ここでの説明は、レプリケーショントポロジを含めたディレクトリサービス全体の設計がすでに完了され、ここに記載された情報を使って、その設計要件を満足できるように Directory Server インスタンスをチューニングすることを前提にしています。ディレクトリサービス全体の設計をまだ行なっていない場合は、『Sun ONE Directory Server Deployment Guide』を参照し、設計方法を確認してください。

パフォーマンスのチューニングには、表 3-1 に示すように、時間と労力をかけて検討することが必要です。

表 3-1　チューニングのプロセス

段階	内容
ゴールの定義	導入要件を基に、具体的に測定可能なチューニング目標を定義する。次のような質問を検討する • どのアプリケーションが Directory Server を使用するのか • システムは Directory Server 専用か。ほかのアプリケーションが実行されていないか。その場合どのアプリケーションか • 導入で必要となるエントリの数はいくつか。それらのエントリの大きさはどのぐらいか • Directory Server がサポートする必要のある 1 秒あたりの検索回数は何回か。どの検索タイプが予想されるか • Directory Server がサポートする必要のある 1 秒あたりの更新回数は何回か。どの更新タイプが予想されるか • 更新や検索の割合のピークはどの程度が予想されるか。平均の割合はどの程度が予想されるか • このシステムでは導入で繰り返し一括インポートの初期化が必要となるか。その場合、インポートはどのぐらいの頻度で実行されるか。一度にインポートされるエントリ数はいくつか。エントリのタイプは何か。初期化は、実行中のサーバーを使用してオンラインで行う必要があるか このリストがすべての質問ではないので、自分に必要な質問事項を確認する
方法の選択	チューニングの最適化の実装計画と、最適化を測定および分析する方法の計画を決定する システムのハードウェア設定は変更可能か。既存のハードウェアを使用して、オペレーティングシステムと Directory Server だけの調整に限定するか。ほかのアプリケーションはどのようにシミュレートするか。テストに使用する典型的なデータのサンプルはどのように生成するか。結果をどのように測定するか。結果をどのように分析するか
テストの実行	計画したテストを実行する。大規模で複雑な導入では、この段階にとても時間がかかることがある

表 3-1 チューニングのプロセス (続き)

段階	内容
結果の検証	最適化されている可能性のあるテストを行なった結果、このプロセスの最初に定義した目標を達成しているかどうかをチェックする
	目標を達成している場合は、結果を記録する
	目標を達成していない場合は、チューニングしている Directory Server をプロファイルし、監視する
プロファイルと監視	可能な変更を適用したら Directory Server の動作をプロファイルし、監視する。関連動作をすべて測定し、収集する
プロットと分析	プロファイルと監視の間に観察された動作をプロットし、分析する。詳細なテストを示唆する証拠やパターンを探してみる
	多くのデータを集めるために、プロファイルと監視の段階に戻ることが必要になる場合がある
変更とチューニング	測定結果の分析から考えられる追加の最適化を適用する
	テストを実施する段階に戻る
結果の記録	適用した最適化によって、プロセスの最初に定義した目標に到達した場合は、同じ結果をすぐに再現できるように記録を残す

この章では、Directory Server インスタンスのほとんどのチューニングで該当する基本的な要件を一覧にして表示しています。ここで提示した要件は一般に有効なものですが、特定の導入に与える影響を理解しないまますぐに適用することは避けてください。この章はチェックリストとして使用されることを想定しています。内容を理解せずにそのまま使用しないでください。

1. キャッシュサイズを調整します。

   サーバーには、Directory Server で使用されるキャッシュをすべて保持するのに十分な利用可能物理メモリがあることが理想的です。この場合、エントリキャッシュにはディレクトリ内のすべてのエントリを保持するのに十分なサイズを設定し、データベースキャッシュにはすべてのインデックスを保持するのに十分なサイズを設定します。

   詳細は、第 6 章「キャッシュサイズのチューニング」を参照してください。

2. インデックスを最適化します。

a. 不要なインデックスを削除し、予想される要求に対応するためのインデックスを追加します。

   新しいアプリケーションからの要求をサポートするインデックスを追加する必要が生じる場合があります。Directory Server の実行中にインデックスを追加、削除、および修正することは可能ですが、既存のデータはその時点から時間経過に従って段階的にインデックスされるだけです。

   詳細は、130 ページの「利点 : 検索でインデックスを使用する方法」と 132 ページの「コスト : 更新がインデックスに与える影響」を参照してください。

b. インデックスされた検索だけを許可します。

   インデックスされていない検索では、サーバーのパフォーマンスに強い悪影響を与えるおそれがあり、サーバーリソースを著しく消費する可能性があります。アプリケーションで実行する特定の検索をサポートするインデックスを追加し、インデックスされていない検索を拒否するようにサーバーで設定することを検討します。

   詳細は、140 ページの「インデックス検索だけを許可する」を参照してください。

c. インデックスリストの最大長を調整します。

   詳細は、140 ページの「インデックスリストの長さを制限する」を参照してください。

3. 基盤となるオペレーティングシステムをチューニングします。

   詳細は、第 5 章「オペレーティングシステムのチューニング」を参照してください。

4. 操作の上限を調整します。

   操作の上限を調整すると、Directory Server で単一の操作に過度のリソースを割り当てることがなくなります。強化された機能を要求するクライアントアプリケーションに一意のバインド DN を割り当て、それらの一意のバインド DN に固有のリソース制限を設定することを検討します。

   詳細は、第 9 章「その他のリソースの使用の管理」を参照してください。

5. 不必要なログを無効にします。

   ディスクアクセスは、メモリアクセスよりもかなり速度が遅いです。頻繁にディスクのログファイルに書き込むと、パフォーマンスに強い悪影響を与えるおそれがあります。可能ならば、アクセス、エラー、監査の各ログは、必要がないときにオフにして、ディスクに書き込まないようにします。少なくとも、別のコントローラを使用する別のディスクにログファイルを配置することで、ログの影響を減らすようにしてください。

   詳細は、第 8 章「ログのチューニング」を参照してください。

6. ディスク活動を分散させます。

   特に大量の更新をサポートする導入の場合、Directory Server では大量のディスク I/O を集中的に使用します。可能ならば、別々のコントローラを使用する複数ディスクに負荷を分散させることを検討します。

   詳細は、88 ページの「ディスクサブシステムのサイジング」を参照してください。

第 4 章

ハードウェアのサイジング

適切にハードウェアをサイジングすることは、ディレクトリサービスの計画と導入において必須事項です。ハードウェアのサイジングには、利用可能なメモリ容量と利用可能なローカルディスクの空き容量がきわめて重要になります。

注 最良の結果を得るために、実際の運用で使用されるエントリに相当するエントリのサブセットを用いて、テストシステムをインストールし、設定します。次に、テストシステムを使用して、実際の運用サーバーに近い動作を行います。

特定のシステムに対して最適化する場合は、Directory Server をサポートするためのチューニングの際に I/O サブシステムの機能を利用できるように、システムバス、周辺機器バス、I/O デバイス、およびサポートされるファイルシステムがどのように動作するかを確認します。

この章では、Directory Server インスタンスのディスクとメモリの要件を見積もる方法を紹介します。また、ネットワーク要件や SSL アクセラレータハードウェアの要件についても説明します。

推奨される最小要件

表 4-1 に、実際の運用環境でソフトウェアをインストールして使用するためのメモリとディスクの最小要件を示します。

指定したエントリ数における最小要件は、実際には表 4-1 で示す要件と異なることがあります。ここでのサイズは、比較的小さなエントリで、デフォルトの設定によるインデックスと最小限チューニングされたキャッシュを用いる場合です。デジタル写真などの大きなバイナリ属性値がエントリに含まれる場合や、インデックスまたはキャッシュに異なる設定を用いている場合は、それに応じてディスク容量とメモリの最小見積もりをそれぞれ上方修正します。

表 4-1　ディスク容量とメモリの最小要件

ケース	ローカルディスクの空き容量	空き RAM
製品の解凍	最低 125M バイト	-
製品のインストール	最低 200M バイト	最低 256M バイト
10,000 ～ 250,000 エントリ	最低 3G バイト追加	最低 256M バイト追加
250,000 ～ 1,000,000 エントリ	最低 5G バイト追加	最低 512M バイト追加
1,000,000 エントリ以上	8G バイト以上追加	1G バイト以上追加

ディスク容量の最小要件には、アクセスログ専用の 1G バイトが含まれています。デフォルトでは、Directory Server は 10 個 (`cn=config` における `nsslapd-accesslog-maxlogsperdir`) のアクセスログファイルをローテーションするように設定されており、それぞれのログファイルは最大 100M バイト (`cn=config` における `nsslapd-accesslog-maxlogsize`) のメッセージを保持します。エラーログと監査ログの大きさは、Directory Server の設定状況によって変わります。ログの設定の詳細は、『Sun ONE Directory Server 管理ガイド』を参照してください。

最小の利用可能なメモリ

メモリの最小見積もりは、通常の導入における Directory Server インスタンスで使用されるメモリを反映しています。見積もりには、システムやほかのアプリケーションで使用されるメモリは含まれていません。より正確な見積もりを得るには、メモリの使用量を経験的に測定する必要があります。詳細は、84 ページの「物理メモリのサイジング」を参照してください。

原則として、利用可能なメモリ量が多ければ多いほど良い状況であるといえます。

最小のローカルディスクの空き容量

ローカルディスクの空き容量の最小見積もりは、典型的な導入における Directory Server インスタンスで必要となる空き容量を反映しています。経験上、ディレクトリエントリが大きい場合、必要な空き容量は、ディスク上にある同等の LDIF のサイズを少なくとも 4 倍したサイズになります。詳細は、88 ページの「ディスクサブシステムのサイジング」を参照してください。

ネットワークディスクにアクセスするサーバーやデータをインストールしないでください。Sun ONE Directory Server では、NFS、AFS、または SMB を介してネットワークに接続されたストレージの使用をサポートしていません。設定、ログ、データベース、およびインデックスファイルはすべて、インストールを終えてからも、常にローカルストレージ上に配置する必要があります。

Windows システムの場合は、ドライブは FAT ではなく NTFS でフォーマットします。FAT は Directory Server での使用がサポートされていません。NTFS では、アクセス制御をファイルやディレクトリに設定することができます。

最小プロセッサ能力

大ボリュームシステムでは、通常、複数の高速プロセッサを搭載することで、複数の同時検索、拡張インデックス、レプリケーション、およびその他の機能に適切な処理能力を実現しています。詳細は、97 ページの「マルチプロセッサシステムのサイジング」を参照してください。

最小ネットワークキャパシティ

テストによると、想定される最大スループットによっては、サービスプロバイダレベルのパフォーマンスでも 100M ビット Ethernet で十分であることが示されています。論理的な最大スループットは、次のようにして見積もることができます。

最大スループット = 返される最大エントリ数 / 秒×平均エントリサイズ

たとえば、Directory Server がピーク時には 1 秒あたり 5000 個の検索に対応しなければならず、平均サイズが 2000 バイトのエントリを返すとします。このとき論理的な最大スループットは 10M バイト、つまり 80M ビットとなります。しかし 80M ビットの方が、100M ビット Ethernet アダプタ 1 つで実現するスループットよりも大きくなる傾向があります。実際に測定されるパフォーマンスは異なることがあります。

WAN (広域ネットワーク) でマルチマスターレプリケーションを実行したい場合は、その接続で、待ち時間が最小でパケット損失がほとんどない十分なスループットが得られることを確認します。

詳細は、98 ページの「ネットワークキャパシティのサイジング」を参照してください。

物理メモリのサイジング

Directory Server では、データベーステクノロジを使用して情報を格納します。データベーステクノロジに依存するあらゆるアプリケーションでは、高速なメモリが十分あることが Directory Server のパフォーマンスを最適化する上で重要です。原則として、利用可能なメモリが多ければ多いほど、すばやくアクセスするためにキャッシュされるディレクトリ情報も多くなります。ディレクトリ内容全体を常にキャッシュするのに十分なメモリが各サーバーにあるのが理想的です。Sun ONE Directory Server 5.2 では 64 ビットメモリアドレッシングをサポートしているため、キャッシュサイズは数 G バイトに制限されていません。64 ビットアーキテクチャでは、合計で 1.5T バイト以上のキャッシュサイズを扱うことが論理的に可能になりました。

注 Directory Server を実際の運用環境で導入するときは、キャッシュサイズを論理的な処理上限よりも小さく設定し、通常のシステム操作で使用できる適切なリソースを確保します。

Directory Server を実行するのに必要なメモリサイズを見積もるには、特定の Directory Server 設定と、Directory Server が実行されるシステムの両方とで必要なメモリを見積もります。

Directory Server 用メモリのサイジング

特定の導入に対して見積もった設定値が決まると、Directory Server インスタンスに必要な物理メモリを見積もることができます。表 4-2 に、この節で説明する計算で使用した値をまとめて表示します。

表 4-2 Directory Server 用メモリのサイジングの値

値	内容[1]
`nsslapd-cachememsize`	サフィックスのエントリキャッシュサイズ エントリキャッシュには形式化されたエントリが含まれており、クライアント要求に応答して送信されることになる。1 インスタンスで複数のエントリキャッシュを処理できる
`nsslapd-dbcachesize`	データベースキャッシュサイズ データベースキャッシュはサーバーで使用されるデータベースとインデックスからの要素を保持する

表 4-2　Directory Server 用メモリのサイジングの値 (続き)

値	内容[1]
nsslapd-import-cachesize	一括インポート用のデータベースキャッシュサイズ インポートキャッシュは、エントリのインポート時にのみ使用される。インポートキャッシュ用に余分なメモリを確保する必要はないが、オフラインインポートだけを実行する場合は、エントリキャッシュやデータベースキャッシュ用に確保したメモリを再利用する
nsslapd-maxconnections	管理される接続の最大数
nsslapd-threadnumber	サーバーの起動時に作成される操作スレッド数

1. 完全な説明は、『Sun ONE Directory Server Reference Manual』を参照してください。

おおよそのメモリサイズを見積もるには、次の手順を実行します。

1. サーバープロセスの基本サイズ slapdBase を見積もります。

```
slapdBase      = 75MB +(nsslapd-threadnumber x 0.5MB) +(nsslapd-maxconnections x
0.5 KB)
```

2. エントリキャッシュサイズの合計 entryCacheSum を決定します。

```
entryCacheSum = Sum全エントリキャッシュ (nsslapd-cachememsize)
```

3. 全キャッシュの合計サイズ cacheSum を決定します。

```
cacheSum       = entryCacheSum + nsslapd-dbcachesize + nsslapd-import-cachesize
```

4. Directory Server プロセスの合計サイズ slapedSize を決定します。

```
slapdSize      = slapdBase + cacheSum
```

Directory Server で使用される物理メモリを測定するのに、Solaris システムの pmap(1) などのユーティリティまたは Windows のタスクマネージャを使用することができます。

5. 着信クライアント要求を処理するのに必要なメモリ slapdGrowth を見積もります。

```
slapdGrowth    = 20% x slapdSize
```

最初の見積もりなので、クライアント要求を処理するためのオーバーヘッドとして 20% を想定します。実際の割合は、その導入の特徴によって異なる可能性があります。この割合は Directory Server を運用環境に置く前に、実際に検証します。

6. Directory Server の合計メモリサイズ slapdTotal を決定します。

```
slapdTotal     = slapdSize + slapdGrowth
```

32 ビットサーバーを含む大規模導入では、slapdTotal の値が実際の上限である約 3.4G バイトを超えたり、論理的な処理上限である約 3.7G バイトを超えることもあります。この場合、第 6 章「キャッシュサイズのチューニング」に示されているように、キャッシュをチューニングするか、システムの制限内で動作させるか、あるいは製品の 64 ビットバージョンを使用することができます。

オペレーティングシステム用メモリのサイジング

オペレーティングシステムを実行するのに必要なメモリを見積もるには、経験的に行うことが必要となり、オペレーティングシステムのメモリ要件は、固有のシステム設定に基づいて大きく異なります。そのため、オペレーティングシステムで必要なメモリ量を見積もろうとする前に、第 5 章「オペレーティングシステムのチューニング」で説明するように、導入に対して典型的なシステムでチューニングすることを検討します。システムをチューニングしたら、最初の見積もりの systemBase に到達するまで、メモリの使用状況を監視します。メモリの使用状況を監視するには、Solaris の sar (1M) などのユーティリティまたは Windows タスクマネージャを使用することもできます。

注 最高のパフォーマンスを得るために、Directory Server を実行するシステムはこのサービス専用で使用します。

その他のアプリケーションやサービスを実行する必要がある場合は、必要な合計メモリをサイジングするときに、使用するメモリも監視します。

さらに、一般的なシステムオーバーヘッドと通常の管理目的とにメモリを割り当てます。systemOverhead の最初の見積もりは、大きさに関係なく、少なくとも数百 M バイト、または合計物理メモリの 10% のいずれかになります。システムが実際の運用環境でメモリのページインやページアウトを行わないよう、systemOverhead に十分な容量を割り当てることが目的です。

オペレーティングシステムで必要な合計メモリ systemTotal は、次のようにして見積もることができます。

```
systemTotal = systemBase + systemOverhead
```

合計メモリのサイジング

前述の節で見積もった `slapdTotal` および `systemTotal` を使用して、必要な合計メモリ `totalRAM` を見積もります。

```
totalRAM = slapdTotal + systemTotal
```

`totalRAM` は必要な合計メモリの見積もりであり、システムが Directory Server プロセス専用であることが前提です。また、この値には、システム上で実行する可能性のあるすべてのアプリケーションとサービスに対するメモリ使用量の見積もりが含まれています。

メモリ不足での処理

多くの場合、Directory Server で使用される全データをキャッシュするのに十分なメモリを用意するのは、費用効率は良くありません。

少なくとも Directory Server を実行するのに十分なメモリをサーバーに搭載すれば、継続的なページスワップは発生しません。継続的にページスワップすることは、パフォーマンスに強い悪影響があります。Directory Server を起動してエントリキャッシュをプライムする前後でメモリ統計を表示するには、Solaris やその他のシステムの `vmstat` (1M) などのユーティリティを使用することができます。Solaris システムの `MemTool` など、個別には利用可能だがサポートされていないユーティリティも、アプリケーションがテストシステムで実行されているときに、メモリの使用状況や割り当て状況を監視するのに役立ちます。

システムに追加のメモリを搭載できない場合は、これまでどおり継続的なページスワップを監視し続け、データベースキャッシュやエントリキャッシュのサイズを減らします。スワップスペースが不足すると、Directory Server はクラッシュします。

全ディレクトリデータをキャッシュするのに十分な物理メモリを確保できないときの可能な代替策は、第 6 章「キャッシュサイズのチューニング」を参照してください。

ディスクサブシステムのサイジング

ディスクの使用と I/O 能力は、パフォーマンスに強く影響します。特に大量の変更をサポートする導入では、ディスクサブシステムは I/O のボトルネックとなる可能性があります。この節では、Directory Server インスタンスで全体のディスク容量を見積もったり、ディスク I/O のボトルネックを軽減したりするための推奨について説明します。

ディスク I/O のボトルネックを軽減する詳細は、第 8 章「ログのチューニング」を参照してください。

ディレクトリサフィックスのサイジング

サフィックスに対するディスク容量の要件は、ディレクトリ内のエントリのサイズや個数によって変化するだけでなく、ディレクトリ設定や、特にサフィックスのインデックス方法によっても変化します。大規模導入で必要なディスク容量を測定するには、次の手順を実行します。

1. 導入で想定される 3 つの代表的なエントリのセットである 10,000 エントリ、100,000 エントリ、1,000,000 エントリの各セットに対して LDIF を生成します。

   生成したエントリは想定されるエントリタイプ (ユーザー、グループ、ロール、拡張スキーマのエントリ) の組み合わせを反映するとともに、特に `userCertificate` や `jpegPhoto` のように単一の大きな属性値が想定される場合の個々の属性値の平均サイズも反映する必要があります。

2. Directory Server のインスタンスを導入で想定されるように設定します。

   特に、実際の運用ディレクトリに対して実行するようにデータベースをインデックスします。後でインデックスを追加する可能性がある場合は、追加するインデックスのスペースも追加する必要があります。

3. 各エントリセットをロードし、各セットで使用されるディスク容量を記録します。
4. 結果をグラフにし、導入におけるサフィックスサイズの見積もりを推定します。
5. エラーや変更のために使用されるディスク容量を追加します。

サフィックスのディスク容量は、全体の一部にすぎません。同様に、Directory Server のディスク使用状況についても検討する必要があります。

Directory Server のディスク使用状況

ディレクトリのサフィックスは、Directory Server によってディスクに格納されるものの一部です。ディスク使用に影響するその他の多くの要因は、導入後の Directory Server の使用状況によっても大きく異なります。そのため、ここでは一般的な状況について説明します。ここで説明する項目を設定する手順は、『Sun ONE Directory Server 管理ガイド』を参照してください。

Directory Server バイナリ

このバージョンの Directory Server をインストールするには、約 200M バイトのディスク容量が必要です。ただし、この見積もりにはデータに対する容量は含まれていません。含まれているのは製品のバイナリに対する容量だけです。

イベントログ

ログファイルのディスク使用の見積もりは、Directory Server 動作の割合、ログのタイプとレベル、およびログローテーションの方法によって変わります。

多くのログ要件は予測でき、事前に計画をたてることができます。Directory Server でログや特に監査ログに書き込みが行われる場合、負荷レベルとともにディスクの使用状況が増大します。高負荷の導入で大規模なログを必要とするときは、高負荷に適応するように、ディスク容量を追加することを計画します。高負荷のログで導入に必要なディスク容量を減らすには、インテリジェントログローテーションおよび保管システムを構築し、頻繁にログをローテーションします。そして古いファイルはテープや、より安価なディスククラスタなど費用がかからず大容量のストレージメディアに自動的に移行させます。

ログ要件は簡単に予測できないこともあります。たとえばログをデバッグすると、一時的ですが急激にエラーログが増大します。大規模で高負荷の導入では、一時的で大量のデバッグログ専用に、数 G バイトのディスク容量を別に設定することを検討します。詳細は、第 8 章「ログのチューニング」を参照してください。

トランザクションログ

トランザクションログの量は、ピーク時の書き込み負荷によって変わります。書き込みが急激に起こる場合は、書き込み負荷が一定である場合と比べて、トランザクションログで使用する容量が多くなります。Directory Server では、トランザクションログを定期的に削除しています。そのため、トランザクションログはチェックなしで増大し続けることはありません。ただし、オンラインバックアップ中は、トランザクションログはフラッシュされません。

Directory Server では、一般的に永続トランザクションが有効な状態で実行されています。永続トランザクション機能が有効であると、Directory Server では、変更操作 (`add`、`delete`、`modify`、および `modrdn`) ごとにトランザクションログへ同期書き込みを行います。この場合、ディスクがビジーの場合は操作がブロックされ、潜在的な I/O ボトルネックとなります。

更新パフォーマンスが重要な場合は、高速な書き込みキャッシュを持つディスクサブシステムをトランザクションログ用に使用することを計画します。詳細は、第 8 章「ログのチューニング」を参照してください。

レプリケーションの更新履歴ログデータベース

Directory Server 導入時にレプリケーションを行う場合は、更新履歴ログを記録します。更新履歴ログのサイズは、変更される量と、使用された更新履歴ログのタイプによって異なります。変更履歴ログの削除方法に応じて、容量を計画します。大規模で高負荷の導入では、変更率が異常に高いときの変更履歴ログの増加を処理するために、数 G バイトのディスク容量を別に設定することを検討します。詳細は、第 8 章「ログのチューニング」を参照してください。

サフィックスの初期化と LDIF ファイル

サフィックスの初期化は一括ロードまたは一括インポートとも呼ばれます。この最中、Directory Server で必要となるディスク容量は、サフィックスデータベースファイルとサフィックスの初期化に使用される LDIF ファイルの分だけではなく、初期化プロセス中に使用される中間ファイルの分も必要となります。サフィックスの初期化中に使用される LDIF ファイルと中間ファイル用に、追加の容量をデータベースファイルと同じディレクトリに一時的に確保することを計画します。

バックアップと LDIF ファイル

バックアップでディスク容量を大量に消費することは珍しくありません。バックアップのサイズは、関連するデータベースファイルのサイズと等しくなります。データベースファイルの量を数倍した容量を割り当てることで、複数のバックアップに対応します。これにより、データベースとそれに対応するバックアップが別々のディスクで維持されます。時間が経過するにつれてバックアップを安価なストレージメディアに移行させるには、優れたストラテジを使用します。

導入でレプリケーションを行う場合、初期化用の LDIF ファイルを保持するためのディスク容量を追加することを計画します。これは、初期化用の LDIF ファイルはバックアップ用の LDIF ファイルと異なるためです。

ディスクベースでなくメモリベースのファイルシステムを使用する

一部のシステムでは、メモリベースの tmpfs ファイルシステムをサポートしています。たとえば Solaris 上では、/tmp は、パフォーマンスを向上させるために、メモリベースのファイルシステムとしてマウントされます。システムのほかのアプリケーションと共用されている /tmp にキャッシュファイルが置かれている場合は、/tmp 下で容量不足が起こらないように注意する必要があります。このようにしないと、メモリ量が低下し、メモリベースのファイルシステム内の Directory Server ファイルが、スワップパーティション専用のディスク容量にページングされる可能性があります。

一部のシステムは、RAM ディスクやその他のメモリベースファイルシステムをサポートしています。メモリベースのファイルシステムを作成および管理する手順については、オペレーティングシステムのマニュアルを参照してください。そのようなファイルシステム内のデータはすべて揮発性であり、システムリブート後にそれらのデータをメモリ内に再ロードする必要があることに注意してください。

