Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)

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

Part No: 816–6235–10
2002 年 9 月

原典: *System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*

Part No: 816-4856-10

Revision A

Adobe PostScript

020808@2851

目次

図目次

はじめに

『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス*: DNS*、*NIS*、*LDAP* 編*)*』では、Solaris 9 オペレーティング環境のネームサービスおよびディレクトリサービスである DNS、NIS、LDAP の設定、構成、管理について説明します。このマニュアルは、Solaris ™ 9 リリースシステム管理マニュアルセットの一部です。

対象読者

このマニュアルは、経験豊富なシステム管理者およびネットワーク管理者を対象としています。

このマニュアルは、 Solaris™ のネームサービスおよびディレクトリサービスに関連したネットワークの概念を紹介するものであり、Solaris オペレーティング環境のネットワークの基礎や管理ツールについては説明しません。

内容の紹介

このマニュアルは、各ネームサービスに対応して複数の部分に分けられています。

パート I:ネームサービスおよびディレクトリサービスの紹介

パート II: DNS の設定と管理

パート III: NIS の設定と管理

パート IV: iPlanet Directory Server 5.1 の構成

関連マニュアル

- 『*DNS and Bind*』Cricket Liu および Paul Albitz 共著 (O′ Reilly, 1992)、浅羽登志也/上水流由香監訳、アスキー出版社、1995 年
- 『*Understanding and Deploying LDAP Directory Services* 』Timothy A. Howes, Ph.D および Mark C. Smith 著

  LDAP ネームサービスに関する総合的な解説を提供することに加え、この本には、大規模な大学、大規模な多国籍企業、およびエクストラネットを備えた企業への LDAP 導入に関する有用なケーススタディが含まれています。
- 『*iPlanet Directory Server 5.1* 導入ガイド』。このマニュアルは、Solaris 9 Documentation CD に含まれています。

  このドキュメントでは、基本的なディレクトリ計画 (ディレクトリ設計、スキーマ設計、ディレクトリツリー、トポロジ、複製、およびセキュリティを含む) が説明されています。最後の章では、簡単な配備に加え、世界中に存在する何百万ものユーザーをサポートする複雑な配備を行う際の参考になる、サンプルの開発シナリオを紹介しています。
- 『*iPlanet Directory Server 5.1* 管理者ガイド』

Sun のオンラインマニュアル

docs.sun.com[SM] では、Sun が提供しているオンラインマニュアルを参照することができます。マニュアルのタイトルや特定の主題などをキーワードとして、検索を行うこともできます。URL は、`http://docs.sun.com` です。

表記上の規則

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

表 **P–1** 表記上の規則

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

コード例は次のように表示されます。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bourne シェルおよび Korn シェル

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

- Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

パート I

ネームサービスとディレクトリサービスについて

ここでは、Solaris オペレーティング環境用のネームサービスとディレクトリサービスについて概要を説明します。また、異なるサービスと組み合わせて使用する際に利用する `nsswitch.conf` ファイルについても説明します。

第 1 章

ネームサービスとディレクトリサービス (概要)

この章では、名前空間とネームサービスの概要および機能について説明します。また、Solaris のネームサービスであるDNS 、NIS 、NIS+、および LDAP ネームサービスについても 簡潔に説明します。NIS+ および FNS の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

ネームサービスとは

「ネームサービス」は、ユーザー、マシン、およびアプリケーションがネットワーク経由で通信するための情報を集中管理することを可能にします。格納される情報には、以下が含まれます。

- マシン (ホスト) 名とアドレス
- ユーザ名
- パスワード
- アクセス権
- グループのメンバーシップ、プリンタなど

集中化されたネームサービスが存在しない場合、マシンごとに、これらの情報のコピーを管理する必要があります。ネームサービス情報はファイルまたはマップ、データベーステーブルの形で格納できます。これらのデータを 1 カ所で管理すれば、大規模なネットワークの管理がより簡単になります。

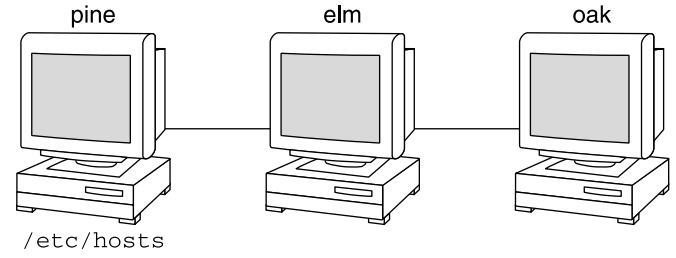
ネームサービスは、どのようなコンピュータネットワークにも欠かせないものです。ネームサービスは、他の機能に加え、次の機能を提供します。

- 名前とオブジェクトを対応付ける (バインドする)
- オブジェクトの名前を解決する
- バインドを解除する
- 名前を一覧表示する

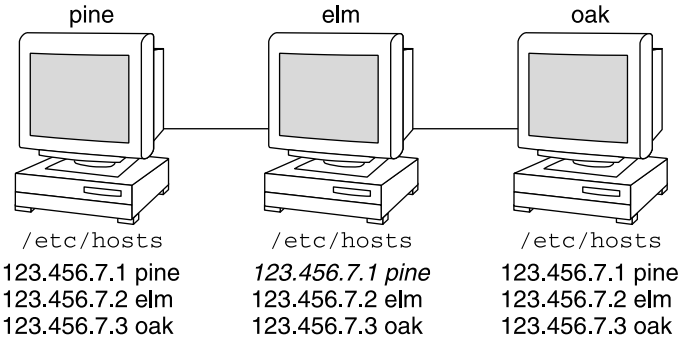
- 名前を変更する

ネットワーク情報サービスを使用すると、数値アドレスの代わりに一般的な名前でマシンを識別できます。これにより、ユーザーは 192.168.00.00 のような扱いにくい数値アドレスを記憶して入力する必要がなくなるため、通信がより簡単になります。

たとえば、pine、elm、oak という 3 台のマシンで構成されるネットワークを考えてみましょう。pine が elm または oak にメッセージを送信するには、それら 2 台のネットワークアドレスを知る必要があります。そのため pine は、自分自身を含めたネットワーク内のすべてのマシンのネットワークアドレスを格納する /etc/hosts ファイルまたは /etc/inet/ipnodes ファイルを保持しています。



同様に、elm や oak が pine と通信したり、お互いに通信するためには、上記のようなファイルを保持している必要があります。



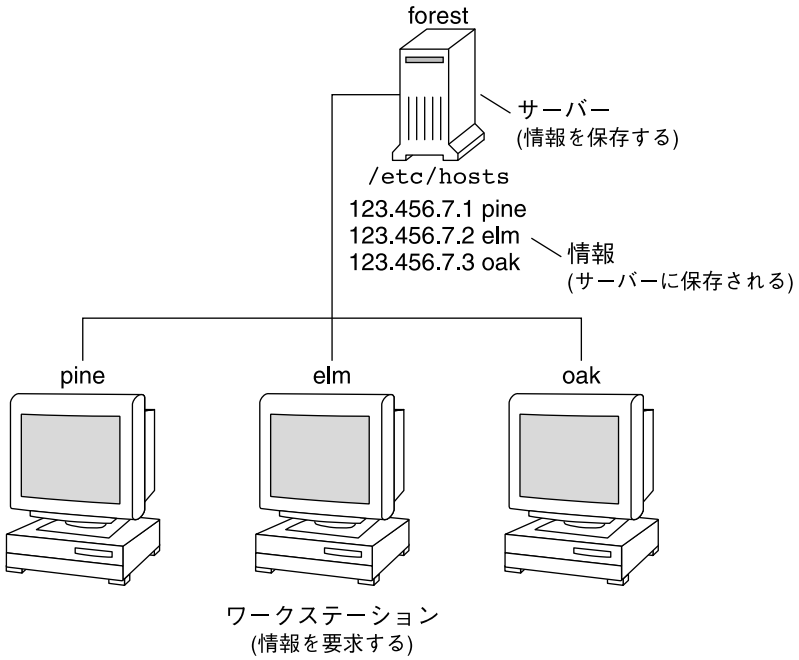
マシンは、アドレスに加え、セキュリティ情報、メールデータ、Ethernet インタフェースについての情報、ネットワークサービスについての情報、ネットワークの使用を許可されたユーザーグループについての情報、ネットワーク上で提供されるサー

ビスについての情報なども格納します。ネットワークによって提供されるサービスが増えるにつれて、格納する情報の種類も増えていきます。その結果、各マシンで /etc/hosts や /etc/inet/ipnodes のようなファイルのセット全部を保持する必要がでてくる可能性があります。

この情報が変更されるたびに、管理者はネットワーク内の各マシン上の情報を最新のものにしなければなりません。小規模なネットワークでは、これは時間のかかる面倒な作業です。中規模または大規模なネットワークでは、これは時間がかかるだけではなく、ほとんど管理不可能な作業となります。

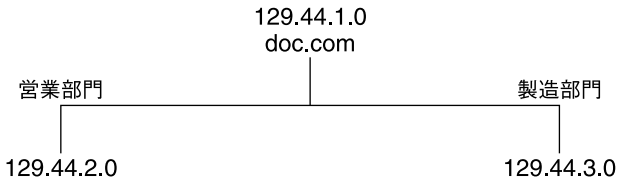
ネットワーク情報サービスがこの問題を解決します。ネットワーク情報サービスは、サーバー上にネットワーク情報を格納し、このサーバーが照会を実行するマシンに情報を提供します。

照会を実行するマシンは、サーバーの「クライアント」と呼ばれます。次の図に、クライアントとサーバーの関係を示します。ネットワークについての情報が変更されるたびに、各クライアントのローカルファイルを変更する代わりに、管理者はネットワーク情報サービスが格納する情報だけを更新します。これによって、エラー、クライアント間の不一致、そして作業量を減らすことができます。

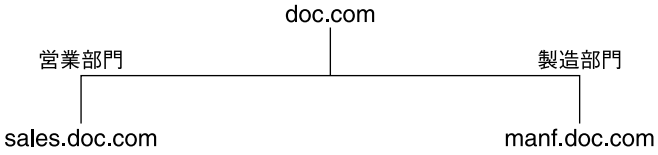


このように、サーバーがネットワークを通してサービスをまとめてクライアントに提供する方法を「クライアントサーバーコンピューティング」と呼びます。

ネットワーク情報サービスの第一の目的は情報の一元管理ですが、もう 1 つの目的はネットワーク名の簡素化です。たとえば、ある会社がネットワークを設定して、インターネットに接続したと仮定します。インターネットはその会社に 192.68.0.0 というネットワーク番号と、doc.com というドメインネームを割り当てました。会社には「営業 (Sales)」と「製造 (Manf)」という 2 つの部門があるため、このネットワークは 1 つのメインネットと、各部門に 1 つずつ、合計 2 つのサブネットに分割されます。各ネットには独自のアドレスがあります。



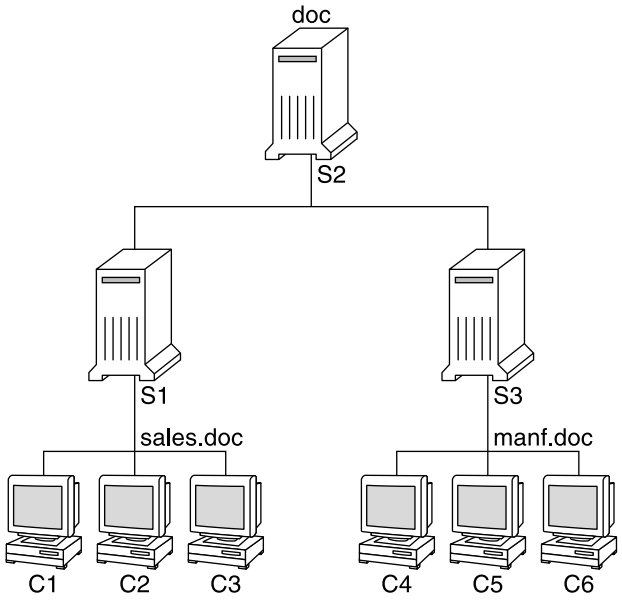
上に示すように、各部はネットワークアドレスで識別することもできますが、ネームサービスによって使用可能となる説明的な名前の方が便利です。



メールやその他のネットワーク通信の送信先を 129.44.1.0 というアドレスで指定する代わりに、単に doc と指定できます。また、192.68.2.0 や 192.68.3.0 と指定する代わりに、sales.doc や manf.doc と指定できます。

名前はまた、物理アドレスよりもはるかに柔軟です。物理的なネットワークはめったに変更されませんが、ネットワークを使用する組織はよく変化します。ネットワーク情報サービスは、組織と物理ネットワーク間のバッファーとして機能します。これは、ネットワーク情報サービスが物理的ネットワークに実際に接続されているのではなく、対応づけられているためです。

次の例でこれを説明します。この doc.com ネットワークが、S1、S2、S3 の 3 台のサーバーによってサポートされ、これらのうち 2 台のサーバー (S1 と S3) がクライアントをサポートしていると仮定します。

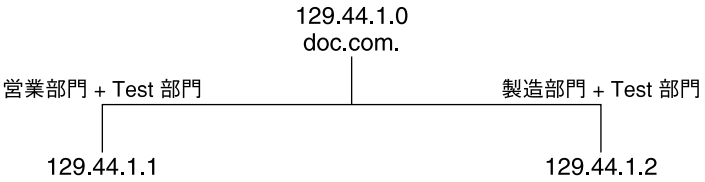


クライアント C1、C2、C3 はネットワーク情報をサーバー S1 から入手します。クライアント C4、C5、C6 はこれをサーバー S3 から入手します。結果として構成されるネットワークの概要を、次の表に示します(表は、前記のネットワークを一般化して表現したもので、実際のネットワーク情報マップとは異なります)。

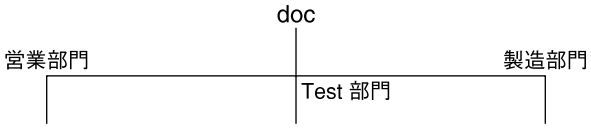
表 **1–1** `doc.com` ネットワークの構成

ネットワークアドレス	ネットワーク名	サーバー	クライアント
192.68.1.0	doc	S1	
192.68.2.0	sales.doc	S2	C1、C2、C3
192.68.3.0	manf.doc	S3	C4、C5、C6

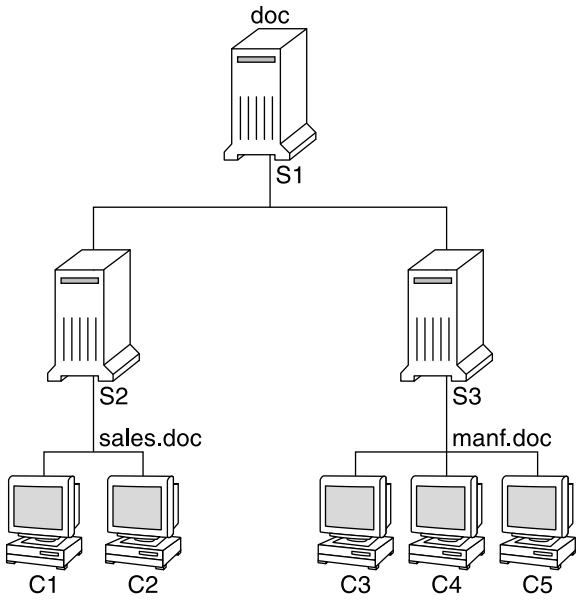
2 つの部門からある人数の人材を借りて第 3 の部門 Test を新設し、第 3 のサブネットは開設しなかったとします。その結果、物理ネットワークは、企業の組織とは対応しなくなります。



Test 部門のトラフィックには専用のサブネットがなく、`192.68.2.0` と `192.68.3.0` に分割されます。ここで、ネットワーク情報サービスを使用することにより、Test 部門のトラフィックにも専用のネットワークを備えることができます。



このように、組織が変更された場合、そのネットワーク情報サービスでは以下に示すようにマッピングを変更できます。



この変更の結果、クライアント C1 と C2 はサーバー S2 から、C3、C4、C5 はサーバー S3 から情報を入手するようになります。

各組織で行われる変更に対しては、ハードウェアのネットワーク構造を再編成することなく、「ソフト」ウェアのネットワーク情報構造を変更することにより対応できます。

Solaris のネームサービス

Solaris オペレーティング環境は、以下のネームサービスを提供します。

- DNS (ドメイン名システム、Domain Name System) - 29 ページの「DNS」を参照してください。
- `/etc` ファイル - 初期の UNIX™ のネームシステム。30 ページの「`/etc` ファイル」 を参照してください。
- NIS (ネットワーク情報サービス、Network Information Service) - 30 ページの「NIS」を参照してください。
- NIS+ (ネットワーク情報サービスプラス、Network Information Service Plus) - 『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。
- FNS (フェデレーテッド・ネーミング・サービス、Federated Naming Service) - 『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*を参照してください。

最近のほとんどのネットワークでは、これらのサービスを 2 つ、またはそれ以上組み合わせて使用します。複数のサービスを使用するときは、`nsswitch.conf` ファイルで調整します。`nsswitch.conf` ファイルについては第 2 章で説明します。

DNS

DNS は TCP/IP ネットワーク用にインターネットが提供するネームサービスです。ネットワーク上のマシンがインターネットアドレスではなく、普通の名前で識別できるように開発されたものです。DNS は、ローカルの管理ドメイン内と、複数の管理ドメインの間でホスト名の管理を行います。

DNS を使用する、ネットワークに接続されたマシンの集合のことを「DNS 名前空間」と呼びます。DNS 名前空間は階層になった複数の「ドメイン」に分けることができます。DNS ドメインは、複数のマシンからなるグループです。各ドメインは複数の「ネームサーバー」、つまり、1 つの主サーバーと 1 つまたは複数の副サーバーよってサポートされます。各サーバーは `in.named` と呼ばれるデーモンを実行することによって DNS を実装しています。クライアント側は、「リゾルバ」によって DNS を実装します。リゾルバの機能は、ユーザーによる参照を解決することです。リゾルバがネームサーバーに照会すると、ネームサーバーは要求された情報か、または他のサーバーに照会する旨を返します。

/etc ファイル

ホストを基本とした初期の UNIX のネームシステムは、スタンドアロンの UNIX マシン用に開発された後、ネットワークで使用されるようになりました。UNIX オペレーティングシステムの旧版の多くや UNIX マシンでは、現在でもこのシステムが使用されていますが、大規模で複雑なネットワークにはあまり適切ではありません。

NIS

ネットワーク情報サービス (NIS) は DNS とは独立して開発され、目的はやや異なっています。DNS が数値 IP アドレスの代わりにマシン名を使うことによって、通信を簡略化することに焦点を当てているのに対して、NIS は、多様なネットワーク情報を集中管理することによりネットワーク管理機能を高めることに焦点を当てています。NIS には、マシンの名前とアドレス、ユーザー、ネットワークそのもの、ネットワークサービスについての情報も格納されます。このようなネットワーク情報の集合体を、「NIS 名前空間」と呼びます。

NIS 名前空間情報は NIS マップに格納されています。NIS マップは、UNIX の `/etc` ファイルおよび他の構成ファイルを置換するように設計されているので、名前およびアドレスよりはるかに多くの情報を保存できます。その結果、NIS 名前空間には非常に大きなマップの集合が含まれることになります。詳細については、160 ページの「NIS マップに関する作業」を参照してください。

NIS は DNS に似たクライアントサーバーの配列を持っています。複製された NIS サーバーは NIS クライアントへサービスを提供します。主サーバーは「マスター」サーバーと呼ばれ、信頼性を保証するためにバックアップつまり「スレーブ」サーバーを持っています。どちらのサーバーも NIS 情報検索ソフトウェアを使用し、NIS マップを格納します。NIS アーキテクチャおよび NIS の管理方法の詳細については、第 8 章および第 9 章を参照してください。

NIS+

ネットワーク情報サービスプラス (NIS+) は、NIS によく似たネットワークネームサービスですが、より多くの機能を備えています。NIS+ は、NIS の拡張機能ではなく、異なるソフトウェアプログラムです。

NIS+ ネームサービスは、ほとんどすべてのネットワーク構成に対して、インストールを実行する組織の形態に適合するように設計されています。NIS とは異なり、NIS+ の名前空間は動的な構成で、正規ユーザーであればいつでも更新できます。

NIS+ はマシンのアドレス、セキュリティ情報、メール情報、Ethernet インタフェース、ネットワークサービスなどの情報を 1 カ所に格納して、ネットワーク上のすべてのマシンからアクセスできるようにします。このように構成されたネットワーク情報を、NIS+「名前空間」と呼びます。

NIS+ 名前空間は階層構造となっていて、UNIX のディレクトリファイルシステムによく似ています。階層構造になっていることから、NIS+ 名前空間を企業組織の階層に合わせて構成できます。名前空間における情報の配置は、物理的な配置とは関係ありません。したがって、NIS+ 名前空間は、独立して管理できる複数のドメインに分割できます。クライアントは、適切なアクセス権があれば、自分のドメインだけではなくほかのドメインの情報にもアクセスできます。

NIS+ では、NIS+ 名前空間への情報の保存やその情報へのアクセスにクライアントサーバーモデルを使用します。各ドメインは複数のサーバーによってサポートされます。メインのサーバーは「主」サーバーと呼ばれ、バックアップサーバーは「副」サーバーと呼ばれます。ネットワーク情報は、NIS+ 内部のデータベースにある 16 個の標準 NIS+ テーブルに格納されます。主サーバーと副サーバーの両方で NIS+ サーバーソフトウェアが動作しており、NIS+ テーブルのコピーを管理しています。マスターサーバー上の NIS+ データの変更は、副サーバーにも自動的に伝達されます。

NIS+ では高機能のセキュリティシステムによって、名前空間の構造と保存されている情報が保護されます。このシステムは、情報にアクセスしようとしているクライアントが正当なものであるかどうかを認証と承認によって確認します。「認証」とは、情報の要求者がネットワークの正当なユーザーであるかどうかを判定することです。「承認」では、特定のユーザーが情報を所有したり修正したりできるかどうかを確認します。NIS+ のセキュリティおよびその管理方法の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

FNS

FNS の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』 を参照してください。

LDAP ネームサービス

Solaris 9 は、iPlanet™ Directory Server 5.1 および他の LDAP Directory Server を使用する場合、LDAP (Lightweight Directory Access Protocol) をサポートします。

詳細については、第 12 章を参照してください。

ネームサービスの比較一覧

	DNS	NIS	NIS+	FNS	LDAP
名前空間	階層	フラット	階層	階層	階層
データ記憶領域	ファイル/リソースレコード	2 列のマップ	複数列のテーブル	マップ	ディレクトリ (可変)
サーバー名	マスター/スレーブ	マスター/スレーブ	ルートマスター/非ルートマスター 主/副 キャッシュ/スタブ	なし	マスター/複製
セキュリティ	SSL	なし (root またはなし)	DES 認証	なし (root または、なし)	SSL
トランスポート	TCP/IP	LAN	LAN	LAN	TCP/IP
スケール	グローバル	LAN	LAN	グローバル (DNS 付) /LAN	グローバル

第 2 章

ネームサービススイッチ (概要)

この章では、ネームサービススイッチ、その機能、およびこのスイッチを使用してクライアントが 1 つまたは複数のソースからネーミング情報を入手する方法について説明します。ネームサービススイッチは、異なるネームサービスの使用方法を調整するために使います。

ネームサービススイッチについて

ネームサービススイッチは `nsswitch.conf(4)` という名前のファイルで、クライアントのマシンやアプリケーションがネットワーク情報を得る方法を管理します。 ネームサービススイッチは、次のような `getXbyY()` インタフェースのいずれかを呼び出すクライアントアプリケーションによって使用されます。

- `gethostbyname()`
- `getpwuid()`
- `getpwnam()`
- `getipnodebyname()`

各マシンの `/etc` ディレクトリには、スイッチファイルがあります。ファイルの各行は、host、passwd、group などの特定タイプのネットワーク情報を識別します。その後には、クライアントがネットワーク情報を探すための 1 つまたは複数のソースが続きます。

クライアントは、1 つまたは複数のスイッチのソースからネーミング情報を入手できます。たとえば、NIS+ のクライアントは、NIS+ テーブルからホスト情報を、ローカルの `/etc` ファイルからパスワード情報をそれぞれ入手できます。さらに、スイッチが各ソースを使用する条件を指定することもできます。表 2–1 を参照してください。

Solaris オペレーティング環境では、インストールの過程で、各マシンの /etc ディレクトリに nsswitch.conf ファイルが自動的にロードされます。LDAP、NIS、NIS+ またはローカルファイル用にスイッチファイルの 4 つの代替 (テンプレート) バージョンも /etc にロードされます。38 ページの「nsswitch.conf テンプレートファイル」 を参照してください。

これら 4 つのファイルは、代替デフォルトスイッチファイルです。各ファイルはそれぞれ /etc ファイル、NIS、NIS+ 、LDAP という異なる主要なネームサービス用に設計されています。 Solaris ソフトウェアをマシンに最初にインストールする時に、インストール担当者はマシンのデフォルトのネームサービス (NIS+、NIS、ローカルファイル、または LDAP) を選択します。 インストール中に、対応するテンプレートファイルが nsswitch.conf ファイルにコピーされます。たとえば、LDAP を使用するクライアントマシンでは、インストールの過程で nsswitch.ldap が nsswitch.conf にコピーされます。特殊な名前空間を持っている場合を除き、通常の操作には nsswitch.conf にコピーされるデフォルトのテンプレートファイルを使用できます。

DNS または IPv6 用のデフォルトファイルは提供されませんが、これら 4 つのファイルを編集して DNS または IPv6 用に使用できます。44 ページの「DNS とインターネットでのアクセス」または44 ページの「IPv6 と Solaris ネームサービス」を参照してください。

マシンの主要なネームサービスを後から変更する場合は、該当する代替スイッチファイルを nsswitch.conf にコピーします。38 ページの「nsswitch.conf テンプレートファイル」を参照してください。NIS 管理者はまた、/etc/nsswitch.conf ファイルの該当行を編集することによって、クライアントで使用する特定タイプのネットワーク情報のソースを変更できます。この操作を行うための構文について、以下に説明します。詳細については、43 ページの「ネームサービススイッチの変更」を参照してください。

nsswitch.conf ファイルのフォーマット

nsswitch.conf ファイルは、基本的には 16 種類の情報とそのソース (getXXbyYY() 関数の情報検索先) のリストです。16 種類の情報は次のとおりです (順序は、必ずしも次のとおりではありません)。

- aliases
- bootparams
- ethers
- group
- hosts
- ipnodes
- netgroup
- netmasks
- networks
- passwd (シャドウ情報含む)
- protocols

- publickey
- rpc
- services
- automount
- sendmailvars

次の表に、上記の情報タイプのスイッチファイルの中に表示できるソースの種類とその説明を示します。

表 **2–1** スイッチファイルの情報ソース

ソース	説明
files	クライアントの /etc ディレクトリに格納されているローカルファイル (/etc/passwd など)
nisplus	NIS+ テーブル (hosts テーブルなど)
nis	NIS マップ (hosts マップなど)
compat	パスワードとグループ情報を対象に、/etc/passwd、/etc/shadow、/etc/group ファイルで旧形式の「+」または「-」構文をサポートする
dns	ホスト情報を DNS から入手するように指定する
ldap	エントリを LDAP ディレクトリから入手するように指定する

検索規準

「単一ソース」。nisplus のような情報のソースが 1 つだけの場合、スイッチを使用している関数は、そのソースだけで情報を検索します。情報が見つかった場合、success という状態メッセージが渡されます。情報が見つからない場合は、検索が停止され「success」 以外の状態メッセージが返されます。状態メッセージに基づいて何をするかは、関数によって異なります。

「複数ソース」。テーブルに複数のソースがある場合、スイッチは最初のソースから情報検索を始めるように関数に指示します。情報が見つかれば success という状態メッセージが返されます。情報が見つからない場合は、次のソースが検索されます。関数は必要な情報が見つかるか、return 処理によって中止されるまで全ソースの検索を続けます。必要な情報がどのソースにもなかった場合、関数は検索を停止し、「non-success」 という状態メッセージを返します。

スイッチ状態メッセージ

関数は情報を見つけると、「success」 という状態メッセージを返します。探している情報が見つからない場合は、その理由によって、3 種類の状態メッセージのいずれかを返します。表示される状態メッセージを次の表に示します。

表 **2–2** スイッチ状態メッセージ

状態メッセージ	意味
SUCCESS	要求されたエントリがソース内で発見された
UNAVAIL	ソースが応答しない、または使用不可。つまり、NIS+ テーブル、NIS マップ、または /etc ディレクトリのファイルが見つからなかったかアクセスできなかった
NOTFOUND	ソースが「エントリなし」と応答した。テーブル、マップ、ファイルにアクセスしたが、必要な情報は見つからなかった
TRYAGAIN	ソース使用中のため再検索の必要あり。テーブル、マップ、ファイルは見つかったが、照会に対して応答しなかった

スイッチの動作に関するオプション

次の表に示すように、状態メッセージに対して 2 つの「動作」のどちらかで応答するようにスイッチに指示できます。

表 **2–3** スイッチ状態メッセージへの応答

作業	意味
return	情報の検索を停止する
continue	次のソースがあれば、それを検索する

デフォルト検索基準

nsswitch.conf ファイルの状態メッセージと動作オプションの組み合わせによって、関数の各ステップでの動作が決まります。この状態と動作の組み合わせのことを、「検索基準」と呼びます。

スイッチのデフォルト検索規準は、どのソースについても同じです。これらを上記の状態メッセージに基づいて説明すると、次のようになります。

- SUCCESS=return。情報の検索を停止し、見つかった情報を使用して処理を続行する
- UNAVAIL=continue。次のソース (nsswitch.conf ファイルに指定されたもの) を使用して検索を続行する。次のソースがなければ「NOTFOUND」という状態メッセージを返す
- NOTFOUND=continue。次のソース (nsswitch.conf ファイルに指定されたもの) を使用して検索を続行する。次のソースがなければ「NOTFOUND」という状態メッセージを返す
- TRYAGAIN=continue。次のソース (nsswitch.conf ファイルに指定されたもの) を使用して検索を続行する。次のソースがなければ「NOTFOUND」という状態メッセージを返す

これらはデフォルトの検索基準であるため、自動的に表示されます。つまり、スイッチファイルで、明示的に指定する必要はありません。ほかの検索基準を明示的に指定してデフォルトの検索基準を変更するには、上記の *STATUS=action* という構文を使用します。たとえば、NOTFOUND 状態に対し、デフォルトの動作では次のソースに対する検索を続行します。networks など特定のタイプの情報を設定して検索すると、検索は NOTFOUND 状態で中止されます。スイッチファイルの networks の行は、次のように編集されていると考えられます。

```
networks: nis [NOTFOUND=return] files
```

networks: nis [NOTFOUND=return] files 行は、NOTFOUND 状態に関してデフォルトでない検索基準を設定するものです。デフォルト以外の設定をするときは [ ] を使用します。

この例では、検索関数は次のような働きをします。

- networks マップが見つかり必要な情報があった場合、関数は「SUCCESS」という状態メッセージを返します。
- networksマップが見つからなかった場合、関数は「UNAVAIL」という状態メッセージを返しデフォルトで適切な /etc ファイルの検索を続行します。
- networksマップは見つかったが必要な情報がなかった場合、関数は「NOTFOUND」という状態メッセージを返します。そして /etc ファイルの検索を続行する (デフォルトの設定) 代わりに検索を停止します。
- networks マップが使用中の場合、関数は「TRYAGAIN」という状態メッセージを返し、デフォルトで適切な /etc ファイルの検索を続行します。

構文が正しくない場合の処理

クライアントのライブラリ関数には、nsswitch.conf ファイルにおいて「必要なエントリがない」、「エントリの構文が誤っている」といった場合に使用される、コンパイル時に組み込まれるデフォルトエントリがあります。これらのエントリは nsswitch.conf ファイルのデフォルトエントリと同じものです。

ネームサービススイッチは、テーブル名やソース名のスペルが正しいものとして処理をします。テーブル名やソース名のスペルが正しくない場合は、デフォルト値が使用されます。

Auto_home と Auto_master

auto_home テーブル、auto_master テーブルとマップのスイッチ検索基準は、automount と呼ばれる 1 つのカテゴリに統合されます。

Timezone とスイッチファイル

timezone テーブルはスイッチを使用しないため、スイッチファイルのリストには含まれていません。

nsswitch.conf ファイル中のコメント

nsswitch.conf ファイル中の行のうち、コメント文字 (#) で始まっているものはコメント行として解釈され、ファイルを検索する関数では無視されます。

行の途中にコメント文字 (#) が含まれる場合、コメント文字の前の文字列は nsswitch.conf ファイルを検索する関数によって解釈されます。コメント文字よりあとの文字列は、コメントとして解釈され、無視されます。

表 **2–4** スイッチファイルのコメント例

行の種類	コメント例
コメント行 (無視される)	#hosts: nisplus [NOTFOUND=return] files
完全に解釈される行	hosts: nisplus [NOTFOUND=return] file
部分的に解釈される行 (「files」の部分は解釈されない)	hosts: nisplus [NOTFOUND=return] # files

スイッチファイルのキーサーバーと publickey エントリ

注意 – nsswitch.conf に変更を加えた後は、キーサーバーを再起動する必要があります。

キーサーバーは、起動時にだけネームサービススイッチ構成ファイルの publickey エントリを参照します。つまり、スイッチ構成ファイルを更新しても再起動しない限り、キーサーバーは publickey への変更を認識しないということになります。

nsswitch.conf テンプレートファイル

Solaris オペレーティング環境では、さまざまなネームサービスに対応できるように、nsswitch.conf(4) テンプレートファイルが 4 つ用意されています。各ファイルでは、デフォルトの情報ソース (一次ソース、および二次以降のソース) として、それぞれ異なる内容が指定されています。

4 つのテンプレートファイルは、次のとおりです。

- 「LDAP テンプレートファイル」(nsswitch.ldap ファイル)。この構成ファイルでは、マシンの情報の一次ソースとして LDAP ディレクトリが指定されています。

注 – LDAP ネームサービスを使用するには、すべての LDAP クライアントマシンを正しく設定し、nsswitch.conf を変更する必要があります。詳細については、第 16 章を参照してください。

- 「NIS+ テンプレートファイル」(nsswitch.nisplus ファイル)。この構成ファイルでは、passwd、group、automount、aliases を除くすべての情報の一次ソースとして NIS+ が指定されています。passwd、group、automount、aliases の一次ソースとして指定されているのは /etc ディレクトリのファイルで、二次ソースとして指定されているのは NIS+ テーブルです。[NOTFOUND=return] という検索基準は、「No such entry」というメッセージを受け取ったら NIS+ テーブルの検索を停止するという意味です。また、ローカルファイルを検索するのは NIS+ サーバーを使用できない場合だけです。
- 「NIS テンプレートファイル」(nsswitch.nis ファイル)。この構成ファイルは、NIS+ テーブルではなく NIS マップを使用するという点を除けば、NIS+ テンプレートファイルとほぼ同じです。passwd と group の情報に関しては files nis という順序で検索するよう指定されているため、/etc/passwd と/etc/group の各ファイルに + エントリを指定する必要はありません。
- 「Files テンプレートファイル」(nsswitch.files ファイル)。この構成ファイルでは、マシンの情報ソースとしてローカルの /etc ディレクトリのファイルだけが指定されています。netgroup に関する files のソースは存在しないため、クライアントがスイッチファイルでこのエントリを使用することはありません。

要件に一番近いテンプレートファイルを nsswitch.conf 構成ファイルにコピーして、必要に応じてファイルを変更します。

たとえば、LDAP テンプレートファイルを使用する場合は、次のコマンドを入力します。

mymachine# **cp nsswitch.ldap nsswitch.conf**

デフォルトスイッチテンプレートファイル

Solaris オペレーティング環境で用意されている 4 つのスイッチファイルは、次のとおりです。

例 **2–1** NIS+ スイッチファイルテンプレート (nsswitch.nisplus)

```
#
# /etc/nsswitch.nisplus:
#
# An example file that could be copied over to /etc/nsswitch.conf;
```

例 **2–1** NIS+ スイッチファイルテンプレート (nsswitch.nisplus)　(続き)

```
# it uses NIS+ (NIS Version 3) in conjunction with files.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file has a "-" for nametoaddr_libs of "inet"
# transports.
# the following two lines obviate the "+" entry in /etc/passwd
# and /etc/group.
passwd: files nisplus
group: files nisplus
# consult /etc "files" only if nisplus is down.
hosts: nisplus [NOTFOUND=return] files
# Uncomment the following line, and comment out the above, to use
# both DNS and NIS+. You must also set up the /etc/resolv.conf
# file for DNS name server lookup. See resolv.conf(4).
# hosts: nisplus dns [NOTFOUND=return] files
services: nisplus [NOTFOUND=return] files
networks: nisplus [NOTFOUND=return] files
protocols: nisplus [NOTFOUND=return] files
rpc: nisplus [NOTFOUND=return] files
ethers: nisplus [NOTFOUND=return] files
netmasks: nisplus [NOTFOUND=return] files
bootparams: nisplus [NOTFOUND=return] files
publickey: nisplus
netgroup: nisplus
automount: files nisplus
aliases: files nisplus
sendmailvars: files nisplus
```

例 **2–2** NIS スイッチファイルテンプレート

```
#
# /etc/nsswitch.nis:
#
# An example file that could be copied over to /etc/nsswitch.conf;
# it uses NIS (YP) in conjunction with files.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file has a "-" for nametoaddr_libs of "inet"
# transports.
#
# the following two lines obviate the "+" entry in /etc/passwd
# and /etc/group.
passwd: files nis
group: files nis
# consult /etc "files" only if nis is down.
hosts: nis [NOTFOUND=return] files
networks: nis [NOTFOUND=return] files
protocols: nis [NOTFOUND=return] files
rpc: nis [NOTFOUND=return] files
ethers: nis [NOTFOUND=return] files
netmasks: nis [NOTFOUND=return] files
bootparams: nis [NOTFOUND=return] files
```

例 **2–2** NIS スイッチファイルテンプレート　(続き)

```
publickey: nis [NOTFOUND=return] files
netgroup: nis
automount: files nis
aliases: files nis
# for efficient getservbyname() avoid nis
services: files nis
sendmailvars: files
```

例 **2–3** Files スイッチファイルテンプレート

```
#
# /etc/nsswitch.files:
#
# An example file that could be copied over to /etc/nsswitch.conf;
# it does not use any naming service.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file has a "-" for nametoaddr_libs of "inet"
# transports.
passwd: files
group: files
hosts: files
networks: files
protocols: files
rpc: files
ethers: files
netmasks: files
bootparams: files
publickey: files
# At present there isn't a 'files' backend for netgroup;
# the system will figure it out pretty quickly, and will notuse
# netgroups at all.
netgroup: files
automount: files
aliases: files
services: files
sendmailvars: files
```

例 **2–4** LDAP スイッチファイルテンプレート

```
 #
# /etc/nsswitch.ldap:
#
# An example file that could be copied over to /etc/nsswitch.conf; it
# uses LDAP in conjunction with files.
#
# "hosts:" and "services:" in this file are used only if the
# /etc/netconfig file has a "-" for nametoaddr_libs of "inet" transports.

# the following two lines obviate the "+" entry in /etc/passwd
and /etc/group.
passwd:     files ldap
```

例 **2–4** LDAP スイッチファイルテンプレート　(続き)

```
group:      files ldap

hosts:      ldap [NOTFOUND=return] files

networks:   ldap [NOTFOUND=return] files
protocols:  ldap [NOTFOUND=return] files
rpc:        ldap [NOTFOUND=return] files
ethers:     ldap [NOTFOUND=return] files
netmasks:   ldap [NOTFOUND=return] files
bootparams: ldap [NOTFOUND=return] files
publickey:  ldap [NOTFOUND=return] files

netgroup:   ldap

automount:  files ldap
aliases:    files ldap

# for efficient getservbyname() avoid ldap
services:   files ldap
sendmailvars:   files
```

nsswitch.conf ファイル

Solaris オペレーティング環境を初めてインストールするときのデフォルトの nsswitch.conf ファイルは、Solaris のソフトウェアをインストールする際に選択したネームサービスで決まります。このファイルの各行は、ネットワーク情報の種類 (ホスト、パスワード、グループなど) と、それに対する 1 つ以上の情報ソース (NIS+ テーブル、NIS マップ、DNS ホストテーブル、同一マシン上の /etc など) を対応させています。クライアントは、この情報ソースから各情報を検索します。ネームサービスを選択すると、そのサービスのスイッチテンプレートファイルがコピーされ新しい nsswitch.conf ファイルが作成されます。たとえば、NIS+ を選択した場合は、nsswitch.nisplus ファイルがコピーされ新しい nsswitch.conf ファイルが作成されます。

/etc/nsswitch.conf ファイルは、Solaris 9 リリースをインストールすると各マシンの /etc ディレクトリに自動的に作成されます。また、次の 4 つの代替テンプレートファイルも作成されます。

- /etc/nsswitch.nisplus
- /etc/nsswitch.nis
- /etc/nsswitch.files
- /etc/nsswitch.ldap

これらの 4 つの代替テンプレートファイルには、それぞれネットワーク情報の情報ソースとして NIS+、NIS、ローカルファイル、 または LDAP を使用する標準的なスイッチ構成が設定されています。DNS 用のデフォルトファイルは提供されませんが、これら 4 つのファイルのどれでも編集して DNS 用に使用できます。第 5 章を参照し

てください。Solaris オペレーティング環境をマシンに最初にインストールするときに、インストール担当者はマシンのデフォルトのネームサービス (NIS+、NIS、ローカルファイル、または LDAP) を選択します。インストール中に、選択されたネームサービスに対応するテンプレートファイルが /etc/nsswitch.conf にコピーされます。たとえば、NIS+ を使用しているクライアントマシンでは、インストールの過程で nsswitch.nisplus が nsswitch.conf にコピーされます。

ネットワークがインターネットに接続されており、ユーザーが DNS を使用してインターネット上のホストにアクセスできるようにする場合は、DNS 転送を有効にする必要があります。

特殊な名前空間を持っている場合を除き、通常の操作には nsswitch.conf にコピーされるデフォルトのテンプレートファイルを使用します。

構成ファイルの変更

マシンのネームサービスを変更するときは、そのマシンのスイッチファイルを新しいネームサービスに対応させて変更する必要があります。たとえば、マシンのネームサービスを NIS から NIS+ に変更する場合、スイッチファイルを NIS+ に対応したものに変更する必要があります。スイッチファイルを変更するには、対応するテンプレートファイルを nsswitch.conf にコピーします。

NIS+ インストールスクリプトを使って NIS+ をマシンにインストールすると、NIS+ テンプレートファイルが自動的に nsswitch.conf にコピーされます。この場合、特にスイッチファイルをカスタマイズしたいというのでなければ、スイッチファイルを明示的に変更する必要はありません。

スイッチファイルを変更する前に、ファイルに列挙されている情報ソースが正しく設定されていることを確認してください。たとえば、NIS+ 用スイッチファイルに変更するのであれば、ワークステーションには NIS+ サービスへのアクセス権が必要になり、ローカルファイル用スイッチファイルに変更するのであれば、それらのローカルファイルがワークステーション上に正しく設定されている必要があります。

▼ ネームサービススイッチの変更

スイッチファイルを変更する場合は、次の手順に従います。

1. スーパーユーザーになります。
2. 使用するネームサービス用のテンプレートファイルを nsswitch.conf にコピーします。

   「NIS+ 用」(NIS+ スクリプトにより自動的にコピーされる)

   client1# **cd /etc**

client1# **cp nsswitch.nisplus nsswitch.conf**

「NIS 用」

client1# **cd /etc**

client1# **cp nsswitch.nis nsswitch.conf**

「ローカルの /etc ファイル用」

client# **cd /etc**

client# **cp nsswitch.files nsswitch.conf**

3. マシンをリブートします。

nscd ネームサービスキャッシュデーモンによってスイッチ情報がキャッシュに書き込まれます。ライブラリ関数の中には、nsswitch.conf ファイルが変更されたかどうかを定期的にチェックしないものがあります。このため、マシンをリブートして、nscd とこれらのライブラリ関数が確実に最新スイッチの情報を持つようにする必要があります。

注 – LDAP ネームサービスを使用するには、すべての LDAP クライアントマシンを正しく設定し、nsswitch.conf を変更する必要があります。詳細については、第 16 章を参照してください。

DNS とインターネットでのアクセス

nsswitch.conf ファイルでは、以下の節で説明しているように、クライアントの DNS 転送も制御しています。DNS 転送によって、クライアントへのインターネットでのアクセスが可能になります。NIS と NIS+ 用に DNS 転送を設定する方法については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』を参照してください。

IPv6 と Solaris ネームサービス

注 – DNS と LDAP は、IPv6 アドレスを格納できるという点で IPv6 との互換性があります。 ただし、Solaris 9 では、クライアントサーバーの DNS または LDAP トラフィックに IPv6 トランスポートを使用できません。LDAP ネームサービスは、IPv6 専用のネットワークではまだ機能しません。

NIS と NIS+ では、IPv6 データを格納できるだけでなく、NIS/NIS+ プロトコルのトラフィックに IPv6 トランスポートを使用することもできます。

nsswitch.conf ファイルは、IPv6 アドレスの検索基準を制御します。IPv6 は、32 ビットから 128 ビットまで IP アドレスサイズを大きくして、より多くのアドレス階層をサポートし、より多くのノードにアドレス指定できるようにします。IPv6 の構成と実装の詳細については、『*Solaris* のシステム管理 *(IP* サービス*)*』を参照してください。

IPv6 アドレスには、新しい ipnodes ソースを使用してください。
/etc/inet/ipnodes ファイルには、IPv4 と IPv6 のアドレスが格納されています。/etc/inet/ipnodes ファイルは、/etc/hosts ファイルと同じフォーマットを使用します。

IPv6 のネームサービスでは、 検索用に新しい ipnodes ソースを使用しています。たとえば、LDAP で IPv6 アドレスを認識させる場合は、次のように指定します。

```
ipnodes: ldap [NOTFOUND=return] files
```

注意 – ipnodes は、デフォルトでは files です。 IPv4 から IPv6 への変更中には、すべてのネームサービスが、IPv6 アドレスを認識できるわけではないので、デフォルトの files を使用します。このデフォルトを使用しない場合には、アドレスの解決中に不必要な遅延が生じることがあります (ブート時の遅延など)。

注意 – アプリケーションは、IPv4 アドレスを ipnodes データベースで検索してから、hosts データベースを検索します。ipnodes を指定する前に、IPv4 アドレスの両方のデータベースを検索する時間を考慮にいれる必要があります。

+/- 構文との互換性を追加する

/etc/passwd、/etc/shadow、/etc/group の各ファイルで +/- 構文を使用する場合は、nsswitch.conf ファイルを変更して互換性を確保する必要があります。