(UNIX プラットフォーム) core ファイル

少なくとも core ファイル 1 ～ 2 個分の空きスペースを残しておきます。Directory Server でのコアダンプはお勧めしませんが、クラッシュ中に生成された core ファイルを調査に使用できる場合は、クラッシュ後の回復とトラブルシューティングが極めて簡単になる場合があります。core ファイルは生成されると、cn=config の場合に nsslapd-errorlog で指定したファイルと同じディレクトリに格納されます。起動中にクラッシュした場合は *ServerRoot*/bin/slapd/server/ に格納されます。

管理用のディスク容量

システムと Directory Server の管理など、予期されるシステムの使用に対して空き容量を確保します。基本となる Directory Server インストール、ローカルインスタンス上に置かれたときの設定サフィックス、設定ファイルなどに十分なディスク容量を確実に割り当てます。

ディスク間のファイルの分散

当然のように更新される Directory Server データベースやログファイルを別のディスクサブシステムに配置することで、複数のディスクスピンドルやコントローラ間に I/O トラフィックの範囲を広げることができ、I/O ボトルネックを回避できます。次の各項目に対して、専用のディスクサブシステムを用意することを検討します。

トランザクションログ

永続トランザクション機能が有効であると、Directory Server では、変更操作ごとにトランザクションログへ同期書き込みを行います。そのため、ディスクがビジーであると、操作がブロックされてしまいます。トランザクションログを専用ディスクに配置することで書き込みパフォーマンスが向上し、Directory Server で処理可能な変更率が増加します。

詳細は、153 ページの「トランザクションログ」を参照してください。

データベース

複数のデータベースをサポートすることで、各データベースをそれ専用の物理ディスクに置くことが可能になります。そのため、個々のデータベースが専用のディスクサブシステム上にある場合、Directory Server の負荷を複数のデータベース間に分散させることができます。データベース操作の I/O 競合を防ぐには、データベースファイルの各セットを個別のディスクサブシステムに配置することを検討します。

最高のパフォーマンスを得るには、データベースファイルを大きい I/O バッファを持つ専用高速ディスクサブシステムに配置します。キャッシュに候補エントリが見つからない場合、Directory Server はディスクからデータを読み込み、定期的に書き込みをフラッシュします。これらの操作で専用の高速ディスクサブシステムを使用すると、潜在的な I/O ボトルネックが軽減されます。

`cn=config,cn=ldbm database,cn=plugins,cn=config` における `nsslapd-directory` 属性では、Directory Server がインデックスファイルを含むデータベースファイルを格納するディスク位置を指定します。これらのファイルはデフォルトで、*ServerRoot*`/slapd-`*ServerID*`/db/` にあります。

当然のことながら、データベース位置の変更では、Directory Server の再起動だけでなく、データベースの完全な再構築も必要となります。実際の運用サーバーでデータベース位置を変更することは、大掛かりな作業になる場合があるため、最も重要なデータベースを特定し、それを別のディスク上に配置した後で、サーバーの運用を開始してください。

ログファイル

Directory Server には、アクセス、エラー、および監査の各ログが用意されており、バッファのあるログ能力を備えています。バッファがあるにもかかわらず、これらのログファイルに書き込みをするには、ほかの I/O 操作と競合する可能性のあるディスクアクセスが必要です。パフォーマンス、容量、および管理を改善するには、ログファイルを別々のディスクに配置することを検討します。

詳細は、第 8 章「ログのチューニング」を参照してください。

メモリベースファイルシステム上のキャッシュファイル

`tmpfs` ファイルシステムでは、たとえば、物理メモリが使い果たされたときだけファイルがディスクにスワップされます。すべてのキャッシュファイルを物理メモリに保持するのに十分なメモリがあれば、`tmpfs` ファイルシステム (Solaris プラットフォーム)、または RAM ディスクなどのその他のメモリベースファイルシステム (その他のプラットフォーム) に同じサイズのディスク容量を割り当て、Directory Server がそのファイルシステムにキャッシュファイルを格納するように `nsslapd-db-home-directory` の値を設定することで、パフォーマンスの向上が得られます。これにより、システムでは、キャッシュファイルにマッピングされたメモリが、不必要にディスクへフラッシュしなくなります。

ディスクサブシステムの選択

「Fast, cheap, safe: pick any two (速度、低コスト、安全性 : いづれか 2 つを選択)」— 『Sun Performance and Tuning』、Cockroft と Pettit 著

速度と安全性

パフォーマンスと稼働時間の両方が重要な導入を実装するときは、大規模のディスクアレイ間に分散した I/O を高速でバッファできるように、不揮発性のメモリキャッシュを備えているハードウェアベースの RAID コントローラを検討します。負荷を多くのディスクで分散させ、高速接続でアクセスをバッファすることで、I/O は最適化され、高パフォーマンスの RAID ストライピングやパリティ ブロックによって極めて安定します。

Sun StorEdge™ 製品で提供されているような大容量で不揮発性 I/O バッファや高パフォーマンスのディスクサブシステムによって、Directory Server のパフォーマンスや稼働時間は大きく向上します。

高速な書き込みキャッシュカードでは、特にトランザクションログ専用としたときに、書き込みパフォーマンスが向上する可能性があります。高速な書き込みキャッシュカードには、ディスクコントローラから独立した不揮発性のメモリキャッシュが備わっています。

速度と低コスト

高速で低コストのパフォーマンスを得るには、十分なディスク容量が複数ディスク間に分散されていることを確認します。回転速度が速く、シークタイムが短いディスクの使用を検討します。最高の結果を得るには、分散されたコンポーネントそれぞれに対して 1 つのディスクを専用にします。シングルポイントのエラーを避けるには、マルチマスターレプリケーションを使用することを検討します。

低コストと安全性

安価で安全な設定には、Solaris Volume Manager のように低コストでソフトウェアベースの RAID コントローラの使用を検討します。

RAID の選択

RAID とは Redundant Array of Inexpensive Disks (安価なディスクを複数使用した冗長アレイ) のことです。名前が示すとおり、RAID の第 1 の目的は回復力を備えることです。アレイ内のディスクで障害が発生しても、そのディスク上のデータは失われずに、アレイ内のほかのディスクで依然として利用できます。回復力を実装するために、RAID では抽象概念を持ち込み、通常は単一のボリュームとして参照される大容量の仮想ディスクとして、複数のディスクドライブを設定することができます。これは、物理ディスクを連結、ミラーリング、またはストライピングすることで実現されます。連結の実装では、あるディスクのブロックの後に別のディスクのブロックを論理的に続けます。たとえば、ディスク 1 にはブロック 0 ～ 99、ディスク 2 にはブロック 100 ～ 199、というようになります。ミラーリングの実装では、あるディスクのブロックを別のディスクにコピーし、その後は同期し続けます。ストライピングでは、複数の物理ディスクに仮想ディスクのブロックを分散させるアルゴリズムを使用します。

ストライピングの目的はパフォーマンスを得ることです。ランダム書き込みでは、書き込まれるデータがストライピングボリュームの複数のディスクに送信される可能性があるため、非常に迅速に処理されます。したがって、ディスクは並列で動作することが可能です。ランダム読み込みにも同じことが言えます。大量のシーケンシャルな読み込みと書き込みの場合は、それほど単純ではありません。しかし、シーケンシャルな I/O パフォーマンスも向上することが確認されています。たとえば、多くの I/O 要求を生成するアプリケーションでは、単一のディスクコントローラに集中することがあります。ストライピングボリュームのディスクすべてで専用のコントローラを使用している場合、このような集中は起こりにくく、パフォーマンスが向上します。

ソフトウェアとハードウェアのどちらの RAID 管理デバイスでも、RAID を実装することができます。それぞれの方法には利点と欠点があります。

- 一般にハードウェア RAID では、ハードウェアに実装されているため、高パフォーマンスが得られる。それにより、ソフトウェア RAID に比べて処理のオーバーヘッドが少なくて済む。さらに、ハードウェア RAID はホストシステムから分離されているため、アプリケーションを実行するためのホストリソースを消費しないで済む
- 一般にハードウェア RAID は、ソフトウェア RAID に比べて高価
- ソフトウェア RAID は、ハードウェア RAID に比べて柔軟性が高い。たとえば、通常、ハードウェア RAID マネージャは単一のディスクアレイや指定されたディスクアレイのセットと関連付けられている。一方、ソフトウェア RAID では、任意の個数のディスクアレイや、必要ならばアレイ内の特定のディスクだけをカプセル化できる

次の節では、レベルとして知られている RAID 設定について説明します。もっともよく使用される RAID レベルは 0、1、1+0、および 5 であり、これらについて詳細を説明しますが、あまり使用されないレベルについては比較と対照だけの対象とします。

RAID 0、ストライピングボリューム

ストライピングでは、データを複数の物理ディスクに分散します。論理ディスクや論理ボリュームは、チャンク (まとまり) やストライプ (しま状) に分割され、ラウンドロビン方式で物理ディスク上に分散されます。ストライプは常に 1 つ以上のディスクブロックから成り、すべてのストライプは同じサイズになります。

RAID 0 という名前は、冗長性がないことと矛盾しています。RAID 0 のストライプでディスク障害が発生すると、論理ボリューム全体が失われてしまいます。ただし、RAID 0 はすべての RAID レベルの中でもっとも安価であり、すべてのディスクをデータ専用にします。

RAID 1、ミラーボリューム

ミラーの目的は冗長性を実現することです。ミラー内のディスクの 1 つがエラーになっても、データは利用可能なままであり、処理を継続することができます。各物理ディスクがミラーされるため、物理ディスク容量の半分がミラー用に当てられてしまうのが欠点です。

RAID 1+0

RAID 10 とも呼ばれます。RAID 1+0 では、高レベルのパフォーマンスと回復力を備えています。その結果、RAID レベルの中では実装にもっとも費用がかかります。障害を起こしたすべてのディスクで別のミラーを作っている限り、最高で 3 ディスクが障害を起こしてもデータは利用可能なままです。RAID 1+0 はセグメントが RAID 1 であるストライピングアレイとして実装されます。

RAID 0+1

RAID 0+1 は RAID 1+0 に比べると回復力の点で劣っています。ストライプは作成されるとミラーされます。ミラーの同じ側にある 1 つ以上のディスクで障害が起きても、データは利用可能なままです。ただし、その後でミラーの別の側にあるディスクが障害を起こすと、論理ボリュームは失われます。RAID 1+0 とのわずかな違いは、RAID 1+0 では、どちらの側のディスクが同時に障害を起こしても、データが利用可能なままであるという点です。RAID 0+0 はセグメントが RAID 0 であるミラーアレイとして実装されます。

RAID 5

RAID 5 にはミラーリングのような回復力はありませんが、それにもかかわらず、単一ディスクが障害を起こしてもデータは利用可能なままであるという、冗長性を備えています。RAID 5 では冗長性を実現するために、論理エクスクルーシブを実行して作成されたパリティストライプや、ほかのディスク上で対応するストライプのバイトにあるパリティストライプを使用します。1 つのディスクが障害を起こすと、そのディスクのデータは、残りのディスクにある対応するストライプのデータとパリティを使用して、もう一度計算されます。ただし、このような補正計算を実行する必要がある場合は、パフォーマンスに影響します。

通常の運用では、RAID 5 は RAID 0、1+0、および 0+1 に比べてパフォーマンスが低いのが普通です。RAID 5 ボリュームでは論理書き込みのたびに 4 回の物理 I/O 操作を実行する必要があります。古いデータとパリティーが読み込まれ、エクスクルーシブまたは操作が 2 回実行され、そして新しいデータとパリティが書き込まれます。読み込み操作では、同じ負荷はかからないため、同数のディスクを使用した標準的なストライプの場合と比べてもパフォーマンスがわずかに低下するだけです。つまり RAID 5 ボリュームでは、パリティ専用のディスク容量があるため、ストライプ内には事実上 1 つ少ないディスクがあることになります。RAID 5 では利用可能なディスク容量以上をデータに使用するため、RAID 5 ボリュームは RAID 1+0 や 0+1 と比べて一般的に安価であるということです。

パフォーマンスの問題から、データが読み取り専用でない限り、またはボリュームへの書き込みが非常に少ない場合を除いて、通常は RAID 5 をお勧めしません。ただし、書き込みキャッシュと高速なエクスクルーシブまたは論理エンジンを備えたディスクアレイでは、パフォーマンスの問題を緩和することができ、導入によっては RAID 5 が安価な、ミラーに代わる実現可能な代替策となります。

RAID レベル 2、3、および 4

RAID レベル 2 および 3 は、ビデオストリーミングのように、大量のデータがシーケンシャルに転送される場合に適しています。どちらのレベルでも、1 度に I/O 操作は 1 回だけ処理可能で、ランダムアクセスを要求するアプリケーションには向いていません。RAID 2 はハミングエラー訂正コーディング (ECC) を使用して実装されます。3 つの物理ディスクドライブに ECC データを格納する必要があり、RAID 5 よりも高価

ですが、ストライプが 3 つより多くのディスクで構成されている場合は RAID 1+0 より安価になります。RAID 3 ではビットワイズパリティ方法を使用して冗長性を実現しています。パリティは、RAID 5 のようには分散されませんが、単一の専用ディスクに書き込まれます。

RAID 2 や RAID 3 とは異なり、RAID 4 では複数のディスクドライブに同時にアクセス可能な独立アクセス技術を使用しています。RAID 5 に似た方法でパリティを使用しますが、パリティが単一ディスクに書き込まれる点が異なります。そのため、書き込みごとにパリティディスクにアクセスされるため、パリティディスクがボトルネックとなり、事実上、複数の書き込みが直列処理されています。

ソフトウェアのボリュームマネージャ

Solaris™ Volume Manager のようなボリュームマネージャは、Directory Server のディスク管理でも使用できます。Solaris Volume Manager は、実際の運用環境への導入において、その他のソフトウェアのボリュームマネージャと好意的に比較されています。

I/O およびディスク使用の監視

通常の運用環境で、ディスクをいっぱいにしないでください。潜在的な I/O ボトルネックを分離するには、Solaris システムやその他のシステムにおける `iostat` (1M) などのユーティリティを使用することもできます。Windows システムの I/O ボトルネックへの対応方法の詳細は、Windows ヘルプを参照してください。

マルチプロセッサシステムのサイジング

Directory Server ソフトウェアは、複数のプロセッサ間でスケールされるように最適化されています。一般に、プロセッサを追加すると、全体的な検索、インデックスの保守、およびレプリケーションパフォーマンスが向上します。

しかし、特定のディレクトリの導入では、プロセッサを追加してもパフォーマンスが大幅に改善しない効果がなくなる点に達していることがあります。検索、インデックス、およびレプリカで多大に要求されるパフォーマンス要件を扱うときは、解決策の一部として、負荷分散やディレクトリプロキシなどの技術を検討します。

ネットワークキャパシティのサイジング

Directory Server はネットワークを集中利用するアプリケーションです。Directory Server インスタンスでネットワークの可用性を向上させるには、システムにネットワークインタフェースを 2 つ以上搭載します。Directory Server ではそのようなハードウェア設定をサポートしており、同一プロセス内の複数のネットワークインタフェースで待機します。

負荷分散のために、同じネットワーク上でディレクトリサーバーをクラスタにする場合は、新たに生じる負荷がネットワークインフラストラクチャで対応できることを確認します。更新率の高いレプリケーションを WAN 環境でサポートする場合は、経験テストを通じて、ネットワーク品質と帯域幅がレプリケーションのスループット要件を満たすことを確認します。

SSL 用のサイジング

ソフトウェアには、Secure Sockets Layer (SSL) プロトコルがデフォルトで実装されています。ソフトウェアベースの SSL 実装を使用すると、Directory Server パフォーマンスに重大な悪影響を及ぼすことがあります。SSL モードでディレクトリを実行するには、全体のパフォーマンス要件を満たすために、複数のディレクトリのレプリカの導入が必要な場合があります。

ハードウェアアクセラレータカードでは SSL を使用する影響を取り除くことはできませんが、ソフトウェアベースの実装の場合と比べてパフォーマンスが大幅に改善されます。Sun ONE Directory Server 5.2 では、サポートされている Sun Crypto Accelerator ハードウェアのような SSL ハードウェアアクセラレータを使用できます。

Sun Crypto Accelerator ボードの使用は、SSL 鍵計算がボトルネックになっている場合に有効です。ただし、SSL 鍵計算がボトルネックでない場合には、そのようなハードウェアを使用しても、パフォーマンスの改善は望めません。ハードウェアが実際に加速するのは、接続確立時の SSL ハンドシェイク中に行われる鍵計算の処理であり、接続確立後のメッセージの暗号化や復号化の処理ではないからです。このようなハードウェアを Directory Server インスタンスで使用する手順は、付録 B「Sun Crypto Accelerator ボードの使用」を参照してください。

第 5 章

オペレーティングシステムのチューニング

デフォルトのシステムとネットワーク設定は、高パフォーマンスディレクトリサービスに適していません。Directory Server のパフォーマンスを最適化するためのシステムのチューニングでは、少なくとも基本的なセキュリティレベルを確保し、システムとネットワーク設定の一部を変更する最新の推奨パッチがシステムにインストールされていることを確認する必要があります。この章では、これらのチューニングの問題について説明します。

製品に付属の `idsktune` ユーティリティ (および Solaris パッケージバージョンの `/usr/sbin/directoryserver idsktune`) は、基本的なシステム設定の欠陥を診断する場合に役立ちます。このユーティリティは、高パフォーマンスディレクトリサービスをサポートするシステムチューニングの推奨事項を提供します。実際には、ユーティリティにこれらの推奨事項は何も実装されていません。チューニングの推奨事項は、認定システム管理者が実装する必要があります。

プラットフォームのサポートの確認

18 ページの表 1-1 は、このリリースでサポートされるプラットフォームと関連するハードウェアーキテクチャを示します。サポートされるプラットフォームの更新リストについては、製品のリリースノートを参照してください。

Windows システムをインストールする場合は、ネットワークセキュリティサービスの依存度を下げるために、既存のドメインまたはワークグループのメンバーではない、スタンドアロンサーバーのコンピュータを指定します。

システムのパッチのインストール

システムのセキュリティ全体を管理して、Sun ONE Directory Server 5.2 の適切なインストールと操作を行うために、最新の推奨システムパッチ、サービスパック、または修正プログラムをインストールします。表 5-1 は、必要なパッチの検索場所を示します。

表 5-1　パッチの入手先 (プラットフォームごと)

プラットフォーム	参照
Sun Solaris™ オペレーティング環境	http://sunsolve.sun.com/
Hewlett Packard HP-UX	http://www.hp.com/support/
IBM AIX	http://www.ibm.com/jp/support/
Microsoft Windows	http://support.microsoft.com/
Red Hat Linux	http://www.jp.redhat.com/

基本的なセキュリティレベルの確保

この節で取り扱う推奨事項によって、すべてのリスクが排除されるわけではありません。ここで取り扱う推奨事項は、最も明らかなセキュリティリスクのいくつかを制限する作業に役立つ簡単なチェックリストとして意図されたものです。

システムの隔離

可能な場合には、ネットワークのファイアウォールを使用して、Directory Server を実行するシステムを、公共のインターネットから隔離します。システムの隔離は、IP ベースの攻撃を防護する必要のある Windows プラットフォームで Directory Server を実行する場合は特に重要です。

デュアルブートを導入しない

Directory Server を実行するシステム上で、デュアルブートあるいは別のオペレーティングシステムを実行しないでください。デュアルブートを導入すると、制限されたファイルに別のシステムがアクセスを許可する場合があります。

強固なパスワード

使用するスーパーユーザーまたは管理者用パスワードは、句読文字、またはアルファベット以外の文字を含む少なくとも 8 文字の長さにします。Windows プラットフォームで Directory Server を実行する場合は、簡単に推測されないパスワードを使用することが特に重要です。

長いオペレーティングシステムのパスワードを使用する場合は、システムがパスワードを処理する方法を設定することが必要になる場合があります。手順については、オペレーティングシステムのマニュアルを参照してください。

ローカルセキュリティポリシー (Windows の場合)

不正なログオンを試行したユーザーをロックアウトする Windows サーバーのローカルセキュリティポリシーを実装します。ポリシーの導入用に、適切なサイズのログを管理するためにイベントログを有効にして設定します。また、ログオン試行に対する監査ログを有効にします。管理者アカウント名を、部外者が推測できないような名前に変更することを検討します。

詳細は、Windows のヘルプを参照してください。

ユーザーとグループ (UNIX プラットフォームの場合)

セキュリティ上の理由から、Directory Server または管理サーバーをスーパーユーザー権限で実行しないことをお勧めします。たとえば、ログイン権限のないユーザーとグループを作成して、このユーザーとグループでサーバーをインストールして実行します。ユーザーとグループをローカルファイルに追加する場合、`/etc/passwd` エントリは次のようになります。

```
server:x:61001:Server User:/dev/null:/dev/null
```

対応する `/etc/group` エントリは次のようになります。

```
servers::61001:
```

デバッグを容易にするために、Solaris システムの `coreadm`(1M) などのユーティリティを使用して、このユーザーとグループ ID を使用してコアダンプを行うプロセスを実行することができます。

特定の導入の場合に、メッセージングサーバーなどほかのサーバーとの Directory Server ファイルの共有が必要な場合は、同じユーザーとグループを使用してこれらのサーバーを実行することを検討します。

管理サーバーをスーパーユーザーとして実行する必要がある場合、使用しないサービスの停止を検討します。

不必要なサービスの無効化

最高のパフォーマンスを達成しリスクを軽減するためには、システムを Directory Server 専用にします。特にネットワークサービスなどの追加サービスの実行は、サーバーのパフォーマンスと拡張性に悪い影響を与え、セキュリティリスクを増加させる場合があります。

できるだけ多くのネットワークサービスを無効にします。Directory Server は TCP/IP だけを使用するため、ファイルの共有やその他のサービスを必要としません。IP 転送、メール、NetBIOS、NFS、RAS、Web パブリッシング、Windows ネットワークライアントサービスなどのサービスを無効にします。特に Windows 上では、Event Log、Plug and Play、Protected Storage、セキュリティアカウントマネージャ、Sun ONE Administration Server、Sun ONE Directory Server、Remote Procedure Call (RPC)、および SNMP 以外のすべてのサービスを停止し無効にします。`telnet` と `ftp` の無効化も検討してください。

多くのネットワークサービスと同じように、`telnet` と `ftp` はセキュリティリスクを増大させます。これらの 2 つのサービスは、ネットワーク上でユーザーパスワードをクリアテキストで送信するため特に危険です。`telnet` および `ftp` サービスが必要な場合は、代わりに Secure Shell (`ssh`) および Secure FTP (`sftp`) などのクライアントを使用して対応することができます。

Directory Server インスタンス自体にネットワークのネームサービスが用意されていない場合は、システムのネームサービスを有効にすることを検討します。Sun ONE サーバーコンソールなどのリモート管理ツールは、IP アドレスとホスト名の変換など操作の一部の面でネームサービスに依存しています。

ネットワークサービスの無効化についての詳細は、オペレーティングシステムのマニュアルを参照してください。

正確な時刻の維持

ログファイルの時刻および日付のタイムスタンプがシステム間でレプリケートされ、連動して使用できるように、システム時刻が正しく同期していることを確認します。特に Windows システム上では、NTP (Network Time Protocol) クライアントを使用して、正しいシステム時刻の設定を検討します。

システム障害発生後の再起動

可能な場合は、『Sun ONE Directory Server 管理ガイド』で説明する手順に従って Directory Server を停止します。システムを適切にシャットダウンしないで、シャットダウン中に突然停止させた場合、データベースが壊れることで、Directory Server の起動が遅れることがあります。データベースの復旧には時間がかかる可能性があります。

(Solaris パッケージの場合) インストールおよび設定の一環として、適切なスクリプトによりブート時に再起動が可能になります。

(Windows の場合) システム障害発生後、自動的に再起動するように Windows を設定します。詳細は、Windows のヘルプを参照してください。

その他のプラットフォームの場合は、ブート時の起動サービスに関する詳細について、オペレーティングシステムのマニュアルを参照してください。

基本的なチューニングの推奨事項の生成

Solaris パッケージバージョンの場合は、`idsktune` ユーティリティ (`/usr/sbin/directoryserver idsktune`)、それ以外のバージョンの場合は、製品のバイナリファイルのあるディレクトリに配置された `idsktune` を使用して、Windows 以外のプラットフォームの基本的なチューニングの推奨事項を生成します。

スーパーユーザーとしてユーティリティを実行する場合は、ユーティリティはシステムに関する情報を収集します。ユーティリティは、推奨する修正作業とともに注意、警告、およびエラーを表示します。たとえば、ユーティリティは次の項目をチェックします。

- オペーレーティングシステムとカーネルのバージョンがこのリリースでサポートされている
- 使用できるメモリとディスク容量が標準使用の最小要件を満たしている
- システムリソースの制限が標準使用の最小要件を満たしている
- 必要なパッチまたはサービスパックがインストールされている

注 実際の運用環境で使用するシステムに Directory Server ソフトウェアをインストールする前に、少なくともすべての `ERROR` 状態を修正します。

個々の導入要件が最小要件を超える場合があります。`idsktune` ユーティリティを使用して、最小システム要件として確認された以上のリソースを用意することができます。

ユーティリティに関する詳細については、Sun ONE Directory Server Resource Kit を参照してください。Sun ONE Directory Server Resource Kit は、12 ページの「Directory Server ツールのダウンロード」で説明する手順に従って、取得できます。