- 「NIS+」。NIS+ で +/- 構文と同じ効果を得るには、passwd および groups のソースを compat に変更し、nsswitch.conf ファイルの passwd あるいは group エントリの後に passwd_compat: nisplus というエントリを追加します (下記参照)。

```
passwd: compat
passwd_compat: nisplus
group: compat
group_compat: nisplus
```

上記の指定により、クライアント関数は、/etc ファイル内の +/- エントリで指定されているとおりに /etc ファイルと NIS+ テーブルからネットワーク情報を入手します。

- 「NIS」。Sun オペレーティング環境 4.x リリースの構文と同じ効果を得るには、passwd と groups の各ソースをcompat に変更します。

```
passwd: compat
group: compat
```

この指定により、クライアント関数は、/etcファイル内の +/- エントリで指定されているとおりに /etc ファイルと NIS マップからネットワーク情報を入手します。

注 – NIS+ サーバーが NIS 互換モードで動作している場合、クライアントマシンでは netgroup テーブルに対して ypcat を実行できません。実行すると、エントリの有無に関わらず「テーブルが空である」という結果が返されます。

スイッチファイルとパスワード情報

注意 – passwd 情報の nsswitch.conf ファイルでは、files を 1 番目のソースにしてください。files が 1 番目のソースでない場合は、ネットワークセキュリティが低くなり、ログの扱いが難しくなります。

たとえば、NIS+ の環境では、nsswitch.conf ファイルの passwd 行は次のようになります。

```
passwd: files nisplus
```

NIS の環境では、nsswitch.conf ファイルの passwd 行は次のようになります。

```
passwd: files nis
```

パート II　DNS の設定と管理

ここでは、Solaris オペレーティング環境における DNS ネームサービスの設定、構成、管理、障害追跡について説明します。

第 3 章

ドメインネームシステム (概要)

この章ではドメインネームシステム (DNS) の構造と概要を説明します。

注 – DNS の利用で最も一般的で重要なことは、ネットワークをグローバルなインターネットに接続することです。インターネットに接続するためには、親ドメインの管理者にネットワークの IP アドレスを登録してもらう必要があります。

この章の内容は次のとおりです。

DNS の基礎

ドメインネームシステム (DNS) は、標準 TCP/IP プロトコル群のひとつのアプリケーション層プロトコルです。このプロトコルによって DNS ネームサービスが実現します。DNS ネームサービスはインターネットで使用されるネームサービスです。

ここでは、DNS の基本的な概念について説明します。説明に際しては、ネットワークの管理 (特に TCP/IP) に精通していて、NIS+ や NIS といった他のネームサービスについての知識があることを前提とします。

DNS の初期設定と構成については、第 4 章を参照してください。

注 – DNS、NIS+、NIS、FNS には同じような機能があり、時として異なる構成要素を定義するのに同じ用語を使用している場合がありますが、この章では、ドメインやネームサーバーといった用語は DNS の機能に合わせて定義しています。なぜなら、DNS の機能は NIS+ や NIS のドメインやサーバーとは大きく異なるからです。

名前のアドレス解決

DNS はインターネット上の複雑で全世界的なコンピュータの階層をサポートしますが、それ自身の基本機能は非常に簡単なものです。すなわち、TCP/IP に準拠したネットワークに「名前のアドレス解決」を提供するということです。名前のアドレス解決は「マッピング」ともいい、あるコンピュータのホスト名をインデックスとして用い、その IP アドレスをデータベースから見つけだすプロセスのことです。

名前のアドレスマッピングは、ローカルマシンで実行されているプログラムが遠隔コンピュータにアクセスする必要があるときに行われます。ほとんどの場合、このプログラムでは遠隔コンピュータのホスト名はわかっても、その場所を特定できません。特に遠隔マシンが自分のサイトから何マイルも離れた他の会社にある場合には場所を特定できません。プログラムは遠隔マシンのアドレスを得るために、ローカルマシン上で動作している DNS ソフトウェアに要求を送ります。このときのローカルマシンを「DNS クライアント」と呼びます。

ローカルマシンは、「DNS ネームサーバー」に要求を送ります。DNS ネームサーバーは 分散型 DNS データベースを保持しています。NIS+ の `host` または `ipnodes` テーブル、あるいはローカルの `/etc/hosts` ファイル、`/etc/inet/ipnodes` ファイルも同様の情報、すなわち、ホスト名、`ipnodes` 名、IPv4、IPv6 のアドレス、その他コンピュータの特定のグループに関する情報を保持していますが、DNS データベースのファイルはそれらと異なる部分が多くあります。ネームサーバーはローカルマシンが遠隔マシンの IP アドレスを検索または解決するための要求の一部として、送信する自分自身のホスト名を使用しますが、ネームサーバーは、そのホスト名も利用します。そして、要求したホスト名が DNS データベースにあれば、その IP アドレスがネームサーバーによってローカルマシンに返されます。

次の図に、DNS クライアントのローカルネットワーク上で行われるクライアントとネームサーバーの間での名前のアドレスマッピングを示します。

図 **3–1** 名前のアドレス解決

ホスト名がネームサーバーの DNS データベースになければ、そのマシンは権限の外側にある、すなわち、DNS の用語でいう「ローカル管理ドメイン」の外側にあることを意味します。このように、各ネームサーバーは、ローカル管理ドメインに対して権限があるとされています。

幸い、ローカルネームサーバーでは、ルートドメインネームサーバーのホスト名と IP アドレスのリストを保持しています。ローカルネームサーバーは、ローカルマシンからの要求を「ルートドメインネームサーバー」に転送します。ルートネームサーバーは、61 ページの「DNS 階層とインターネット」で詳しく説明しているように、大きな組織のドメインに対して権限があります。その階層は、上下逆のツリー構造で編成された UNIX のファイルシステムに似ています。

各ルートネームサーバーでは、会社、大学、またはその他大規模な組織における最上位のドメインネームサーバーのホスト名と IP アドレスを管理しています。ルートネームサーバーは、ホスト名と IP アドレスがわかっているすべての最上位のネームサーバーにローカルマシンからの要求を送信します。要求されたホストの IP アドレスを最上位のネームサーバーのどれかが持っている場合、その情報がローカルマシンに返されます。最上位のサーバーが要求されたホストに関する情報を持っていない場合、情報がわかっている第 2 レベルのネームサーバーに要求が渡されます。このようにローカルマシンからの要求は組織の巨大なツリー構造の下へ向かって順に渡されます。最終的に、ローカルマシンからの要求に関する情報がデータベースに格納されているネームサーバーが IP アドレスを返します。

次の図に、ローカルドメインの外側での名前のアドレス解決を示します。

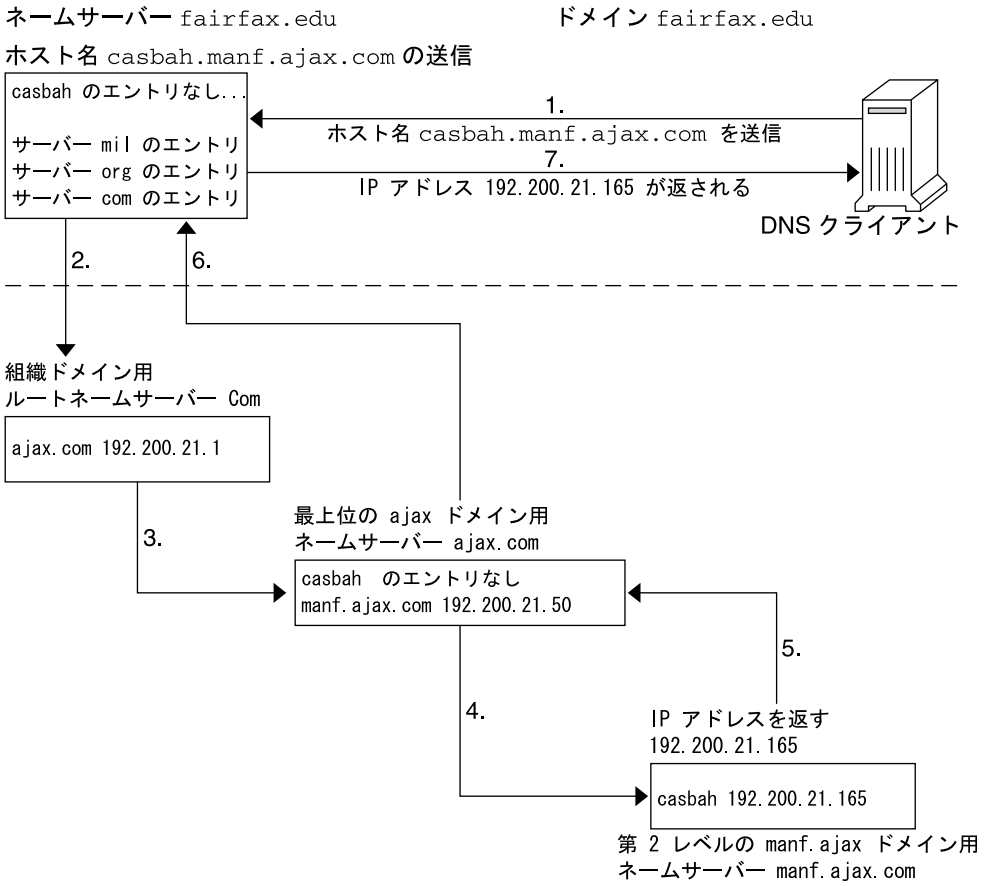


図 **3–2** 遠隔ホストに対する名前のアドレス解決

DNS 管理ドメイン

DNS から見ると「管理ドメイン」は 1 つの単位として管理されるマシン群を構成しています。このドメインに関する情報は、ドメインに対して権限を持つ 2 つ以上のネームサーバーによって管理されます。DNS ドメインはマシンを論理的にグループ分けしたものですが、小規模のビジネスにおいて全マシンが Ethernet に接続されている場合のように、ドメインのグループがマシンの物理的グループと対応している場合もあります。同様に、ローカルの DNS ドメインには、大学の大規模なネットワークのすべてのマシンが入っていることがあります。これらのマシンは、コンピュータサイエンスの学科や管理部門に属しています。

たとえば、Ajax という会社がサンフランシスコとシアトルの 2 つのサイト持っているとします。そして、`Retail.Sales.Ajax.com.` ドメインがシアトルにあり、`Wholesale.Sales.Ajax.com.` がサンフランシスコにあるとします。また、`Sales.Ajax.com.` ドメインが 2 つの市に分かれているとします。

各管理ドメインは、固有のサブドメイン名を持たなければなりません。さらに、ネットワークをインターネットに接続するのであれば、そのネットワークの属する管理ドメインはすでに登録されたものでなければなりません。ドメイン名とドメインの登録の詳細については、62 ページの「インターネットへの参加」の節を参照してください。

in.named と DNS ネームサーバー

すでに説明したように、管理ドメイン内のネームサーバーは、DNS データベースを保持しています。また、このネームサーバーは in.named デーモンを実行して DNS サービスを実装します。in.named は、パブリックドメインの TCP/IP プログラムであり、Solaris オペレーティング環境に含まれています。

注 – in.named デーモンは、カリフォルニア大学バークレー校で開発されたので、Berkeley Internet Name Domain service (BIND) と呼ばれることもあります。

DNS ネームサーバーには、次の 3 種類があります。

- マスターサーバー
- スレーブサーバー
- スタブサーバー

各ドメインには、マスターサーバーが 1 つと、バックアップ用のスレーブサーバーが 1 つ以上必要です。マスターサーバーとスレーブサーバーの詳細については、89 ページの「DNS の実装」を参照してください。

サーバーの構成とデータファイルの名前

in.named デーモンを正しく機能させるには、1 つの構成ファイルと 4 つのデータファイルが必要です。

構成ファイル

マスターサーバーの成ファイルは、/etc/named.conf . です。この構成ファイルには、ドメイン名と、ホスト情報を含むファイル名が記述されています。named.conf ファイルの詳細については、101 ページの「named.conf ファイル」を参照してください。

DNS データファイルの名前

内部で一貫性が取れていれば、ゾーンデータファイルには何でも好きな名前を付けることができます。名前の付け方への柔軟性が高いために、他のサイトで作業をしたり、DNS 間連の各種マニュアルを参照する場合に、混乱を招くことがあります。

たとえば、Sun のマニュアルや大多数の Solaris サイトで使われているファイル名は、『*DNS and BIND*』(Paul Albeltz & Criclcet Liu 著、浅羽登志也／上水流由香監訳 、アスキー出版局、1995年) で使われているファイル名とは異なります。そしてこれら 2 派の命名方法は、『*Name Server Operations Guide for BIND*』に記載されているハブリックドメインの命名方法とも若干の相違があります。

さらに、本書とその他の DNS 関連のマニュアルでは、説明にはファイルの主な役割を表す総称名を使い、コード例ではファイルに対して具体的な固有の名前を使っています。たとえば、Solaris のネームサービスに関するマニュアルでは、ファイルの機能や役割を説明する場合は `hosts` という総称名を使い、コード例では `db.doc` や `db.sales` といった名前を使っています。

必要なデータファイルは次のとおりです。

- `/var/named/named.ca`。`named.ca` ファイルの詳細については、103 ページの「`named.ca` ファイル」を参照してください。内部で一貫性が取れていれば、このファイルには任意の名前を付けることができます。
- `/var/named/hosts`。`hosts` ファイルの詳細については、106 ページの「`hosts` ファイル」を参照してください。

  `hosts` という名前はファイルの役割や内容を表す総称名です。ただし、この総称名をそのまま使うと `/etc/hosts` と紛らわしいので、この種のファイルは `hosts` 以外の名前にします。最も一般的な名前の例は、`db.`*domainname* です。たとえば、`doc.com` ドメインの `hosts` ファイルの名前は `db.doc` となります。

  ドメイン内に複数のゾーンがある場合は、各ゾーンに 1 つずつ `hosts` ファイルを置き、各ゾーンの `hosts` ファイルには一意の名前を付けなければなりません。たとえば、`doc.com` と `sales.doc.com` に分けられている DNS ドメインであれば、一方の `hosts` ファイルの名前は `db.doc`、もう一方の名前は `db.sales` とします。
- `/var/named/hosts.rev`。`hosts.rev` ファイルの詳細については、107 ページの「`hosts.rev` ファイル」を参照してください。

  `hosts.rev` という名前は、ファイルの役割や内容を表す総称名です。ドメイン内に複数のゾーンがある場合は、各ゾーンに 1 つずつ `hosts.rev` ファイルを置き、各ゾーンの `hosts.rev` ファイルには一意の名前を付けなければなりません。たとえば、DNS ドメイン内に `doc.com` と `sales.doc.com` という 2 つのゾーンがある場合は、1 つを `doc.rev`、もう 1 つを `sales.rev` という名前にするとよいでしょう。
- `/var/named/named.local`。`named.local` ファイルの詳細については、108 ページの「`named.local` ファイル」を参照してください。内部で一貫性が取れていれば、このファイルには任意の名前を付けることができます。

$INCLUDE ファイル

インクルードファイルは、DNS データファイルの中の $INCLUDE() 文で指定された任意のファイルです。$INCLUDE ファイルを使ってデータを型ごとに別々のファイルに分割しておくと便利です。109 ページの「$INCLUDE ファイル」を参照してください。

参考のため、次の表では上で述べた BIND ファイル名を比較します。

表 **3–1** ファイル名

Solaris	**O'Reilly** その他	カリフォルニア州立大学バークレイ校	ファイルの内容と役割
/etc/named.conf (全て同じファイル名)			BIND 8.1 では新たに named.conf ファイルを追加して、従来の named.boot ファイルと置きかえる。この構成ファイルにはセキュリティ、スタートアップオプション、ロギングが追加されている。このファイルによって、稼動中のサーバーの種類を指定して、すべてのゾーンまたはサーバーではなく、ゾーンごとまたはサーバーごとにオプションを選択的に適用する。ドメイン名とデータファイル名が記述されている
/etc/resolv.conf (全て同じファイル名)			各クライアント (DNS サーバーを含む) 上に存在するファイル。DNS 情報を探すためにクライアントが照会するサーバーを示す
named.ca	db.cache db.root	root.cache	ルートサーバー名とそのアドレスがリストされている
総称名: hosts 例:db.docdb.sales	総称名: db.domain 例:db.movie、 db.fx	総称名: hosts 例:ucbhosts	サーバーがサービスを提供するローカルゾーン内のマシンに関する全データが格納されている
総称名: hosts.rev 例:doc.rev	総称名: db.ADDR 例: db.192.249.249 db.192.249.253	hosts.rev	逆変換 (アドレスから名前への変換)を行うための特殊ドメイン in-addr.arpa. 内のゾーンを指定する
named.local	総称名: db.cache 例:db.127.0.0	named.local	ローカルループバックインタフェース (ローカルホスト) 用のアドレスを指定する
$INCLUDE ファイル (全てで同じ規則)			データファイル内の $INCLUDE() 文によって指定されるファイル

ドメイン名

「ドメイン名」とは、ローカルネットワークの中で DNS 管理ファイルを共有する複数のシステムを 1 つのグループとして扱って、そのグループに付けた名前のことです。ドメイン名は、ネットワーク情報サービスデータベースが正常に動作するために必要です。

デフォルトのドメイン名

DNS がデフォルトで使用するドメイン名は `resolv.conf` ファイルに指定されています。

- `resolv.conf` ファイルがない場合、あるいは `resolve.conf` ファイルにデフォルトのドメイン名が指定されていない場合で、しかも、エンタープライズレベルで使っているネームサービスが NIS+ または NIS のどちらかである場合は、Sun が実装する DNS はそれらのサービスからデフォルトのドメイン名を取得します。
- `resolv.conf` ファイルがない場合、あるいは `resolv.conf` ファイルにドメイン名が指定されていない場合で、しかも、NIS+ と NIS のどちらも使っていない場合は、そのドメインを指定するすべてのマシンに `resolv.conf` ファイルを追加するか、`LOCALDOMAIN` 環境変数を設定しなければなりません。

ドメイン名の末尾のドットについて

各種 DNS 関連ファイルを使用する場合、ドメイン名の末尾のドットには次のような規則があります。

- `hosts`、`hosts.rev`、`named.ca`、`named.local` の各データファイルの中では、ファイル名の末尾にドットを付けます。たとえば、`sales.doc.com` は、これらのファイルの中では有効です。
- `named.boot` ファイルまたは `resolv.conf` ファイルの中では、ドメイン名の末尾にドットを付けません。たとえば、`sales.doc.com` は、これらのファイルの中では有効です。

DNS クライアントとリゾルバ

あるマシンを DNS クライアントにするには、「リゾルバ」を実行する必要があります。リゾルバはデーモンでも単一のプログラムでもなく、アプリケーションによって使用される動的ライブラリルーチンの集合です。マシン名を知る必要があるときに、このライブラリが使用されます。リゾルバの機能はユーザーの照会を解決することで

す。このために、リゾルバはネームサーバーに照会します。するとネームサーバーは要求された情報か、または他のサーバーに照会する旨を返します。一度リゾルバを設定すれば、そのマシンはネームサーバーに DNS サービスを要求できるようになります。

DNS ネームサーバーは、いくつかのファイルを使用して、そのデータベースを読み込みます。リゾルバのレベルでは、必要な情報を取得できるサーバーのアドレスを登録するファイル (/etc/resolv.conf) が必要です。リゾルバは resolv.conf ファイルを読み取り、ローカルドメインの名前とネームサーバーの位置を見つけます。リゾルバはローカルドメイン名を設定し、リゾルバルーチンに指示して、登録されたネームサーバーに情報を照会させます。通常、ネットワーク上の各 DNS クライアントシステムの /etc ディレクトリには、resolv.conf ファイルがあります。クライアントに resolv.conf ファイルがない場合は、デフォルトにより IP アドレス 127.0.0.1 にあるサーバーが使用されます。

リゾルバがホストのアドレス (またはアドレスに対応する名前) を探さなければならないときには、照会パッケージを構築し、/etc/resolv.conf に登録されたネームサーバーにこれを送信します。サーバーは、その照会にローカルに応答するか、または他のサーバーのサービスを使ってリゾルバに答えを返します。

あるマシンの /etc/nsswitch.conf ファイルで hosts:dns (または hosts 行に dns を含む別のパターン) が指定されていると、リゾルバのライブラリが自動的に使用されます。nsswitch.conf ファイルが dns より前に、他のネームサービスを指定した場合、最初にそのネームサービスに対してホスト情報を問い合わせ、要求されたホストの情報が見つからなかった場合にだけリゾルバのライブラリが使用されます。

たとえば、nsswitch.conf ファイル内の hosts の行で hosts:nisplus dns と指定されている場合、ホストの情報を得るために NIS+ ネームサービスが最初に検索されます。NIS+ で情報が見つからない場合は、DNS リゾルバが使用されます。NIS+ や NIS といったネームサービスではそれ自身のネットワーク内にあるホストに関する情報だけを保持しているため、スイッチファイルに hosts:nisplus dns 行を指定すると、ローカルホストの情報に対しては NIS+ が使用され、インターネット上の遠隔ホストの情報に対しては DNS が使用されることになります。

DNS クライアントには、次の 2 種類があります。

- クライアント専用

  クライアント専用の DNS クライアントは in.named を実行せず、代わりにリゾルバを参照します。リゾルバはドメインに対するネームサーバーのリストを保持しているので、リゾルバに問い合わせが転送されます。

- クライアント兼サーバー

  クライアント兼サーバーの DNS クライアントは、クライアントマシンのリゾルバによって転送されてきた問い合わせを解決するために in.named で提供されるサービスを使用します。

resolv.conf ファイル

resolv.conf ファイルの機能については、resolv.conf(4) のマニュアルページを参照してください。

resolv.conf ファイルの設定方法については、67 ページの「resolv.conf ファイルの設定」を参照してください。

named.conf ファイル

BIND 8.1 では、新たに構成ファイル /etc/named.conf を追加して、/etc/named.boot ファイルと置き換えました。/etc/named.conf ファイルは、マスター、スレーブ、キャッシュ専用のネームサーバーを確立します。また、サーバーが権限を持つゾーンを指定し、どのデータファイルから初期データを取得するかを指定します。

/etc/named.conf ファイルには、次の機能を実装するための文が含まれます。

- アクセス制御リスト(ACL) によるセキュリティ。このリストには、NIS+ ホストが読み取り権と書き込み権を持っている IP アドレスの集まりが定義されている
- ロギング動作の指定
- すべてのゾーンにではなく、ゾーンのセットに対して選択的に適用されるオプション

この構成ファイルは、サーバーの起動スクリプト /etc/init.d/inetsvc によってデーモンが起動されるとき、 in.named によって読み取られます。そして、in.named が他のサーバーや、指定されたドメインのローカルデータファイルに対して実行されるようにします。

named.conf ファイルは、いくつかの文とコメントで構成されています。文はセミコロンで終わります。一部の文には、文のブロックを記述することができます。ブロック内の各文もセミコロンで終わります。

表 **3–2** named.conf で使用する文

acl	アクセス制御に使用する、IP アドレスの一致リストを名前を付けて定義する。アドレスの一致リストは、1 つ以上の IP アドレス (ドット形式の 10 進表記) または IP 接頭辞 (ドット形式の 10 進表記の後にスラッシュとネットマスクのビット数が付く) を示す。名前を付けたIP アドレスの一致リストは、他の場所で使用する前に acl 文で定義されている必要がある。前方参照は不可
include	include 文がある箇所にインクルードファイルを挿入する。include を使用することで、より管理しやすいまとまりに構成情報を分割することができる
key	特定のネームサーバーでの認証と承認に使用される鍵の ID を指定する。server 文を参照
logging	サーバーが記録するログの種類とログメッセージの送り先を指定する
options	グローバルなサーバー構成のオプションを制御して、他の文に対するデフォルト値を設定する
server	遠隔ネームサーバーに関して、指定された構成オプションを設定する。すべてのサーバーに対してではなく、サーバーごとに選択的にオプションを適用する
zone	ゾーンを定義する。すべてのゾーンに対してではなく、ゾーンごとに選択的にオプションを適用する

例 **3–1** マスターサーバー用のマスター構成ファイルの例

```
options {
         directory "/var/named";
         datasize 2098;
         forward only;
         forwarders {
                  99.11.33.44;
         };
         recursion no;
         transfers-in 10;
         transfers-per-ns 2;
         allow-transfer {
                  127.0.1.1/24;
         };
};

logging {
         category queries { default_syslog; };
};

include "/var/named/abcZones.conf"


// マスターファイルの名前
zone "cities.zn" {
         type master;
```

例 **3–1** マスターサーバー用のマスター構成ファイルの例　(続き)

```
         file "db.cities.zn";
};

zone "0.0.127.in-addr.arpa" {
         type master;
         file "db.127.cities.zn";
};

zone "168.192.in-addr.arpa" {
         type master;
         file "db.cities.zn.rev";
};

zone "sales.doc.com" {
         type slave;
         file "slave/db.sales.doc";
         masters {
                  192.168.1.151;
         };
};


zone "168.192.in-addr.arpa" {
             type slave;
         file "slave/db.sales.doc.rev";
         masters {
                  192.168.1.151;
         };
};
```

ローカルドメイン内の DNS 階層

大規模な会社であれば、多くのドメインをサポートしており、ローカルな名前空間が構成されていることでしょう。次の図に、ある会社に存在するドメインの階層構造の例を示します。この会社の最上位のドメイン、すなわちルートドメインは `ajax.com` で、その下に `sales.ajax.com`、`test.ajax.com`、`manf.ajax.com` の 3 つのサブドメインがあります。

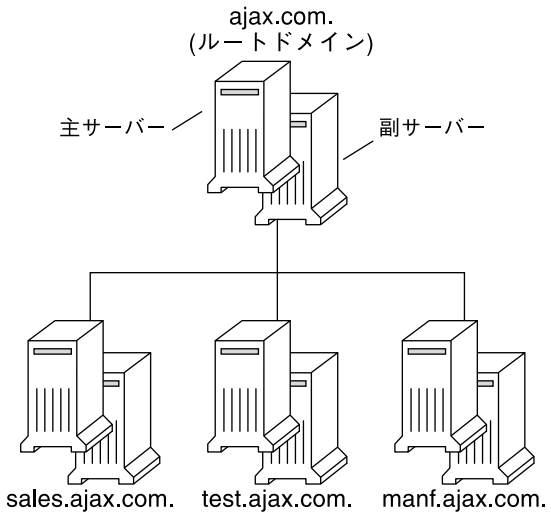


図 **3–3** ある組織での DNS ドメインの階層

DNS クライアントは、そのドメインをサポートしているサーバーだけにサービスを要求します。クライアントの必要としている情報がそのドメインサーバーにない場合、要求は親サーバーに転送されます。親サーバーは、1 つ上の階層のドメインサーバーです。要求が最上位のサーバーに達した場合、最上位のサーバーはクライアントのドメインが有効かどうか調べます。ドメインが有効でない場合、サーバーは「not found」というメッセージをクライアントに返します。ドメインが有効な場合、最上位のサーバーはそのドメインをサポートしているサーバーに要求を転送します。

DNS 階層とインターネット

次の図に示したドメインの階層は、グローバルなインターネット上でサポートされる巨大な DNS 名前空間の「枝葉」のようなものです。

それは、ピリオド (.) で表されるルートディレクトリと、最上位のドメインの階層 2 つ (組織的なものと地理的なもの) で構成されます。次の図の com ドメインは、インターネットに存在する最上位の組織ドメインの 1 つです。

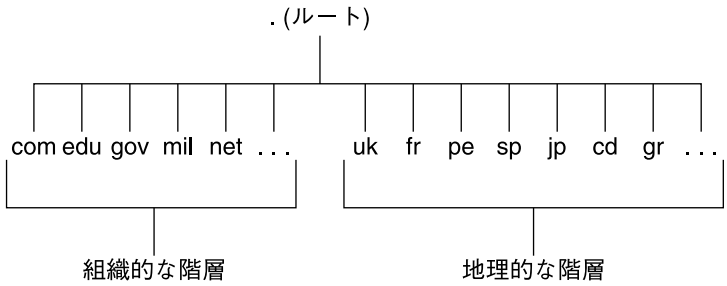


図 **3–4** インターネットのドメインの階層

現在、組織的な階層の名前空間は、次の表に示した最上位のドメインに分けられます。将来、これ以外にも最上位の組織ドメインが追加される可能性があります。

表 **3–3** インターネットの組織ドメイン

ドメイン	目的
com	営利団体
edu	教育機関
gov	行政機関
mil	軍事組織
net	ネットワークサポートセンター
org	非営利団体
int	国際組織

地理的な階層においては、各国に対して 2、3 文字の識別子が割り当てられ、それが、各国に対する公式名として用いられます。たとえば、イギリス国内のドメインは、uk という最上位のドメインのサブドメインになります。日本国内のドメインは、jp のサブドメインで、他の国に関しても同様です。

インターネットへの参加

インターネットのルートドメイン、すなわち最上位のドメイン (組織的および地理的) は、様々なインターネット運営体によって管理されています。規模にかかわらずネットワークを有する人は、そのネットワークのドメイン名を組織的な階層か地理的な階層に登録することによってインターネットに参加できます。

DNS ドメインはドメイン名を持つ必要があります。インターネットに接続しないで、ネームサービスとして DNS を使用する場合は、ドメインやサブドメインに任意の名前を付けることができます。ただし、インターネットに参加する場合は、そのドメイン名をインターネット運営体に登録する必要があります。

インターネットに参加する場合は、次の手順に従います。

- DNS ドメイン名を、適切なインターネット運営体に登録する
- そのインターネット運営体から、ネットワークの IP アドレスを入手する

上記のことを行うには、次の 2 つの方法があります。

- 適切なインターネット運営体またはその代理団体と直接連絡をとる方法
- インターネットサービスプロバイダ (ISP) と契約する方法。ISP は、コンサルティングから実際の接続まで広範なサービスを提供している

ドメイン名

ドメイン名は、DNS 名前空間全体でのドメインの位置を表します。それは UNIX のファイルシステムでパス名がファイルの位置を表しているのと同じです。ローカルドメインが登録されると、そのドメイン名は所属するインターネットの階層の名前に追加されます。たとえば、図 3–5 で示した Ajax ドメインはインターネットの階層である com の一部として登録されました。したがって、そのインターネットドメイン名は `ajax.com` となります。

次の図に、`ajax.com` ドメインがインターネット上の DNS 名前空間のどこに位置しているのかを示します。

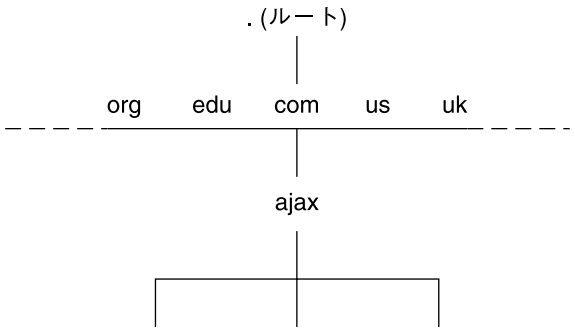


図 **3–5** DNS 名前空間における Ajax ドメインの位置

`ajax.com` のサブドメインには、次のような名前があります。

```
sales.ajax.com
test.ajax.com
 manf.ajax.com
```

DNS ではドメイン名を大文字にすることは可能ですが、必須ではありません。次に、マシン名とドメイン名の例を挙げます。

```
boss.manf.ajax.com
quota.sales.ajax.com
```

インターネットでのドメインの管理は、ドメイン内のホスト名に対する権限を各ドメインに保証し、各ドメインがそれより下位のレベルに権限を委任することによって行われます。したがって、com ドメインはそのドメイン内のホスト名に対して権限があります。また com ドメインは、ajax.com ドメインに対して編成の権限を与え、そのドメイン内の名前に対する権限を委任します。それを受けて ajax.com はドメイン内のホストに名前を割り当て、sales.ajax.com、test.ajax.com、manf.ajax.com の各ドメインの編成を承認します。

完全指定ドメイン名 *(FQDN)*

ドメイン名にローカルドメインから DNS のルートドメイン (.) までのすべての DNS のドメインが含まれているとき、そのドメイン名は「完全指定されている」といいます。概念的には、完全指定ドメイン名は、UNIX ファイルの絶対パス名と同様に、ルートへのパスを示しています。しかし、完全指定ドメイン名を読む場合、左から右に進むにしたがって最下位から最上位となります。したがって、完全指定ドメイン名は次のような構文になっています。

local_domain_name>.<Internet_Org_name>.

ルートドメイン

ajax ドメインとそのサブドメインの完全指定ドメイン名は、次のとおりです。

```
ajax.com.
sales.ajax.com.
test.ajax.com.
manf.ajax.com.
```

ここで、各名前の一番右に付けられたドット (.) に注意してください。

ゾーン

ドメインの DNS サービスは、ネームサーバーの集合で管理されます。ネームサーバーは、単一のドメインまたは複数のドメイン、あるいは複数のドメインとその下のサブドメインの一部または全部を管理できます。あるネームサーバーによって管理される名前空間の一部は、「ゾーン」と呼ばれます。したがって、ネームサーバーはゾーンに対して権限があるといわれます。ネームサーバーの責任者は、「ゾーン管理者」とも呼ばれます。

ネームサーバーのデータベース内のデータは、「ゾーンファイル」と呼ばれます。ゾーンファイルの 1 つには、IP アドレスとホスト名が格納されています。ftp または telnet のようなユーティリティでホスト名を用いて遠隔ホストに接続しようとすると、DNS は名前のアドレスマッピングを実行し、ゾーンファイルの中でホスト名を探して、IP アドレスに変換します。

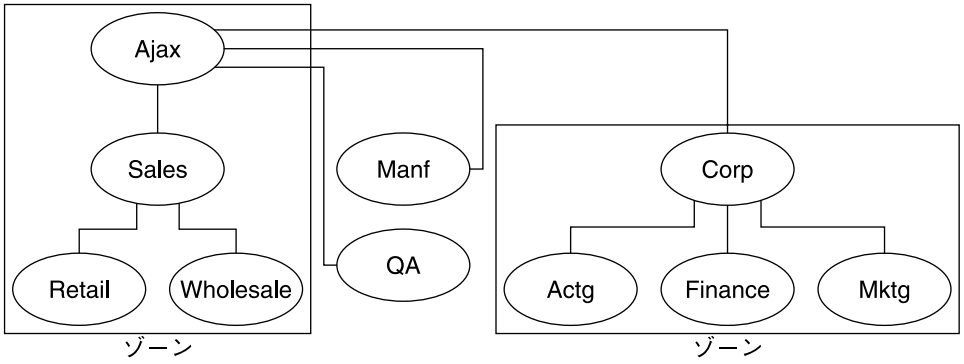


図 **3–6** ドメインとゾーン

たとえば、上の図で示した Ajax ドメインは、最上位のドメイン (Ajax) 、4 つのサブドメイン、そして 5 つのサブサブドメインから構成されます。このドメインは、4 つのゾーンに分けられます。そして、Ajax ネームサーバーは、Ajax 、Sales 、Retail 、Wholesale の各ドメインからなるゾーンを管理します。Manf と QA の両ドメインは別のゾーンで、自分自身のネームサーバーからサービスを受けます。Corp ネームサーバーは Corp、Actg、Finance、Mktg ドメインからなるゾーンを管理します。

逆マッピング

DNS データベースには、マシンのホスト名を見つけだすキーとして IP アドレスを用いるゾーンファイルもあります。このファイルで IP アドレスのホスト名解決が可能になります。このプロセスを、「逆解決」または、より一般的には「逆マッピング」と呼びます。逆マッピングは基本的にはメッセージを送ってきたマシンの識別情報を確認したり、ローカルホスト上での遠隔操作を許可したりするために使用されます。

in-addr.arpa ドメイン

in-addr.arpa ドメインは、DNS 名前空間において概念的な存在であり、認証 (許可) のためにドメインでなく IP アドレスを用います。このドメインは、ゾーンの一部ですが、これによってアドレスから名前のマッピングが可能になります。

DNS ドメイン名は、一番左が最下位のサブドメイン、一番右がルートとして読まれるので、in-addr.arpa ドメインでも、IP アドレスは最下位のレベルからルートになるように理解されます。すなわち、IP アドレスは逆転します。たとえば、あるホスト

の IP アドレスが 192.168.21.165 であるとします。in-addr.arpa ゾーンファイルでは、そのアドレスは 165.21.200.168.in-addr.arpa. と記述されます。ここで最後のドットは、in-addr.arpa ドメインのルートを示しています。

第 4 章

DNS の管理 (手順)

この章では、DNS (Domain Name System) の管理方法について説明します。

この章の内容は次のとおりです。

resolv.conf ファイルの設定

ここでは、doc.com ドメインのサーバーで使用する簡単な resolv.conf(4) ファイルの例を示します。

例 **4–1** DNS サーバー用 resolv.conf ファイルの例

```
;
; /etc/resolv.conf file for dnsmaster (sirius)
;
domain             doc.com
nameserver         192.168.0.0
nameserver         192.168.0.1
```

このファイルの最初の行では、ドメイン名を次の書式で指定します。

`domain` *domainname*

ここで、*domainname* は インターネット管理組織（このドキュメントの執筆時点では InterNIC。日本では JPNIC) に登録されている名前です。

注 – ドメイン名の末尾にスペースまたはタブを使うことはできません。ドメイン名の最後の文字を入力したら、必ずキャリッジリターンで強制改行してください。

---

2 行目には、サーバー自体を次の書式で指定します。

```
nameserver 192.168.0.0
```

それ以降の行では、スレーブ DNS ネームサーバーまたはキャッシュ専用ネームサーバー の IP アドレスを 1 つまたは 2 つ指定します。リゾルバはこれらの行を照会して該当するアドレスを識別します。各行の書式は次のとおりです。

`nameserver` *IP_address*

IP_address には、スレーブ DNS ネームサーバーまたはキャッシュ専用ネームサーバーの IP アドレスを指定します。リゾルバは、必要な情報が見つかるまで、ここに指定されている順番どおりにネームサーバーを探していきます。

DNS 用ネットワークの構成

DNS 用のネットワークを構成する場合は、クライアントとサーバーを設定する必要があります。

DNS クライアントの設定

DNS サーバーを設定する前に、クライアントを設定します。

▼ DNS クライアントを設定する方法

1. `/etc/resolv.conf` ファイルを作成します。

   次に、`doc.com` ドメインのクライアント (非サーバー) マシン用の簡単な `resolv.conf` ファイルの例を示します。

   例 **4–2** `resolv.conf` ファイルの例

   ```
   ; Sample resolv.conf file for the machine polaris
   domain doc.com
   ; try local name server
   ```

例 **4–2** resolv.conf ファイルの例　（続き）

```
nameserver 10.0.0.1
; if local name server down, try these servers
nameserver 192.168.16.6
nameserver 192.168.16.7
; sort the addresses returned by gethostbyname(3c)
sortlist
130.155.160.0/255.255.240.0
130.155.0.0
```

/etc/resolv.conf ファイルの最初の行では、ドメイン名を次の書式で指定します。

`domain` *domainname*

ここで、*domainname* は インターネット管理組織（このドキュメントの執筆時点では InterNIC。日本では JPNIC) に登録されている名前です。

注 – ドメイン名の末尾にスペースまたはタブを使うことはできません。ドメイン名の最後の文字を入力したら、必ずキャリッジリターンで強制改行してください。

---

2 行目では、ループバックネームサーバーを次の書式で指定します。

`nameserver 10.0.0.1`

それ以降の行では、DNS マスターネームサーバー、DNS スレーブネームサーバー、またはキャッシュ専用ネームサーバー の IP アドレスを最大 3 つまで指定します。4 つ以上指定することはできません。各行の書式は次のとおりです。

`nameserver` *IP_address*

IP_address には、マスターDNS ネームサーバーまたはスレーブ DNS ネームサーバーの IP アドレスを指定します。リゾルバは、必要な情報が見つかるまで、ここに指定されている順番どおりにネームサーバーを探していきます。

/etc/resolv.conf ファイルの 5 行目では、アドレス `sortlist` を次の書式で指定します。

`sortlist`
addresslist

addresslist は、`gethostbyname(3c)` によって戻されるアドレスのソート順序を示します。上記の列では、`gethostbyname` は、IP アドレス `130.155.0.0` より先に 1 組のネットマスク `130.155.160.0/ 255.255.240.0` を戻します。

2. /etc/nsswitch.conf ファイルを変更します。

「NIS」。エンタープライズレベルで主として使っているネームサービスが NIS で、設定に問題がない場合、DNS はすでに使用可能になっています。

「ファイルベース」。エンタープライズレベルで主として使っているネームサービスが /etc ファイルベース、または NIS+ の場合は、次の手順に従います。

a. スーパーユーザーになります。

b. /etc/nsswitch.conf ファイルを開きます。

c. **DNS** は、ホスト情報のソースとして、「唯一の」ソースとしても「追加の」ソースとしても使用できます。/etc/nsswitch.conf の hosts 行を次のいずれかのように変更して、DNS をホスト情報のソースとすることを指定します。

```
hosts: files dns
```

または

```
hosts: nis dns [NOTFOUND=return] files
```

または

```
hosts: dns nis [NOTFOUND=return] files
```

NIS クライアントの場合は、上記の指定をしないでください。この指定をすると、名前を見つけることができない場合に 2 度 DNS から検索することになります。

d. ホスト情報のソースとして **DNS** を指定します。

e. ファイルを保存してリブートします。

DNS サーバーの設定

▼ DNS サーバーを設定する方法

1. スーパーユーザーになります。

2. サーバーを **DNS** クライアントとして設定します **(**サーバーの resolv.conf ファイルの設定も含む**)**。詳細については、**68** ページの「**DNS** クライアントの設定」を参照してください。

3. 起動ファイルを設定します。詳細については、**90** ページの「起動ファイルの例」を参照してください。

4. データファイルを設定します。次の **4** つのデータファイルを設定する必要があります。

- named.ca
- hosts
- hosts.rev
- named.local

5. サーバーの初期設定を行います。詳細については、**75** ページの「サーバーの初期設定」を参照してください。

6. サーバーをテストします。詳細については、**76** ページの「インストール結果の確認」を参照してください。

注 – DNS の最も一般的な役割は、ローカルなネットワークをインターネットに接続することです。インターネットに接続するためには、親ドメインの管理者にネットワークの IP アドレスを登録してもらう必要があります。管理者は、ネットワークの地理的な位置と親ドメインの種類によって異なります。ドメイン管理者にネットワークを登録してもらう方法については、本書では説明をしていません。

マスターサーバーを指定する方法

マスターサーバーには、次の 2 つの種類があります。

- 「ゾーンマスターサーバー」。各ゾーンには、そのゾーンの「マスター」マスターサーバーが 1 つあります。ゾーンの「マスター」マスターサーバーは、そのゾーンの「正規」サーバーです。
- 「ゾーンスレーブサーバー」。 ゾーンには、1 つ以上の「スレーブ」マスターサーバーがあります。「スレーブ」マスターサーバーは、その DNS データをゾーンのマスターサーバーから入手します。

サーバーをある特定のゾーンのマスターサーバーに指定する場合は、そのサーバーの `named.boot` ファイルに 3 つのマスターレコードを作成します。

1. ゾーンの「マスター」レコードを作成します。

   このレコードは、そのサーバーをマスターサーバーとして使用するゾーンを指定し、正規の `hosts` ファイルの場所を示すものです。この「マスター」レコードは、次の 3 つのフィールドで構成されます。

   - 第 1 フィールド - サーバーを「マスター」サーバーとして指定する
   - 第 2 フィールド - 「マスター」サーバーが機能するゾーンを指定する
   - 第 3 フィールド - `hosts` ファイルを指定する

   次に示す起動ファイルの行は、あるサーバーを `doc.com` ゾーンで「マスター」サーバーとして使い、正規の `hosts` ファイルとして `db.doc` を使うことを示すものです。

   ```
   master    doc.com    db.doc
   ```

2. ゾーンの逆マッピング用の「マスター」レコードを作成します。

   このレコードは、そのサーバーを逆アドレスマッピング (つまり、`doc.com` の逆アドレスドメイン) の「マスター」サーバーとして使うことを指定し、正規の `hosts` ファイルの場所を示すものです。この「マスター」レコードは、次の 3 つのフィールドで構成されます。第 1 フィールドではサーバーを「マスター」サーバーとして指定します。第 2 フィールドでは対象のゾーンを指定します。第 3 フィールドでは `hosts.rev` ファイルを指定します。

   あるゾーンにおける逆アドレスドメインは、そのゾーンにおける IP アドレスを逆にならべ、最後に `in-addr.arpa` を配したものです。たとえば、`doc.com` ゾーンの IP アドレスが `10.0.0.` だとすると、逆アドレスドメインは `0.0.10.in-addr.arpa` になります。

次に示す起動ファイルの行は、そのサーバーを doc.com ゾーンの逆アドレスドメインで「マスター」サーバーとして使い、正規の hosts ファイルとして doc.rev を使うことを示すものです。

```
master   0.0.10.in-addr.arpa    doc.rev
```

3. ローカルループバックインタフェースまたはホストの逆アドレス関連の「マスター」レコードを作成します。

   このレコードは、そのサーバーをループバックホストの「マスター」サーバーとして使うことを指定し、正規の hosts ファイルの場所を示すものです。この「マスター」レコードは、3 つのフィールドで構成されます。第 1 フィールドではサーバーを「マスター」サーバーとして指定します。第 2 フィールドではループバックホストの逆アドレスを指定します。第 3 フィールドでは hosts ファイルを指定します。

注 – ループバックホストは常に、0.0.10.in-addr.arpa といった書式で識別されます。

次に示す起動ファイルの行は、そのサーバーをループバックホストの逆アドレスドメインで「マスター」サーバーとして使い、正規の hosts ファイルとして named.local を使うことを示すものです。

```
master  0.0.10.in-addr.arpa   named.local
```

スレーブサーバーを指定する方法

「スレーブ」サーバーは、ゾーンに関するデータのコピーを保持しています。 マスターサーバーはそのデータをスレーブサーバーに送り、権限を任せます。クライアントは、DNS 情報をスレーブサーバーに照会できます。スレーブサーバーを使用することによって、負荷が複数のマシンに分散され応答時間を短縮してネットワークのオーバーヘッドを減らすことができます。また、スレーブサーバーは、マスターサーバーが使用できないときに代わりの機能を果たします。

スレーブサーバーは、in.named の起動時に所定のゾーンに関するすべてのデータをマスターサーバーに要求します。以降、スレーブサーバーはデータベースを更新する必要があるかどうかを調べるためにマスターサーバーを定期的にチェックします。最新のゾーンデータベースをマスターサーバーからスレーブサーバーに送信するプロセスを「ゾーン転送」と呼びます。このため、スレーブサーバー上のデータファイルを変更するのではなく、ゾーンのマスターサーバー上のデータファイルを変更します。その後、スレーブサーバーのファイルがマスターサーバーから更新されます。