システム設定のチューニング

idsktune ツールを使用して、現在のシステム設定および推奨する変更を読み込むことができます。一般的には、推奨事項の実装により、Directory Server を実行する専用のシステムおよび追加アプリケーションを実行するシステムの両方のパフォーマンスが最適化されます。

特定の推奨事項を実装する前に、ローカルネットワークの状態およびその他のアプリケーションを確認します。追加するネットワークのチューニングのヒントについては、オペレーティングシステムのマニュアルを参照してください。

表 5-2 導入前に確認する設定ファイル

プラットフォーム	ファイル	内容
Solaris オペレーティング環境	/etc/init.d/inetinit	チューニング用 ndd ステートメントを追加する
	/etc/system	システム制限を確認する
	/etc/vfstab	ファイルがローカルにあることを確認する
HP-UX	/etc/rc.config.d/nddconf	チューニング用 ndd ステートメントを追加する
		代わりに、sam(1M) を使用する
Red Hat Linux	/etc/fstab	ファイルがローカルにあることを確認する
	/etc/security/limits.conf	nofile ハードリミット指令を追加する
	/etc/sysctl.conf	カーネルパラメタを確認および設定する
	/proc/sys/fs/file-max	ファイル記述子の制限を確認する

DPC (Deferred Procedure Call) (Windows の場合)

マルチプロセッサシステムの着信ネットワークトラフィックを処理する Windows の遅延割り込み要求のデフォルト DPC (Deferred Procedure Call) 処理は、パフォーマンスに悪い影響を与える可能性があります。実際には、DPC となる遅延割り込みは、あるプロセッサから別のプロセッサに再スケジュールされて、大幅なオーバーヘッドの増加をもたらします。このような DPC オーバーヘッド問題を防止するには、次のレジスタキーで `ProcessorAffinityMask` の値を `0` に設定します。

```
HKEY_LOCAL_MACHINE¥System¥CurrentControlSet¥Services¥NDIS¥Parameters
```

ファイル記述子

Directory Server は、同時クライアント接続を処理するときにファイル記述子を使用します。システムで使用可能な、またはプロセスで使用可能なファイル記述子の最大数を小さくして、同時接続数を制限することができます。したがって、ファイル記述子の数に関連する推奨事項は、Directory Server がシステムで処理できる同時接続の数に関連します。

Solaris システムでは、利用できるファイル記述子の数は、`/usr/sbin/directoryserver idsktune` (パッケージバージョンの場合) または `idsktune` (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、`rlim_fd_max` パラメータを使用して設定します。利用可能なファイル記述子の数の変更に関する詳細手順は、オペレーティングシステムのマニュアルを参照してください。

システムで使用できるファイル記述子の最大数を変更した後、変更後のファイル記述子を使用するための Directory Server の設定に関する詳細については、159 ページの表 9-2 を参照してください。

大規模ファイルのサポート (HP-UX の場合)

一部の HP-UX システムでは、デフォルトで大規模ファイルがサポートされません。Directory Server をインストールするファイルシステムで大規模ファイルのサポートを有効にする手順については、HP-UX の製品マニュアルを参照してください。手順は、`idsktune` ユーティリティにあるドキュメントにも記載されています。

タイムアウトを保留状態にするスレッド (HP-UX の場合)

一部の HP-UX システムでは、タイムアウトを保留状態にするスレッドの最大数が、idsktune ユーティリティの指定に対して不適切に設定される場合があります。タイムアウトを保留状態にするスレッドの最大数を増やす手順については、HP-UX の製品マニュアルを参照してください。手順は、idsktune ユーティリティにあるドキュメントにも記載されています。

プロセス当たりのスレッド (HP-UX の場合)

一部の HP-UX システムでは、プロセス当たりのスレッドの最大数が、idsktune ユーティリティの指定に対して小さすぎる場合があります。プロセス当たりのスレッドの最大数を増やす手順については、HP-UX の製品マニュアルを参照してください。手順は、idsktune ユーティリティにあるドキュメントにも記載されています。

TCP (伝送制御プロトコル) の設定

特定のネットワーク設定はプラットフォームに依存します。一部のシステムでは、TCP 設定を変更することによって、Directory Server のパフォーマンスを強化することができます。この節では、TCP 設定に関する idsktune ユーティリティの推奨事項の背後にある理論について説明します。

TIME-WAIT 状態の閉じられた接続

一部のシステムでは、接続解除後カーネル内のテーブルに TCP 接続が保持される時間を設定することができます。接続が保持されている間は、迅速に接続を再開することができます。この設定が大きすぎると、システムは長時間に渡ってカーネルのテーブル内に多数の接続を保持するため、Directory Server へ接続できる数が少なくなります。ほとんどの導入では、このパラメータの値を 30 秒 (30,000 ミリ秒) に設定すると、Directory Server への同時接続を増やすことができます。

Solaris システムでは、この時間間隔は、/usr/sbin/directoryserver idsktune (パッケージバージョンの場合) または idsktune (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、tcp_time_wait_interval パラメータを使用して設定します。

接続保留状態の受け入れ

一部のシステムでは、Directory Server などの TCP リスナーが TCP 接続の保留状態を受け入れる数を設定することができます。この設定が小さすぎると、Directory Server が受け入れ可能な保留状態の接続の数が制限されます。ほとんどの導入では、このパラメータの値を少なくとも 1024 に設定すると、Directory Server が処理できる同時接続要求を増やすことができます。

Solaris システムでは、許可される保留状態の接続の数は、`/usr/sbin/directoryserver idsktune` (パッケージバージョンの場合) または `idsktune` (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、`tcp_conn_req_max_q` パラメータを使用して設定します。`tcp_conn_req_max_q0` を 2048 に増やすことを検討します。

確認応答の遅延

一部のシステムでは、システムに直接接続されていないホストに対する TCP 確認応答を遅延させる時間を設定することができます。遅延時間を直接設定する代わりに、159 ページの表 9-2 で説明しているように、`cn=config` 上の `nsslapd-nagle` を `off` に設定します。

接続の無効化

一部のシステムでは、キープアライブパケットの転送間隔を設定することができます。この設定では、接続が無効化したり解除された可能性があるときに、TCP 接続を維持する時間を決定することができます。設定が大きすぎると、キープアライブ間隔によって、解除されたクライアントの接続を活動状態に維持するために、システムは不必要なリソースを使用してしまいます。ほとんどの導入では、このパラメータの値を 600 秒 (600,000 ミリ秒 = 10 分) に設定すると、Directory Server への同時接続を増やすことができます。

Solaris システムでは、この時間間隔は、`/usr/sbin/directoryserver idsktune` (パッケージバージョンの場合) または `idsktune` (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、`tcp_keepalive_interval` パラメータを使用して設定します。

着信接続

一部のシステムでは、確認応答を送信しない着信接続の待機時間を設定できます。設定が大きすぎると、接続の失敗を検出する場合に長時間の遅延を引き起こします。高速で信頼性の高いネットワークにイントラネットを導入する場合、このパラメータを 600 秒 (600,000 ミリ秒 = 10 分) に設定するとパフォーマンスを向上させることができます。

Solaris システムでは、この時間間隔は、/usr/sbin/directoryserver idsktune (パッケージバージョンの場合) または idsktune (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、tcp_ip_abort_interval パラメータを使用して設定します。

発信接続

一部のシステムでは、確立される発信接続の待機時間を設定できます。設定が大きすぎると、迅速に応答しない送信先サーバー (レプリカなど) に対する発信接続を確立する場合に長時間の遅延を引き起こします。高速で信頼性の高いネットワークにイントラネットを導入する場合、このパラメータを 10 秒に設定するとパフォーマンスを向上させることができます。

Solaris システムでは、この時間間隔は、/usr/sbin/directoryserver idsktune (パッケージバージョンの場合) または idsktune (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、tcp_ip_abort_cinterval パラメータを使用して設定します。

再転送のタイムアウト

一部のシステムでは、パケットの再転送の初期時間間隔を設定することができます。この設定は、確認応答のないパケットの再転送前の待機に影響を与えます。この設定が大きすぎると、クライアントは失われたパケットで待機させられる場合があります。高速で信頼性の高いネットワークにイントラネットを導入する場合、このパラメータを 500 ミリ秒に設定するとパフォーマンスを向上させることができます。

Solaris システムでは、この時間間隔は、/usr/sbin/directoryserver idsktune (パッケージバージョンの場合) または idsktune (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、tcp_rexmit_interval_initial パラメータを使用して設定します。

Windows では、TCP の高速な再転送と回復のための Van Jacobson アルゴリズムを実装し、ACK 受信時に、再送タイマーの時間切れを待つことなく、失われたセグメントを迅速に転送し直すことができます。Van Jacobson アルゴリズムを実装するには、次のレジストリキーを編集します。

HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Services/Tcpip/Parameters

REG_DWORD タイプを含む TcpMaxDupAcks を追加します。ACK の数に対応する値を設定します。有効な値は 1 から 3 で、デフォルトは 2 です。

シーケンス番号

一部のシステムでは、システムが初期シーケンス番号生成を処理する方法を設定することができます。エクストラネットおよびインターネットを導入する場合、シーケンス番号攻撃を防ぐために、このパラメータを RFC 1948 に基づく初期シーケンス番号生成などに設定します。

Solaris システムでは、この動作は、`/usr/sbin/directoryserver idsktune` (パッケージバージョンの場合) または `idsktune` (パッケージバージョンではない場合) ユーティリティが提供するドキュメントで説明しているように、`tcp_strong_iss` パラメータを使用して設定します。

第 6 章

キャッシュサイズのチューニング

Directory Server では、クライアントの要求にすばやく対応できるように、ディレクトリ情報をメモリやディスクにキャッシュします。適切にキャッシュをチューニングすることで、クライアント要求を処理するときにディスクサブシステムにアクセスする必要が最小限になります。

注 キャッシュをチューニングして適切に動作する前に、ほかのチューニングを行なってもパフォーマンスに与える効果は制限される場合があります。

キャッシュのタイプ

Directory Server では、表 6-1 で説明するように 3 タイプのキャッシュを処理します。

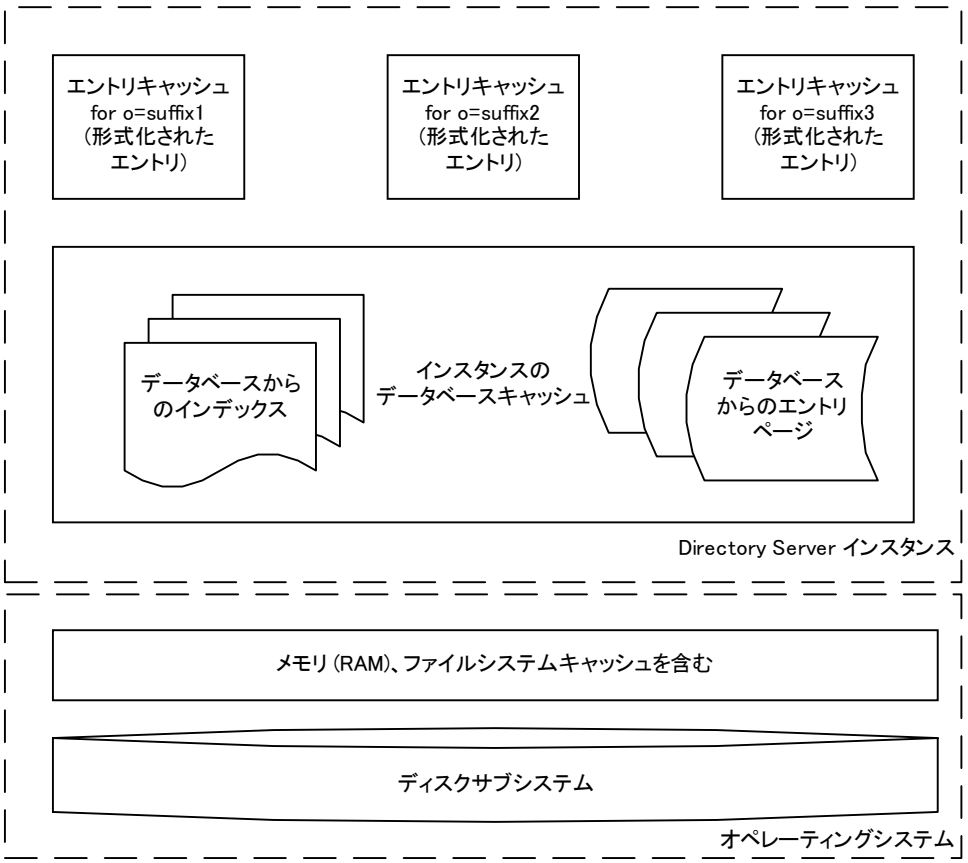
表 6-1 キャッシュ

キャッシュのタイプ	内容
データベース	各 Directory Server インスタンスにはデータベースキャッシュが 1 つあり、データベースの形式でインデックスとエントリの両方を保持する
エントリ	各サフィックスにはエントリキャッシュがあり、前の操作でデータベースから取得し、クライアントアプリケーションにすばやく配信するように形式化されたエントリを保持する
インポート	各 Directory Server インスタンスにはインポートキャッシュがある。構造的にデータベースキャッシュと似ていて、一括ロード中に使用される

Directory Server では、オペレーティングシステムが処理するファイルシステムキャッシュやディスクサブシステムの I/O バッファも有益です。

図 6-1 に 3 つのサフィックスを処理する Directory Server のキャッシュを示します。それぞれのサフィックスには専用のエントリキャッシュがあります。インスタンスは重要なディスクのアクティビティを処理するように設定されており、トランザクションログ、データベース、ほかのファイルやログなどは第 8 章「ログのチューニング」で推奨するように、別のディスクサブシステムに配置されています。

図 6-1 コンテキストにおけるエントリキャッシュとデータベースキャッシュ



データベースキャッシュ

各 Directory Server インスタンスにはデータベースキャッシュが 1 つあります。データベースキャッシュはインデックスやエントリを含むデータベースからのページを保持します。ページはエントリではなく、データベースの一部が含まれるメモリの断片です。データベースキャッシュのサイズはユーザーが指定します (`nsslapd-dbcachesize`)。データベースキャッシュのサイズ変更は、サーバーの再起動後、サーバーの起動時に割り当てられるデータベースキャッシュ領域で有効になります。

Directory Server はデータベースファイルとデータベースキャッシュとの間でページを移動し、データベースキャッシュのサイズが最大になるように維持します。Directory Server によってデータベースキャッシュ用として実際に使用されるメモリの量は、指定したサイズよりも最大で 25% 大きくなることがあります。これはデータベースキャッシュ自体を管理するのに追加のメモリが必要となるためです。

非常に大きなデータベースキャッシュを使用する場合、経験テストを実施したり、Solaris システム上の `pmap`(1) などのツールを使ってメモリ使用量を監視したりして、Directory Server によるメモリ使用量が利用可能な物理メモリのサイズを超えないかどうかを検証します。利用可能な物理メモリを超えると、システムがページングを繰り返し行うようになり、深刻なパフォーマンスの低下を招きます。

また、Directory Server がサポートする UNIX プラットフォーム上に存在する `ps`(1) ユーティリティで `-p` *pid* オプションと `-o` *format* オプションを指定しても、Directory Server (`ns-slapd`) などといった特定のプロセスによる現在のメモリ使用量を表示することができます。Windows システムの場合、「タスク マネージャ」の「プロセス」タブページに、各プロセス (`slapd.exe` など) のメモリ使用量が一覧表示されます。詳細は、オペレーティングシステムのマニュアルを参照してください。

32 ビットサーバーの場合、データベースキャッシュのサイズは実際には 2G バイト以下にする必要があります。

注　Windows や AIX プラットフォームでは、1G バイト (1,073,741,824 バイト) 以上をデータベースキャッシュに割り当てないでください。

`nsslapd-dbcachesize` の有効な値の範囲についての詳細は、『Sun ONE Directory Server Reference Manual』を参照してください。

エントリキャッシュ

エントリキャッシュには、最近アクセスしたエントリが、クライアントアプリケーションに配信できる形式で保持されます。サフィックス (nsslapd-cachememsize) と最大エントリ数 (nsslapd-cachesize) のキャッシュサイズはユーザーが指定します。エントリキャッシュは必要に応じて割り当てられます。

Directory Server では、このキャッシュに格納されたエントリはフォーマットされているので、最大限に効率的にエントリキャッシュからエントリを戻すことができます。データベース内のエントリは、バイトの raw 文字列として格納されますが、クライアントアプリケーションに配信される前に形式化してエントリキャッシュに格納される必要があります。

エントリキャッシュのサイズを指定するときに、nsslapd-cachememsize には、基盤となる記憶域割り当てライブラリに Directory Server が要求するメモリー量を指定します。記憶域割り当てライブラリが処理するメモリー要求の量によっては、実際に使用されるメモリー量は、Directory Server がエントリキャッシュに実際に割り当てるメモリー量よりもかなり大きくなります。

Directory Server プロセスが使用する実際のメモリー量は、主として、使用される記憶域割り当てライブラリとキャッシュされるエントリ数によって決まります。多数の小さな属性値を持つエントリは、少数の大きな属性値を持つエントリよりも、通常は負荷が大きくなります。

32 ビットサーバーの場合、エントリキャッシュのサイズは実際には 2G バイト以下にする必要があります。

注 AIX プラットフォームでは、Directory Server が maxdata = 0x50000000 で作成されるため、データベースキャッシュとエントリキャッシュの両方に 1G バイトずつ割り当てることができます。maxdata の値を変更する必要がある場合は、ご購入先までお問い合わせください。

---

nsslapd-cachememsize と nsslapd-cachesize の有効な値の範囲についての詳細は、『Sun ONE Directory Server Reference Manual』を参照してください。

インポートキャッシュ

インポートキャッシュは、サフィックスの初期化中にだけ作成されて使用されます。サフィックスの初期化のことを一括ロードまたはインポートとも言います。導入時にオフラインのサフィックス初期化だけが行われる場合は、インポートキャッシュとデータベースキャッシュが同時に使用されることはありません。そのため、「総計キャッシュサイズ」で説明しているように、キャッシュサイズをまとめるときに、インポートキャッシュとデータベースキャッシュを足し合わせる必要はありません。インポートキャッシュのサイズはユーザーが指定します (`nsslapd-import-cachesize`)。インポートキャッシュのサイズ変更は、次にサフィックスをリセットして初期化したときに有効になります。インポートキャッシュは初期化で割り当てられ、初期化後に解放されます。

Directory Server ではインポートキャッシュの処理をデータベースキャッシュの処理と同じように行います。そのため、スワップを避けるために、十分な物理メモリが利用できることを確認してください。

32 ビットサーバーの場合、インポートキャッシュのサイズを実際には 2G バイト以下にする必要があります。`nsslapd-import-cachesize` の有効な値の範囲についての詳細は、『Sun ONE Directory Server Reference Manual』を参照してください。

ファイルシステムキャッシュ

オペレーティングシステムでは、Directory Server キャッシュやほかのアプリケーションで使用しない利用可能なメモリをファイルシステムキャッシュに割り当てます。このキャッシュにはディスクから最近読み込んだデータを保持しているため、後続の要求ではディスクからもう一度読み込むのではなく、コピーされたデータをキャッシュから取得することができます。メモリアクセスはディスクアクセスより高速なので、物理メモリをファイルシステムキャッシュに利用できるようにしておくことで、パフォーマンスが向上します。

ファイルシステムキャッシュについての詳細は、オペレーティングシステムのマニュアルを参照してください。

総計キャッシュサイズ

同時に使用されるすべてのキャッシュの合計が、利用可能な物理メモリの合計サイズ未満となる必要があるため、ファイルシステムキャッシュに割り当てるメモリは少なく設定します。32 ビットサーバーの場合、総計キャッシュのサイズは実際には 2G バイト以下に制限されます。使用されるキャッシュの合計が、ユーザーが指定したサイズを大幅に上回る可能性があります。キャッシュサイズと Directory Server プロセスサイズが、利用可能な物理メモリのサイズを超えていないかどうかをチェックする方法については、113 ページの「データベースキャッシュ」を参照してください。

注 Windows プラットフォームで、アプリケーションが使用可能な最大アドレス空間は 2G バイトです。総計キャッシュサイズがこの制限を超えると、Directory Server はエラーメッセージを表示して終了します。

Directory Server がオンラインのときにサフィックスが初期化 (一括ロード) される場合は、データベース、エントリ、インポートの各キャッシュサイズの合計が、利用可能な物理メモリの合計サイズよりも小さく設定される必要があります。

表 6-2 サフィックスの初期化 (インポート) 操作とキャッシュの使用

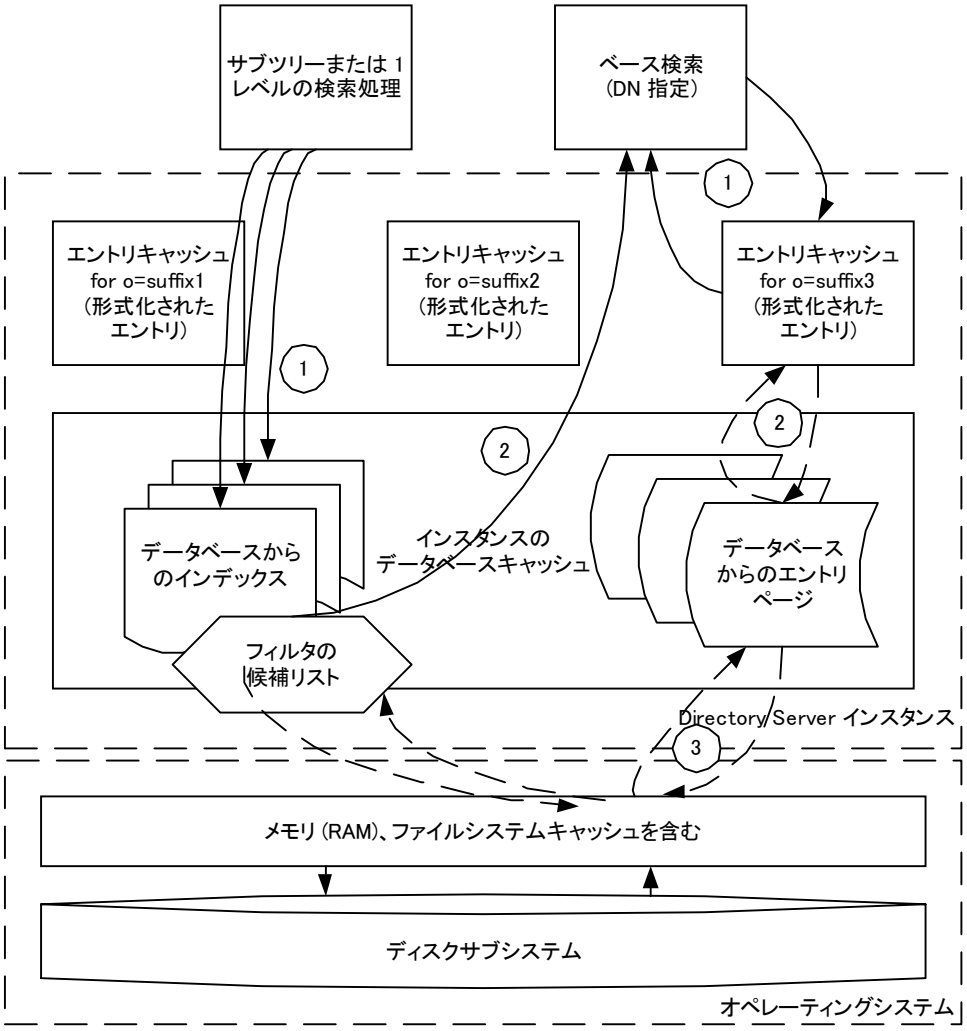
キャッシュのタイプ	オフラインインポート	オンラインインポート
データベース	未使用	使用
エントリ	使用	使用
インポート	使用	使用

Directory Server の停止中にすべてのサフィックス初期化がオフラインで行われる場合は、この制限を回避することができる場合があります。この場合、インポートキャッシュはデータベースキャッシュと共存しないため、同じメモリをオフラインのサフィックス初期化ではインポートキャッシュに割り当て、オンラインでの使用にはデータベースキャッシュに割り当てることができます。このような特殊な場合の実装では、実際の運用システムでオンライン一括ロードを実行しないようにしてください。同時に使用するキャッシュの合計は、利用可能な物理メモリの合計サイズ未満であることが必要です。

検索でキャッシュを使用する方法

図 6-2 に Directory Server がベース DN を使用した検索と、フィルタを使用した検索の両方を処理する方法を示します。それぞれの線は、メモリの異なるレベルにアクセスするスレッドを表しています。破線は、効率的なチューニングによって最小化されるステップを表します。

図 6-2　検索とキャッシュ



ベース検索のプロセス

図で示すように、ベース検索、つまりベース DN を指定する検索は、Directory Server が処理する検索のうち、もっともシンプルなタイプです。この検索を行うために、Directory Server では次を行います。

1. 指定したベース DN を持つエントリをエントリキャッシュから取得しようとします。

   ここで、エントリが見つかった場合、Directory Server は検索用に指定したフィルタとその候補が一致するかどうかをチェックします。

   エントリが一致する場合は、Directory Server によって、形式化されキャッシュされたエントリがクライアントアプリケーションにすばやく返されます。

2. エントリをデータベースキャッシュから取得しようとします。

   ここで、エントリが見つかった場合、Directory Server はそのサフィックス用のエントリキャッシュにエントリをコピーします。次に、エントリがエントリキャッシュで見つかったときと同様の処理をします。