サーバーを所定のゾーンのスレーブサーバーに指定する場合は、そのサーバーの named.boot ファイルに「スレーブ」レコードを作成します。別々のレコードで、サーバーをそのゾーン、逆アドレスドメイン、およびループバックホストのスレーブサーバーとして指定できます。

この「スレーブ」レコードは、次の3 つのフィールドで構成されます。

- 第 1 フィールド - サーバーを「スレーブ」サーバーとして指定する
- 第 2 フィールド - 対象のゾーンを指定する
- 第 3 フィールド - そのゾーンのマスターサーバーの IP アドレスを指定する。スレーブサーバーはマスターサーバーから正規データを取得する。

「スレーブ」レコードでは、必須フィールドに続けて 1 つまたは複数の任意指定のフィールドを設けることができます。任意指定のフィールドには次の種類があります。

- 「スレーブサーバー」

  マスターサーバーの IP アドレスに続けて、他のスレーブサーバーの IP アドレスを指定できます。この指定により、スレーブサーバーが情報を入手できるソースが増えます。一方、状況によっては、スレーブサーバーの IP アドレスを指定することで、パフォーマンスが低下することも考えられます (IP アドレスがマルチホームマスターサーバーの別のネットワークアドレスである場合を除く)。

- 「バックアップファイル」

  マスターサーバー (および任意指定のスレーブサーバー) の IP アドレスに続けて、バックアップ用 `hosts` ファイルの名前を指定できます。バックアップファイル名を指定すると、スレーブサーバーはそのバックアップファイルからデータをロードし、続いてマスターサーバー (および任意指定のスレーブサーバー) をチェックしてバックアップファイルのデータが最新のものであるかどうかを確認します。その結果、最新ではないことが分かった場合は、マスターサーバーから受け取った情報に基づいてバックアップファイルのデータが更新されます。

次に示す起動ファイルの行は、あるサーバーを `doc.com` ゾーンとその逆アドレスドメインのスレーブサーバーとして使うことを示します。さらに、そのスレーブサーバーが IP アドレス `172.16.0.1` のマスターサーバーから正規データを受け取り、サーバー `172.16.0.2` をゾーンデータのスレーブ情報源として使い、最初に `doc.com.bakup` ファイルからデータをロードすることを示します。

```
slave   doc.com   129.146.168.119  192.146.168.38  doc.com.bakup
slave   4.0.32.128.in-addr.arpa       129.146.168.119
```

この章で紹介するさまざまなサンプルファイルの中で、上記のサンプル起動ファイル行はサーバー `dnsslave` (IP アドレス `192.146.168.38` の `sirius` マシンのエイリアス) の起動ファイルに対応しています。

注 – 1 台のサーバーは、1 つまたは複数のゾーンのマスターサーバーとして機能でき、さらに 1 つまたは複数のゾーンのスレーブサーバーとしても機能できます。起動ファイル内のエントリの組み合わせによって、サーバーがマスターサーバーになるかスレーブサーバーになるかが決まります。

キャッシュ専用 (スタブ) サーバーを指定する方法

DNS データのキャッシュを保持するという意味では、すべてのサーバーがキャッシュサーバーであるといえます。キャッシュ専用 (スタブ) サーバーは、in-addr.arpa. ドメイン以外のどのゾーンのマスターサーバーでもないサーバーです。

キャッシュ専用サーバーは正規データは一切保持しません。キャッシュ専用サーバーは照会を行い、in.named ファイルにリストされているホストを照会して必要な情報を探します。つまり、キャッシュ専用サーバーは、正規のネームサーバーと同様の照会を行いますが、正規データそのものは一切保持しません。

次に、キャッシュ専用サーバーの起動ファイルの例を示します。

例 **4–3** キャッシュ専用サーバーの起動ファイルの例

```
;
; Sample named.boot file for caching-only name server
;
; type                   domain                  source file or host
;
directory /var/named
cache                    .                       named.ca
master                  0.0.127.in-addr.arpa    named.local
```

サーバーをキャッシュ専用サーバーとして指定するための行は特に必要ありません。起動ファイル内に slave または master など、権限に関する行がないということが、キャッシュ専用サーバーであると判断する根拠になります。

ただし、以下が必要です。

- 起動ファイルの directory 行
- 起動ファイルの master 0.0.127.in-addr.arpa 行
- ブートファイルの cache. named.ca 行

+/- 構文との DNS 互換性を追加する方法

この節では、マスターネームサービスとして NIS または NIS+ を使用する場合に、/etc/passwd、/etc/shadow、/etc/group の各ファイルで使用される +/- 構文との互換性を確保する方法について説明します。

1. スーパーユーザーになります。
2. /etc/nsswitch.conf ファイルをオープンします。
3. passwd と groups の各ソースを compat に変更します。

- NIS を使う場合は次のように入力します。

```
passwd: compat
group: compat
```

- NIS+ を使う場合は次のように入力します。

```
passwd: compat
passwd_compat: nisplus
group: compat
group_compat: nisplus
```

これにより Solaris 1.x リリースと同じ構文を使用できます。ファイル内の +/- エントリに従って、/etc ファイルと NIS マップ (または NIS+ テーブル) を検索します。

4. -+ または -+ netgroup を /etc/passwd、/etc/shadow、/etc/group の各ファイルに追加します。

注 – -+ または -+ netgroup のエントリを /etc/shadow および /etc/passwd に追加できないと、ローカルファイルに登録されていないユーザーは以後ログインできなくなります。

---

5. ファイルを保存して、システムをリブートします。
   ライブラリ関数には nsswitch.conf ファイルが変更されてもスイッチ情報を読み直さないものがあります。そのため、マシンをリブートして、nscd とこれらのライブラリ関数が最新スイッチの情報を持つようにする必要があります。

DNS サーバーの設定

サーバーの初期設定

サーバーの初期設定を行う場合は、次の手順に従います。

▼ サーバーの初期設定を行う方法

1. スーパーユーザーになります。
2. 前出の節の説明に従って、named.conf 構成ファイルとその他必要なファイルをインストールします。
3. in.named を実行します。

#/usr/sbin/in.named

コマンド行から in.named を実行する代わりに、リブートするという方法もあります。

インストール結果の確認

起動ファイルとデータファイルを設定し、in.named を実行したら、インストールが正しく行われたかどうかを確認してください。

▼ インストール結果を確認する方法

1. スーパーユーザーになります。
2. syslog ファイルをオープンして、エラーメッセージが書き込まれていないかどうか確認します。

   一般的な DNS エラーメッセージと障害追跡情報については、第 6 章を参照してください。
3. nslookup コマンドを使用して、ローカルドメインのホスト名を確認します。

```
dnsmaster% nslookup altair
 Server:  dnsmaster.doc.com
 Address: 192.146.168.5
 Name:  altair.doc.com
 Address: 192.146.168.10
```

   - ルックアップが正常に実行できれば、ネームサーバーは正常に機能していると推定されます。
   - 「Can't find」または「can't initialize address」といったメッセージがサーバーに表示された場合、あるいは「Non-existent domain」といったメッセージが表示された場合は、サーバーが起動ファイルまたはホストファイルに正しく設定されていない可能性があります。
   - 「can't find *name*」または「non-existent domain」といったメッセージがサーバーに表示された場合は、検索したサーバーがサーバーの hosts ファイルに書き込まれていないか、resolv.conf ファイルのドメイン情報に誤りがある、あるいは、それ以外のサーバーの問題がある可能性があります。
4. nslookup を実行して遠隔ドメイン名を検索します。

   インターネットに接続されているネットワークの場合、遠隔ドメイン名を検索します。インターネットに接続されていないネットワークの場合は、他のゾーンにサブドメインがあれば、その名前を検索します。

   たとえば、インターネット上の遠隔ドメイン名 internic.net を検索するには、次のように入力します。

   dnsmaster% **nslookup internic.net**

```
Server:  dnsmaster.doc.com
Address: 192.168.168.
```

```
Name:  internic.net
 Addresses: 192.168.0.9,  192.168.0.6,  192.168.0.5,  192.168.0.8
```

- 正常に実行できれば、ネームサーバーはおそらく正常に機能しています。
- 上記のコマンドを実行しても遠隔ドメイン名が表示されない場合は、インターネットとの接続に問題があることが原因の 1 つとして考えられます。
- あるいは、named.ca ファイルが正しくインストールまたは設定されていないことも考えられます。

もう一度 nslookup を使用してドメインを検索すると、「non-authoritative」というメッセージが出るはずですが、これは無視してかまいません。2 回目の実行では、遠隔ネームサーバーからではなく、キャッシュから応答が来ているからです。

5. 遠隔ドメインから自分のドメインのホスト名を検索します。

   インターネットに接続されているネットワークの場合、遠隔ドメインから自分のドメインのホスト名を検索します。インターネットに接続されていないネットワークの場合は他のゾーンがあれば、そこから自分のドメインのホスト名を検索します。

   たとえば、インターネット上の遠隔ドメインから自分のドメインにあるホスト名を検索するには、nslookup コマンドに続けて、引数を 2 つ指定します。1 つめの引数は検索対象のホスト名、2 つめの引数は nslookup コマンドを実行するネームサーバー名です。

   remotemachine9% **nslookup altair remotemaster.foo.org.**

```
Server:  remotemaster.foo.org
 Address: 192.168.0.1
 Name:  altair.doc.com
 Addresses: 192.168.1.2
```

- 正常に実行できれば、ネームサーバーはおそらく正常に機能しています。
- 上記のコマンドを実行しても探しているマシンが見つからない場合は、ドメインが親ドメイン (上記の例では .com) の管理元に正しく登録されていないことが原因の 1 つとして考えられます。

サーバーの追加

ネットワークにマスターサーバーやスレーブサーバーを追加できます。

▼ サーバーを追加する方法

1. スーパーユーザーになります。
2. **DNS** クライアントとしてサーバーを設定します。**79** ページの「クライアントの追加」を参照してください。
3. 次のファイルを設定します。

   起動ファイル

named.ca
hosts
hosts.rev
named.local

詳細については、75 ページの「DNS サーバーの設定」を参照してください。

DNS データファイルの変更

DNS マスターサーバー内の DNS データファイルの 1 つに対して、ホストの追加および削除、またはそれ以外の何らかの変更、あるいは DNS データファイルの修正を行なったときには、以下も行なってください。

- スレーブサーバーがそのデータを変更するように、SOA リソースレコードのシリアル番号を変更します (78 ページの「SOA のシリアル番号を変更する方法」を参照)。
- データファイルを再度読み込んで内部データベースを更新するようにマスターサーバーの in.named に情報を与えます (79 ページの「in.named に DNS データを強制的に再度読み込ませる」 を参照)。

SOA のシリアル番号を変更する方法

すべての DNS データベースファイルには権限の開始 (SOA) リソースレコードがあります。DNS データベースのデータを変更したときは必ず、SOA シリアル番号を 1 増加させる必要があります。

たとえば、データファイルの SOA のシリアル番号が現在 101 で、ファイルのデータに変更を加えた場合は、シリアル番号を 101 から 102 に変更する必要があります。SOA のシリアル番号を変更しないと、ドメインのスレーブサーバーは、新しい情報でデータベースファイルのコピーを更新しません。その結果、マスターサーバーとスレーブサーバーが同期しなくなります。

hosts ファイル例の一般的な SOA レコードは、以下のとおりです。

```
; sample  hosts  file
@ IN       SOA     nismaster.doc.com. root.nismaster.doc.com. (
   109 ; Serial
   10800 ; Refresh
   1800 ; Retry
   3600000 ; Expire
   86400 ) ; Minimum
```

したがって、hosts ファイルを変更すると、シリアル番号は 109 から 110 に変更して、次にファイルを変更した場合、110 から 111 に変更します。

in.named に DNS データを強制的に再度読み込ませる

in.named が無事起動すると、デーモンはそのプロセス ID を /etc/named.pid ファイルに書き込みます。in.named で named.conf を再び読み込み、データベースを再度読み込ませる場合は、次の手順に従います。

▼ in.named に DNS データを強制的に再度読み込ませる方法

1. スーパーユーザーになります。
2. **# kill -HUP `cat /etc/named.pid`**

   この操作によって以前のキャッシュはすべて削除され、キャッシュの処理が再スタートします。

注意 – inetd から in.named を実行しないでください。これを行うとネームサーバーは繰り返し再起動され、キャッシュを持つ意味がなくなります。

クライアントの追加と削除

クライアントを追加または削除するときは必ず、DNS マスターサーバーに格納されたデータファイルを変更してください。スレーブサーバーのファイルを変更または編集しないでください。スレーブサーバーは、SOA のシリアル番号の変化に基づいて、マスターサーバーから自動的に更新されます。

クライアントの追加

DNS ドメインにクライアントを追加するには、新しいマシンを DNS クライアントとして設定してから、新しいマシンのレコードを該当する hosts と hosts.rev の各ファイルに追加します。

たとえば、rigel というホストを doc.com ドメインに追加する場合は、次の手順に従います。

▼ クライアントを追加する方法

1. スーパーユーザーになります。
2. /etc/resolv.conf ファイルを rigel 上に作成します。
3. rigel の /etc/nsswitch.conf ファイルの hosts の行に dns を追加します。
   44 ページの「DNS とインターネットでのアクセス」を参照してください。
4. マスターサーバーの hosts ファイルに、rigel 用のアドレス (A) レコードを追加します。

   ```
   rigel  IN  A  192.168.112
   ```

5. マスターサーバーの hosts ファイルに、rigel 用の任意指定のレコードを追加します。
   任意指定のレコードには、次のものがあります。
   - エイリアス (CNAME)
   - メール交換 (MX)
   - 既知サービス (WKS)
   - ホスト情報 (HINFO)
6. hosts.rev ファイルに rigel 用の PTR レコードを追加します。
7. マスターサーバーの hosts ファイルと hosts.rev ファイルの SOA シリアル番号を増やします。
8. サーバーのデータを再度読み込みます。
   サーバーをリブートするか、次のように入力します。

   ```
   # kill -HUP `cat /etc/named.pid`
   ```

クライアントの削除

DNS ドメインからクライアントを取り除く場合は、次の手順に従います。

▼ クライアントを削除する方法

1. スーパーユーザーになります。
2. 削除するマシンの nsswitch.conf ファイルの hosts の行から dns を削除します。
3. マシンの /etc/resolv.conf ファイルを削除します。
4. マスターサーバーの hosts ファイルと hosts.rev ファイルからそのマシンのレコードを削除します。

5. そのマシンを示す CNAME レコードがある場合は、その CNAME レコードも hosts ファイルから削除する必要があります。

6. 削除されるマシンによってサポートされていたサービスの代替を設定します。

   マシンがマスターサーバー、メールホスト、その他必要なプロセスまたはサービスのホストである場合、そのマシンが行なっていたサービスを他のマシンで実行できるように設定する必要があります。

クライアントで IPv6 を使用できるようにする

表 4–1 マシンで IPv6 を使用できるようにする

作業	説明	参照先
マシンで IPv6 を使用できるようにする	/etc/nsswitch.conf ファイルを変更して、NIS+ クライアントで IPv6 を使用できるようにする	以下を参照

▼ クライアントで IPv6 を使用できるようにする方法

1. スーパーユーザーになります。

2. /etc/nsswitch.conf ファイルを編集します。

3. 新しい ipnodes ソースを追加して、ネームサービス **(LDAP** など**)** を指定します。

   ```
   ipnodes: ldap [NOTFOUND=return] files
   ```

   ipnodes は、デフォルトでは files です。IPv4 から IPv6 への変更中すべてのネームサービスが IPv6 のアドレスを認識できるわけではないので、 デフォルトの files を使用してください。デフォルトを使用しない場合は、アドレスの解決中に不必要な遅延が生じることがあります。

4. ファイルを保存して、マシンをリブートします。

   nscd デーモンはこの情報をキャッシュに保存して起動時にこの情報を読み取るため、 ここでマシンをリブートする必要があります。

DNS サブドメインの作成

ネットワークは大きくなっていくので、ネットワークを複数の DNS サブドメインに分割すると便利です (DNS のドメインの階層と構造については、98 ページの「DNS 名前空間の階層」を参照)。

ネットワークを親ドメインと 1 つ以上のサブドメインに分割すると、負担が複数のドメインに分散して、各 DNS サーバーの負荷は減ります。この方法で、ネットワークのパフォーマンスを改善できます。たとえば、ネットワークに 900 台のマシンがあり、すべてがひとつのドメインにあるとします。この場合、1 台のマスターサーバー、1 台以上のスレーブサーバーとキャッシュ専用サーバーからなる DNS のサーバーの集合では、900 台のマシンをサポートしなければなりません。このネットワークを各ドメインが 300 台ずつの 1 つの親ドメインと 2 つのサブドメインに分割すると、マスターサーバーとスレーブサーバーの組は 3 つとなり、それぞれ 300 台だけを担当することになります。

ネットワークを物理的または組織的な構成に合うように複数のドメインに分割することによって、DNS ドメイン名はどこに対象とするマシンがあるか、電子メールのアドレスが組織のどこにあたるのかを指し示すことになります。たとえば、`rigel@alameda.doc.com` は、マシン `rigel` が Alameda というサイトにあることを意味し、電子メールのアドレス `barnum@sales.doc.com` は、ユーザー `barnum` が営業 (sales) 部署の者であることを意味します。

ネットワークを複数のドメインに分けると、すべてをひとつのドメインに置く場合よりも設定作業が増えます。また、ドメインを互いにつなぐ委託データを保持しなければなりません。一方で、複数のドメインを持つと各ドメインの保守をそれぞれの管理者またはチームに分散させることができます。

サブドメインの設計

次に、ネットワークを 1 つの親ドメインと 1 つ以上のサブドメインに分割するにあたって考慮すべき点をいくつかあげます。

- 「サブドメインの数」。作成するサブドメインが多ければ、それだけ初期設定の作業が増え、運用時の親ドメインでの管理者の調整作業も増えます。多くのサブドメインがあれば、親ドメインのサーバーのために委任される作業もその分増加します。一方で、ドメインの数が少ないということは各ドメインが大きいということを意味し、ドメインが大きければそれをサポートするためにサーバーのスピードやメモリーが余計必要になります。
- 「ネットワークの分割方法」。ネットワークはどのようにでも複数のドメインに分割できます。最も一般的な方法が 3 つあります。まず、組織の構成によるもので、各部署 (営業、研究開発、製造など) に別々のサブドメインを持つ方法です。次に、地理的なもので、各サイトに別々のサブドメインを持つ方法です。最後に、ネットワークの構造によるもので、大きなネットワーク構成要素ごとに別々のサブ

ドメインを持つ方法です。もっとも重要なのは、ドメインの構造が統一されていて論理的かつ自明であれば、管理も利用もより簡単になるということです。

- 「将来に対する考慮」。もっとも問題が多いのは、新しいサイトや部が増えるに従って、時間の経過とともにサブドメインが無計画に追加されて大きくなったドメインです。可能な限り、将来のドメインの拡大を見越してドメインの階層を設計してください。安定性も考慮に入れてください。最も安定しているものを基礎として、サブドメインを分けるようお勧めします。たとえば、地理的なサイトは比較的安定しているけれども、部や課が頻繁に再編成されるなら、サブドメインは組織よりも地理的な位置に基づいて決定する方が良いでしょう。一方、組織は比較的安定しているがサイトが頻繁に追加されたり変更されたりする場合は、サブドメインは組織の階層に基づいて決定する方が良いでしょう。
- 「広域ネットワーク *(WAN)* とのリンク」。ネットワークがモデムまたは専用回線で接続された複数のサイトに広がっている場合、広域ネットワーク (WAN) のリンクにドメインが広がっていないなら、パフォーマンスと信頼性は高くなります。多くの場合、WAN のリンクは連続的なネットワーク接続より遅く失敗に終わる傾向があります。サーバーが、WAN のリンク経由だけで接続できるマシンをサポートしなければならないときは、より遅いリンクを経由するトラフィックファネリングが増加することになります。この場合、もし 1 つのサイトで停電や何か他の問題が起こった場合には他のサイトのマシンにも影響が及ぶ可能性があります。(同様のパフォーマンスと信頼性の配慮は DNS ゾーンについても必要です。一般的な方法として、ゾーンを WAN のリンクに広げないようにすることをお勧めします。)
- 「*NIS+* 互換性」。エンタープライズレベルのネームサービスが NIS+ である場合は、DNS と NIS+ のドメイン構造とサブドメイン構造が一致すると管理がより簡単になります。
- 「サブドメイン名」。可能な限り、サブドメインに名前を付けるのに一貫したポリシーを確立してそれを守るのが理想的です。ドメイン名が一貫していれば、ユーザーは簡単に記憶し正しく指定できます。ドメイン名はすべての DNS データファイルで重要な要素であること、サブドメインを変更すると古いドメイン名があるすべてのファイルを修正する必要があることに注意してください。したがって、安定していて変更の必要がないと思われるサブドメイン名を選択するようにしてください。サブドメイン名として `manufacturing` のように完全指定することもできますし、`manf` のように短縮して指定することもできます。しかし、あるサブドメインが短縮形で指定され他が完全指定された場合、利用者は混乱します。短縮形を使用するのであれば、名前を識別するのに十分な文字数にしてください。短かすぎる名前は利用や記憶が困難だからです。最上位のインターネットのドメイン名として使用されている名前をサブドメイン名として使わないで下さい。すなわち、`org`、`net`、`com`、`gov`、`edu` と、2 文字の国のコード `jp`、`uk`、`ca`、`it` 等は、すでに使用されているのでサブドメイン名として使うことはできません。

サブドメインの設定

ほとんどの場合、新しいサブドメインは、新しいネットワークやマシンを接続するか、既にあるドメインを分ける場合に作成されます。どちらの場合も、プロセスはよく似ています。

新しいサブドメインを設計した後、次の手順に従って設定します。

▼ サブドメインを設定する方法

1. 新しいサブドメイン内のすべてのマシンが **DNS** クライアントとして正しく設定されていることを確認してください。

   既存のドメインから新しいサブドメインを分ける場合、ほとんどのマシンは DNS クライアントの設定が既にされているはずです。新しいサブドメインを最初から構築する場合、あるいは既存のネットワークに新しいマシンを追加する場合は、正しく設定された `resolv.conf` ファイルと `nsswitch.conf` ファイルを各マシンにインストールする必要があります。

2. 正しく設定された起動ファイルと **DNS** データファイルをサブドメインのマスターサーバーにインストールします。

   - `/etc/named.conf`.
   - `/var/named/named.ca`.
   - `/var/named/hosts`.
   - `/var/named/hosts.rev`.
   - `/var/named/named.local`.