3. エントリをデータベース自体から取得しようとします。

   ここで、エントリが見つかった場合、Directory Server はデータベースキャッシュにエントリをコピーします。次に、エントリがデータベースキャッシュで見つかったときと同様の処理をします。

サブツリーまたは 1 レベルの検索処理

また 117 ページの図 6-2 に示すように、サブツリーや 1 レベルの検索では、エントリのセットを処理するのに追加の処理が行われます。この検索を行うために、Directory Server は次の処理を行います。

1. フィルタに一致する候補エントリのセットを、データベースキャッシュのインデックスから作成しようとします。

   適切なインデックスが見つからない場合は、データベース自体にある関連エントリから候補エントリのセットが作成されます。

2. 各候補エントリは、次のように処理されます。

   a. ベース検索を実行してエントリを取得します。

   b. 検索で指定したフィルタとエントリが一致するかどうかチェックします。

   c. エントリがフィルタに一致する場合は、クライアントアプリケーションに返されます。

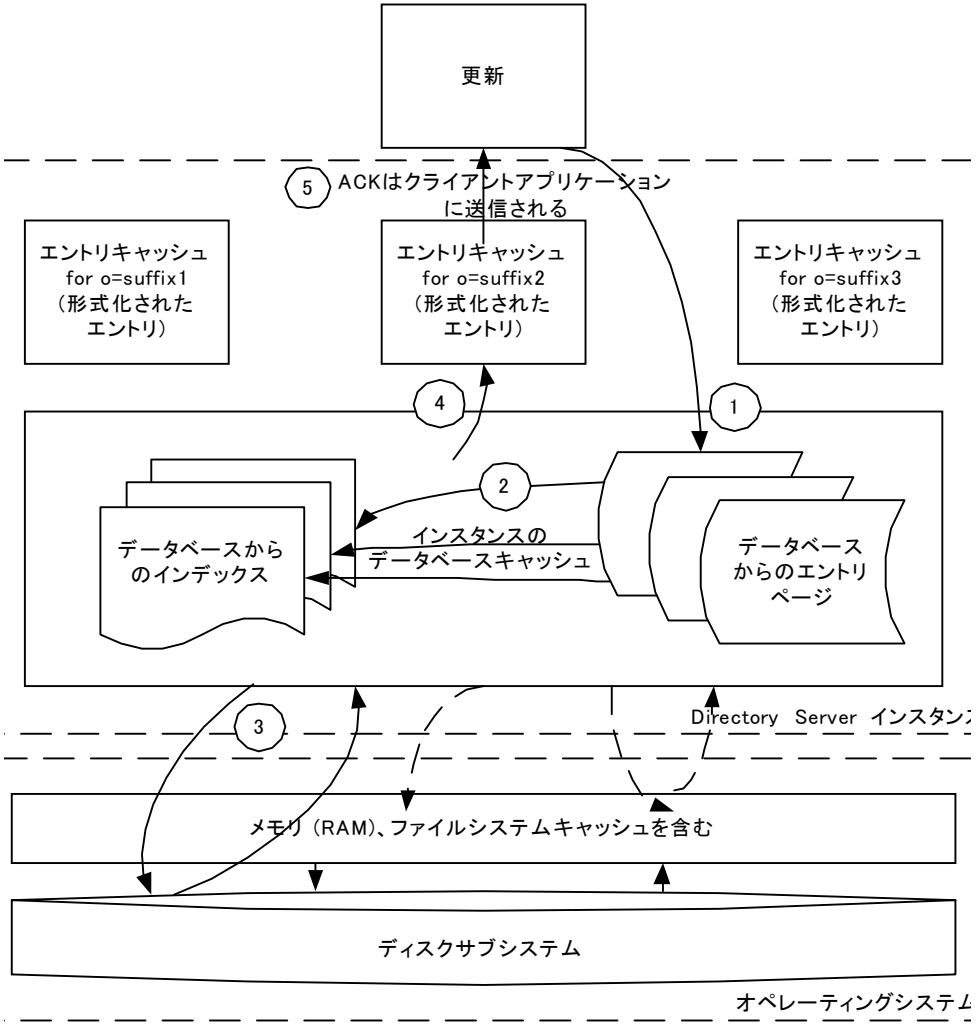
   この方法で、Directory Server は、メモリにセットを構成することを回避します。

Directory Server をチューニングする前に、期待する検索内容を知っておくことが理想的ですが、実際にテストを行なって仮定した内容を検証してください。

更新でキャッシュを使用する方法

図 6-3 に、Directory Server で更新を処理する方法を示します。それぞれの線は、メモリの異なるレベルにアクセスするスレッドを表しています。破線は、効率的なチューニングによって最小化されるステップを表します。

図 6-3　更新とキャッシュ



更新では検索の場合よりも多くの処理が行われます。Directory Server は、次の手順で更新を実行します。

1. ベース DN 検索を実行して、更新するエントリを取得します。追加操作の場合は、エントリがまだ存在していないことを確認します。
2. データベースキャッシュを変更し、特に、更新の影響を受けるあらゆるインデックスを更新します。

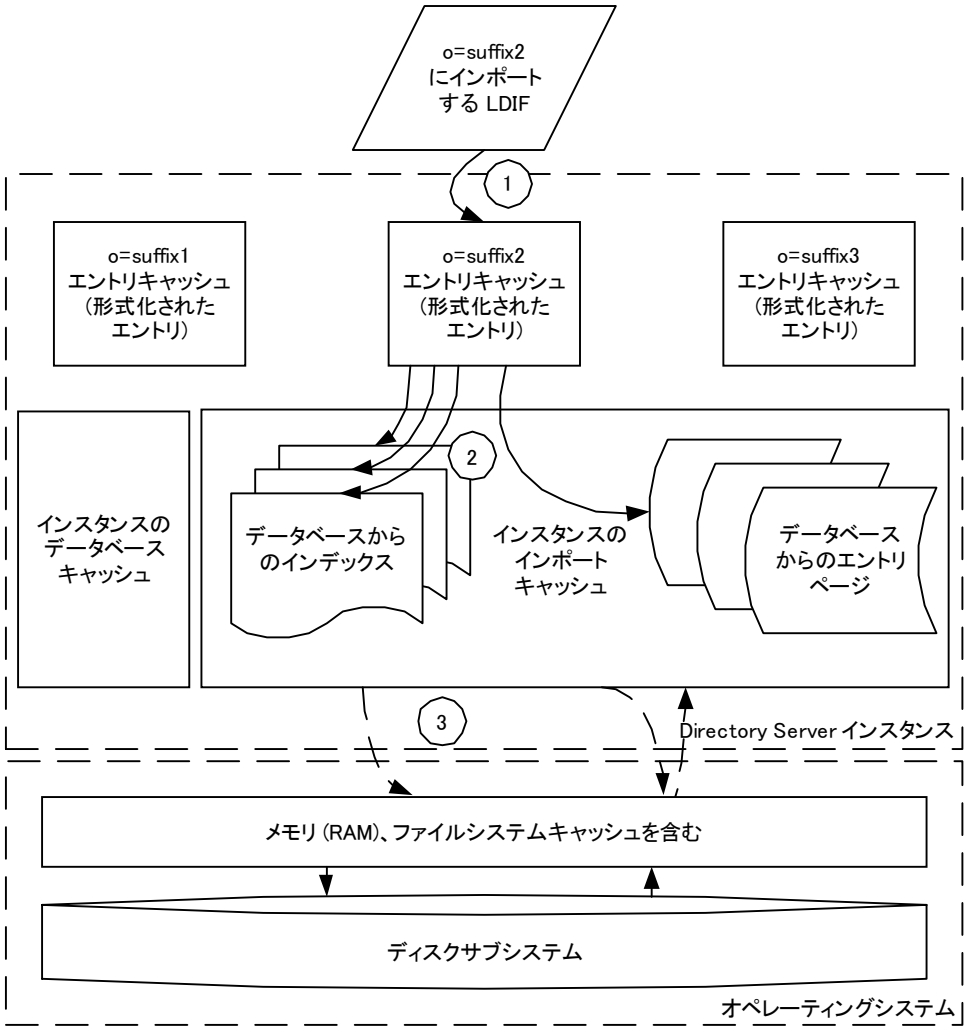
   更新の影響を受けるデータがデータベースキャッシュにロードされていない場合、このステップはディスクアクティビティとなります。関連データはキャッシュに読み込まれます。
3. トランザクションログに変更内容についての情報を書き込み、情報がディスクにフラッシュされるまで待機します。

   詳細は、153 ページの「トランザクションログ」を参照してください。
4. 更新されたエントリを形式化し、そのサフィックスのエントリキャッシュにコピーします。
5. 更新に成功した通知をクライアントアプリケーションに返します。

サフィックスの初期化でキャッシュを使用する方法

図 6-4 に、Directory Server でサフィックスの初期化を処理する方法を示します。サフィックスの初期化は一括ロードインポートとも呼ばれます。それぞれの線は、メモリの異なるレベルにアクセスするスレッドを表しています。破線は、効率的なチューニングによって最小化されるステップを表します。

図 6-4　サフィックスの初期化 (一括ロード) とキャッシュ



Directory Server は、次の手順でサフィックスを初期化します。

1. LDIF からデータをバッファとして使用するエントリキャッシュに与えるスレッドを起動します。
2. 影響を受ける各インデックス用のスレッドと、インポートキャッシュにエントリを作成するスレッドを起動します。これらのスレッドは、エントリキャッシュに入れられたエントリを使用します。

3. インポートキャッシュを使い切った場合、データベースファイルの読み書きを行います。

   Directory Server ではサフィックスの初期化中にログメッセージの書き出しを行うこともありますが、トランザクションログには書き出しません。

Directory Server に同梱されている `ldif2db` (`/usr/sbin/directoryserver ldif2db`) などのサフィックス初期化ツールでは、キャッシュのヒット率やインポートのスループットに関するフィードバックがあります。キャッシュのヒット率とインポートのスループットの両方が低下するということは、インポートキャッシュが小さすぎる可能性があるということです。インポートキャッシュのサイズを増やすことを検討してください。

検索の最適化

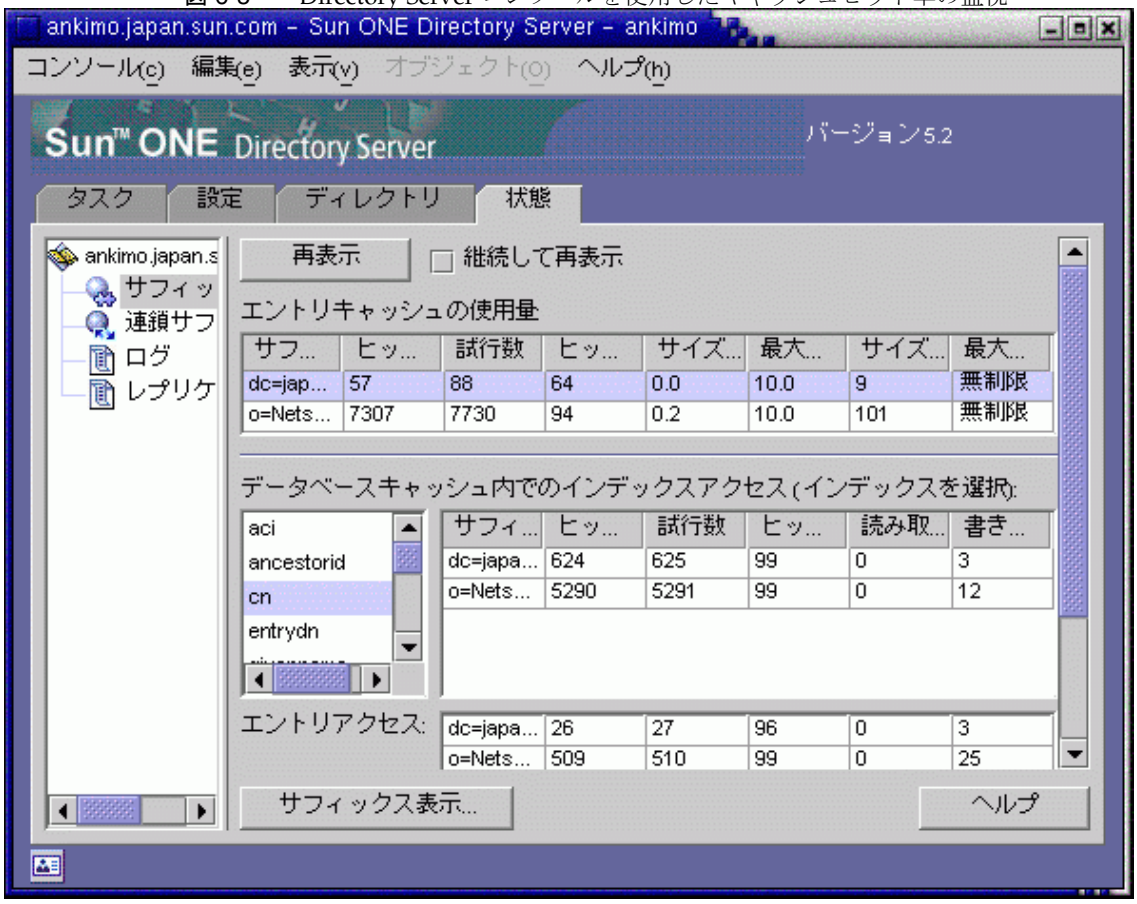
最高のパフォーマンスを得るために、できるだけ多くのディレクトリデータをメモリにキャッシュしてください。ディレクトリがディスクから情報を読み込まないように、ディスクの I/O ボトルネックを制限します。さまざまな方法で行うことができますが、ディレクトリツリーのサイズ、利用可能なメモリ量、使用しているハードウェアなどによって変わります。導入によっては、エントリキャッシュやデータベースキャッシュに割り当てるメモリを多くしたり少なくしたりして、検索パフォーマンスを最適化します。また、複数のサーバーの Directory Server コンシューマに検索を配布すると、検索パフォーマンスが最適化されます。

メモリにすべてのエントリとインデックスがある場合

最適な場合を考えてみます。データベースキャッシュとエントリキャッシュの合計サイズは、利用可能な物理メモリの範囲内です。エントリキャッシュは、ディレクトリのエントリすべてを保持できるサイズです。データベースキャッシュは、少なくともインデックスすべてを保持できるサイズです。この場合は、検索するとすべてがキャッシュで見つかります。Directory Server はファイルシステムキャッシュやディスクからエントリを取得する必要はありません。

従って、更新後でもデータベースキャッシュには必ずすべてのデータベースインデックスが含まれています。データベースキャッシュでインデックス用の空きがなくなった場合、Directory Server では検索要求ごとにディスクからインデックスを読み込む必要があり、スループットに深刻な影響を与えます。図 6-5 に示すように、Directory Server コンソール では「状態」タブにヒット率やほかの有益な情報が表示されます。

図 6-5　Directory Server コンソールを使用したキャッシュヒット率の監視



代わりに次のようにコマンド行から検索を利用して、ページングとキャッシュの状態を監視することもできます。

```
$ ldapsearch -D admin -w password ¥
-b cn=monitor,cn=database_name,cn=ldbm database,cn=plugins,cn=config
```

.db3 ファイルのすべてのデータベースインデックスをデータベースキャッシュに収めるために必要なメモリ量を概算するには、次の式を使用します。写真のような大きなバイナリ属性を持たない典型的なエントリを使用した、デフォルトのインデックス設定では、この式はほぼ正確です。

nsslapd-dbcachesize = 1.2 x SUM すべての .db3 ファイル (ファイルサイズ)

すべてのエントリをエントリキャッシュに収めるために必要となるエントリキャッシュのスロット数とメモリ量を概算するには、次の公式を使用します。写真のような大きなバイナリ属性を持たない典型的なエントリを使用した、デフォルトのインデックス設定では、この公式はほぼ正確です。

```
nsslapd-cachesize = 4.5 x (LDIF 内のエントリ数)
```

```
nsslapd-cachememsize = 3.8 x (id2entry.db3 のファイルサイズ)
```

テスト経験を通じて、見積もりを検証し、訂正してください。特に、エントリキャッシュは、割り当てられた量以上のメモリを使用する可能性があります。

32 ビット Directory Server で十分なメモリを確保する場合

ここでは、エントリキャッシュとデータベースキャッシュ内のすべてのデータを格納するのに十分なメモリを搭載しているが、64 ビット Directory Server プロセスをサポートしないシステムを考えてみます。たとえば、ハードウェア上の制約により Solaris システム上に導入できない場合に、32 ビットプロセスのメモリ制限に見合う適切なキャッシュサイズを決めた後、残りの利用可能なメモリをファイルシステムキャッシュ用として残しておくことが大切です。

注

ファイルシステムキャッシュは、システム上のほかのプロセスと共有されます。ファイルベースの操作を行う場合は特にそうです。したがって、ファイルシステムキャッシュの管理は、ほかのキャッシュの場合よりも困難になります。特に、Directory Server 専用でないシステム上では困難です。

システムは、ファイルシステムキャッシュをほかのプロセスに割り当て直す可能性があります。

メモリは少ないがファイルシステムキャッシュはある場合

エントリキャッシュとデータベースキャッシュにすべてのデータを保持するには利用可能なメモリが十分ではないが、それでも多くの利用可能なメモリがあるシステムを考えてみます。この場合の重要な点は、エントリキャッシュとデータベースキャッシュの合計サイズが、利用可能な物理メモリを超えないことです。合計サイズが物理メモリを超えると、仮想メモリの頻繁なページングが発生し、システムの仮想停止が引き起こされる可能性があります。

利用可能なメモリはファイルシステムキャッシュで使用し、エントリキャッシュとデータベースキャッシュのサイズは 500K バイト程度のサイズに抑えます。こうすることで、Directory Server でディスクからエントリやインデックスを繰り返して読み込む必要が生じる前に、検索をファイルシステムキャッシュで完結するのに十分なデータを、ファイルシステムキャッシュに保持することができます。

代わりに、特定の導入におけるほとんどの検索ではディレクトリにある全エントリの同一の小さなサブセットがアクセスされることや、そのような検索のためにエントリやインデックスをキャッシュしたことによる利点が時折発生する特殊な検索要求を処理するためのコストを補ってくれるものと仮定すると、検索パターンがランダムでなければ、エントリキャッシュやデータベースキャッシュを多めに設定することもできます。テスト経験を通じて、仮定を検証し、訂正してください。

メモリが少なく、ファイルシステムキャッシュも少ない場合

エントリキャッシュにもデータベースキャッシュにもデータを保持するための利用可能なメモリが不足しており、システムがファイルシステムキャッシュにデータをキャッシュするにもメモリが不足しているシステムを考えてみます。この場合に重要な点は、利用可能なメモリ量を最大にすることです。

エントリキャッシュとデータベースキャッシュのサイズをできるだけ小さくし、ファイルシステムキャッシュにできるだけ多くのメモリを残します。ファイルシステムキャッシュにできるだけ多くのメモリを残しておくことで、少なくとも 3 ～ 4.5 のエントリをデータベースキャッシュやエントリキャッシュに展開する必要がなくなり、論理的にディスク I/O アクティビティを制限することができます。特定の導入において、テスト経験を通じて仮定した内容を検証します。

更新用の最適化

最高の更新パフォーマンスを得るには、まず監視用のトランザクションログのボトルネックを取り除きます。詳細は、153 ページの「トランザクションログ」を参照してください。

次に、更新をメモリで処理してディスクアクティビティを最小化するのに十分なメモリ量をデータベースキャッシュに割り当ててみます。Directory Server コンソールに表示されるヒット率を読み取ることで、データベースキャッシュの効率性を監視することができます。123 ページの図 6-5 に示すように、Directory Server コンソールでは「状態」タブにサフィックスのヒット率が表示されます。

同様に、ファイルシステムキャッシュに利用できるメモリを大きめに残すようにします。Directory Server が数回処理を行なったら、ファイルシステムキャッシュにはディスクを読み込まずに済むだけの十分なエントリとインデックスが格納されているはずです。更新ではメモリ内のデータベースキャッシュに影響を与え、大きいデータベースキャッシュからのデータはフラッシュされることがあります。

データをディスクにフラッシュすること自体もボトルネックであるため、Sun StorEdge™ ディスクアレイのような別の RAID システムにデータベースを格納することで、更新パフォーマンスが改善されます。潜在的な I/O ボトルネックを分離するには、Solaris システムの `iostat` (1M) などのユーティリティを使用することもできます。Windows システムの I/O ボトルネックへの対応方法の詳細は、Windows ヘルプを参照してください。

キャッシュのプライムと監視

キャッシュのプライムとは、後続の Directory Server の動作が、通常の操作パフォーマンスに反映されるようにキャッシュにデータを取り入れることです。潜在的な最適化を測定し分析する前にキャッシュのプライムを行います。

サフィックスのエントリキャッシュのプライムを行うには `ldapsearch` ユーティリティを使用します。たとえば、次のようにします。

```
$ ldapsearch -D directoryManager -w password -b suffix objectclass=¥* > /dev/null
```

検索を実行してデータベースキャッシュのプライムを行い、実際にインデックスをキャッシュにロードします。`(mail=*)` などのフィルタを使用して検索を実行することで、現在のインデックスのプライムを行えます。ほかのインデックスの場合は、Sun ONE Directory Server Resource Kit の `searchrate` ユーティリティを使用して、インデックス化するために、各属性の考えられるすべての値を検索するのにフィルタ形式を適用することを検討します。言い換えると、たとえば `mail` 属性に対する同等

の検索のパフォーマンスをチェックするには、各メールアドレスについて 1 行に 1 メールアドレスという形式のファイルを作成します。次に searchrate ユーティリティを使用して、このファイルを使用した検索を実行します。たとえば、次のようにします。

```
$ searchrate -b suffix -f "(mail=%s)" -i mail.file -K -t 10
```

-K および -t を使用して時間を節約することを検討してください。-K オプションを使用すると、searchrate では接続を開いたままにして、検索のたびにバインドするのではなく、1 度だけバインドします。-t オプションでは、使用するスレッド数を指定できます。searchrate ユーティリティの詳細は、Sun ONE Directory Server Resource Kit マニュアルを参照してください。Sun ONE Directory Server Resource Kit は、12 ページの「Directory Server ツールのダウンロード」で説明する手順に従って、取得できます。

ほかのキャッシュがプライムされた後は、利用可能なファイルシステムのキャッシュのプライムを行えます。ファイルシステムキャッシュ内の情報が絶対にフラッシュされないという保証はありませんが、ファイルシステムキャッシュのプライムにより、増加時間が改善される可能性があります。UNIX システム上でファイルシステムのキャッシュのプライムを行うには、dd(1M) コマンドをスーパーユーザーとして使用します。データベースファイルがデフォルトの場所に格納されている Solaris システム上における例を、次に示します。

```
# for db in ServerRoot/slapd-serverID/db/*/*.db3
> do
> dd if=`pwd`/$db of=/dev/null bs=512k
> done
0+1 records in
0+1 records out
...
```

キャッシュのプライムを行なったら、テストを実行し、キャッシュのチューニングによって望ましい結果が得られるかを監視します。123 ページの図 6-5 に示すように、Directory Server コンソールでは「状態」タブの「サフィックス」ノードを選択すると、キャッシュの監視情報が表示されます。代わりに次のようにコマンド行から検索を利用して、ページングとキャッシュアクティビティを監視することもできます。

```
$ ldapsearch -D admin -w password ¥
-b cn=monitor,cn=database_name,cn=ldbm¥ database,cn=plugins,cn=config
```

データベースキャッシュのサイズが十分に大きく、またキャッシュのプライムが行われている場合は、ヒット率 (dbcachehitratio) は高く、読み込まれたページ数 (dbcachepagein) と書き出されたページ数 (dbcacheroevict) は少なくなります。ヒット率の高低を分析するときには、導入の制限も考慮する必要があります。

サフィックスのエントリキャッシュが十分大きく、またキャッシュのプライムが行われている場合、ヒット率 (entrycachehitratio) は高くなります。エントリキャッシュのサイズ (currententrycachesize) は最大サイズ (maxentrycachesize) の 80% を超えないようにします。結果として、エントリのサイズ (currententrycachecount) は、サフィックス内のエントリの総数と等しいか非常に近い値になります。

その他の最適化

キャッシュサイズのチューニングは、検索、更新、一括ロードの速度を改善する単なる 1 つの方法です。キャッシュをチューニングすると、パフォーマンスのボトルネックが、キャッシュからシステムの別の部分へと移ります。詳細は、このマニュアルの別の章を参照してください。

第 7 章

インデックスのチューニング

Directory Server で扱うエントリ数が多くなると、検索で潜在的に消費する時間やシステムリソースも多くなります。インデックスは検索パフォーマンスを向上するツールの 1 つです。この章では、特定の導入で特定のインデックスを使用する場合のコストと利点について理解するために、Directory Server インデックスの操作について説明します。

インデックスについて

インデックスは、検索情報を Directory Server エントリと関連付けます。インデックスはファイルとして Directory Server データベースに格納されています。この場合にデータベースはサフィックスの物理表現です。ほとんどの導入の場合、サフィックス 1 つはデータベース 1 つに対応しています。1 つのサフィックスが複数のデータベースに分割可能な導入もあります。Directory Server では、データベースをデフォルトで *ServerRoot*/slapd-*ServerID*/db/ に格納します。デフォルト値は nsslapd-directory です。このディレクトリに、インデックスが作成された属性ごとにインデックスファイルを 1 つ保持する単一のデータベースがあります。たとえば、サフィックス dc=example,dc=com からのエントリを保持するデータベース example の CN インデックスファイルの場合は、*ServerRoot*/slapd-*ServerID*/db/example/example_cn.db3 になります。

インデックスする対象は、クライアントアプリケーションがディレクトリデータにアクセスする方法によって異なります。表 7-1 に、標準的なインデックスのタイプについて簡単な説明をまとめます。

表 7-1 標準的なインデックスのタイプ

インデックスのタイプ	次の質問に回答する
近似	この属性の値で foobar に似た値を持つエントリはどれか
ブラウズ	仮想リスト表示の検索に適合するエントリはどれか

表 7-1　標準的なインデックスのタイプ (続き)

インデックスのタイプ	次の質問に回答する
等価	この属性の値で `foobar` を持つエントリはどれか
国際化	この国際化マッチングルールに基づいて一致するエントリはどれか
実在	この属性を持つエントリはどれか
部分文字列	この属性の値で `*foo*` に一致する値を持つエントリはどれか

CN など特定の属性のインデックスファイルには、複数のタイプのインデックスが格納されていることがあります。たとえば、CN が等価インデックスと部分文字列インデックスで `example` データベースにインデックスが作成されている場合、`example_cn.db3` には等価インデックスと部分文字列インデックスの両方が含まれます。

次については、『Sun ONE Directory Server 管理ガイド』を参照してください。

- 各インデックスタイプの概要
- インデックスの作成や削除の手順
- Directory Server によって作成されるデフォルトインデックスのリスト
- Directory Server で必要とされるシステムインデックスのリスト

デフォルトインデックスは、多くの場合で検索パフォーマンスを向上させます。また、メッセージングなどほかのアプリケーションもいくつかサポートしています。パフォーマンス上の理由から、特定のデフォルトインデックスを無効にしたり、場合によっては削除したりすることも選択できます。システムインデックスは、Directory Server に必要なインデックスです。削除したり変更したりしないでください。

利点 : 検索でインデックスを使用する方法

インデックスを使用すると検索速度が向上します。インデックス 1 つに値のリスト 1 つが含まれており、それぞれ値に対応するエントリ識別子のリストに関連付けられています。Directory Server ではインデックス内のエントリ識別子のリストを使用して、エントリをすばやく検索することができます。エントリのリストを管理するのにインデックスがないと、Directory Server では検索に一致するエントリを探すために、エントリすべてをチェックすることが必要になる場合があります。

インデックスを使用した検索がインデックスを使用しない検索より極めて少ない処理で済む理由は、検索要求処理を説明すると明白です。Directory Server では、次の手順で、検索要求を処理します。

1. クライアントアプリケーションは Directory Server に検索要求を送信します。
2. Directory Server では、処理可能なサフィックスに検索ベースが対応していることを確認するために、要求を調べます。対応していない場合は、エラーをクライアントに返し、場合によっては別の Directory Server インスタンスへの参照を返します。
3. Directory Server では検索に適したインデックス (複数の場合あり) があるかどうか判断します。

   そのようなインデックスが存在する場合、Directory Server では 117 ページの図 6-2 に示すように、インデックス内で候補エントリ (検索要求に一致するかもしれないエントリ) を検索します。

   そのようなインデックスが存在しない場合、Directory Server によってデータベース内の全エントリから候補エントリのセットが生成されることに注意してください。大規模な導入の場合、検索によっては、この手順にかなりの時間とシステムリソースが消費される可能性があります。
4. Directory Server では、検索基準に一致するかどうかを判断するために、各候補エントリが調べられます。一致するエントリは見つかるごとに、Directory Server によってクライアントに返されます。

   すべての候補が調べ終わるか、155 ページの「クライアントが使用できるリソースの制限」で説明するように `nsslapd-lookthroughlimit`、`nsslapd-sizelimit`、または `nsslapd-timelimit` などのリソース制限に達するまで、Directory Server では候補を調べ続けます。

手順 3 で明らかなように、インデックスを使用すると、クライアントからの検索要求に対応するために Directory Server で実行する処理を大幅に減らすことができます。

コスト : 更新がインデックスに与える影響

更新ではエントリ自体が変更されるだけでなく、そのエントリを参照するインデックスも更新されます。インデックス内のエントリへの参照が増えるほど、更新中にインデックスを変更するための潜在的な処理コストも増えていきます。特に Directory Server では 119 ページの図 6-3 に示すように、更新通知をクライアントアプリケーションに送信する前に、影響のあるインデックスをすべて変更します。

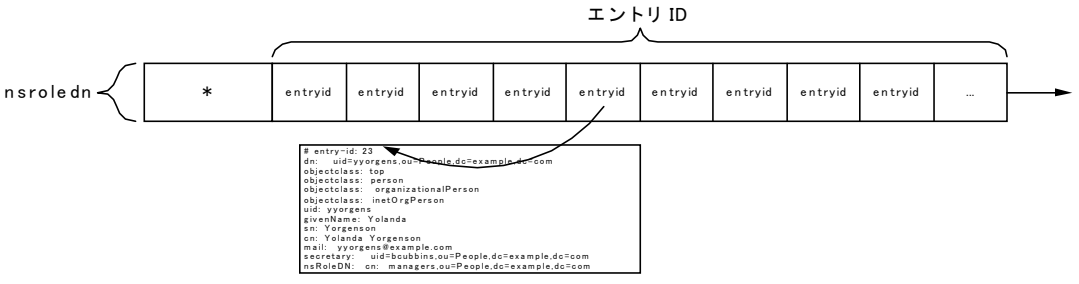
インデックスには、その維持にかかる処理コストに加えて、ディスクの空き容量や場合によってはメモリの空き容量に関してコストがかかることもあります。122 ページの「検索の最適化」で説明するように、検索に使用するデータベースキャッシュのサイズを最適化するときは、データベースキャッシュにエントリとインデックスの両方を保持できるだけのメモリを用意するように選択できます。インデックスが大きくなると、それだけ必要な空き容量も増えます。64 ビットのインデックスでは、32 ビットのインデックスよりも多少多くの空き容量が必要になります。

一般に、Directory Server のインスタンスのインデックスをチューニングすることは、高速検索処理から得られる利点で、多くの更新処理にかかるコストや多くの空き容量を必要とするコストが埋め合わせされるようなインデックスだけを維持することを意味します。有用なインデックスを維持するのは利点になりますが、クライアントで検索する頻度が少ない属性に対して使用することもないインデックスを維持することは無益です。

実在インデックス

図 7-1 は nsRoleDN 属性に対する実在インデックスです。このインデックスは属性の値には依存せず、データベース内の nsRoleDN 属性を持つすべてのエントリを含んでいることを示しています。この属性のすべての値は、* を使用して参照します。

図 7-1　実在インデックスの表現



図に示すように、内部の entryid 属性値を使用すると Directory Server によって、すばやい取得に備えてエントリへの参照を格納できます。Directory Server では、*dbinstance*_id2entry.db3 インデックスを使用するエントリを実際に取得します。ここで *dbinstance* は 129 ページの「インデックスについて」で説明するように、データベース識別子に依存しています。

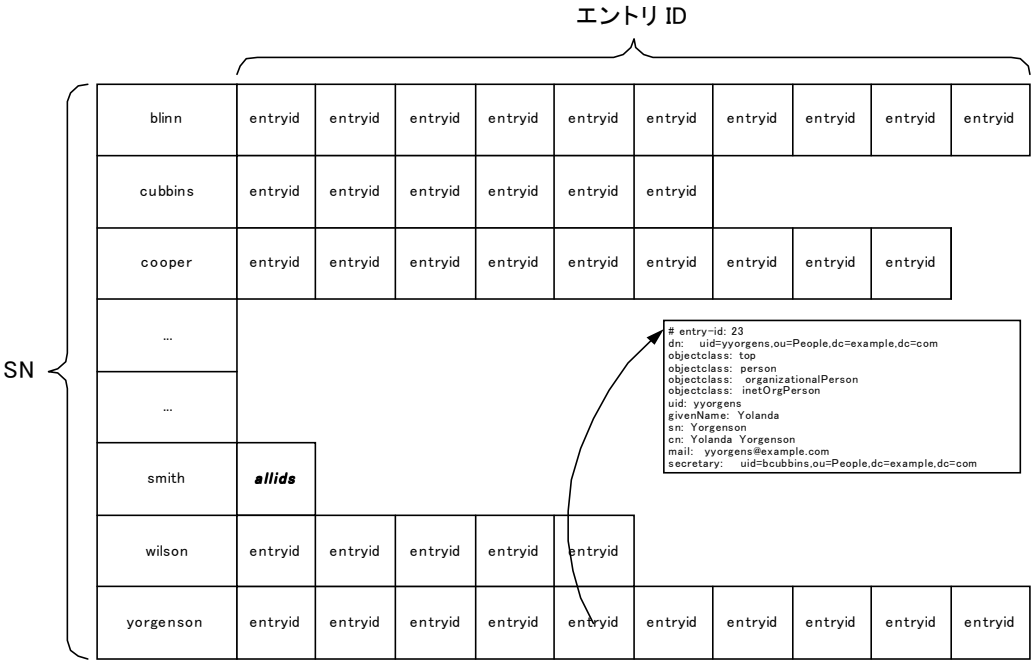
Directory Server では実在にインデックスされた属性を持つエントリの更新要求を受信すると、エントリをインデックスから削除する必要があるかどうかを判断し、更新通知をクライアントアプリケーションに返す前に、必要な変更を実行する必要があります。

実在インデックスにおけるコストは、一般にその他のインデックスタイプよりも低いですが、実在インデックスに保持されるインデックスのリストは長くなることがあります。

等価インデックス

図 7-2 は SN (姓) 属性の等価インデックスです。このインデックスに SN 属性の属性値を持つエントリの属性ごとにリストを保持する様子を示します。

図 7-2　等価インデックスの表現



Directory Server では等価にインデックスが作成された属性を持つエントリの更新要求を受信すると、エントリをインデックスから削除する必要があるかどうかを判断する必要があります。次に、インデックスに対してリストを追加または削除する必要があるかどうかを判断し、更新通知をクライアントアプリケーションに返す前に、必要な変更を実行する必要があります。

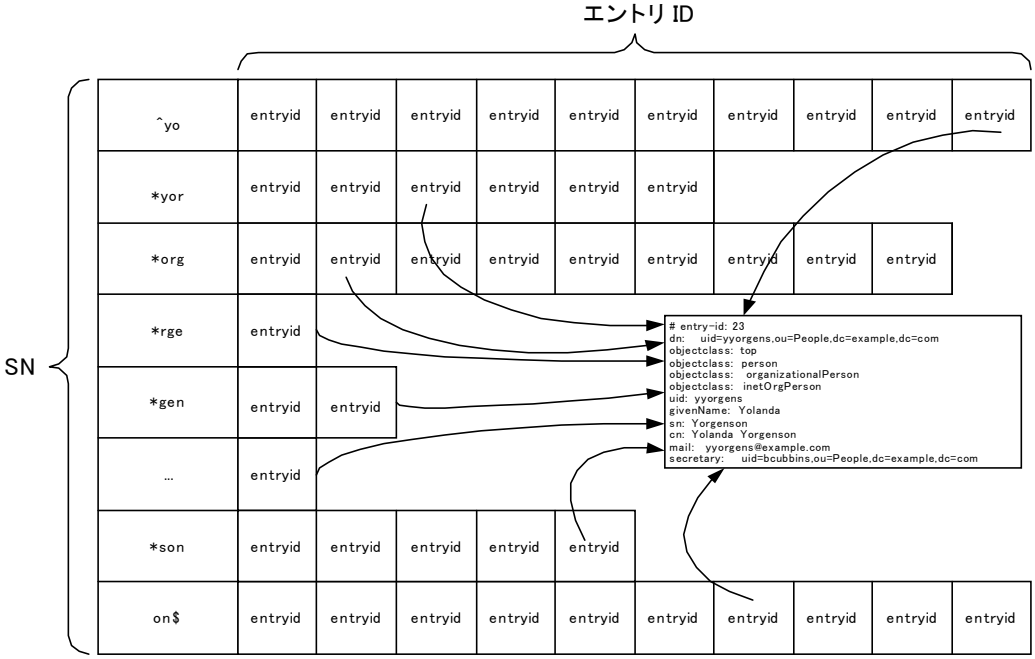
等価インデックスにおけるコストはたとえば部分文字列インデックスよりも一般に低くなりますが、空き容量は実在インデックスよりも多く必要になります。しかし、メッセージングサーバーのようなクライアントアプリケーションでは、検索パフォーマンスが最高であることから等価インデックスを使用します。写真や暗号化パスワードなど、大きなバイナリ属性に等価インデックスを使用することは避けてください。

部分文字列インデックス

図 7-3 は SN (姓) 属性の部分文字列インデックスです。このインデックスが属性値ごとの一連のリストを保持する様子の一部を示します。

Directory Server では、部分文字列 2 文字による検索がインデックスに見つかるように部分文字列のインデックスを作成します。そのため（sn=*ab*）という検索はインデックスを使用して高速化できますが、（sn=*a*）という検索は高速化できません。

図 7-3　部分文字列インデックスの表現



Directory Server では、ワイルドカードの前に 1 文字だけあるような前方部分文字列検索を可能とする高度な最適化も用意しています。つまり、たとえば（sn=*a*）や（sn=*a）ではなく、（sn=a*）の場合も部分文字列インデックスが利用できるときは高速化されます。

Directory Server では、組み込みのルールに準拠して部分文字列インデックスを作成します。これらの部分文字列は、システム管理者が設定することはできません。

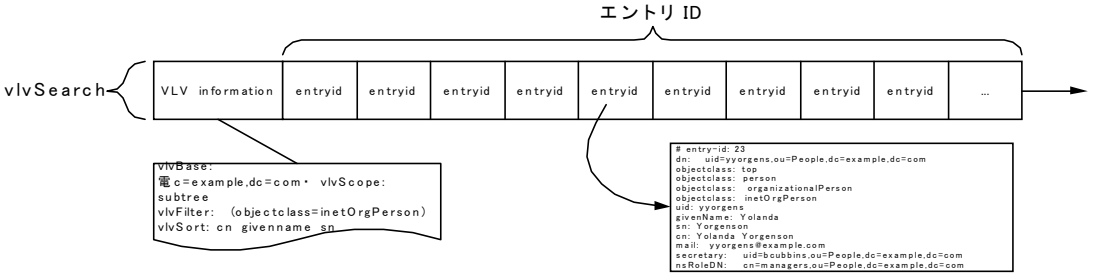
Directory Server では部分文字列にインデックスが作成された属性を持つエントリの更新要求を受信すると、エントリをインデックスから削除する必要があるかどうかと、エントリの変更がインデックスに与える影響を判断する必要があります。次に、インデックスに対してエントリ ID またはエントリ ID のリストを追加または削除する必要があるかどうかを判断し、更新通知をクライアントアプリケーションに返す前に、必要な変更を実行する必要があります。更新する回数は、属性値の文字列の長さによって異なります。

部分文字列インデックスを維持するコストは、一般にとても高くなります。コストはインデックス化された文字列の長さの関数となるため、`description` のように、長い文字列値となる可能性のある属性に対しては、不必要な部分文字列インデックスは避けてください。部分文字列インデックスは写真のようなバイナリ属性に適用することはできません。

ブラウズ (仮想リスト表示) インデックス

図 7-4 は仮想リスト表示のブラウズインデックスです。このインデックスが仮想リスト表示情報に依存している様子を示します。つまり、ブラウザインデックスの `vlvBase`、`vlvScope`、`vlvFilter`、および `vlvSort` の各属性値に依存しています。このタイプのインデックスのエントリ ID は `vlvSort` 基準に基づいて並べられています。

図 7-4　ブラウズインデックスの表現



Directory Server では `vlvFilter` に一致するエントリの更新要求を受信すると、エントリをインデックスから削除する必要があるかどうかを判断する必要があります。次に、エントリのリスト内の位置が正しいことを確認し、更新通知をクライアントアプリケーションに返す前に、必要な変更を実行する必要があります。

近似インデックス

Directory Server では、*metaphone* 音声アルゴリズムのバリエーションを使用した近似インデックスを維持します。このアルゴリズムでは、属性の文字列値を英語の音声発音の近似に分割します。着信した検索要求で一致する値は、同じアルゴリズムを使用して処理されます。アルゴリズムは音節に対しては厳密でないため、電話番号のように数値を含む属性には効果的ではありません。

アルゴリズムによって、属性値の文字列ごとにターゲット文字列が生成されます。そのため、この英語の文字列に「似た音」のインデックスにおけるコストは、等価インデックスのコストと同程度になります。

国際化インデックス

国際化インデックスでは、インデックスの維持に、ある特定のロケール用のマッチングルールを使用します。そのため、国際化インデックスにおけるコストは、部分文字列インデックスや等価インデックスを使用する場合のコストと同程度になります。

カスタムマッチングルールのサーバープラグインを使用すると、国際化インデックスやほかのタイプのインデックスに対する標準サポートを拡張できます。カスタムマッチングルールのプラグインについての詳細は、『Sun ONE Directory Server Plug-In API Programming Guide』を参照してください。

例 : エントリのインデックス

次に示すような、インデックスが作成されたサフィックスに追加されるユーザーエントリを考えてみます。ここで `uid` は等価検索、共通名 (`cn`) と姓 (`sn`) 属性は等価、部分文字列、および近似検索、`mail` 属性は等価検索、`telephoneNumber` 属性は等価および部分文字列検索、そして `description` 属性は部分文字列検索のために、それぞれインデックスが作成されています。

コード例 7-1 ユーザーエントリの例

```
dn:uid=yyorgens,ou=People,dc=example,dc=com
objectclass:top
objectclass:person
objectclass:organizationalPerson
objectclass:inetOrgPerson
uid:yyorgens
givenName:Yolanda
sn:Yorgenson
cn:Yolanda Yorgenson
mail:yolanda.yorgenson@example.com
```

コード例 7-1 ユーザーエントリの例 (続き)

```
telephoneNumber: 1-650-960-1300
description:Business Development Manager, Platinum Partners
```

このエントリを追加する際、Directory Server では cn、sn、mail、telephoneNumber、および description の各インデックスを変更する必要があります。表 7-2 に予想されるエントリの数を示します。

表 7-2 ユーザーエントリの例におけるインデックスの更新

属性	近似	等価	部分文字列[1]	インデックス更新の合計
uid		1		1
cn	1	1	17	19
sn	1	1	9	11
mail		1		1
telephoneNumber		1	11	12
description			47	47

1. ここでの description 文字列と同じぐらいの長さの文字列に対して、部分文字列インデックスを使用することは、ほとんどの場合でお勧めできません。

description に対する部分文字列インデックスの更新数 (47) が、ほかの全属性の更新数を合わせた数 (44) よりも大きいことに注意してください。また、description 文字列をさらに変更すると、更新回数が上限に達したり、新しい文字列への依存度が増加したりします。ほとんどの場合、description 値と同程度の長さの文字列に対して部分文字列インデックスを適用しないことで、この量の部分文字列インデックスを回避してください。

パフォーマンスに対するインデックスのチューニング

多くの場合、パフォーマンスに対してインデックスをチューニングするということは、インデックスをアクティブ化して頻繁に行われる検索の速度を上げること、および維持にコストがかかり、あまり頻繁に使用されないインデックスを無効にすることを指します。

注 データベースバックアップにはインデックスも含まれるため、Directory Server 設定に一致する必要があります。

インデックスの設定方法を変更したら、設定とデータの両方をバックアップしてください。

特定のアプリケーション専用のレプリカを含む大規模な導入では、Directory Server インスタンスごとに異なるインデックスを設定するように選択することができます。たとえば、次のようなトポロジがあるとします。

- 書き出しだけを処理するマスタ
- コンシューマに対するレプリカの負荷を処理するハブ
- メッセージングなど特定のアプリケーションを使用するコンシューマ

この場合のマスタは検索を処理しないため、たとえばコストのかかる部分文字列インデックスをマスタで保持しないようにすることができます。また、これまでほとんど使われていないほかのインデックスを無効にすることもできます。

ハブでは基本的に管理要求以外のクライアント要求を受信しません。そのため、Directory Server 自体で必要とするシステムインデックス以外のインデックスは無効にできます。

個別のアプリケーションだけを使用する特定のコンシューマに対しては、そのアプリケーションで使用しないインデックスをすべて無効にすることもできます。無効にするインデックスは、そのアプリケーションで実行される検索によって異なります。

インデックス検索だけを許可する

Directory Server では、コストのかかる、インデックスを使用しない検索を行わないようにすることが可能です。インデックスを使用しない検索を要求するクライアントには LDAP_UNWILLING_TO_PERFORM を返します。

特定のデータベースに対してインデックスを使用しない検索を許可しないようにするには、そのデータベースの nsslapd-require-index 属性値を on にします。

```
$ ldapmodify -h host -p port -D "cn=directory manager" -w password
dn:cn=example,cn=ldbm database, cn=plugins, cn=config
changetype:modify
replace:nsslapd-require-index
nsslapd-require-index:on
^D (Windows システムでは ^Z)
```

変更内容はすぐに有効になります。Directory Server を再起動する必要はありません。

インデックスリストの長さを制限する

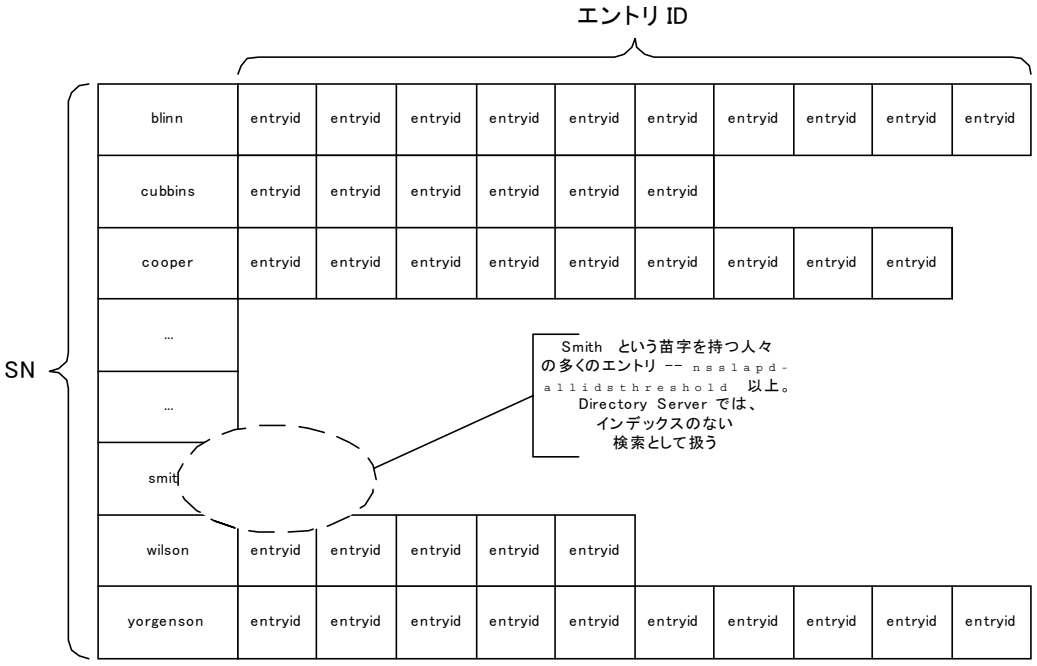
大規模で成長の速いディレクトリの導入では、特定のインデックスキーが効果がなくなる点に到達する可能性があります。効果がなくなる点では、あるキーに関連付けられたリストがとても長くなり、候補エントリのそのキーに対してときどき発生するインデックスされていない検索を実行するコストよりも、リストの維持にかかるコストの方がかかってしまいます。姓で等価インデックスされた大規模な電話帳アプリケーションを考えてみます。電話帳で Smith の個数が多すぎて、Smith のインデックスを維持するコストが、検索で得られる利益を上回ってしまうとします。このとき Directory Server では、Smith を姓でインデックスの作成を停止する必要があります。しかし Directory Server は別の姓のインデックスは継続する必要があります。

Directory Server にはこのような状況を処理するメカニズムが備わっています。設定属性にしきい値を設定します。あるキーに対するリスト内のエントリ数が設定した値になると、Directory Server では、そのキーの候補エントリを検索するために、インデックスされていない検索を実行する必要があると指定するトークンでキーのリストを置き換えます。156 ページの表 9-1 で説明するように、この値は検索でチェックした候補エントリの最大数 (nsslapd-lookthroughlimit で設定) に近いが小さな値になります。

このメカニズムは 全 ID しきい値 (all IDs threshold) と呼ばれ、cn=config,cn=ldbm database,cn=plugins,cn=config の場合の nsslapd-allidsthreshold というグローバルしきい値を設定するための設定属性にちなんで名付けられています。この値は現在 Directory Server インスタンスに対してグローバルになっている点に注意してください。この値をインデックスごとに別々に設定することはできません。

nsslapd-allidsthreshold よりも多い個数の Smith で姓をインデックスを作成する例を、図 7-5 に示します。

図 7-5　インデックスキーの全 ID しきい値に到達する



このしきい値は、このインデックステーブルでは 1 リストだけに影響します。ほかのキーのリストには影響しません。

不適切なインデックスリストサイズによる現象

クライアントが第 6 章「キャッシュサイズのチューニング」で説明するように、最初にインデックスされた検索を実行し、キャッシュサイズが適切にチューニングされている場合でも検索パフォーマンスが低いときは、不適切なしきい値がその原因である可能性があります。インデックスを使用した検索で検索パフォーマンスが低いときは、まずキャッシュサイズが適切にチューニングされていることを確認してください。次に、Directory Server が頻繁に全 ID しきい値に達しているかを確認するために、アクセスログを調べます。