   サーバーのホストファイルには、サブドメイン内のマシンごとにアドレス (`A`) レコードと、場合によっては `CNAME` レコードが必要です。また、サーバーの `hosts.rev` ファイルには、サブドメイン内のマシンごとにポインタ (`PTR`) レコードが必要です。任意指定の `HINFO` と `WKS` レコードも追加できます。

3. 既存のドメインを分割する場合は、新しいサブドメインのマシン用のレコードを親ドメインのマスターサーバーの `hosts` ファイルと `hosts.rev` ファイルから削除してください。

   そのためには、現在新しいサブドメイン内にあるマシン用の `A` レコードを古いドメインサーバーのホストファイルから削除します。また、同じマシンの `PTR` レコードを古いドメインサーバーの `hosts.rev` ファイルから削除します。移動するマシン用の任意指定の `HINFO` レコードと `WKS` レコードも、削除する必要があります。

4. 既存のドメインを分割する場合は、新しいサブドメイン名を、親ドメインのマスターサーバーのホストファイル内の `CNAME` レコードに追加します。

   たとえば、`aldebaran` というマシンをファックスサーバーとして使っているとします。また親ドメインのサーバーのホストファイル内の `CNAME` レコードが次のとおりであるとします。

   ```
   faxserver   IN   CNAME   aldebaran
   ```

   新しい `faxserver` の `CNAME` レコードを、`aldebaran` のために、新しいサブドメインのマスターサーバーにあるホストファイル内に作成するとともに、下記のように `aldebaran` のサブドメインが含まれるように、親ドメインのホストファイル内の `CNAME` レコードも変更する必要があります。

   ```
   faxserver   IN   CNAME   aldebaran.manf.doc.com
   ```

5. 新しいサブドメインのサーバーの **NS** レコードを、親ドメインのホストファイルに追加します。

   たとえば、親ドメインは `doc.com` で、`manf.doc.com` という新しいサブドメインを作成しているとします。また、このとき `rigel` というマシンを `manf` のマスターサー

バーとして指定し、aldebaran をスレーブサーバーとして指定するとします。この場合、次のレコードを doc.com のマスターサーバーの hosts ファイルに追加することになります。

```
manf.doc.com 99999 IN NS rigel.manf.doc.com
             99999 IN NS aldebaran.manf.doc.com
```

6. 新しいサブドメインのサーバー用の A レコードを、親ドメインの hosts ファイルに追加します。

   引き続き上の例で考えると、次のレコードを doc.com のマスターサーバーの hosts ファイルに追加することになります。

```
rigel.manf.doc.com       99999  IN  A  1.22.333.121
aldebaran.manf.doc.com   99999  IN  A  1.22.333.136
```

7. サブドメインのサーバー上の named を起動します。

   # /usr/sbin/in.named

   コマンド行から in.named を実行する代わりに、リブートします。53 ページの「in.named と DNS ネームサーバー」を参照してください。

Solaris DNS BIND 8.2.4 の実装

Solaris オペレーティング環境では、コンパイル版の Berkeley Internet Name Domain (BIND) 8.2.4 を提供します。コンパイルにあたっては、より多くのサイトのニーズを満たすように各種オプションを設定しました。このコンパイル済みの BIND が要件に合わない場合は、公開されているソースコードから独自にコンパイルすることができます。

Solaris オペレーティング環境で提供される BIND バージョンのコンパイルでは、以下の選択が行われました。

- 「RFC1535」。実装によって暗黙の検索リストが削除されるため、現在は実装されていません。
- 「逆参照」。これがないと SunOS 4 の nslookup が機能しないため許可されています。
- 「デフォルトのドメイン名」。DNS ドメイン名が /etc/resolv.conf 内または LOCALDOMAIN 環境変数で設定されていない場合、libresolv は、デフォルトのドメイン名を NIS または NIS+ のドメイン名から引用します。
- 「ユーティリティスクリプト」。BIND のユーティリティスクリプトは、今回の Solaris リリースには含まれていません。
- 「テストプログラム」。BIND のテストプログラムである dig、dnsquery、host は、今回の Solaris リリースには含まれていません。これらのテストプログラムの目的は、nslookup と nstest の目的と同様であるためです。

▼ BIND 4.9.x から BIND 8.2.4 に移行する方法

1. スーパーユーザーになります。
2. **Korn** シェルスクリプト `/usr/sbin/named-bootconf` を実行し、**BIND 4.9.x** の `named.boot` ファイルを **BIND 8.2.4** の `named.conf` ファイルに変換します。

注 – Solaris 9 では `named.boot` ファイルは無視されます。

DNS の転送

`nsswitch.conf` ファイルは、クライアントの DNS 転送とインターネットへのアクセスを管理します。NIS クライアントには、転送機能が含まれています。NIS+ クライアントにはこの機能がありません。次の手順を参照してください。

▼ NIS+ クライアントで DNS 転送機能を使用できるようにする方法

1. スーパーユーザーになります。
2. `/etc/resolve.conf` ファイルの `hosts` 行を次のように正しく設定します。

   ```
   hosts:nisplus dns files
   ```

   この NIS 実装では、該当するサーバー上に `/etc/resolve.conf` ファイルが存在する場合は、`ypstart` が `-d` オプションで「自動的に」 `ypserv` デーモンを起動して DNS に要求を転送します。DNS への転送を停止する場合は、`/usr/lib/netsvc/yp/ypstart` スクリプトを編集して、`-d` オプションを `ypserv` コマンドから削除してください。その後マシンをリブートする必要があります。

▼ 以前の NIS クライアントで DNS 転送機能を使用できるようにする方法

1. スーパーユーザーになります。
2. `hosts.byname` マップと `hosts.byaddr` マップに `YP_INTERDOMAIN` キーを設定します。`Makefile` の次の行 **(**ファイルの先頭の行**)** を次のように変更してください。

   ```
   #B=-b
   B=
   ```

上記の行を次のように変更します。

```
B=-b
#B=
```

これで、マップの作成時に makedbm が -b フラグを使って起動されるようになるため、YP_INTERDOMAIN が ndbm ファイルに挿入されます。

3. マップを作成し直します。

   # /usr/ccs/bin/make hosts

4. 有効な名前のサーバーを指定している /etc/resolv.conf ファイルが **NIS** サーバーに存在することを確認します。

5. 各サーバーを ypstop コマンドで停止します。

   # /usr/lib/netsvc/yp/ypstop

6. 各サーバーを ypstart コマンドで再起動します。

   # /usr/lib/netsvc/yp/ypstart

注 – Solaris 2 以降が実行されていない NIS サーバーを使用している場合は、ホストマップに YP_INTERDOMAIN キーが存在することを確認してください。また、マスターサーバーとスレーブサーバーが「異なる」バージョンの Solaris を実行している場合は、問題が発生することがあります。 次の表に、このような問題を回避するためのコマンドがまとめてあります。「4.0.3+」という表記は、「SunOS のリリース 4.0.3 以降」であることを意味します。makedbm -b コマンドは、Makefile の「-B」変数への参照です。

表 4–2 異機種システムが混在する NIS ドメインにおける NIS/DNS

スレーブサーバー	マスターサーバー		
	4.0.3+	**Solaris NIS**	
4.0.3+	マスターサーバー: makedbm -b スレーブサーバー: ypxfr	マスターサーバー: makedbm -b スレーブサーバー: ypxfr -b	マスターサーバー: ypserv -d スレーブサーバー: ypxfr -b
Solaris NIS	マスターサーバー: makedbm -b スレーブサーバー: ypxfr	マスターサーバー: makedbm -b スレーブサーバー: ypxfr	マスターサーバー: ypserv -d スレーブサーバー:ypxfr が存在する resolve.conf または ypxfr -b

Solaris オペレーティング環境には、リゾルバを構成している動的ライブラリルーチンが含まれています。

第 5 章

DNS の管理 (参照情報)

この章の内容は次のとおりです。

DNS の実装

実例

この節では、この章で説明する例を基にして、サンプルのインターネット接続ネットワークを想定し、そこで使う DNS を設定するために必要なファイルを示します。

注意 – この例で使われている IP アドレスとネットワーク番号、および本書で使われているサンプルコードは、説明に具体性を持たせるために仮に決めたものです。これらの情報は、実際のネットワークやホストに使われていることがありますので、そのまま使うのは避けてください。

この実例では、次のことを前提としています。

- インターネットに接続されている

- 2 つのネットワークが存在していて、それぞれ個別のドメインを持つ (doc.com と sales.doc.com)。DNS ゾーンも別々に管理
- doc.com ドメインおよび doc.com ゾーンが sales.doc.com サブドメインおよび sales.doc.com ゾーンの上の最上位ゾーンである
- 2 つのネットワークはそれぞれ個別のネットワーク番号を持つ

表 **5–1** ネットワークドメインとゾーン構成の例

名前/ゾーン	番号
doc.com	123.45.6
sales.doc.com	111.22.3

- 各ゾーンにマスターサーバーとスレーブサーバーが 1 台ずつあり、doc.com のマスターサーバーが sales.doc.com のスレーブサーバーを兼ねている

表 **5–2** ネットワーク DNS サーバーの例

ゾーン	ホスト名	機能	アドレス	正規名
doc.com	sirius	doc.com のマスターサーバー	123.45.6.1	dnsmaster
doc.com	deneb	doc.com のスレーブサーバー	111.22.3.5	dnssecond
sales.doc.com	altair	sales.doc.com のマスターサーバー	111.22.3.4	dnssales
sales.doc.com	altair	sales.doc.com のスレーブサーバー	123.45.6.1	dnsmaster

起動ファイルの例

次に示すのは、2 つのネットワークで使われている 3 つのサーバーの起動ファイルです。

例 **5–1** dnsmastr サーバー用起動ファイルの例

```
; named.boot file on the dnsmastr (sirius)
;
; files required by in.named are located here
directory /var/named
; here are the names of the master files
cache          .                        named.ca
master       doc.com                   db.doc
master       0.0.127.in-addr.arpa      named.local
master       6.45.123.in-addr.arpa     doc.rev
;This system is also the slave for the sales.doc.com domain
slave     sales.doc.com             111.22.3.4   db.sales
```

例 **5–1** dnsmastr サーバー用起動ファイルの例　(続き)

```
slave     3.22.111.in-addr.arpa     111.22.3.4   sales.rev
```

例 **5–2** dnssales サーバー用起動ファイルの例

```
; named.boot file on the dnssales (altair)
;
; in.named is located here
directory /var/named
; here are the names of the master files
cache      .                        named.ca
master    sales.doc.com            db.sales
master    0.0.127.in-addr.arpa     db.127.0.0
master    3.22.111.in-addr.arpa    db.192.168.8
```

例 **5–3** dnssecond サーバー用起動ファイルの例

```
; named.boot file on the dnsecond (deneb)
directory /var/named
cache           .              named.ca
slave       doc.com         123.45.6.1 doc.com
slave       6.45.123.in-addr.arpa    123.45.6.1 doc.123.45.6
```

resolv.conf ファイルの例

次に示すのは、2 つのネットワークで使われている 3 つのサーバーの resolv.conf ファイルです。そのホストが in.named を起動していない場合、そのローカルホストのアドレスをネームサーバーとして使用しないでください。

例 **5–4** dnsmaster サーバー用 resolve.conf ファイルの例

```
;
; /etc/resolv.conf file for dnsmaster (sirius)
;
domain           doc.com
nameserver       0.0.0.0
nameserver       111.22.3.5
```

例 **5–5** dnssales サーバー用 resolve.conf ファイルの例

```
;
; /etc/resolv.conf file for dnssales (altair)
;
domain           sales.doc.com
nameserver       111.22.3.4
nameserver       123.45.6.1
```

例 **5–6** dnssecond サーバー用 resolve.conf ファイルの例

```
;
; /etc/resolv.conf for dnssecond
;
```

例 **5–6** dnssecond サーバー用 resolve.conf ファイルの例　(続き)

```
domain          doc.com
nameserver      111.22.3.5
nameserver      123.45.6.1
```

named.local ファイルの例

次に示すのは、2 つのネットワーク上の 2 つのマスターサーバーで使われている named.local ファイルです。どちらのサーバーも同じファイルを持っています。

例 **5–7** マスターサーバー用 named.local ファイルの例

```
; SOA rec
0.0.127.in-addr.arpa. IN SOA siriusdoc.com. sysop.centauri.doc.com.(
                          19970331    ; serial number
                          10800       ; refresh every 3 hours
                          10800       ; retry every 3 hours
                          604800      ; expire after a week
                          86400 )     ; TTL of 1 day
; Name Servers
0.0.127.in-addr.arpa.  IN  NS   sirius.doc.com.
0.0.127.in_addr.arpa   IN  NS   dnssecond.doc.com
1  IN  PTR localhost.
```

hosts ファイルの例

次に示すのは、2 つのネットワーク上の 2 つのマスターサーバーで使われている db.doc ファイルと db.sales ファイルです。

例 **5–8** dnsmastr サーバー用 db.doc ファイルの例

```
; SOA rec
doc.com. IN SOA sirius.doc.com. sysop.centauri.doc.com. (
                          19970332    ; serial number
                          10800       ; refresh every 3 hours
                          10800       ; retry every 3 hours
                          604800      ; expire after a week
                          86400 )     ; TTL of 1 day
; Name Servers
doc.com.               IN  NS  sirius.doc.com.
sales.doc.com.         IN  NS  altair.sales.doc.com.
; Addresses
localhost              IN  A  127.0.0.1
sirius                 IN  A  123.45.6.1
rigel                  IN  A  123.45.6.112
antares                IN  A  123.45.6.90
polaris                IN  A  123.45.6.101
procyon                IN  A  123.45.6.79
tauceti                IN  A  123.45.6.69
```

例 **5–8** dnsmastr サーバー用 db.doc ファイルの例　*(続き)*

```
altair.sales.doc.com.   N  A   111.22.3.4
; aliases
dnsmastr               IN  CNAME    sirius.doc.com.
dnssecond.doc.com      IN  CNAME   deneb.doc.com
```

例 **5–9** dnssales サーバー用 db.sales ファイルの例

```
; SOA rec
sales.doc.com.  IN SOA altair.sales.doc.com. sysop.polaris.doc.com. (
                           19970332    ; serial number
                           10800          ; refresh every 3 hours
                           10800          ; retry every 3 hours
                           604800        ; expire after a week
                           86400 )        ; TTL of 1 day
; Name Servers
doc.com.                IN  NS  sirius.doc.com.
sales.doc.com.          IN  NS  altair.sales.doc.com.
; Addresses
altair                  IN  A  111.22.3.4
localhost               IN  A  127.0.0.1
sirius.doc.com.         IN  A  123.45.6.1
luna                    IN  A  192.168.8.22
phoebus                 IN  A  192.168.8.24
deimos                  IN  A  192.168.8.25
ganymede                IN  A  192.168.8.27
europa                  IN  A  192.168.8.28
callisto                IN  A  192.168.8.29
;
; aliases
dnssales.sales.doc.com  IN  CNAME    altair.sales.doc.com
```

hosts.rev ファイルの例

次に示すのは、2 つのネットワーク上の 2 つのマスターサーバーで使われている hosts.rev ファイルです。

例 **5–10** dnsmastr サーバー用 hosts.rev ファイルの例

```
; SOA rec
6.45.123.in-addr.arpa.  IN SOA sirius.doc.com. sysop.centauri.doc.com. (
                           19970331    ; serial number
                           10800       ; refresh every 3 hours
                           10800       ; retry every 3 hours
                           604800      ; expire after a week
                           86400 )     ; TTL of 1 day
; Name Servers
6.45.123.in-addr.arpa.  IN  NS  sirius.doc.com.
;Pointer records for 123.45.6
1                       IN  PTR sirius.doc.com.
112                     IN  PTR rigel.doc.com.
```

例 **5–10** dnsmastr サーバー用 hosts.rev ファイルの例　(続き)

```
90                      IN  PTR antares.doc.com.
101                     IN  PTR polaris.doc.com.
79                      IN  PTR procyon.doc.com.
69                      IN  PTR tauceti.doc.com.
```

例 **5–11** dnssales サーバー用 hosts.rev ファイルの例

```
; SOA rec
3.22.111.in-addr.arpa.  IN SOA altair.sales.doc.com. \
sysop.polaris.doc.com.(
                           19970331    ; serial number
                           10800       ; refresh every 3 hours
                           10800       ; retry every 3 hours
                           604800      ; expire after a week
                           86400 )     ; TTL of 1 day
; Name Servers
3.22.111.in-addr.arpa.  IN  NS  altair.sales.doc.com.; \
Pointer records for 111.22.3
22                      IN  PTR  luna
23                      IN  PTR  deneb
24                      IN  PTR  phoebus
25                      IN  PTR  deimos
26                      IN  PTR  altair
27                      IN  PTR  ganymede
28                      IN  PTR  europa
29                      IN  PTR  callisto
```

name.ca ファイルの例

次に示すのは、2 つのネットワーク上の 2 つのマスターサーバーにそれぞれ格納される named.ca ファイルです。どちらのサーバーも同じ named.ca ファイルを使用します。

例 **5–12** named.ca ファイルの例

```
;
; formerly NS1.ISI.EDU
.                       3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     3600000      A     128.9.0.107
;
; formerly C.PSI.NET
.                       3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
.                       3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.     3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;.                      3600000      NS    E.ROOT-SERVERS.NET.
```

例 **5–12** named.ca ファイルの例　(続き)

```
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
;
; formerly NS.NIC.DDN.MIL
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
;
; temporarily housed at NSI (InterNIC)
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     198.41.0.10
;
; temporarily housed at NSI (InterNIC)
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     198.41.0.11
;
; temporarily housed at ISI (IANA)
.                        3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000      A     198.32.64.12
;
; temporarily housed at ISI (IANA)
.                        3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A     198.32.65.12
; End of File
```

データファイルの設定

DNS デーモン in.named が使用するすべてのデータファイルは、標準リソースレコード書式で記述されます。標準リソースレコード書式では、ファイルの各行は、リソースレコード (RR) と呼ばれるレコードです。各 DNS データファイルには決められたリソースレコードが必要です。

リソースレコードのタイプ

最も一般的に使用されるリソースレコードのタイプを 表 5–7 に列挙します。 通常、表 5–7 に並んだ順で入力しますが、この順序は必須ではありません。

表 **5–3** 一般的に使用されるリソースレコードのタイプ

形式	説明
SOA	権限の開始
NS	ネームサーバー
A	IPv4 インターネットアドレス (名前からアドレス)
AAAA	IPv6 インターネットアドレス (名前からアドレス)
PTR	ポインタ (アドレスから名前)
CNAME	正規名 (ニックネーム)
TXT	テキスト情報
MX	メール交換

これ以降に示すサンプルファイルでは、@ は現在のゾーンまたは現在の起点を示します。セミコロン (;) で始まる行はコメントです。

サブドメインの設定

単一ゾーンのサブドメインの設定

最も簡単な方法は、サブドメインを親ドメインのゾーンに含めることです。こうすると、1 セットの DNS サーバーとデータファイルでドメインに関係なくすべてのマシンを管理できます。

単一ゾーン方式の長所は、管理が簡素化され簡単なことです。短所は 1 セットのサーバーですべてのゾーンのドメインにあるマシンを管理しなければならないということです。マシンの数が多すぎると、サーバーの負荷が大きくなり過ぎ、パフォーマンスが低下することがあります。

複数のドメインで構成されているゾーンのデータファイルには、そのゾーンに含まれる各ドメインのすべてのマシンとサーバーに関わるレコードが必要です。

複数のドメインで構成されているゾーンを設定するのも、単一ドメインで構成されているゾーンを設定するのも、必要な作業は基本的に同じです。唯一の相違は、遠隔ドメインのマシンを識別できるようにするために、hosts ファイルには完全指定のドメイン名を使用しなければならないということです。サーバーのローカルドメインにあるマシンであれば、hosts ファイルにマシン名しか指定されていなくても識別できます。しかし、他のドメインにあるマシンを識別するには、完全指定のドメイン名、つまり *machine.domain* という書式で指定しなければなりません。

hosts.rev ファイルと named.local ファイルに指定するサーバー名やマシン名にも、完全指定のドメイン名を使用する必要があります。ただし、これはゾーンがいくつのドメインで構成されているかには関係ありません。

複数ゾーンのサブドメインの設定

複数ゾーン方式の長所は、ドメインごとにその中のマシンを管理するサーバーセットを変更できるということです。つまり、サーバーの負荷を分散させ、1 セットのサーバーに負荷が集中するのを防ぐことができます。短所は、設定時の作業がより複雑になることです。

異なるゾーンのサブドメインを設定するのは、1 つのゾーンに複数のドメインを含めるのよりも複雑です。これは、さまざまなゾーンにあるクライアントが他のゾーンの DNS 情報を得る方法を指定しなければならないからです。

ネットワークを複数のドメインに分ける場合、ドメインを階層化します。必ず最上位のドメインがあって、その下に 1 つまたは複数のサブドメインがあります。サブドメインの下にサブドメインを作ることもできます。しかし、どのサブドメインにも階層構造の中で最上位のドメインから相対的に決まった場所があります。ドメイン名は左から右に読んでいくと、階層内におけるドメインの位置を示していることがわかります。たとえば、doc.com ドメインは sales.doc.com の上位にあり、west.sales.doc.com ドメインは sales.doc.com ドメインの下位にあることがわかります。

DNS ゾーンはそのゾーンが含むドメインから階層を取り込みます。ネットワークの最上位ドメインを含むゾーンは最上位のゾーンになります。最上位のドメインの下のサブドメインを 1 つ以上含むゾーンは、ゾーンの階層でいえば最上位のゾーンの下のゾーンになります。DNS 情報をあるゾーンから別のゾーンへ移動させるということは、このゾーン階層の中を上下に移動させるということです。つまり、各ゾーンでは、すぐ上のゾーンに情報を渡すにはどうするか、すぐ下のゾーンに情報を渡すにはどうするかを専用のデータファイルのレコードに指定しておく必要があります。

複数のゾーンで構成されているネットワークの中で、DNS 情報をあるゾーンから別のゾーンへ正確に転送させるために必要なことを以下に示します。

- hosts.rev ファイル。すぐ上のゾーンにある 1 つまたは複数のマスターサーバー名を指し示す PTR レコードが各 hosts.rev ファイルに必要です。上位ゾーンのサーバーを指し示すということを除けば、この種の PTR レコードは、ファイル内のその他の PTR レコードとまったく同じのものです。

- hosts ファイルの NS レコード。すぐ下の各ゾーンにあるネームサーバー名を指し示すゾーン NS レコードが各 hosts ファイルに必要です。この種の NS レコードは、その最初のフィールドに下のゾーン名が指定されていなければなりません。ゾーンの名前は、そのゾーンの host ファイルの SOA レコードに指定されています。
- hosts ファイルの A レコード。すぐ下の各ゾーンにあるネームサーバーの IP アドレスを指し示す A レコードが各 hosts ファイルに必要です。この種の A レコードは、その最初のフィールドに下のゾーン名が指定されていなければなりません。ゾーン名は、そのゾーンの host ファイルの SOA レコードに指定されています。

この章のファイル例に、2 つのゾーンを持つネットワークが示してあります。

DNS 名前空間の階層

全世界の DNS 管理ドメインの集合全体は「DNS 名前空間」と呼ばれる階層構造を形成しています。ここでは、名前空間の組織がどのようにローカルドメインやインターネットに影響するのか説明します。

ドメインとサブドメイン

DNS ドメインは、UNIX™ のファイルシステムと同様、木の根に似た下に向きの枝分かれで構成されています。各枝分かれがドメインであり、そこから分かれる各枝が「サブドメイン」です。「ドメイン」と「サブドメイン」は相対的な関係を示します。階層の中で、あるドメインは、その上にあるドメインに対するサブドメインになり、その下にあるサブドメインの親ドメインになります。

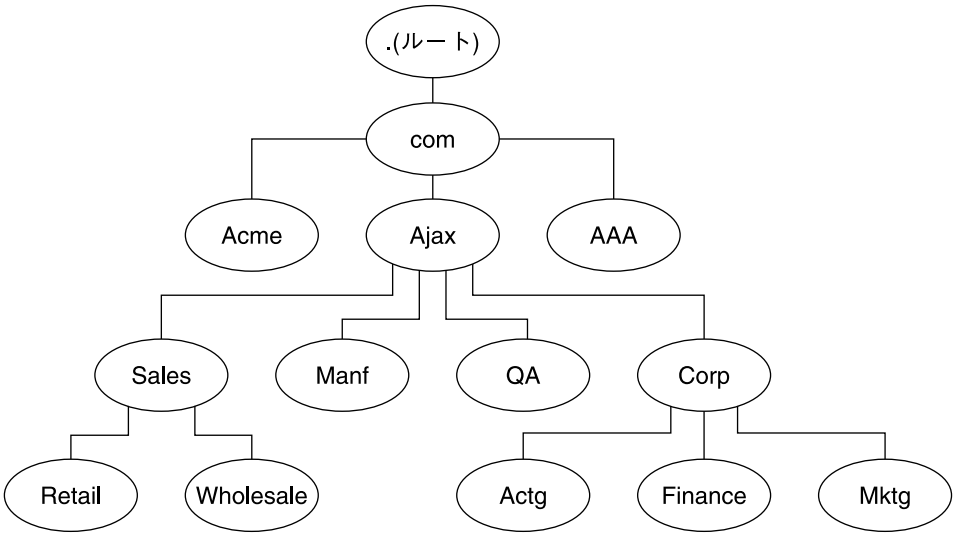


図 **5–1** ドメインとサブドメイン

たとえば、図 5–1 において com は Acme、Ajax、AAA の各ドメインの親ドメインです。あるいは、これらのドメインは com ドメインのサブドメインということもできます。このように考えると、Ajax ドメインは 4 つのサブドメイン (Sales、Manf、QA、Corp) の親ドメインになっています。

あるドメインには、1 つの親 (または最上位) ドメインと、(存在する場合) その下のサブドメインが含まれます。ドメインの名前は、最下位 (階層の底) のサブドメインから始まり、最後がルートドメインとなっています。

DNS のメール配信への影響について

DNS は、50 ページの「名前のアドレス解決」で説明しているように、名前からアドレス (またはその逆のアドレスから名前) のマッピングを行う他に、インターネット上でメールを配信する sendmail や POP といったメール配信エージェントの役にも立っています。

インターネット上でメールを配信するのに、DNS は「メール交換レコード」(MX レコード) を用います。ほとんどの組織は、その組織内にあるホストに宛てられたインターネットから来るメールを直接配信することを許可しません。そのかわりに、1 台の中央メールホスト (またはメールホストの集合) を使用して、入ってくるメールメッセージを途中で止めて宛先に振り分けます。

メール交換レコードで、ドメイン内の各マシンにサービスを提供しているメールホストが識別されるため、メール交換レコードには遠隔組織の DNS ドメイン名と、その IP アドレスまたは対応するメールホストのホスト名のいずれかが列挙されています。

DNS の構成ファイルとデータファイル

DNS のネームサーバーには、`in.named` デーモンに加えて、`named.conf` という起動ファイル、`resolv.conf` というリゾルバファイル、4 種類のゾーンデータファイルがあります。

DNS データファイルの名前

内部で一貫性が取れていれば、ゾーンデータファイルには何でも好きな名前を付けることができます。このため、異なるサイトで作業をしようとする場合や DNS 関連のマニュアルや本を参照する場合に、混乱するかもしれません。

たとえば、Sun のマニュアルや大多数の Solaris サイトで使われているファイル名は、『*DNS and BIND*』(Paul Albeltz & Criclcet Liu 著、浅羽登志也／上水流由香監訳 、アスキー出版局、1995年) で使われているファイル名とは異なります。そしてこれら 2 派の命名方法は、『*Name Server Operations Guide for BIND*』(カリフォルニア州立大学刊、パブリックドメイン) の命名方法とも若干の相違があります。

さらに、本書とその他の DNS 関連のマニュアルでは、説明にはファイルの主な役割を表す総称名を使い、コード例には具体的な固有の名前を使っています。たとえば、Solaris のネームサービスに関するマニュアルでは、ファイルの機能や役割を説明する場合は `hosts` という総称名を使い、コード例では `db.doc` や `db.sales.doc` といった名前を使っています。

参考のため、次の表で上で述べた 3 種類の BIND ファイル名を比較します。

表 **5–4** BIND ファイル名の例

Solaris	**O'Reilly** その他	カリフォルニア州立大学バークレイ校	ファイルの内容と役割
`/etc/named.conf`	`/etc/named.conf`	`/etc/named.conf`	構成ファイルは、それが実行されるサーバーのタイプ、および「マスター」、「スレーブ」、または「スタブ」として機能するゾーンを指定する。また、セキュリティ、ロギング、およびゾーンに適用されるオプションの細かい細分性を定義する

表 **5–4** BIND ファイル名の例　(続き)

Solaris	**O'Reilly** その他	カリフォルニア州立大学バークレイ校	ファイルの内容と役割
/etc/resolv.conf	/etc/resolv.conf	/etc/resolv.conf	各クライアント (DNS サーバーを含む) 上に存在するファイル。DNS 情報を探すためにクライアントが照会するサーバーを示す
named.ca	db.cache db.root	root.cache	ルートサーバー名とそのアドレスがリストされている
総称名: hosts 例:db.doc db.sales	総称名: db.domain 例:db.movie db.fx	総称名: hosts 例: ucbhosts	サーバーがサービスを提供するローカルゾーン内のマシンに関する全データが格納されている
総称名: hosts.rev 例: doc.rev	総称名: db.ADDR例: db.192.249.249 db.192.249.253	hosts.rev	逆マッピング (アドレスから名前) j を行うための特殊なドメイン in-addr.arpa. のゾーンを指定する
named.local	総称名: db.cache 例:db.127.0.0	named.local	ローカルループバックインタフェースまたはローカルホスト用のアドレスを指定する
$INCLUDE ファイル	$INCLUDE ファイル	$INCLUDE ファイル	データファイル内の $INCLUDE() 文によって識別されるファイル

注意 – このマニュアル内の例やコード例で使われている IP アドレスとネットワーク番号は、説明に具体性を持たせるために仮に決めたものです。これらの情報は、実際のネットワークやホストに使われていることがありますので、そのまま使うのは避けてください。

named.conf ファイル

BIND 8.2.4 構成ファイル /etc/named.conf は、サーバーをマスターサーバー、スレーブサーバー、またはキャッシュ専用サーバーとして設定します。また、サーバーが権限を持つゾーンを指定し、どのデータファイルから初期データを取得するかを指定します。

/etc/named.conf ファイルには、次の機能を実装する文が含まれています。

- アクセス制御リスト(ACL) によるセキュリティ。ACL には、NIS+ ホストが読み取り/書き込み権を持つ IP アドレスの集まりが定義されている。
- ログ仕様
- すべてのゾーンにではなく、ゾーンのセットに対して選択的に適用されるオプション

構成ファイルは、サーバーの起動スクリプト /etc/init.d/inetsvc によってデーモンが起動されるとき、 in.named によって読み取られます。構成ファイルにより、他のサーバー (マスター、スレーブまたはキャッシュ専用サーバー) として設定されるか、あるいは初期データを取得する構成ファイルが示されます。

named.conf 文

named.conf ファイルは、いくつかの文とコメントで構成されています。文はセミコロンで終わります。一部の文は、文のブロックを含むことができます。ブロックの中の各文もセミコロンで終わります。

named.conf ファイルは、以下の文をサポートします。

表 **5–5** named.conf 文

acl	アクセス制御に使用する、IP アドレスの一致リストを名前を付けて定義する。アドレスの一致リストは、1 つ以上の IP アドレス (ドット形式の 10 進表記) または IP 接頭辞 (ドット形式の 10 進表記の後にスラッシュとネットマスクのビット数が付く) を示す。名前を付けたIP アドレスの一致リストは、他の場所で使用する前に acl 文で定義されている必要がある。前方参照は不可
include	include 文がある箇所にインクルードファイルを挿入する。include を使用することで、管理しやすいまとまりに構成情報を分割することができる
key	特定のネームサーバーでの認証と承認に使用される鍵の ID を指定する。server 文を参照
logging	サーバーが記録するログ情報と、ログメッセージの送り先を指定する
options	グローバルなサーバー構成のオプションを制御して、他の文に対するデフォルト値を設定する
server	遠隔ネームサーバーに関して、指定された構成オプションを設定する。すべてのサーバーに対してではなく、サーバーごとに選択的にオプションを適用する
zone	ゾーンを定義する。すべてのゾーンに対してではなく、ゾーンごとに選択的にオプションを適用する

例 **5–13** マスターサーバー用マスター構成ファイルの例

```
options {
         directory "/var/named";
         datasize 2098;
         forward only;
         forwarders {
                  99.11.33.44;
         };
         recursion no;
         transfers-in 10;
```

例 **5–13** マスターサーバー用マスター構成ファイルの例　(続き)

```
        transfers-per-ns 2;
        allow-transfer {
                127.0.1.1/24;
        };
};

logging {
        category queries { default_syslog; };
};

include "/var/named/abcZones.conf"

// これはマスターファイルの名前です
zone "cities.zn" {
        type master;
        file "db.cities.zn";
};

zone "0.0.127.in-addr.arpa." {
        type master;
        file "db.127.cities.zn";
};

zone "168.192.in-addr.arpa" {
        type master;
        file "db.cities.zn.rev";
};

zone "sales.doc.com" {
        type slave;
        file "slave/db.sales.doc";
        masters {
                192.168.1.151;
        };
};

zone "168.192.in-addr.arpa" {
        type slave;
        file "slave/db.sales.doc.rev";
        masters {
                192.168.1.151;
        };
};
```

named.ca ファイル

named.ca ファイルによって、ルートサーバー名が確立され、そのアドレスが列挙されます。ネットワークがインターネットに接続されている場合は、named.ca には、インターネットのネームサーバーが表示されます。接続されていなければ、ローカル

ネットワークのルートドメインネームサーバーが表示されます。in.named デーモンは、サーバーの 1 つに接続できるまで、サーバーのリストを一巡します。そして、そのサーバーから現在のルートサーバーのリストを入手します。デーモンは、このリストを named.ca の更新のために用います。

named.ca ファイルの設定

ルートサーバー名は NS レコードに、アドレスは A レコードに示されています。この named.ca ファイルを使用するサーバーごとに、NS レコードと A レコードを追加する必要があります。

named.ca ファイルの入手方法または作成方法は、ネットワークがインターネットに接続されているかどうかによって異なります。

インターネット named.ca ファイル

ネットワークがインターネットに接続されている場合は、InterNIC Registration Service (本書執筆の時点) から次の手段で named.ca ファイルを入手できます。

- 匿名 FTP。FTP サイトは ftp.rs.internic.net 、ファイルは /domain/named.root です。
- Gopher。Gopher サイトは rs.internic.net です。 rs.internic.net. ファイルは named.root であり、これは「InterNIC Registration Service」メニューの「InterNIC Registration Archives」サブメニューで見つけることができます。

本書で説明した命名規則に従う場合、named.root を /var/named/named.ca に移動します。

例 **5–14** インターネット named.ca ファイルの例

```
;
; formerly NS1.ISI.EDU
.                       3600000    NS   B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     3600000    A    128.9.0.107
;
; formerly C.PSI.NET
.                       3600000    NS   C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     3600000    A    192.33.4.12
;
; formerly TERP.UMD.EDU
.                       3600000    NS   D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.     3600000    A    128.8.10.90
;
; formerly NS.NASA.GOV
;.                      3600000    NS   E.ROOT-SERVERS.NET.

E.ROOT-SERVERS.NET.     3600000    A    192.203.230.10
;
; formerly NS.ISC.ORG
```

例 **5–14** インターネット named.ca ファイルの例　(続き)

```
.                        3600000    NS   F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000    A    192.5.5.241
;
; formerly NS.NIC.DDN.MIL
.                        3600000    NS   G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000    A    192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
.                        3600000    NS   H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000    A    128.63.2.53
;
; formerly NIC.NORDU.NET
.                        3600000    NS   I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000    A    192.36.148.17
;
; temporarily housed at NSI (InterNIC)
.                        3600000    NS   J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000    A    198.41.0.10
;
; temporarily housed at NSI (InterNIC)
.                        3600000    NS   K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000    A    198.41.0.11
;
; temporarily housed at ISI (IANA)
.                        3600000    NS   L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000    A    198.32.64.12
;
; temporarily housed at ISI (IANA)
.                        3600000    NS   M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000    A    198.32.65.12
; End of File
```

非インターネット named.ca ファイル

ネットワークがインターネットに接続されていない場合は、独自の named.ca ファイルを作成する必要があります。そのためには、サーバーのどれか 1 つをルートサーバーとし、DNS サーバーごとにそのルートサーバーを指す named.ca ファイルを作成します。

たとえば、private というドメインで ourroot というマシンを非インターネットルートサーバーとして指定する場合を想定します。ourroot の IP アドレスが 192.1.1.10 であるとすると、named.ca ファイルには次の行を書き込みます。

```
ourroot.private.  999999  IN  A  192.1.1.10
```

キャッシュファイルも SOA レコード、各ドメインおよびサブドメインの NS レコード、各サーバーの A レコードを必要とします。

たとえば、ourroot の他に、ourmaster と ourslave という 2 つの DNS ネームサーバーがあるとします。その場合、DNS サーバー上の named.ca ファイルはすべて次のようになります。

例 **5–15** named.ca ファイル (非インターネット) の例

```
;
@    IN    SOA  ourroot.private.   hermit.ourroot.private  (
                 1997071401       ;  serial number (YYYYMMDD##)
                 10800            ;  refresh after 3 hours
                 3600             ;  retry after 1 hour
                 604800           ;  expire after 1 week
                 86400 )          ;  minimum TTL of 1 day
;
ourroot.private.      999999     IN    A    192.1.1.10
;
private.                         IN    NS   ourmaster.private.
1.1.192.in-addr.arpa             IN    NS   ourmaster.private.

ourprivate.private.              IN    A    192.1.1.1
;
private.                         IN    NS   ourslave.private.
1.1.192.in-addr.arpa             IN    NS   ourslave.private.
ourslave.private.            IN    A    192.1.1.2
```

hosts ファイル

hosts ファイルには、ローカルゾーン内のマシンに関するすべてのデータが含まれています。このファイル名は、起動ファイル内で指定します。/etc/hosts との混同を避けるために、hosts 以外の名前を付けます。たとえば、これらのファイルに db.*domain* パターンを使用して名前を付けることができます。この命名方法により、doc.com と sales.doc.com ドメインのホストファイルは db.doc と db.sales になります。

hosts ファイルの設定

hosts ファイルには、ゾーン内にある各マシンの全データが収められています。ゾーンが複数のドメインにまたがっている場合は、そのゾーンを構成する全ドメインの全マシンがそのゾーンのホストファイルに列挙されます。106 ページの「hosts ファイルの設定 」を参照してください。

注 – hosts という名前はファイルの役割や内容を表す総称名です。この総称名をそのまま使うと /etc/hosts と紛らわしいので、この種のファイルは hosts 以外の名前にすることをお勧めします。 ドメイン内に複数のゾーンがある場合は、各ゾーンに 1 つずつ hosts ファイルを置き、各ゾーンの hosts ファイルには一意の名前を付けなければなりません。たとえば、DNS ドメイン内に doc.com と sales.doc.com という 2 つのゾーンがある場合は、1 つを db.doc、もう 1 つを sales.db.doc という名前にするとよいでしょう。

各ゾーンには個別の、一意の名前を持つ hosts ファイルが必要です。複数のゾーンが存在する場合は、各ゾーンの hosts ファイルには他のゾーンのマスター (マスター、スレーブ) サーバーに関する情報も含める必要があります。詳細については、例 5–16 を参照してください。

例 **5–16** hosts ファイルの例

```
;
; SOA rec
doc.com.  IN SOA sirius.doc.com. sysop.centauri.doc.com. (
                 1997071401       ;  serial number (YYYYMMDD##)
                      10800       ;  refresh every 3 hours
                      10800       ;  retry every 3 hours
                      604800      ;  expire after a week
                      86400 )     ;  TTL of 1 day
; Name Servers
doc.com.                    IN  NS  sirius.doc.com.
sales.doc.com.              IN  NS  altair.sales.doc.com.
; Addresses
localhost.                  IN  A  127.0.0.1

sirius                     IN  A  192.168.6.1
rigel                      IN  A  192.168.6.112
antares                    IN  A  192.168.6.90
polaris                    IN  A  192.168.6.101
procyon                    IN  A  192.168.6.79
tauceti                    IN  A  123.45.6.69
altair.sales.doc.com.      IN  A   111.22.3.4
; aliases
durvasa                    IN  CNAME sirius.doc.com.
dnsmastr                   IN  CNAME sirius.doc.com.
dnssales                   IN  CNAME altair.sales.doc.com.
```

hosts ファイルは通常、次の 5 つの要素で構成されています。

- 1 つの SOA (権限の開始) レコード
- 1 つまたは複数の NS (ネームサーバー) レコード。マスターおよびスレーブの DNS ネームサーバーを識別する
- A (アドレス) レコード。ゾーン内の各ホストに必要
- CNAME (正規名) レコード。ゾーン内の各ホストのエイリアスに必要
- 1 つまたは複数の MX (メール交換) レコード

hosts.rev ファイル

hosts.rev ファイルで、逆 (アドレスから名前) マッピングを行うための特別な in-addr.arpa. ドメインのゾーンを指定します。このファイル名は、起動ファイル内で指定します。

hosts.rev ファイルの設定

hosts.rev は逆マッピングを設定するファイルです。

注 – hosts.rev という名前は、ファイルの役割や内容を表す総称名です。ドメイン内に複数のゾーンがある場合は、各ゾーンに 1 つずつ hosts.rev ファイルを置き、各ゾーンの hosts.rev ファイルには一意の名前を付けなければなりません。たとえば、DNS ドメイン内に doc.com と sales.doc.com という 2 つのゾーンがある場合は、1 つを doc.rev、もう 1 つを sales.rev という名前にするとよいでしょう。

例 **5–17** hosts.rev ファイルの例

```
; SOA rec
6.45.123.in-addr.arpa.  IN SOA sirius.doc.com. sysop.centauri.doc.com. (
                  1997071401       ;  serial number (YYYYMMDD##)
                       10800       ;  refresh every 3 hours
                       10800       ;  retry every 3 hours
                       604800      ;  expire after a week
                       86400 )     ;  TTL of 1 day
; Name Servers
6.45.123.in-addr.arpa.   IN  NS  sirius.doc.com.
1                        IN  PTR sirius.doc.com.
```

hosts.rev ファイルは、次の要素で構成されています。

- 1 つの SOA (権限の開始) レコード
- 1 つまたは複数の NS (ネームサーバー) レコード。マスターおよびスレーブの DNS ネームサーバーを識別する。サーバー名は完全指定する必要がある
- PTR レコード。ゾーン内の各ホストに 1 つ必要。マシン名は完全指定する必要がある

(これらのリソースレコードの詳細については、96 ページの「リソースレコードのタイプ」を参照してください。)

named.local ファイル

named.local では、ローカルループバックインタフェースのアドレスまたはローカルホストをネットワークアドレス 127.0.0.1 で指定します。このファイル名は、起動ファイル内で指定します。他のファイルと同様、このマニュアルで使われていない名前を付けることもできます。

named.local ファイルの設定

named.local ファイルは、ネームサーバーのローカルループバックインタフェースを設定します。

例 **5–18** named.local ファイルの例

```
; SOA rec
0.0.127.in-addr.arpa. IN SOA sirius.doc.com sysop.centauri.doc.com (
                          1997071401       ;  serial number (YYYYMMDD##)
                           10800           ;  refresh every 3 hours
                           10800           ;  retry every 3 hours
                           604800          ;  expire after a week
                           86400 )         ;  TTL of 1 day
; Name Servers
0.0.127.in-addr.arpa.      IN  NS     sirius.doc.com
1                          IN  PTR  localhost.
```

named.local ファイルは通常、次の 3 つの要素で構成されています。

- 1 つの SOA (権限の開始) レコード。ゾーンの始まりを示す。named.local データファイルが置かれるホストの名前を含む
- 1 つまたは複数の NS (ネームサーバー) レコード。マスターおよびスレーブの DNS ネームサーバーを識別する。サーバー名およびドメイン名は完全指定する必要がある
- 1 つの PTR レコード。localhost に必要

$INCLUDE ファイル

インクルードファイルは、DNS データファイル内の $INCLUDE() 文で指定されているファイルのことです。$INCLUDE ファイルを使ってデータを型ごとに別々のファイルに分割しておくと便利です。

たとえば、データファイルに次のような行が含まれているとします。

```
$INCLUDE /etc/named/data/mailboxes
```

この行によって、/etc/named/data/mailboxes ファイルがその時点で読み込まれます。この例では、/etc/named/data/mailboxes が、$INCLUDE ファイルです。$INCLUDE ファイルは必要に応じて、必要な数だけ使用します。必要がなければ使う必要はありません。

データファイルのリソースレコード書式

DNS のデーモン in.named によって使用されるすべてのデータファイルは標準リソースレコード書式で書かれます。各 DNS データファイルは、必ずリソースレコードを含む必要があります。ここでは、DNS データファイルと各ファイルに含む必要があるリソースレコードについて説明します。

標準リソースレコード書式

標準リソースレコード書式では、データファイルの各行は、「リソースレコード」(RR) と呼ばれます。リソースレコードには空白で区切られた次のようなフィールドがあります。

namettlclassrecord-typerecord-specific-data

フィールドの順は常に同じですが、最初の 2 行はオプション (カッコ付きで示す) です。また、最後は *record-type* フィールドによって変化します。

name フィールド

最初のフィールドは、そのレコードに適用するドメイン名のフィールドです。RR でこのフィールドが空白のままであれば、デフォルトとして直前の RR の *name* フィールドの値が用いられます。

ゾーンファイルのドメイン名は、ドットで終わる完全指定名でも、相対名でもかまいません。相対名の場合、現在のドメインが付加されます。

ttl フィールド

2 番目のフィールドは、任意指定の有効期限フィールドです。このフィールドでは、データを破棄する前にデータベース内にデータをキャッシュしておく時間 (秒)、すなわちサーバーに新しい情報を次回要求するまでの時間を指定します。このフィールドを空白のままにすると、*ttl* には、権限の開始 (SOA) リソースレコードで指定された最小時間がデフォルトとして用いられます。

ttl の設定値があまりにも小さいと、サーバーはデータ更新のための要求を頻繁に繰り返します。逆に、*ttl* の設定値があまりにも大きいと、情報の変更がタイムリーに反映されなくなります。

ほとんどの場合、*ttl* の値は、初期値として 1 日 (86400) から 1 週間 (604800) の間に設定するとよいでしょう。そのあとで、実際の情報の変更の頻度にあわせて *ttl* の値を適切な値に変更してください。また、ほとんど変化することがないデータと関連しているということで *ttl* の値を大きく設定していた場合、そのデータが変更されるとわかった時点で、*ttl* の値を、データの変更が行われるまで小さな値 (3600 - 86400) にし、その後またもとの大きな値に戻すこともできます。

同じ名前、クラス、タイプを持つすべての RR では、*ttl* は同じ値に設定してください。

class フィールド

3 番目のフィールドは、レコードのクラスです。現在のところ、1 つのクラスだけがあります。それは、TCP/IP プロトコルのファミリーであることを示す IN です。

record-type フィールド

4 番目のフィールドでは、リソースレコードのタイプを記述します。RR にはたくさんのタイプがあります。最も一般的に使用されるタイプは、96 ページの「リソースレコードのタイプ」に説明されています。

record-specific-data フィールド

record-specific-data フィールドの内容は、そのリソースレコードのタイプによって異なります。

ネームフィールドとデータフィールドの大文字と小文字の区別は、ネームサーバーに読み込まれたときには保存されていますが、ネームサーバーのデータベースを比較して検索する際には大文字と小文字の区別はしません。ただし、これは将来的には変更される可能性がありますので、大文字と小文字の使用に関しては一貫性を保つように心がけてください。