アクセスログの RESULT メッセージの末尾にある notes=U フラグは、Directory Server がインデックスを使用しない検索を実行したことを示しています。同じ接続と操作における前の SRCH メッセージでは、使用する検索フィルタを指定します。次の 2 行では、10000 エントリを返す (cn=Smith) に対するインデックスを使用しない検索をトレースします。タイムスタンプはメッセージから削除されています。

```
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
```

インデックスされているはずの検索でこのようなペアが多くある場合は、しきい値を増やすことで検索パフォーマンスを向上させることができる可能性があります。

インデックスリストのしきい値サイズを変更する

nsslapd-allidsthreshold の適切な値は、通常、ディレクトリのエントリ総数のうち、5% 程度です。たとえば、デフォルト値 4000 は、80,000 エントリ以下を処理する Directory Server インスタンスに対してほぼ適切です。近いうちに多数のエントリをディレクトリに追加する予定がある場合、またはディレクトリが非常に大きくなることが予想される場合は、合計の 5% よりも大幅に多い値を設定できます。また、ほとんどの場合で書き出しだけをサポートするマスタよりも多くの検索をサポートするコンシューマのレプリカに、異なるしきい値を設定することもできます。近いうちに大きなディレクトリを LDIF から再初期化する予定がある場合、再初期化の直前に nsslapd-allidsthreshold の値を調整することもできます。ただしこの属性の値に対する変更には、すべてのインデックスを再構築する必要があります。どのような場合でも、全 ID しきい値を非常に高く (50,000 より高く) 設定することは、適切で明確な理由がない限り、大規模導入であっても避けてください。

全 ID しきい値の変更は次のように行います。変更を行なっている間、Directory Server インスタンスはサービスを中断します。

1. 当該の Directory Server インスタンスを停止します。
2. すべてのディレクトリデータベースを LDIF にエクスポートします。

   詳細は、『Sun ONE Directory Server 管理ガイド』を参照してください。
3. *ServerRoot*/slapd-*ServerID*/config/dse.ldif の nsslapd-allidsthreshold 属性の値を慎重に調整します。
4. すべてのディレクトリデータベースを LDIF から再初期化します。

   詳細は、『Sun ONE Directory Server 管理ガイド』を参照してください。

5. データベースキャッシュのサイズが古い全 ID しきい値でチューニングされていて、サーバーに十分な物理メモリがある場合は、データベースキャッシュのサイズをしきい値の増加分の 25% 増やしてみます。

   つまり、全 ID しきい値を 4000 から 6000 に増やす場合は、データベースキャッシュのサイズを増加したインデックスリストのサイズの 12.5% 程度増やすことができます。変更を実際の運用サーバーに適用する前に、最適なサイズを経験的に見つけてください。データベースキャッシュのチューニングについて詳細は、第 6 章「キャッシュサイズのチューニング」を参照してください。

6. Directory Server インスタンスを再起動します。

インデックスの断片化のトラブルシューティング

大規模インデックスと高い更新速度をサポートする Directory Server インスタンスでは、極端なインデックスキーの断片化を起こすおそれがあります。極端なインデックスキーの断片化が発生すると、データベースサイズが一定であってもパフォーマンスが低下することがあります。極端なインデックスキーの断片化がサーバーのパフォーマンスに深刻な影響を与えていると思われる理由がある場合は、影響を受けているインデックスを再生成して断片化を減らすことを検討してください。

インデックスを作成する方法については、『Sun ONE Directory Server 管理ガイド』を参照してください。

第 8 章

ログのチューニング

Directory Server には、表 8-1 に示すように、複数のログタイプが用意されています。この章では、異なるタイプのログを扱う方法について説明します。

表 8-1 Directory Server で使用されるログのタイプ

ログ	タイプ	使用方法
アクセス	フラットファイル	ディレクトリの評価でパターンを使用して、設定を検証し、アクセス問題を診断する
監査	フラットファイル	セキュリティとデータ整合性の監査証跡を提供する
更新履歴ログ	データベース	レプリカ間の同期を有効にする
エラー	フラットファイル	ディレクトリの導入をデバッグする
旧バージョン形式の更新履歴ログ	データベース	旧バージョンとの下位互換性を許可する
トランザクション	データベース	データベースの整合性を管理する

大規模な導入では、ログへの書き込みによりディスクが大量に使用されるため、パフォーマンスに重大な悪影響が生じます。大規模なシステムにおける頻繁なログ記録に伴なう潜在的な I/O ボトルネックに対しては、ログを別のディスクコントローラを使用する別の物理ディスク上に配置することを検討してください。

アクセスログ

アクセスログには、クライアントの接続と実行された操作に関する詳しい情報が記録されます。アクセスログは、アクセス問題の診断、サーバー設定の検証、およびサーバー使用パターンの評価には不可欠です。ただし、デフォルトのログレベルではほとんどの導入で多大なディスクアクセスが発生し、ディスク活動の量がサーバーのパフォーマンスに悪影響を及ぼすおそれがあります。

アクセスログでは有用なトラブルシューティング情報が得られますが、I/O ボトルネックとなる可能性もあります。ディレクトリを導入したらアクセスログは無効にし、エラーやパフォーマンスの問題なく実行することを検討します。アクセスログが実際の運用環境で必要となるときは、ログレベルを最低限必要なレベルに設定します。さらに、アクセスログを専用の物理ディスクか、大きな I/O バッファを持つ高速のディスクサブシステムに配置することを検討します。表 8-2 に、特定の属性に対する詳細な要件を示します。

表 8-2 アクセスログのチューニング要件

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-accesslog	アクセスログファイルのパスとファイル名を指定する 小規模な導入では、アクセスログは監査ログやエラーログとディスクを共有することも可能 大規模な導入では、アクセスログを専用のコントローラを持つ専用のディスクやディスクサブシステムに配置することを検討する。大きな I/O バッファを持つディスクを選択する
nsslapd-accesslog-level	ログで使用される情報のレベルを指定する より高いレベルのログが必要ないなら 0 に変更すると、アクセスログを記録しない (デフォルトは 256 でエントリへのアクセスをログに記録)
nsslapd-accesslog-logbuffering	アクセスログをバッファに入れるかどうかを決める 生成されたアクセスログメッセージを確認するために、バッファを無効にする必要があるまでは、on (デフォルト) にしておくこと。バッファを無効にすると、全体のパフォーマンスが低下する

表 8-2　アクセスログのチューニング要件 (続き)

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-accesslog-logging-enabled	アクセスログを有効または無効にする 最大のパフォーマンスを得るには off にする (デフォルトは on) 導入でアクセスログが有効であることが必要な場合は、nsslapd-accesslog-level を可能な限り低い値に設定にし、アクセスログを専用のディスクやディスクサブシステムに配置する。アクセスログを頻繁に (毎日または毎週) ローテーションし、nsslapd-accesslog-logmaxdiskspace および nsslapd-accesslog-logminfreediskspace を使用してディスク容量の使用状況を管理する
nsslapd-accesslog-logmaxdiskspace	すべてのアクセスログ、つまり現在のログとローテーションされたログが消費する可能性のある最大ディスク容量を指定する この値はアクセスログ専用のディスク容量の合計よりも低く設定すること 監査、アクセス、エラーの各ログで同じディスクを使用する場合は、これらの 3 つのログに対して十分なディスク容量があること アクセスログが専用のディスクにある場合は、この変数をディスクのサイズに設定する
nsslapd-accesslog-logminfreediskspace	古いログがパージされる前に利用可能な最低限のディスク空き容量を指定する ディスクの空き容量がこの属性で指定した値を下回った場合、ディスクの空き容量がこの属性の設定に相当するまで、もっとも古いアクセスログから順番に削除される。ディスクがいっぱいでアクセスログに書き込みできない場合、サーバーはシャットダウンする

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

Sun ONE Directory Server Resource Kit のドキュメントではアクセスログからの情報の抽出について扱っています。詳細は、12 ページの「Directory Server ツールのダウンロード」を参照してください。

監査ログ

監査ログには、サーバーの設定だけでなく、各データベースに対するすべての変更に関する詳細情報が記録されます。監査ログはデフォルトで無効に設定されています。

変更が大量になる導入で有効にすると、監査ログが有効になるため、パフォーマンスが全体的に著しく低下してしまいます。導入で必要となるまで、監査ログは無効にしておきます。監査ログを必要とする大規模な導入の場合は、監査ログ用に別のコントローラ上にある別のディスクを割り当てることを検討します。表 8-3 に、特定の属性に対する詳細な要件を示します。

表 8-3 監査ログのチューニング要件

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-auditlog	監査ログファイルのパスとファイル名を指定する 小規模な導入では、監査ログはアクセスログやエラーログとディスクを共有することも可能 大規模な導入では、監査ログを専用のコントローラを持つ専用のディスクに配置することを検討する。大きな I/O バッファを持つディスクを選択する
nsslapd-auditlog-logging-enabled	監査ログを有効または無効にする 監査ログが必要となるまで、off (デフォルト設定) のままにしておく
nsslapd-auditlog-logmaxdiskspace	すべての監査ログ、つまり現在のログとローテーションされたログが消費する可能性のある最大ディスク容量を指定する この値は監査ログ専用のディスク容量の合計よりも低く設定すること 監査、アクセス、エラーの各ログで同じディスクを使用する場合は、これらの 3 つのログに対して十分なディスク容量があること 監査ログが専用のディスクにある場合は、この変数をディスクのサイズに設定する

表 8-3 監査ログのチューニング要件 (続き)

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-auditlog-logminfreediskspace	古いログがパージされる前に利用可能な最低限のディスク空き容量を指定する ディスクの空き容量がこの属性で指定した値を下回った場合、ディスクの空き容量がこの属性の設定に相当するまで、もっとも古い監査ログから順番に削除される。ディスクがいっぱいで監査ログに書き込みできない場合、サーバーはシャットダウンする

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

エラーログ

Directory Server インスタンスのエラーログには、通常のサーバー運用中に発生した詳細エラー、警告、および情報メッセージが含まれます。デフォルトのログレベルは低いため、比較的小さなディスクアクティビティが行われます。

ログレベルを高く設定するとデバッグ情報が生成されますが、Directory Server では、大量のメッセージをディスクに書き込み始めます。書き込みにおける負荷によって、パフォーマンスが全体的に著しく低下します。パフォーマンスの低下を避けるには、一度にすべてのコンポーネントのログレベルをアクティブにするのではなく、ログレベルをコンポーネントごとに段階的に上げていきます。

エラーログではログのバッファをサポートしていません。すべてのメッセージは即座にディスクにフラッシュされます。大規模な導入では、デバッグが必要になるときに、別のコントローラにある別のディスクをエラーログ用に割り当てることを検討します。表 8-4 に、特定の属性に対する詳細な要件を示します。

表 8-4　エラーログのチューニング要件

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-errorlog	エラーログファイルのパスとファイル名を指定する 小規模な導入では、エラーログはアクセスログや監査ログとディスクを共有することも可能 大規模な導入では、エラーログを専用のコントローラを持つ専用のディスクに配置することを検討する。大きな I/O バッファを持つディスクを選択する
nsslapd-errorlog-logging-enabled	エラーログを有効または無効にする on (デフォルト設定) のままにしておくこと
nsslapd-errorlog-logmaxdiskspace	すべてのエラーログ、つまり現在のログとローテーションされたログが消費する可能性のある最大ディスク容量を指定する この値はエラーログ専用のディスク容量の合計よりも低く設定すること 監査、アクセス、エラーの各ログで同じディスクを使用する場合は、これらの 3 つのログに対して十分なディスク容量があること エラーログが専用のディスクにある場合は、この変数をディスクのサイズに設定する
nsslapd-errorlog-logminfreediskspace	古いログがパージされる前に利用可能な最低限のディスク空き容量を指定する ディスクの空き容量がこの属性で指定した値を下回った場合、ディスクの空き容量がこの属性の設定に相当するまで、もっとも古いエラーログから順番に削除される。ディスクがいっぱいでエラーログに書き込みできない場合、サーバーはシャットダウンする
nsslapd-infolog-area	情報メッセージがログに記録されるコンポーネントを指定する コンポーネントをデバッグするまでは 0 (デフォルト) のままにしておくこと。実際の運用サーバーでは一度に複数のコンポーネントを設定しないこと

表 8-4 エラーログのチューニング要件 (続き)

設定属性 (dn: cn=config の場合)	簡単な説明とチューニングの推奨事項
nsslapd-infolog-level	ログで使用される情報のレベルを指定する nsslapd-infolog-area 単独の設定では十分な詳細を得られないコンポーネントをデバッグするまでは、0 (デフォルト) のままにしておくこと

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

マルチマスターレプリケーションの更新履歴ログ

Directory Server は、レプリケーションの更新履歴ログを使ってレプリカ間の同期を実現します。更新履歴ログについては『Sun ONE Directory Server Deployment Guide』を、設定の詳細については『Sun ONE Directory Server Reference Manual』を、それぞれ参照してください。表 8-5 に、特定の属性に対する詳細な要件を示します。

表 8-5 マルチマスター更新履歴ログのチューニング要件

設定エントリの DN と設定属性	簡単な説明とチューニングの推奨事項
dn:cn=changelog5,cn=config nsslapd-cachememsize	更新履歴ログデータベースのキャッシュサイズを指定する 大規模な導入では、この属性のデフォルト値 10M バイトの変更を検討する
dn:cn=changelog5,cn=config nsslapd-changelogdir	更新履歴ログデータベースのパスとファイル名を指定する 更新履歴ログを専用のコントローラを持つ専用のディスクやディスクサブシステムに配置することを検討する。大きな I/O バッファが有用
dn:cn=changelog5,cn=config nsslapd-changemaxage	更新履歴ログ内のエントリの最長保存期間を指定する この値を 0 (デフォルト、期限なし) から、レプリケートされたサーバー間で同期が完全にとられ、更新履歴ログが削除可能になるまでの期間に変更する

表 8-5　マルチマスター更新履歴ログのチューニング要件 (続き)

設定エントリの DN と設定属性	簡単な説明とチューニングの推奨事項
dn:cn=changelog5,cn=config nsslapd-changemaxentries	更新履歴ログ内のエントリの最大数を指定する この値を 0 (デフォルト、上限なし) から、更新履歴ログが削除されるまでにレプリケートされたサーバー間の同期が完全にとれるだけの数に変更する
dn:cn=changelog5,cn=config nsslapd-cachesize	更新履歴ログのデータベースキャッシュ内のエントリの最大数を指定する この値を -1 (デフォルト、上限なし) から、エントリがフラッシュされるまでに更新履歴ログ内に保持されるエントリの最大数に変更する

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

旧バージョン形式の更新履歴ログ

Directory Server には旧バージョン形式の更新履歴ログのプラグインが同梱されており、サプライヤのサーバーで起こった変化を、Directory Server 4.x リリースと互換のある、LDAP を通したアクセス可能な形式で記録することができます。旧バージョン形式の更新履歴ログのプラグインはデフォルトでは無効になっており、互換性の理由から必要となるまでは有効にすることをお勧めしません。詳細は、『Sun ONE Directory Server Reference Manual』を参照してください。表 8-6 に、特定の属性に対する詳細な要件を示します。

表 8-6　旧バージョン形式の更新履歴ログのチューニング要件

設定エントリの DN と設定属性	簡単な説明とチューニングの推奨事項
dn:cn=Retro Changelog Plugin,cn=plugins,cn=config nsslapd-changelogdir	旧バージョン形式の更新履歴ログファイルのパスとファイル名を指定する 旧バージョン形式の更新履歴ログを専用のコントローラを持つ専用のディスクやディスクサブシステムに配置することを検討する。大きな I/O バッファが有用

表 8-6　旧バージョン形式の更新履歴ログのチューニング要件 (続き)

設定エントリの DN と設定属性	簡単な説明とチューニングの推奨事項
dn:cn=Retro Changelog Plugin,cn=plugins,cn=config	旧バージョン形式の更新履歴ログ内のエントリの最長保存期間を指定する
nsslapd-changelogmaxage	この値を 0 (デフォルト、期限なし) から旧バージョン形式の更新履歴ログを使用するクライアントが、生成したログエントリを処理するまでの期間に変更する
dn:cn=Retro Changelog Plugin,cn=plugins,cn=config	旧バージョン形式の更新履歴ログ内のエントリの最大数を指定する
changelogmaxentries	この値を 0 (デフォルト、上限なし) から削除までに旧バージョン形式の更新履歴ログに保持されるエントリの最大数に変更する

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

トランザクションログ

Directory Server では、トランザクションログを通じてデータベースの整合性を管理しています。add、modify、delete、または modrdn の各更新操作を受け入れる際、Directory Server では操作に関するログメッセージをトランザクションログに書き込みます。永続トランザクションログ (デフォルトで有効) では、データの整合性が保証されます。この保証は、更新操作の結果コードがクライアントアプリケーションに返される前に、更新操作がディスク上のトランザクションログに確実にコミットされることで実現しています。システム障害時、Directory Server ではデータベースの回復にトランザクションログを使用します。データベースが異常にシャットダウンしたときの回復にトランザクションログを利用するには、トランザクションログとディレクトリデータベースを別々のディスクサブシステムに格納することを検討します。

トランザクションログは、特に永続性を有効にしている場合にディスクを著しく消費します。これは、更新のパフォーマンスで大きなボトルネックとなる可能性があります。システム障害に備えてデータ整合性をより適切に保護するだけでなく、トランザクションログとデータベースを Sun StorEdge ディスクアレイのような RAID システムを別々に用意して格納することで、更新パフォーマンスを向上させることができます。表 8-7 に、特定の属性に対する詳細な要件を示します。

表 8-7　トランザクションログのチューニング要件

設定エントリの DN と設定属性	簡単な説明とチューニングの推奨事項
dn:cn=config,cn=ldbm database,cn=plugins,cn=config nsslapd-db-checkpoint-interval	Directory Server がトランザクションログを検査し、データベースシステム全体を確実にディスクに同期させ、トランザクションログをクリーンにする頻度を指定する 経験テストを基にしたデータベースのパフォーマンス最適化で異なる値が必要となるまでは、60 (デフォルトの間隔 / 秒) のままにしておくこと。この属性の値を増やすと更新操作のパフォーマンスが向上するが、同時に、異常なシャットダウンからの回復に時間がかかったり、トランザクションログでより多くのディスク容量を使用することになったりする
dn:cn=config,cn=ldbm database,cn=plugins,cn=config nsslapd-db-durable-transaction	更新操作が、結果コードがクライアントに送信される前に、ディスク上のトランザクションログにコミットされるかどうかを指定する 高レベルのデータ整合性を必要とする導入では、on (デフォルト) のままにしておくこと。導入によっては、パフォーマンスを向上させるために、永続トランザクションログを無効にする場合がある。ただし、無効にすると、ログメッセージをファイルシステムにフラッシュしているが、ディスクにフラッシュしていない場合、システム障害時にログメッセージが失われるおそれがある。したがって、場合によっては、永続トランザクションログが off のとき、クライアントが更新成功の結果コードを受信した後でも回復できない更新がある
dn:cn=config,cn=ldbm database,cn=plugins,cn=config nsslapd-db-logdirectory	トランザクションログファイルのパスとファイル名を指定する トランザクションログを専用のコントローラを持つ非常に高速の専用のディスクやディスクサブシステムに格納することを検討する

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

第 9 章

その他のリソースの使用の管理

キャッシュのサイズ、属性値のインデックス付け、およびログ管理の最適化を行うと、クライアントアプリケーションが使用できる Directory Server のリソースを制限する方法や Directory Server がシステムリソースを使用する方法を調整することが有用であることがわかります。また、Directory Server プラグインとして提供される機能の一部を再設定したり、無効にしたりすることも有用だとわかるでしょう。

クライアントが使用できるリソースの制限

デフォルトでは、実際に必要となる Directory Server リソースよりも多くのリソースをクライアントが使用できるように設定されている場合があります。これは、クライアントアプリケーションが、多数の接続を開いてそのままアイドル状態にし、あるいは未使用のままにしたり、コストがかかる不必要なインデックスを使用しない検索を行なったり、ディレクトリに大量で無計画なバイナリ属性値の格納を行なったりして、無意識のうちに、あるいは故意にサーバーのパフォーマンスに悪い影響を与える可能性をそのまま残しておくことになります。

導入状況によっては、デフォルト設定の変更をお勧めできない場合があります。この節で説明する設定属性値を変更しないような導入の場合は、Sun ONE Directory Proxy Server ソフトウェアを使用して、外部から制限を設定し、サービスを不能にする攻撃を防ぐことを検討します。

導入状況によっては、Directory Server のインスタンスが、メッセージングサーバーなどディレクトリに負荷のかかるクライアントアプリケーションとユーザーメールアプリケーションなど一時的にディレクトリを使用するクライアントの両方をサポートする必要があります。このような場合には、『Sun ONE Directory Server 管理ガイド』で説明するように、バインド DN リソース制限を使用して、ディレクトリに負荷のかかるアプリケーションに対して個別に制限を高めることを検討します。

表 9-1 の推奨事項は、すべてのクライアントアプリケーションが使用できるリソースを制限するための設定を示しています。これらの制限は Directory Manager には適用されないため、クライアントアプリケーションが決して Directory Manager として接続しないようにしてください。

表 9-1 クライアントが使用できるリソースを制限する場合のチューニングの推奨事項

設定エントリの DN と属性	簡単な説明とチューニングの推奨事項
dn:cn=config nsslapd-idletimeout	Directory Server がアイドル状態のクライアント接続を閉じる時間を、秒単位で設定する。ここで、「アイドル状態」とは、接続が開いたままの状態にもかかわらず何も操作が要求されないことを意味する。デフォルトでは、時間制限は設定されていない メッセージングサーバーなど一部のアプリケーションでは、トラフィックが少ないときにアイドル状態のままにする接続のプールを開く場合があるが、これは閉じてはならない。この場合は、アプリケーションをサポートするレプリカを専用にすることが望ましい。レプリカを専用にできない場合は、バインド DN 制限を検討する いずれの場合でも、ほかのアプリケーションが開いていると予測する接続を閉じないだけの大きな値を設定する。ただし、不正に接続がアイドル状態のままにならないよう小さな値を設定する。最適化テストの開始点として、設定を 120 秒 (2 分) にすることを検討する
dn:cn=config nsslapd-ioblocktimeout	Directory Server が停止状態のクライアント接続を閉じる時間を、ミリ秒単位で設定する。ここで、「停止状態」とは、サーバーがクライアントへの出力の送信またはクライアントからの入力の読み込みのいずれかをブロックされていることを意味する 特にサービスを不能にする攻撃にさらされている Directory Server インスタンスの場合は、この値をデフォルトの 1,800,000 ミリ秒 (30 分) より小さく設定することを検討する

表 9-1 クライアントが使用できるリソースを制限する場合のチューニングの推奨事項 (続き)

設定エントリの DN と属性	簡単な説明とチューニングの推奨事項
dn:cn=config,cn=ldbm database,cn=plugins,cn=config	検索中のマッチングをチェックする候補エントリの最大数を設定する
nsslapd-lookthroughlimit	メッセージングサーバーなど一部のアプリケーションでは、ディレクトリ全体の検索が必要になる場合がある。この場合は、アプリケーションをサポートするレプリカを専用にすることが望ましい。レプリカを専用にできない場合は、バインド DN 制限を検討する いずれの場合でも、この値をデフォルトの 5000 エントリよりも小さくすることを検討する。ただし、nsslapd-sizelimit のしきい値より小さくしないようにする
dn:cn=config nsslapd-maxbersize	着信メッセージの最大サイズを、バイト単位で指定する。Directory Server は、この制限より大きいエントリを追加する要求を拒否する ディレクトリデータの最大エントリサイズを正確に予測できる自信がある場合は、この値をデフォルトの 2097152 (2M バイト) から最大予測ディレクトリエントリのサイズに変更することを検討する
dn:cn=config nsslapd-maxthreadsperconn	クライアント接続当たりのスレッドの最大数を設定する メッセージングサーバーなど一部のアプリケーションでは、接続のプールを開き、各接続で多数の要求を行う場合がある。この場合は、アプリケーションをサポートするレプリカを専用にすることが望ましい。レプリカを専用にできない場合は、バインド DN 制限を検討する 一部のアプリケーションが行う接続当たりの要求が多くなると予測する場合は、この値をデフォルトの 5 より大きくすることを検討する。ただし、この値を 10 より大きくしてはならない。通常、接続当たり 10 スレッドを超える指定は推奨されない

表 9-1 クライアントが使用できるリソースを制限する場合のチューニングの推奨事項 (続き)

設定エントリの DN と属性	簡単な説明とチューニングの推奨事項
dn:cn=config nsslapd-sizelimit	Directory Server が、検索要求に応答して返すエントリの最大数を指定する メッセージングサーバーなど一部のアプリケーションでは、ディレクトリ全体の検索が必要になる場合がある。この場合は、アプリケーションをサポートするレプリカを専用にすることが望ましい。レプリカを専用にできない場合は、バインド DN 制限を検討する いずれの場合でも、この値をデフォルトの 2000 エントリより小さくすることを検討する
dn:cn=config nsslapd-timelimit	Directory Server が検索要求の処理に使用できる最大時間を、秒単位で設定する メッセージングサーバーなど一部のアプリケーションでは、非常に大きな検索の実行が必要になる場合がある。この場合は、アプリケーションをサポートするレプリカを専用にすることが望ましい。レプリカを専用にできない場合は、バインド DN 制限を検討する いずれの場合でも、導入要件を満たす限り、この値をできるだけ小さく設定する。デフォルト値 3600 秒 (1 時間) は、多くの導入の場合、必要以上に大きな値になる。最適化テストの開始点として、設定を 600 秒 (10 分) にすることを検討する

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

使用可能なシステムリソースの使用

導入要件によっては、Directory Server インスタンスがシステムリソースとネットワークリソースを使用する方法、アクセス制御を管理する方法、およびサーバープラグインを設定する方法を調整することができます。表 9-2 の推奨事項は、システムリソースの設定を示しています。

表 9-2 システムリソースの使用を設定する場合のチューニングの推奨事項

属性 (dn: cn=config)	簡単な説明とチューニングの推奨事項
nsslapd-listenhost	Directory Server が待機する IP インタフェースのホスト名を設定する。この属性は単一値である デフォルトの動作はすべてのインタフェースで待機する。このデフォルトの動作は、可用性とスループットのために、冗長ネットワークインタフェースを使用する大規模システムの導入に適合する マルチホームシステムを導入したり、あるいは別のインタフェースを介して個々のプロトコルをサポートするシステムの IPv4 や IPv6 だけを待機したりする場合に、この値の設定を検討する。SSL を使用する場合は、nsslapd-securelistenhost の設定を検討する
nsslapd-maxdescriptors	Directory Server が使用を試みるファイル記述子の最大数を設定する Directory Server は、ファイル記述子を使用して、クライアント接続を処理し、ファイルを内部的に管理する。ファイル記述子が不足しているため、Directory Server が新しい接続の待機を中止する場合があることを、エラーログが示す場合は、この属性の値を大きくして、Directory Server が同時に処理できるクライアント接続の数を増加する 105 ページの「ファイル記述子」で説明する手順に従って、システムで使用可能なファイル記述子の数を増加させた場合は、それに応じてこの属性値を設定する。この属性値は、システムで使用可能なファイル記述子の最大数以下にする必要がある
nsslapd-nagle	TCP パケットの送信を、ソケットレベルで遅延させるかどうかを設定する off (デフォルト) のままにして、クライアントアプリケーションへの結果の送信がプロトコルレベルで遅延しないようにする

表 9-2　システムリソースの使用を設定する場合のチューニングの推奨事項 (続き)

属性 (dn: cn=config)	簡単な説明とチューニングの推奨事項
nsslapd-reservedescriptors	Directory Server がインデックス付け、レプリケーション、およびその他の内部処理を管理するために保持するファイル記述子の数を設定する。これらのファイル記述子は、クライアント接続を処理するために Directory Server で使用されることはない 次の条件がすべて当てはまる場合は、この属性の値をデフォルトの 64 から増やすことを検討する • Directory Server が 10 を超えるコンシューマをレプリケートするか、または Directory Server が 30 を超えるインデックスファイルを管理する • Directory Server が、多数のクライアント接続を処理する • エラーログのメッセージによると、クライアント接続に無関係の操作のために、Directory Server がファイル記述子を使い果たしている 予備ファイル記述子の数が増加するにつれて、クライアント接続の処理に使用できるファイル記述子の数が減少することに注意する。この属性の値を増加させる場合は、システムで使用可能なファイル記述子の数の増加、および nsslapd-maxdescriptors の値の増加を検討する この属性を変更する場合、予備のファイル記述子数の最初の見積もりとして、nsslapd-reservedescriptors の値を次の計算式のように設定することを試みる 20 + 4 * (データベース数) + (インデックス総数) + (nsoperationconnectionslimit の値) * (連鎖バックエンド数) + *ReplDescriptors* + *PTADescriptors* + *SSLDescriptors* ここで、*ReplDescriptors* は、サプライヤレプリカの数 + レプリケーションが使用されている場合 8、*PTADescriptors* は、PTA (パススルー認証) プラグインが有効の場合 3 (無効の場合は 0)、*SSLDescriptors* は、SSL が使用されている場合 5 (使用されていない場合 0) を示す インスタンスがサフィックス当たり 1 データベースを超えて使用するように設定されていない限り、データベースの数はインスタンスのサフィックスの数と同一である。経験的テストを通じて見積もりを検証する

表 9-2 システムリソースの使用を設定する場合のチューニングの推奨事項 (続き)

属性 (dn: cn=config)	簡単な説明とチューニングの推奨事項
nsslapd-securelistenhost	Directory Server が SSL 接続を待機する IP インタフェースのホスト名を設定する。この属性は単一値である デフォルトの動作ではすべてのインタフェースで待機する。この属性は nsslapd-listenhost と同じ方法で検討する
nsslapd-threadnumber	Directory Server が使用するスレッドの数を設定する 次の条件のいずれかに当てはまる場合は、この属性の値を調整することを検討する • クライアントアプリケーションが、更新や複雑な検索を同時に行う場合など多くの時間のかかる操作を実行する • Directory Server が多くの同時クライアント接続をサポートする • Directory Server が 5,000,000 を超えるエントリを処理する マルチプロセッサシステムは、単一プロセッサシステムより大きなスレッドプールを保持できる。この属性の値を最適化するときの最初の見積もりとして、「プロセッサ数の 2 倍」または「同時更新数 + 20」を使用する。表 9-1 の説明に従って、クライアント接続当たりのスレッドの最大数 nsslapd-maxthreadsperconn の調整も検討する。クライアント接続を処理するこれらのスレッドの最大数は、システムで使用可能なファイル記述子の最大数を超えることはできない。この属性の値は、増やすよりもむしろ減らしたほうが、有効である場合がある 経験的テストを通じて見積もりを検証する。結果は、特定の導入状況だけではなく基盤となるシステムにも依存する

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

アクセス制御の管理

Directory Server は、メモリ管理の改善やマクロ ACI (Access Control Instruction) のサポートなど ACI のパフォーマンスと拡張性の向上を実現しました。この改善にもかかわらず、Directory Server は複雑な ACI を評価するためにかなりのシステムリソースを使用します。したがって、複雑な ACI の大量の使用はパフォーマンスに悪い影響を与えることになります。

マクロ ACI は使用する ACI の数の制限に役立ちます。ACI の数を制限することによって、アクセス制御の管理をより簡単にして、システムの負荷を軽減します。マクロは、ACI の中で DN、または DN の一部を表現する可変部分です。マクロは、ACI の対象、ACI のバインドルール、またはその両方で使用できます。Directory Server が要求を受け取ると、要求によって実行される操作のターゲットとなるリソースに一致する ACI マクロをチェックします。マクロが一致する場合、Directory Server は、マクロを実際の DN の値に置き換えます。次に、Directory Server は通常どおり ACI を評価します。ACI の詳細については、『Sun ONE Directory Server 管理ガイド』を参照してください。

テストの結果、Directory Server は 50,000 を超える ACI をサポートできることが実証されました。さまざまな導入例のパフォーマンスに与える影響については、現在分析中です。パフォーマンスに悪い影響を与えず、アクセス制御の管理における複雑さを軽減するために、できるだけ ACI の数を少なくします。複雑な ACI 環境を含む導入の場合は、複数のアクセス制御機能を提供する Sun ONE Directory Proxy Server の使用を検討します。

サーバープラグインの設定

Directory Server は、アクセス制御、レプリケーション、構文検査、および属性一意性検査プラグインなど多くの重要な機能を実装しています。特定の導入では、プラグインの一部を再設定することが有効な場合があります。表 9-3 の推奨事項は、標準プラグインの設定例を示しています。

表 9-3 標準プラグインのチューニング例の推奨事項

名前および DN	簡単な説明とチューニングの推奨事項
7 ビット検査プラグイン dn:cn=7-bit check,cn=plugins,cn=config	Directory Server が、属性値が 7 ビットクリーンであるかどうかを検査できるようにする。つまり、与えられた属性値には 7 ビット符号化に適合する文字だけが含まれていることを検査する インフラストラクチャが、たとえば日本語の文字など幅広い符号化をサポートするように設計されている場合は、このプラグインの無効化を選択できる (デフォルトは on)
旧バージョンのレプリケーションプラグイン dn:cn=Legacy Replication Plugin,cn=plugins,cn=config	Directory Server が、4.X サプライヤのコンシューマとして機能できるようにする たとえば、アップグレードの間に Directory Server を 4.X サプライヤのコンシューマとして使用しない場合は、このプラグインを `off` に設定する (4.x レプリケーション機能が必要な場合のデフォルトは on)
参照整合性検査プラグイン dn:cn=referential integrity postoperation,cn=plugins,cn=config	関連するエントリ間の関係が保持されていることを、Directory Server が確認できるようにする。たとえば、ユーザーエントリがディレクトリから削除されたり、名前が変更されたりしたときに、ユーザーが属するグループがユーザーの介入なしで必要に応じて更新される すべてのマスターでこのプラグインを有効にして設定する プラグインを有効にする場合は、プラグインで使用できるように設定するすべての属性の等価インデックスを作成する。プラグインは、更新するエントリを検索するときにこのインデックスを使用する。使用する属性の等価インデックスがない場合、プラグインはパフォーマンスに悪い影響を与え、コストがかかるインデックスを使用しない検索を実行する必要がある プラグインの設定と有効化の手順については、『Sun ONE Directory Server 管理ガイド』を参照

個々の設定属性に関する詳細については、『Sun ONE Directory Server Reference Manual』を参照してください。

インストール製品のレイアウト

この付録では、標準インストール後の製品ソフトウェアのレイアウトについて要約して説明します。インストールされたファイルのうち、この付録にリストされているファイルおよび製品マニュアルで説明されているファイルだけが、サポートの対象となる公開インタフェース製品に属します。

注

この付録で示されている例は、Solaris オペレーティング環境での製品インストールを反映しています。ファイル名および拡張子は、ほかのプラットフォーム上のインストールでは異なる場合があります。

製品の Solaris パッケージバージョンをインストール後、`pkgchk`(1M) ユーティリティの `pkgchk -v` *package-name* コマンドを使用して、特定のパッケージに対するすべてのインストールパス名のリストを取得することもできます。

Solaris オペレーティング環境など一部のプラットフォームには、ディレクトリサービスを管理する統合ツールが用意されています。Sun ONE Directory Server もツールを提供します。この付録でリストされているツールの詳細は、『Sun ONE Directory Server 管理ガイド』と『Sun ONE Directory Server Reference Manual』を参照してください。

ServerRoot ディレクトリ

ServerRoot ディレクトリには、さまざまなシステム管理ユーティリティがあります。プラットフォーム、設定、およびインストール内容の *ServerRoot* ディレクトリに対応するパスの指定方法については、10 ページの「デフォルトパスおよびファイル名」を参照してください。

表 A-1　*ServerRoot* の下にあるユーティリティ

ユーティリティ	内容
ServerRoot/restart-admin	管理サーバーの再起動
ServerRoot/start-admin	管理サーバーの起動
ServerRoot/startconsole	Sun ONE サーバーコンソールの起動
ServerRoot/stop-admin	管理サーバーの停止
ServerRoot/uninstall	製品ソフトウェアのアンインストール

ServerRoot/bin ディレクトリには、サーバーインスタンスの作成時に内部的に使用される製品のバイナリファイルと設定テンプレートがあります。

表 A-2　*ServerRoot*/bin　の下にあるファイル

ファイル	内容
ServerRoot/bin/	次のファイル以外は、内部処理に使用
ServerRoot/bin/admin/admconfig	管理サーバーの設定
ServerRoot/bin/https/bin/ns-httpd	Sun ONE 管理サーバー
ServerRoot/bin/https/bin/uxwdog	管理サーバーのウォッチドッグ
ServerRoot/bin/slapd/server/ns-ldapagt	LDAP ベースの SNMP サブエージェント
ServerRoot/bin/slapd/server/ns-slapd	Sun ONE Directory Server

ServerRoot/lib ディレクトリには、プラグインを含む製品ライブラリがあります。

表 A-3　*ServerRoot*/lib　の下にあるライブラリ

ライブラリ	内容
ServerRoot/lib/	内部処理およびプラグイン
ServerRoot/lib/libnspr4.so	NSPR、バージョン 4.x

ServerRoot/manual ディレクトリには、コンソールのオンラインヘルプのサポートがあります。

表 A-4 *ServerRoot*/manual の下にあるオンラインヘルプのサポート

ディレクトリ	内容
ServerRoot/manual/	オンラインヘルプのサポート

ServerRoot/plugins ディレクトリには、サーバープラグインのサンプル、プラグイン開発用ヘッダーファイル、および SNMP サポート用プラグインがあります。

表 A-5 *ServerRoot*/plugins の下にあるプラグインのサポート

ディレクトリまたはファイル	内容
ServerRoot/plugins/	サンプル、ヘッダー、SNMP サポート
ServerRoot/plugins/slapd/slapi/examples/	サンプルプラグイン
ServerRoot/plugins/slapd/slapi/include/	プラグインのヘッダーファイル
ServerRoot/plugins/snmp/magt/magt	管理エージェントの設定
ServerRoot/plugins/snmp/mibs/	SNMP MIB
ServerRoot/plugins/snmp/sagt/sagt	SNMP エージェントの設定

ServerRoot/shared/bin ディレクトリには、サーバー管理用ツールがあります。

表 A-6 *ServerRoot*/shared/bin の下にあるツールおよびクライアント

ディレクトリまたはファイル	内容
ServerRoot/shared/bin	次のファイル以外は、内部処理に使用
ServerRoot/shared/bin/admin_ip.pl	IP アドレスの変更
ServerRoot/shared/bin/entrycmp	レプリケーション用エントリの比較
ServerRoot/shared/bin/fildif	フィルタを適用した LDIF のダンプ
ServerRoot/shared/bin/insync	レプリケーション同期のチェック
ServerRoot/shared/bin/ldapcompare	属性値の比較
ServerRoot/shared/bin/ldapdelete	ディレクトリエントリの削除

表 A-6　*ServerRoot*/shared/bin の下にあるツールおよびクライアント (続き)

ディレクトリまたはファイル	内容
ServerRoot/shared/bin/ldapmodify	ディレクトリエントリの変更
ServerRoot/shared/bin/ldapsearch	ディレクトリエントリの検索
ServerRoot/shared/bin/modutil	PKCS #11 モジュールの管理
ServerRoot/shared/bin/uconv	ISO から UTF-8 への変換
ServerRoot/shared/bin/repldisc	レプリケーショントポロジの検出

ServerRoot/shared/config ディレクトリには、ディレクトリエントリへの証明書のマッピング用設定ファイルがあります。

表 A-7　*ServerRoot*/shared/config　の下にある証明書のマッピング用設定ファイル

ディレクトリまたはファイル	内容
ServerRoot/shared/config	次のファイル以外は、内部処理に使用
ServerRoot/shared/config/certmap.conf	エントリへの証明書のマッピング

ServerRoot/setup5 ディレクトリには、サイレントインストールおよびサイレントアンインストール用のサンプルテンプレートがあります。

表 A-8　*ServerRoot*/setup5　の下にあるサイレントインストールおよびサイレントアンインストール用テンプレート

ディレクトリまたはファイル	内容
ServerRoot/setup5	次のファイル以外は、内部処理に使用
ServerRoot/setup5/typical.ins	サイレントインストール用テンプレートファイル
ServerRoot/setup5/uninstall.ins	サイレントアンインストール用テンプレートファイル

サーバーインスタンスのディレクトリ

slapd-*ServerID* ディレクトリには、サーバーインスタンスの *ServerID* に対応するファイルがあります。*ServerRoot*/slapd-*ServerID* ディレクトリ自体には、コマンド行で管理を行うための複数のスクリプトがあります。

表 A-9 サーバーインスタンスのスクリプト

スクリプト	内容
ServerRoot/slapd-*ServerID*/	サーバーインスタンス
ServerRoot/slapd-*ServerID*/bak2db	データベースの復元 (オフライン)
ServerRoot/slapd-*ServerID*/bak2db.pl	データベースの復元 (オンライン)
ServerRoot/slapd-*ServerID*/db2bak	データベースのバックアップ (オフライン)
ServerRoot/slapd-*ServerID*/db2bak.pl	データベースのバックアップ (オンライン)
ServerRoot/slapd-*ServerID*/db2index.pl	インデックスの生成 (オンライン)
ServerRoot/slapd-*ServerID*/db2ldif	データベースの LDIF へのダンプ (オフライン)
ServerRoot/slapd-*ServerID*/db2ldif.pl	データベースの LDIF へのダンプ (オンライン)
ServerRoot/slapd-*ServerID*/getpwenc	暗号化したパスワードの出力
ServerRoot/slapd-*ServerID*/ldif2db	LDIF のインポート (オフライン)
ServerRoot/slapd-*ServerID*/ldif2db.pl	LDIF のインポート (オンライン)
ServerRoot/slapd-*ServerID*/ldif2ldap	LDIF の LDAP 経由のインポート
ServerRoot/slapd-*ServerID*/monitor	監視情報の取得
ServerRoot/slapd-*ServerID*/ns-accountstatus.pl	アカウント状態の確立
ServerRoot/slapd-*ServerID*/ns-activate.pl	エントリの有効化
ServerRoot/slapd-*ServerID*/ns-inactivate.pl	エントリの無効化
ServerRoot/slapd-*ServerID*/restart-slapd	ディレクトリサーバーの再起動

表 A-9　サーバーインスタンスのスクリプト (続き)

スクリプト	内容
ServerRoot/slapd-*ServerID*/restoreconfig	管理サーバーの設定の復元
ServerRoot/slapd-*ServerID*/saveconfig	管理サーバーの設定の保存
ServerRoot/slapd-*ServerID*/start-slapd	ディレクトリサーバーの起動
ServerRoot/slapd-*ServerID*/stop-slapd	ディレクトリサーバーの停止
ServerRoot/slapd-*ServerID*/suffix2instance	サフィックスのバックエンドへのマッピング
ServerRoot/slapd-*ServerID*/vlvindex	仮想リスト表示インデックスの作成

ServerRoot/slapd-*ServerID* のサブディレクトリには、設定、ログ、およびバックアップデータがあります。

表 A-10　サーバーインスタンスのサブディレクトリ

ディレクトリ	内容
ServerRoot/slapd-*ServerID*/	サーバーインスタンス
ServerRoot/slapd-*ServerID*/bak/	ディレクトリデータベースのバックアップ
ServerRoot/slapd-*ServerID*/confbak/	管理サーバーの設定のバックアップ
ServerRoot/slapd-*ServerID*/conf_bk/	ディレクトリサーバーの設定のバックアップ
ServerRoot/slapd-*ServerID*/config/	ディレクトリサーバーの設定
ServerRoot/slapd-*ServerID*/config/schema/	ディレクトリスキーマの設定
ServerRoot/slapd-*ServerID*/db/	ディレクトリデータベース
ServerRoot/slapd-*ServerID*/ldif/	サンプル LDIF ファイル
ServerRoot/slapd-*ServerID*/locks/	実行時プロセスのロック
ServerRoot/slapd-*ServerID*/logs/	サーバーのログファイル
ServerRoot/slapd-*ServerID*/tmp/	実行時の一時ファイル

付属のツールを使用してサーバーインスタンスを管理します。ディレクトリの内容を手動で変更しないでください。

内部処理専用

次のディレクトリの内容は、Directory Server で内部処理に使用されます。これらの内部コンポーネントは、サポートされる公開インタフェース製品には属しません。

- *ServerRoot*/adminacl/
- *ServerRoot*/admin-serv/
- *ServerRoot*/admserv
- *ServerRoot*/alias/
- *ServerRoot*/dist/
- *ServerRoot*/httpacl/
- *ServerRoot*/include/
- *ServerRoot*/install/
- *ServerRoot*/java/
- *ServerRoot*/userdb/

これらのディレクトリまたは内容を変更しないでください。

Sun Crypto Accelerator ボードの使用

この付録では、Directory Server で Sun Crypto Accelerator ボードを使用して、証明書ベースの認証と Secure Sockets Layer (SSL) プロトコルを使用した接続のパフォーマンスを向上させる手順について説明します。

始める前に

Sun Crypto Accelerator ボードを使用して SSL 接続のパフォーマンスを向上させる前に、表 B-1 に示した項目を完了しておく必要があります。

表 B-1 ボードを使用するための前提条件

前提条件	内容
ボードのインストール	ハードウェア、ドライバ、パッチ、および管理ユーティリティをホストにインストールするときは、ボードに付属の製品マニュアルを参照
Directory Server インストール	手順は、第 1 章「Sun ONE Directory Server のインストール」を参照
サーバー証明書 (PKCS#12 形式)	Directory Server のサーバー証明書を `.p12` ファイルで取得する
CA 証明書 (PEM 形式)	証明書発行局 (CA) の CA 証明書を PEM (Privacy Enhanced Mail) 形式ファイルで取得する

SSL プロトコルと SSL 証明書、および Sun ONE サーバーコンソール管理をサポートする Sun ONE サーバーでプロトコルを使用する方法については、『Sun ONE Server Console Server Management Guide』を参照してください。

トークンの作成

Directory Server ではトークンとパスワードを使用して、アクセラレータボード上の適切な暗号化キーマテリアルにアクセスします。トークンは *user@realm* という形式になります。ここで *user* は、アクセラレータボードのユーザーで、暗号化キーマテリアルの所有者です。*realm* は、アクセラレータボードのレルムで、ユーザーとそのキーマテリアルの論理パーティションです。アクセラレータボードのユーザーは、システムのユーザーアカウントとは何も関係を持つ必要はありません。このユーザーはボード専用です。ユーザーとレルムについての詳細は、アクセラレータボードの製品マニュアルを参照してください。

トークンのユーザーとレルムは、ボードで使用するように用意された `secadm` (1M) ユーティリティを使用して作成できます。また、アクセラレータボードでは、複数のアプリケーションのトークンを管理するために、複数のスロットを作成することもできます。ここでは、パフォーマンス上の理由から、ホストを Directory Server 専用にし、1 スロットだけを使用します。これはデフォルトの設定です。複数のソフトウェアアプリケーションでボードを使用する詳細は、アクセラレータボードの製品マニュアルを参照してください。

デフォルトのスロットにアクセスするためのトークンのユーザーとレルムを作成するには、次の手順を実行します。

1. `secadm` ユーティリティを起動します。

   ```
   $ CryptoPath/bin/secadm
   ```

   デフォルトの *CryptoPath* は `/opt/SUNWconn/crypto` です。

2. トークンのレルムを作成します。

   ```
   secadm> create realm=dsrealm
   System Administrator Login Required
   Login:super-user
   Password:
   Realm dsrealm created successfully.
   ```

3. ユーザーを作成するレルムを設定します。

   ```
   secadm> set realm=dsrealm
   secadm{dsrealm}> su
   System Administrator Login Required
   Login:super-user
   Password:
   secadm{root@dsrealm}#
   ```

4. デフォルトのスロットを使用するユーザー `nobody` を作成し、SSL を設定した Directory Server を再起動するときに使用するパスワードを指定します。

```
secadm{root@dsrealm}# create user=nobody
Initial password: password
Confirm password: password
User nobody created successfully.
secadm{root@dsrealm}# exit
```

この時点で、トークン nobody@dsrealm のユーザーとレルムが作成され、Directory Server を再起動するときに使用するパスワードを指定しています。

ボード用バインドの生成

アクセラレータボード用のバインドは、生成する外部セキュリティモジュールの形式をとるため、Directory Server はボードにバインドできます。複数の SSL アルゴリズムをサポートした、外部セキュリティモジュールと Directory Server 証明書データベースとの間のバインドを生成するには、次の手順を実行します。

1. modutil を使用する前に LD_LIBRARY_PATH を設定します。

   ```
   $ set LD_LIBRARY_PATH=ServerRoot/lib ; export LD_LIBRARY_PATH
   ```

2. セキュリティモジュールデータベースが存在しない場合は作成します。

   ```
   $ cd ServerRoot/shared/bin
   $ ./modutil -create -dbdir ../../alias -dbprefix "slapd-serverID"
   ```

3. 外部セキュリティモジュールをセキュリティモジュールデータベースに追加します。

   ```
   $ ./modutil -add "Crypto Mod" -dbdir ../../alias -nocertdb ¥
   -libfile CryptoPath/lib/libpkcs11.so ¥
   -mechanisms "RSA:DSA:RC4:DES" -dbprefix "slapd-serverID"
   ```

   デフォルトの *CryptoPath* は /opt/SUNWconn/crypto です。

4. セキュリティモジュールを一覧にして、追加に成功したことを確認します。

   ```
   $ ./modutil -list -dbdir ../../alias -dbprefix "slapd-serverID"
   ```

   手順 3 で追加した Crypto Mod のエントリが表示されます。

5. 外部セキュリティモジュールを RSA、DSA、RC4、および DES のデフォルトにします。

   ```
   $ ./modutil -default "Crypto Mod" -dbdir ../../alias ¥
   -mechanisms "RSA:DSA:RC4:DES" -dbprefix "slapd-serverID"
   ```

   デフォルトのセキュリティモジュールの変更が成功します。

この時点で、アクセラレータボード用バインドを生成し、証明書をインポートできるようになりました。

証明書のインポート

SSL を設定する前に、173 ページの表 B-1 で説明したようにして取得したサーバー証明書と CA 証明書をインポートする必要があります。証明書をインポートするには、次の手順を実行します。

1. サーバー証明書 .p12 ファイルをインポートします。

```
$ cd ServerRoot/shared/bin
$ ./pk12util -i ServerCert.p12 -d ../../alias -P "slapd-serverID" ¥
-h "nobody@dsrealm"
Enter Password or Pin for "nobody@dsrealm":password
Enter Password for PKCS12 file:password
```

2. CA 証明書をインポートします。

```
$ ./certutil -A -n "Crypto CA Cert" -t CT -i CACert.txt ¥
-d ../../alias -P "slapd-serverID" -h "nobody@dsrealm"
```

3. トークンに関連付けられた証明書を一覧にして、インポートが成功したことを確認します。

```
$ ./certutil -L -d ../../alias -P "slapd-serverID" ¥
-h "nobody@dsrealm"
```

手順 1 と手順 2 で追加した証明書のエントリが表示されます。

この時点で、証明書がインポートされ、Directory Server が SSL 接続を待機するように設定できるようになります。

SSL の設定

作成したトークンとパスワード、外部セキュリティモジュールと Directory Server 証明書データベースの間に生成したバインド、およびインポートした証明書を使用すると、Directory Server をセキュリティ保護されたモードで起動できます。SSL を設定して Directory Server をセキュリティ保護されたモードで再起動するには、次の手順を実行します。

1. SSL 関連の Directory Server 設定エントリを変更するための修正ファイル ssl.ldif を作成します。

コード例 B-1 ボードを使用した SSL をアクティブにするための変更 (`ssl.ldif`)

```
dn:cn=RSA,cn=encryption,cn=config
changetype:add
objectclass:top
objectclass:nsEncryptionModule
cn:RSA
nsSSLToken:nobody@dsrealm
nsSSLPersonalitySSL:ServerCertNickname[1]
nsSSLActivation:on

dn:cn=encryption,cn=config
changetype:modify
replace:nsSSL3
nsSSL3:on
-
replace:nsSSLClientAuth
nsSSLClientAuth:allowed
-
replace:nsSSL3Ciphers
nsSSL3Ciphers:-rsa_null_md5,+rsa_rc4_128_md5,+rsa_rc4_40_md5,
 +rsa_rc2_40_md5,+rsa_des_sha,+rsa_fips_des_sha,+rsa_3des_sha,
 +rsa_fips_3des_sha,+fortezza,+fortezza_rc4_128_sha,
 +fortezza_null,+tls_rsa_export1024_with_rc4_56_sha,
 +tls_rsa_export1024_with_rc4_56_sha,
 +tls_rsa_export1024_with_des_cbc_sha
-
replace:nsCertfile
nsCertfile:alias/slapd-serverID-cert7.db
-
replace:nsKeyFile
nsKeyFile:alias/slapd-serverID-key3.db

dn:cn=config
changetype:modify
replace:nsslapd-secureport
nsslapd-secureport:port
-
replace:nsslapd-security
nsslapd-security:on
```

1. このニックネームは Directory Server の証明書に含まれている

ここで `nsslapd-secureport` の値である *port* は、Directory Server がセキュリティ保護されたモードで起動した後、SSL 接続を待機するためのポートです。

2. Directory Server 設定を変更するために、変更内容を適用します。

```
$ ldapmodify -p currPort -D "cn=directory manager" -w password -f ¥
ssl.ldif
```

ここで *currPort* は、Directory Server がクライアント要求を待機している現在のポートの番号です。

3. Directory Server をセキュリティ保護されたモードで再起動します。

```
$ ServerRoot/slapd-serverID/restart-slapd
Enter PIN for nobody@dsrealm:password
```

ここで *password* は nobody のユーザーパスワードで、トークン nobody@dsrealm を作成したときに指定しています。

この時点で、Directory Server は指定したポートで SSL トラフィックを待機します。そのポートを使用して SSL によって Directory Server にアクセスするために、Sun ONE 管理サーバーとクライアントアプリケーションを設定することができます。詳細は、『Sun ONE Directory Server 管理ガイド』を参照してください。

付録 C

Sun Cluster HA for Directory Server のインストール

この付録では、Sun Cluster HA for Directory Server データサービスと、関連する管理サーバーデータサービスの両方のインストールと設定について説明します。Sun Cluster のインストール手順や主要な概念については、Sun Cluster 3.0 の製品マニュアルを参照してください。

データサービスは、フェイルオーバーサービスとして設定する必要があります。

始める前に

インストールと設定を行う前に、『Sun Cluster 3.0 Release Notes』のワークシートをチェックリストとして使用して、この節の内容を確認します。

インストールに先立って、次の質問を検討します。

- 複数の Directory Server インスタンスを同じノードで実行するか

  その場合、`cn=config` の `nsslapd-listenhost` に適切なネットワークリソース (`dirserv.example.com` などの論理ホスト名) を各インスタンスの IP アドレスとして設定することができます。Directory Server のデフォルトの動作では、すべてのネットワークインタフェースで待機します。

- 複数のデータサービスを Sun Cluster 設定で実行するか

  複数のデータサービスを任意の順番で設定できますが、例外が 1 つあります。Sun Cluster HA for DNS を使用するときは、Sun Cluster HA for Directory Server の設定前に Sun Cluster HA for DNS を設定する必要があります。

表 C-1 に Sun Cluster HA for Directory Server のインストールと設定のプロセスを示します。

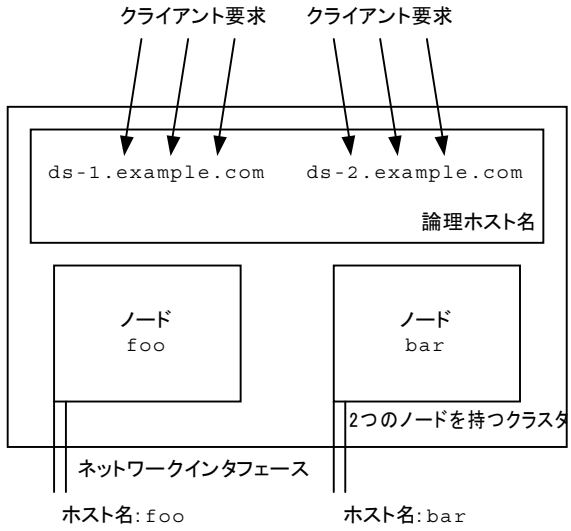
表 C-1　インストールと設定のプロセス

作業	必要な知識
181 ページの「ネットワークリソースの設定」	データサービスを管理するクラスタノードの名前 Directory Server にアクセスするクライアントで使用される論理ホスト名。`ds1.example.com`、`ds2.example.com` など 論理ホスト名の設定手順は、Sun Cluster 3.0 の製品マニュアルを参照
183 ページの「サーバーのインストール」	グローバルファイルシステム上の *ServerRoot* の位置で Directory Server のインストール場所。`/global/ds` など インストールの詳細は、21 ページの表 1-2 に示す
184 ページの「データサービスパッケージのインストール」	`SUNWdsha` パッケージと `SUNWasha` パッケージにはデータサービス用の管理インタフェースが備わっている。そのため Directory Server と管理サーバーをクラスタ内のほかのデータサービスと同じツールで管理することが可能
185 ページの「サーバーの設定」	Directory Server データサービス `SUNW.dsldap` および管理サーバーデータサービス `SUNW.mps` の各リソースタイプ名 データサービスを管理するクラスタノードの名前 Directory Server と管理サーバーにアクセスするクライアントで使用される論理ホスト名 グローバルファイルシステム上の *ServerRoot* の位置で Directory Server のインストール場所 Directory Server でクライアント要求を待機するポート 管理サーバーでクライアント要求を待機するポート 181 ページの「ネットワークリソースの設定」で定義されたリソースグループの名前
187 ページの「拡張プロパティの設定」	(詳細は、この節を参照)

ネットワークリソースの設定

Sun Cluster ソフトウェアでは、個別のネットワークインタフェースに対してノード名ともホスト名とも異なる論理ホスト名が管理されます。図 C-1 を見ると、2 つのノードのクラスタで管理される論理ホスト名がどちらかのノードに永続的に関連付けられているわけではないことがわかります。

図 C-1 2 つのノードのクラスタ



Sun Cluster HA for Directory Server データサービスをインストールするときは、論理ホスト名インタフェースで待機するように Directory Server と管理サーバーを設定します。そのため、クラスタ内の特定のノードと結びつかず、Sun Cluster ソフトウェアでフェイルオーバーを管理することができます。図 C-1 で、ノードの名前は `foo` および `bar` です。ただし、インストール中に使用する論理ホスト名は 図 C-1 で示されるように、`ds-1.example.com` と `ds-2.example.com` であり、`foo` や `bar` ではありません。使用される論理ホスト名は、完全修飾ドメイン名です。

これらの主要な概念についての詳細と論理ホスト名の設定手順については、Sun Cluster 3.0 の製品マニュアルを参照してください。

論理ホスト名の設定が済んだら、次の手順を実行します。

1. クラスタ内の特定のノード上でスーパーユーザーになります。

2. 使用するすべてのネットワークアドレスがネームサービスデータベースに追加済みであることを確認します。

   ネームサービスの検索中に失敗しないように、すべての完全修飾ドメイン名、完全修飾論理ホスト名、および共有 IP アドレスが、各クラスタノードの `/etc/hosts` ファイルに記載されていることも確認します。また、ほかのネームサービスにアクセスしようとする前にローカルファイルをチェックするように、各クラスタノードで `/etc/nsswitch.conf` のネームサービスマッピングを設定します。

3. ネットワークおよびアプリケーションのリソースを保持するフェイルオーバーリソースグループを作成します。たとえば、次のようにします。

   ```
   # scrgadm -a -g resource-group [-h node-list]
   ```

   ここで *resource-group* はグループの名前です。

   省略可能な *node-list* は、クラスタの潜在的マスタノードを指定する、物理ノードの名前または ID のカンマ区切りのリストです。ノード名の順番は、フェイルオーバー中に重要であると考えられるノードの順番にします。クラスタ内の全ノードでマスタになる可能性がある場合は、*node-list* を指定する必要はありません。

4. 論理ホスト名リソースをリソースグループに追加します。

   ```
   # scrgadm -a -L -g resource-group -l logical-host-names [-n netif-list]
   ```

   ここで *logical-host-names* は、論理ホスト名として使用される完全修飾ドメイン名のカンマ区切りのリストです。Directory Server インスタンスごとに論理ホスト名を 1 つ使用します。

   省略可能な *netif-list* は、各ノードの NAFO グループを指定するカンマ区切りのリストです。このオプションを指定しない場合、`scrgadm`(1M) では、論理ホスト名ごとに使用されるサブネット上のネットワークアダプタを検出しようとします。この論理ホスト名は、手順 3 で指定した *node-list* 内の各ノードで指定されます。

5. 手順 4 で論理ホスト名として指定したすべての完全修飾ドメイン名が、ネームサービスデータベースに追加されていることを確認します。

6. リソースグループを有効にし、オンラインにします。

   ```
   # scswitch -Z -g resource-group
   ```

リソースグループをオンラインにすると、サーバーをインストールできます。

サーバーのインストール

Sun Cluster HA for Directory Server では Directory Server と管理サーバーの両方が、Sun Cluster の制御下で実行されています。これは、インストール中に完全修飾ドメイン名を物理ノードに指定する代わりに、別のノードにフェイルオーバー可能な完全修飾論理ホスト名を指定するということです。

まず、ディレクトリクライアントアプリケーションが使用する論理ホスト名に対してオンラインになっているノード上でインストールを実行します。続いて、Directory Server データサービスを管理するその他のクラスタノードすべてに対して、それと同じ処理を繰り返します。

アクティブノード上でのインストール

ディレクトリクライアントアプリケーションが使用する論理ホスト名に対してオンラインになっているクラスタノード上で、次の手順を実行します。

1. Directory Server と管理サーバー両方の Solaris パッケージをインストールします。手順は、24 ページの「Solaris パッケージのインストール」を参照してください。
2. Directory Server を設定します。手順は、29 ページの「Directory Server の設定」を参照してください。

   この手順を実行するときは、次のようにします。

   - Directory Server インスタンスをグローバルクラスタファイルシステム上に配置します。
   - ノード名ではなく論理ホスト名を使用します。
3. 管理サーバーを設定します。手順と、Directory Server を設定するために論理ホスト名で同じものを使用する方法については、29 ページの「管理サーバーの設定」を参照してください。
4. Directory Server をセキュリティ保護されたモードのみで使用するときは、*ServerRoot*/slapd-*serverID*/keypass という名前の空のファイルを作成します。このファイルは、Directory Server インスタンスがセキュリティ保護されたモードで実行されるクラスタを示しています。

   また、*ServerRoot*/alias/slapd-*serverID*-pin.txt ファイルを作成します。このファイルには、インスタンスを自動的にセキュリティ保護されたモードで起動するのに必要なパスワードを記述します。これにより、クラスタがユーザーの介入なしにデータサービスを再起動できるようになります。

その他のノード上でのインストール

Directory Server データサービスを管理するノードごとに、次の手順を実行します。

1. Directory Server と管理サーバー両方の Solaris パッケージをインストールします。手順は、24 ページの「Solaris パッケージのインストール」を参照してください。
2. 183 ページの「アクティブノード上でのインストール」と同じ設定を使って Directory Server を設定します。
3. 183 ページの「アクティブノード上でのインストール」と同じ設定を使って管理サーバーを設定します。
4. 最初のノード上の *ServerRoot*/alias/slapd-*serverID*-pin.txt を *ServerRoot*/alias/ にコピーします。

注 グローバルファイルシステムに配置したファイルを削除したり、位置を変更したりしないでください。

データサービスパッケージのインストール

データサービスパッケージ SUNWdsha および SUNWasha には、クラスタ内のデータサービスとしてサーバーを管理するための管理インタフェースが用意されています。

- Directory Server データサービスをサポートする各クラスタノード上で、pkgadd(1M) ユーティリティを使ってデータサービスパッケージをインストールします。

  ```
  # pkgadd -d dirContainingPackages SUNWasha SUNWdsha
  ```

サーバーの設定

Directory Server が使用している論理ホスト名に対してオンラインになっているクラスタノード上でのみ、次の手順を実行します。

1. スーパーユーザーになります。
2. Directory Server および管理サーバーを停止します。

   ```
   # /usr/sbin/directoryserver stop
   # /usr/sbin/mpsadmserver stop
   ```

3. 各データサービスのリソースタイプを登録します。

   ```
   # scrgadm -a -t SUNW.dsldap -f /etc/ds/v5.2/cluster/SUNW.dsldap
   # scrgadm -a -t SUNW.mps -f /etc/mps/admin/v5.2/cluster/SUNW.mps
   ```

   ここで `SUNW.dsldap` および `SUNW.mps` はデータサービスのリソースタイプ名で事前に定義されています。`/etc/ds/v5.2/cluster/SUNW.dsldap` および `/etc/mps/admin/v5.2/cluster/SUNW.mps` は、データサービスを定義しているファイルです。

4. 181 ページの「ネットワークリソースの設定」で作成したフェイルオーバーリソースグループにサーバーを追加します。

   ```
   # scrgadm -a -j resource-name-ds -g resource-group -t SUNW.dsldap ¥
   -y Network_resources_used=logical-host-name ¥
   -y Port_list=port-number/tcp ¥
   -x Confdir_list=ServerRoot/slapd-serverID

   # scrgadm -a -j resource-name-as -g resource-group -t SUNW.mps ¥
   -y Network_resources_used=logical-host-name ¥
   -y Port_list=port-number/tcp ¥
   -x Confdir_list=ServerRoot
   ```

   ここで新しい *resource-name-ds* は Directory Server インスタンスを指定し、新しい *resource-name-as* は管理サーバーインスタンスを指定します。

   resource-group パラメータは、181 ページの「ネットワークリソースの設定」で指定したグループの名前です。

   logical-host-name は、現在の Directory Server インスタンスに使用される論理ホスト名です。

   port-number は、183 ページの「サーバーのインストール」で説明したように、サーバーインスタンスがクライアント要求を待機しているポート番号です。各コマンドの `Port_list` パラメータには 1 エントリしかありません。

   ServerRoot および *ServerRoot*`/slapd-`*serverID* は、183 ページの「サーバーのインストール」で指定しているパスです。各コマンドの `Confdir_list` パラメータには 1 エントリしかありません。

5. サーバーリソースを有効にし、監視します。

```
# scswitch -e -j resource-name-ds
# scswitch -e -j resource-name-as
```

ここで *resource-name-ds* および *resource-name-as* は、手順 4 でサーバーを指定するために使用した名前です。

注 サーバーを設定した後は、db2bak、db2ldif、back2db、および ldif2db などのバックアップおよび復元コマンドを、アクティブでないクラスタノード上で実行しないでください。代わりに、アクティブノード上で、すべてのバックアップおよび復元処理を実行します。

6. 190 ページの「HA ストレージとデータサービスの同期」で説明している手順の実行を検討してください。フェイルオーバー時のパフォーマンスの向上が図れます。

登録と設定の例

コード例 C-1 に、181 ページの図 C-1 で図示したクラスタにデータサービスを登録し、設定する方法を示します。

コード例 C-1 データサービスの登録と設定

```
(オンラインになっているノード上でフェイルオーバーリソースグループを作成する)
# scrgadm -a -g ds-resource-group-1 -h foo,bar

(論理ホスト名リソースをリソースグループに追加する)
# scrgadm -a -L -g ds-resource-group-1 -l ds-1.example.com

(リソースグループをオンラインにする)
# scswitch -Z -g ds-resource-group-1

(クラスタ内の各ノード上でパッケージをインストールする)

(オンラインになっているノード上でサーバーを停止する)
# /usr/sbin/directoryserver stop
# /usr/sbin/mpsadminserver stop

(SUNW.dsldap および SUNW.mps リソースタイプを登録する)
# scrgadm -a -t SUNW.dsldap -f /etc/ds/v5.2/cluster/SUNW.dsldap
# scrgadm -a -t SUNW.mps -f /etc/mps/admin/v5.2/cluster/SUNW.mps

(サーバーのリソースを作成し、リソースグループに追加する)
# scrgadm -a -j ds-1 -g ds-resource-group-1 ¥
-t SUNW.dsldap -y Network_resources_used=ds-1.example.com ¥
-y Port_list=389/tcp ¥
-x Confdir_list=/global/ds/slapd-ds-1
```

コード例 C-1 データサービスの登録と設定 (続き)

```
# scrgadm -a -j as-1 -g ds-resource-group-1 ¥
-t SUNW.mps -y Network_resources_used=ds-1.example.com ¥
-y Port_list=5201/tcp ¥
-x Confdir_list=/global/ds

( アプリケーションリソースを有効にする )
# scswitch -e -j ds-1
# scswitch -e -j as-1
```

拡張プロパティの設定

拡張プロパティを使用すると、クラスタソフトウェアがアプリケーションソフトウェアを処理する方法を設定できます。たとえば、データサービスによるフェイルオーバー実行のタイミングをクラスタが決定する方法を調整できます。

設定可能なプロパティ

通常、リソース拡張プロパティを設定するには、Sun Management Center の Cluster Module を使用するか、`scrgadm` ユーティリティを使用します。表 C-2 に記載された拡張プロパティを変更するには、`scrgadm` ユーティリティを `-x` *parameter=value* オプションを指定して実行します。

表 C-2 `SUNW.dsldap` リソース拡張プロパティ

プロパティ	内容	デフォルト	範囲
`Monitor_retry_count`	`Monitor_retry_interval` プロパティの値で指定されたタイムウィンドウ中に、プロセスモニター装置 (PMF) が障害モニターを再起動する回数を示す整数値	4 回	-1 ～ 2,147,483,641 回 -1 は無期限のやり直しを意味する
`Monitor_retry_interval`	障害モニターのエラーがカウントされた時間 (分) を示す整数値 障害モニターがエラーになった回数が、この期間内に `Monitor_retry_count` で指定した値を超えた場合、PMF は障害モニターを再起動できない	2 分	-1 ～ 2,147,483,641 分 -1 は無限のやり直し期間を示す

表 C-2　SUNW.dsldap リソース拡張プロパティ (続き)

プロパティ	内容	デフォルト	範囲
Probe_timeout	Directory Server インスタンスを検査するために障害モニターで使用するタイムアウト値 (秒) を示す整数値	30 秒	0 ～ 2,147,483,641 秒

Sun Cluster プロパティの詳細は、Sun Cluster 3.0 の製品マニュアルを参照してください。

障害モニターの動作方法

クラスタソフトウェアは、データサービスが正常に動作しているかどうかを、障害モニターを使用して判断します。障害モニターは、データサービスを検査し、その検査結果に基づいて、サービスが正常に動作しているかどうか、再起動すべきかどうかを判断します。

表 C-3 障害モニターによる検査結果の解釈方法

Directory Server の実行モード	使用される検査	アルゴリズム
通常モード	`ldapsearch`	1. 検索を試みる 2. 検索を実行した結果、 • `LDAP_SUCCESS` が返された場合、そのサービスは正常に動作していると見なされる • LDAP エラーが発生した場合、そのサービスは再起動する必要がある • タイムアウト以外の問題が発生した場合、障害モニターは、`Monitor_retry_count` と `Monitor_retry_interval` に基づいて検査を再度実行する • `Probe_timeout` 期間を超えた場合、障害モニターは、`Monitor_retry_count` と `Monitor_retry_interval` に基づいて検査を再実行する タイムアウトの潜在的な原因としては、システム、ネットワーク、または Directory Server インスタンスにおける高負荷が考えられる。また、`Probe_timeout` の設定値が、監視対象の Directory Server インスタンスの数に比べて低すぎる可能性もある
セキュリティ保護されたモード (SSL)	TCP 接続	1. 接続を試みる 2. 接続処理を実行した結果、 • 接続が成功した場合、そのサービスは正常に動作していると見なされる • 接続が失敗した場合、そのサービスは再起動する必要がある • `Probe_timeout` を超えた場合、そのサービスは再起動する必要がある

障害モニターは、185 ページの「サーバーの設定」で設定された IP アドレスとポート番号を使って、検査処理を実行します。Directory Server が 2 つのポート (SSL トラフィック用のポートと通常のトラフィック用のポート) で待機するように設定されている場合、障害モニターは両ポートを検査する際に、TCP 接続を使用して、セキュリティ保護されたモードポート用の障害監視アルゴリズムに従います。

HA ストレージとデータサービスの同期

SUNW.HAStorage リソースタイプでは、HA ストレージとデータサービスのアクションが同期されており、Directory Server などのディスクを集中使用するデータサービスがフェイルオーバーに対処するときでも高パフォーマンスを提供することができます。

Directory Server データサービスと HA ストレージを同期させるには、データサービスが使用する論理ホスト名に対してオンラインになっているノード上で、次の手順を実行します。

1. HA ストレージのリソースタイプを登録します。

   ```
   # scrgadm -a -t SUNW.HAStorage
   ```

2. ストレージリソースが同期されたままになるように設定します。

   ```
   # scrgadm -a -j HAStorage-resource-name -g HAStorage-resource-group ¥
   -t SUNW.HAStorage -x ServicePaths=volume-mount-point ¥
   -x AffinityOn=True
   ```

   ここで、*volume-mount-point* は、Directory Server がデータを格納するディスクボリュームを表します。

3. ストレージリソースを有効にし、監視します。

   ```
   # scswitch -e -j HAStorage-resource-name
   ```

4. 既存の Directory Server リソースに依存関係を追加します。

   ```
   # scrgadm -c -j resource-name-ds ¥
   -y Resource_Dependencies=HAStorage-resource-name
   ```

バックグラウンド情報は、SUNW.HAStorage(5) を参照してください。また新しいリソース用に SUNW.HAStorage リソースタイプを設定する手順についての詳細は、Sun Cluster 3.0 製品マニュアルを参照してください。

追加の Directory Server インスタンスの作成

次の手順を実行します。

1. Sun ONE サーバーコンソールを使用して、追加の Directory Server インスタンスを作成します。

   手順は、『Sun ONE Server Console Server Management Guide』を参照してください。

2. データサービスが使用する論理ホスト名に対してオンラインになっているノード上で、新しく作成した Directory Server インスタンスを停止します。

   ```
   # /usr/sbin/directoryserver -server serverID stop
   ```

3. 181 ページの「ネットワークリソースの設定」で作成したフェイルオーバーリソースグループに Directory Server を追加します。

   ```
   # scrgadm -a -j resource-name-ds -g resource-group -t SUNW.dsldap ¥
   -y Network_resources_used=logical-host-name ¥
   -y Port_list=port-number/tcp ¥
   -x Confdir_list=ServerRoot/slapd-serverID
   ```

   ここで、新しい *resource-name-ds* は Directory Server インスタンスを指します。

   resource-group パラメータは、181 ページの「ネットワークリソースの設定」で指定したグループの名前です。

   logical-host-name は、インスタンスに使用される論理ホスト名です。

   port-number は、インスタンスがクライアント要求を待機しているポート番号で、183 ページの「サーバーのインストール」で指定しています。`Port_list` パラメータには 1 エントリしかありません。

   ServerRoot および *ServerRoot*`/slapd-`*serverID* は、183 ページの「サーバーのインストール」で指定しているパスです。`Confdir_list` パラメータには 1 エントリしかありません。

4. サーバーリソースを有効にし、監視します。

   ```
   # scswitch -e -j resource-name-ds
   ```

   ここで *resource-name-ds* は、手順 3 で Directory Server を指定するために使用した名前です。

アンインストール

Sun Cluster HA for Directory Server と、それに関連する管理サーバーをクラスタから削除するには、次の手順を実行します。

1. サーバーインスタンスを停止します。

```
# scswitch -n -j resource-name-ds
# scswitch -n -j resource-name-as
```

2. リソースを削除します。

```
# scrgadm -r -j resource-name-ds
# scrgadm -r -j resource-name-as
```

3. リソースタイプをクラスタデータベースから削除します。

```
# scrgadm -r -t SUNW.dsldap
# scrgadm -r -t SUNW.mps
```

4. サーバー設定を削除します。

```
# /usr/sbin/mpsadmserver unconfigure
# /usr/sbin/directoryserver unconfigure
```

5. pkgrm(1M) ユーティリティを使用して、SUNWdsha や SUNWasha などのインストールしたパッケージを各ノードから削除します。

索引

S

あ

い

ろ