特殊なリソースレコード文字

次に挙げる文字には特別な意味があります。

表 5–6 特殊なリソースレコード文字

文字	定義
.	ネームフィールドで、1 つのドットだけが指定された場合は現在のドメインを指す
@	ネームフィールドで、1 つの @だけが指定された場合は現在の起点を示す
. .	2 つのドットがネームフィールドに指定された場合は NULL ドメイン名を表す
\ *X*	*X* は数字 (0 - 9) 以外の任意の文字で、\ を付けることによって文字に特別な意味を持たせないようにする。たとえば、\ . を指定して、ラベルにドット文字を入れることができる。
\ *DDD*	各 *D* は一桁の数字で、\ を付けることによって *DDD* で表される 10 進数に対応する 8 進数を表現する。結果的に得られる 8 進数は、テキストとみなされ、そのテキストに特別な意味があるかどうかはチェックされない
( )	データが 1 行に収まらないとき、データをグループ化するのにカッコを使用する。結果的に、カッコの間では行の終わりが認識されない
;	セミコロンでコメントが開始する。その行でセミコロン以降は無視される

表 **5–6** 特殊なリソースレコード文字　*(続き)*

文字	定義
*	アスタリスクはワイルドカードを表す

ほとんどのリソースレコードには、現在の起点があり、名前の最後にドット (.) が付いていなければ、現在の起点が名前に追加されます。この機能は、マシン名などのデータに現在のドメイン名を追加する際には便利ですが、追加したくない場合には、問題を引き起こすかもしれません。データファイルを作成しているドメイン内に名前がない場合は、ピリオドで終わる完全指定名を使用してください。

制御エントリ

データファイルで制御エントリの行だけは標準 RR 書式に従わない行です。制御エントリには、`$INCLUDE()` と `$ORIGIN()` の 2 つのタイプがあります。

$INCLUDE

インクルード行は 1 列目の `$INCLUDE` で始まり、その後にファイル名 (`$INCLUDE` ファイル) が続きます。次の例に示すように、この機能は異なるタイプのデータを複数のファイルに分けるのに特に便利です。

```
$INCLUDE /etc/named/data/mailboxes
```

この行は、`/etc/named/data/mailboxes` ファイルを読み込む要求として解釈されます。`$INCLUDE` コマンドでは、異なるゾーンまたはツリーにデータは読み込まれません。このコマンドを使用しても、あるゾーンのデータが別々のファイルに格納されるだけです。たとえば、メールボックスのデータはこの機能を使ってホストデータとは別に保存できます。

`$INCLUDE` の文とファイルは必要に応じて必要な数だけ使用できます。

$ORIGIN()

`$ORIGIN` コマンドによって、データファイル内の起点を変更できます。この行は 1 列目から始まり、ドメイン名が続きます。これによって、相対ドメイン名 (たとえば、完全指定されていないドメイン名) の現在の起点を指定の名前に変更します。これは、1 つのデータファイルに複数のドメインを入れるのに便利です。

注 – 1 つのデータファイルに複数のゾーンを入れるために `$ORIGIN()` を使うことはできません。

$ORIGIN コマンドは、必要に応じて必要な数だけデータファイルで使用できます。$ORIGIN() 文がない場合、DNS データファイルのデフォルトの起点は、named.conf ファイルの master または slave の各行の 2 番目のフィールドに指定されているドメイン名となります。

リソースレコードのタイプ

最も一般的に使用されるリソースレコードのタイプを 表 5–7 に列挙します。 通常、表 5–7 に並んだ順で入力しますが、この順序は必須ではありません。

表 **5–7** 一般的に使用されるリソースレコードのタイプ

形式	説明
SOA	権限の開始
NS	ネームサーバー
A	インターネットアドレス (名前からアドレス)
PTR	ポインタ (アドレスから名前)
CNAME	正規名 (ニックネーム)
TXT	テキスト情報
WKS	既知サービス
HINFO	ホスト情報
MX	メール交換

権限の開始レコード (SOA)

例 5–19 に権限の開始 (SOA) リソースレコードの構文を示します。

例 **5–19** SOA レコードの書式

```
name class   SOA  origin person-in-charge  (  serial number
        refresh
 retry
 expire
 ttl)
```

SOA レコードは、ゾーンの開始を示します。次の SOA レコードでそのゾーンは終了します。以下に、SOA レコードの各フィールドについて説明します。

name

ゾーン名を指定するフィールドです。ゾーン名の後にはドットを付ける必要があります。たとえば : doc.com. は正しいですが、doc.com は誤りです。

class

アドレスクラスのフィールドです。たとえば IN はインターネットを示し、最も一般的に用いられるクラスです。

SOA

このリソースレコードのタイプを示します。

origin

このデータファイルが存在するホスト名のフィールドです。ホスト名の後にはドットを付ける必要があります。たとえば、dnsmaster.doc.com. は正しいですが、dnsmaster.doc.com は誤りです。

person-in-charge

ネームサーバーの責任者のメールアドレスのフィールドです。たとえば、kjd.nismaster.doc.com. です。この名前も終わりにドットを付ける必要があります。

serial

データファイルのバージョン番号のフィールドです。データを変更するたびにこの番号を増やしてください。スレーブサーバーは serial フィールドを使って、最後にマスターサーバーからデータファイルをコピーしてから変更があったかどうかを検出します。

refresh

更新が必要かどうかを調べるためにスレーブネームサーバーがマスターネームサーバーをチェックする頻度を秒単位で指定します。たとえば、7200 は 2 時間を意味します。

retry

リフレッシュのためのチェックに失敗した後、スレーブサーバーが再試行する時間を秒単位で指定します。

expire

リフレッシュが頻繁に行われない場合に、データの期限が切れる前に、スレーブネームサーバーがそのデータを使用する上限の時間を秒単位で指定します。

`ttl`

リソースレコードの `time-to-live` フィールドで使用されるデフォルトの秒数を指定します。このデフォルト値はリソースレコードで他に *ttl* が指定されていないときに適用されます。

`SOA` レコードは、各ゾーンに 1 つだけ指定してください。例 5–20に、`SOA` リソースレコードの例を示します。

例 **5–20** `SOA` リソースレコードの例

```
;name class          SOA     origin                  person-in-charge
doc.com. IN         SOA    dnsmaster.doc.com. root.nismaster.doc.com. (
                             101             ;Serial
                             7200         ;Refresh
                             3600         ;Retry
                             432000         ;Expire
                             86400)         ;Minimum             )
```

ネームサーバー (NS)

例 5–21 に、ネームサーバー (NS) リソースレコードの構文を示します。

例 **5–21** `NS` レコードの書式

domainname `[optional` *TTL*`]` *class* `NS` *name-server-name*

ネームサーバーレコードは、対象としているドメインを受け持つサーバーの名前を示します。*name* フィールドには、指定したネームサーバーからサービスを受けるドメインを指定します。*name* フィールドを指定しない場合は、デフォルトで、最後に指定された名前になります。`NS` レコードは、そのドメインのマスターサーバーとスレーブサーバーにそれぞれ 1 つずつ必要です。例 5–22 に、`NS` リソースレコードの例を示します。

例 **5–22** `NS` リソースレコードの例

```
;domainname    [TTL]      class    NS    nameserver
doc.com        90000    IN    NS     sirius.doc.com.
```

アドレス (A)

例 5–23 に、アドレス (A) リソースレコードの構文を示します。

例 **5–23** アドレス (`A`) レコードの書式

machinename `[optional` *TTL*`]` *class* `A` *address*

A レコードは、対象としているマシンのアドレスを示します。*name* フィールドは、ホスト名のフィールドです。*address* は、IP アドレスです。A レコードは、マシンの各アドレスに 1 つ必要です。つまり、ルーターやゲートウェイには 2 つ以上のエントリが必要で、IP アドレスを含む個々のエントリが各ネットワークインタフェースに割り当てられます。

例 **5–24** アドレスレコードの例

```
;machinename    [TTL]    class    A    address
sirius        IN        A    123.45.6.1
```

ホスト情報 (HINFO)

例 5–25 に、ホスト情報 (HINFO) リソースレコードの構文を示します。

例 **5–25** HINFO レコードの書式

```
[optional name] [optional TTL]      class     HINFO hardware     OS
```

HINFO には、ホスト固有のデータが含まれ、ハードウェアとこのホストで動作しているオペレーティング環境を指定します。マシン名や hardware フィールドのエントリに空白を含めるには、エントリを引用符で囲む必要があります。*name* フィールドでは、ホスト名を指定します。名前が指定されなければ、in.named での最後のホストがデフォルトになります。HINFO レコードは、各ホストに 1 つ必要です。例 5–26に、HINFO リソースレコードの例を示します。

例 **5–26** HINFO リソースレコードの例

```
;[name]    [TTL] class HINFO   hardware    OS
                 IN    HINFO   Sparc-10    UNIX
```

注意 – HINFO フィールドにはネットワーク上のマシンについての情報が含まれているので、多くのサイトでは、この情報はセキュリティ上危険であると考えられ、現在はほとんど使われていません。

既知サービス (WKS)

例 5–27 に、既知サービス (WKS) リソースレコードの構文を示します。

例 **5–27** WKS レコードの書式

```
[Optional name] [TTL] class WKS address protocol-list-of-services
```

WKS レコードは、指定されたアドレスの特定のプロトコルでサポートされているよく知られたサービスを示します。サービスのリストとポート番号は、services データベースで指定されたサービスのリストから得られます。WKS レコードは、各アドレスの各プロトコルに 1 つだけ存在している必要があります。例 5–28 に、WKS リソースレコードの例を示します。

例 **5–28** WKS リソースレコードの例

```
;[name]    [TTL]    class    WKS    address      protocol-list-of-services
altair        IN    WKS    123.45.6.1     TCP ( smtp discard rpc
 sftp uucp-path systat daytime
 netstat qotd nntp doc.com )
```

注意 – WKS レコードは任意指定です。セキュリティ上の理由から、ほとんどのサイトでは現在この情報は提供していません。

正規名 (CNAME)

例 5–29 に、正規名 (CNAME) リソースレコードの構文を示します。

例 **5–29** CNAME レコードの書式

nickname [optional *TTL*] *class CNAME canonical-name*

CNAME は、正規名のニックネームまたはエイリアスを示します。ニックネームは一意である必要があります。他のすべてのリソースレコードは、ニックネームではなく正規名と結びつけるようにする必要があります。ニックネームを作成して他のリソースレコードで使用することはしないでください。ニックネームは、マシン名が変更されたけれども古い名前でのアクセスを許可する移行期に特に有用です。また、ニックネームはメールサーバーなど特定の目的で使用するマシンを識別するためにも使用できます。例 5–30に、CNAME リソースレコードの例を示します。

例 **5–30** CNAME リソースレコードの例

```
;nickname      [TTL]    class    CNAME    canonical-name
mailhost        IN    CNAME antares.doc.com
```

ポインタレコード (PTR)

例 5–31に、PTR リソースレコードの構文を示します。

例 **5–31** PTR レコードの書式

special-name [optional *TTL*] *class PTR-real-name*

ポインタレコードを使用すると、特殊な名前でドメイン内の他の場所を指すことができます。次の例では、アドレス (特殊な名前) を実名に変換するために、PTR は主に in-addr.arpa. レコードで使用されます。アドレスを解釈するときはドメインが完全指定であれば、指定する必要があるのはマシンの識別番号だけです。PTR の名前は、ゾーンに対して一意である必要があります。例 5–32の PTR レコードでは、特殊なドメイン in-addr.arpa. に対して逆ポインタを設定します。

例 **5–32** PTR リソースレコードの例

```
;special name   [TTL]    class    PTR-real-name
1            IN    PTR sirius.doc.com.
```

メール交換 (MX)

例 5–33 に、メール交換 MX リソースレコードの構文を示します。

例 **5–33** MX レコードの書式

name [optional *TTL*] *class* MX *preference-value mailer-exchanger*

MX リソースレコードは、あるドメインまたはドメイン内の特定のマシンにメールを配信するマシンを指定するために使用します。対象としている名前に対して複数の MX リソースレコードが作成される場合もあります。例 5–34では、Seismo.CSS.GOV (完全指定のドメイン名) は、Munnari.OZ.AU にメールを配信するメールゲートウェイです。ネットワーク上の他のマシンは、Munnari に直接メールを配信できません。Seismo と Munnari は、専用接続を持っている場合も、異なるトランスポート媒体を使用している場合もあります。*preference-value* フィールドでは、メールプログラムが従う順序を指定します。このフィールドは、単一のマシンにメールを配信する方法が複数ある場合に指定します。値が 0 (ゼロ) は最優先であることを意味します。同じ名前に対して複数の MX リソースレコードがある場合、そのレコードの優先値 (*preference-value*) は同じであることも、同じでないこともあります。

メールを配信するために、MX レコードでワイルドカードであるアスタリスク (*) を名前に使うこともできます。あるドメイン宛のメールがすべてリレー経由で配信されるサーバーがネットワーク上にはよくあります。例 5–34では、foo.com ドメイン内のホスト宛のメールはすべて RELAY.CS.NET を経由して送られます。これを指定するには、ワイルドカードを用いて MX リソースレコードを作成し、*.foo.com のメール交換が RELAY.CS.NET. により行われることを指定します。アスタリスクは foo.com のどのホストまたはサブドメインにも一致します。ただし、foo.com 自体には一致しません。

例 **5–34** MX リソースレコードの例

```
;name          [TTL]    class        MX     preference mailer-exchanger
Munnari.OZ.AU.    IN        MX     0    Seismo.CSS.GOV.
foo.com.        IN        MX     10    RELAY.CS.NET.
*.foo.com.    IN        MX     20    RELAY.CS.NET.
```

第 6 章

DNS の障害追跡 (参照情報)

この章では、DNS に関する一般的な問題とその解決方法について説明します。

クライアントは名前でマシンを見つけられるが、サーバーは見つけられない

「症状」

DNS クライアントは、IP アドレスかホスト名でマシンを見つけられますが、サーバーは IP アドレスでしか見つけることができません。

「考えられる原因と対策」

サーバーの `nsswitch.conf` ファイルの `hosts` 行から `DNS` を省略したために発生する可能性があります。たとえば、不完全な `hosts` 行は、`hosts: files` のようになります。

DNS の使用時は、各マシンの `nsswitch.conf` ファイルの `hosts` レコード内に `dns` を入れる必要があります。たとえば、次のようにします。

```
hosts: dns nisplus files
```

または

```
hosts: nisplus dns files
```

変更が反映されないか、その効果が一定しない

「症状」

マシンやサーバーを追加または削除しても、変更が認識されないか反映されません。あるいは、変更が認識されたり認識されなかったりします。

「考えられる原因」

考えられる最初の原因は、変更を加えた後にマスターサーバー上の SOA のシリアル番号を増やし忘れたことです。新しいSOA 番号がないので、スレーブサーバーはそのデータをマスターサーバーのデータと一致させるためのデータ更新を行いません。このため、古い未変更のデータファイルを使用しています。

この他に考えられる原因は、マスターサーバー上の 1 つ以上のデータファイルの SOA のシリアル番号が、スレーブサーバー上の対応するシリアル番号よりも小さい値に設定されたことです。この状態はたとえば、マスターサーバー上のファイルを削除してから、ある種の入力ファイルを使って最初から作成し直した場合に発生します。

考えられる 3 番目の原因は、マスターサーバーのデータファイルに変更を加えた後で、HUP 信号をマスターサーバーに送信し忘れたことです。

「診断と対策」

まず、変更したデータファイルの SOA のシリアル番号とスレーブサーバー上の対応するファイルをチェックします。

- マスターサーバーのファイルの SOA シリアル番号がスレーブサーバーのファイルのシリアル番号と同じかそれ以下の場合は、マスターサーバーのファイルのシリアル番号を増やしてスレーブサーバーのファイルの番号よりも大きくなるようにします。たとえば、両方のファイルの SOA のシリアル番号が 37 の場合は、マスターサーバーのファイルの番号を 38 に変更します。次回、スレーブサーバーがマスターサーバーをチェックすると、新しいデータがロードされます。マスターサーバーに、スレーブサーバーへのデータ転送を即座に強制するユーティリティがあります。このユーティリティがある場合には、マスターサーバーのチェックを待たずにスレーブサーバーを更新できます。
- 最新の `named` *nnnn* `restarted` または、`named` *nnn* `reloading nameserver` エントリに対する `syslog` 出力を確認します。そのエントリのタイムスタンプが、ファイルへの変更を終了した時間よりも前の場合には、サーバーをリブートするか、79 ページの「`in.named` に DNS データを強制的に再度読み込ませる」で説明しているように、新しいデータの読み取りを強制します。

DNS クライアントが短縮名を検索できない

「症状」

クライアントは完全指定名は検索できますが、短縮名は検索できません。

「考えられる原因と対策」

クライアントの `/etc/resolv.conf` ファイルで、ドメイン名の最後にスペースがないかをチェックします。スペースやタブはドメイン名の最後では使用できません。

逆ドメインデータがスレーブサーバーに正しく転送されない

ゾーンのドメイン名の付いたデータは、ゾーンのマスターサーバーからゾーンのスレーブサーバーに正しく転送されますが、逆ドメインデータは転送されません。つまり、スレーブサーバーの `host.rev` ファイルがマスターサーバーから正しく更新されていません。

「考えられる原因」

スレーブサーバーの起動ファイルの構文エラー

「診断と対策」

スレーブサーバーの起動ファイルをチェックします。マスターサーバーの IP アドレスが、ホストデータの場合と同じように、逆ゾーンエントリに対してリストされていることを確認してください。

サーバーが失敗してゾーンが期限切れになる

スレーブサーバーがそのマスターサーバーから更新を得られないときは、「master unreachable」というメッセージがログに記録されます。問題が修正されない場合、スレーブサーバーはゾーンを期限切れにして、クライアントからの要求への応答を停止します。この状況が発生すると、「server failed」というメッセージが表示されます。

「症状」

- syslog に「Masters for slave zone *domain* unreachable」というメッセージが表示される
- syslog に「slave zone *domain* expired」というメッセージが表示される
- ユーザーに「*** *domain* Can't find *name*:server failed」というメッセージが表示される

問題がスレーブサーバーにある場合、一部のユーザーはマスターサーバーから DNS 情報を獲得できるため問題なく操作できます。

「考えられる原因」

これらの問題に対して考えられる主な原因は 2 つあります。1 つはネットワーク障害であり、もう 1 つはスレーブサーバーの起動ファイル内に指定したマスターサーバーの IP アドレスが間違っていることです。

「診断と対策」

- スレーブサーバーの構成ファイルに、マスターサーバーの IP アドレスが正しく設定されているかどうか確認します。次の行をチェックしてください。

```
zone "someone" {
                                    type slave;
file "somefile":
master [IPaddress; };
};
```

hosts ファイルで指定したマスターサーバーの IP アドレスが実際の IP アドレスと一致することを確認してください。IP アドレスが間違っている場合は、それを修正してからスレーブサーバーをリブートします。

- マスターサーバーの IP アドレスが正しい場合は、そのアドレスを ping して、マスターサーバーが正しく起動しているかどうかを確認します。たとえば、マスターサーバーを IP アドレス 192.168.0.1 で ping する場合は、次のように入力します。

  % **ping 192.168.0.1 -n 10**

- マスターサーバーが ping に応答しない場合は、マスターサーバーが正しく起動しているかどうかを確認します。
- マスターサーバーが正しく起動している場合は、ps を使用して、マスターサーバーが named を実行しているかどうかを確認します。named を実行していない場合は、リブートします。
- マスターサーバーが named を正しく実行している場合は、ネットワークに障害が発生している可能性があります。

rlogin、rsh、ftp の問題

「症状」

- インターネットで、別のドメインのマシンに rlogin を試みると、パスワードの入力を求められる
- インターネットで、別のドメインのマシンに ftp を試みると、アクセスを拒否される
- ローカルネットワーク上のマシンに rlogin や rsh を試みると、アクセスを拒否される

「考えられる原因」

- ユーザーが作業しているマシンの PTR レコードがマスターサーバーの hosts.rev ファイルにありません。
- hosts.rev ファイル内のサブドメインがないか、不正な委任が行われています。

「診断と対策」

該当するhosts.rev ファイルをチェックして、ユーザーのマシン用の PTR レコードが存在することを確認します。たとえば、192.168.0.1 の IP アドレスを持つマシン altair.doc.com で作業をしている場合は、doc.com マスターサーバーの doc.rev ファイルに次のようなエントリを追加する必要があります。

```
46      IN      PTR     altair.doc.com.
```

レコードがない場合は、hosts.rev ファイルに追加してからサーバーをリブートするか、79 ページの「in.named に DNS データを強制的に再度読み込ませる」の指示に従ってデータをロードし直します。

hosts.rev ファイルの NS エントリをチェックおよび修正してから、サーバーをリブートするか、79 ページの「in.named に DNS データを強制的に再度読み込ませる」の指示に従ってデータをロードし直します。

その他の DNS 構文エラー

「症状」

次のような表現が含まれるコンソールまたは `syslog` のエラーメッセージは、たいてい DNS データや起動ファイルの構文エラーによるものです。

- `No such...`
- `Unknown field...`
- `Non-authoritative answer:`
- `Database format error...`
- `illegal` または `(illegal)`
- `error receiving zone transfer`

関連ファイルにスペルや構文のエラーがないかどうか チェックしてください。

一般的な構文エラーは、ドメイン名で後ろに付けるドットの誤用 (禁じられている場合に使い、必要な場合に使わないなど) が原因で発生します。75 ページの「DNS サーバーの設定」を参照してください。

パート III　NIS の設定と管理

ここでは、NIS ネームサービスの概要、および Solaris オペレーティング環境での NIS の設定、管理、障害追跡について説明します。

第 7 章

ネットワーク情報サービス (NIS) (概要)

この章では、ネットワーク情報サービス (NIS) の概要を説明します。

NIS とは分散型ネームサービスであり、ネットワーク上のオブジェクトおよびリソースを識別し、探索するメカニズムです。NIS は、ネットワーク全体の情報に関する一様な記憶領域と検索方法を、トランスポートプロトコルやメディアに依存しない形式で提供します。

この章の内容は次のとおりです。

NIS の概要

システム管理者は、NIS を実行することにより、「マップ」 と呼ばれる管理データベースをさまざまなサーバー (「マスター」と「スレーブ」) に分散させることができます。 さらに、これらの管理データベースを一元管理により自動的かつ確実な方法で更新できるため、どのクライアントもネットワーク全体を通して一貫した方法で同じネームサービス情報を共有できます。

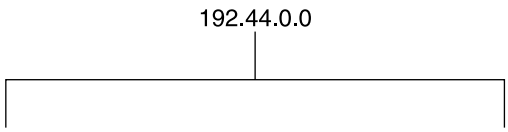
NIS は DNS とは別に開発されたため、その目的もやや異なっています。DNS が数値 IP アドレスの代わりにマシン名を使うことによって、通信を簡略化することに焦点を当てているのに対して、NIS は、多様なネットワーク情報を集中管理することによりネットワーク管理機能を高めることに焦点を当てています。NIS には、マシン名とアドレスだけでなく、ユーザー、ネットワークそのもの、ネットワークサービスについての情報も格納されます。このようなネットワーク「情報」の集まりを NIS の「名前空間」と呼びます。

注 – 「マシン」名の代わりに「ホスト」名が使われることがあります。この解説では「マシン」名が使われていますが、一部の画面メッセージまたは NIS マップ名では「ホスト」名または「マシン名」が使われています。

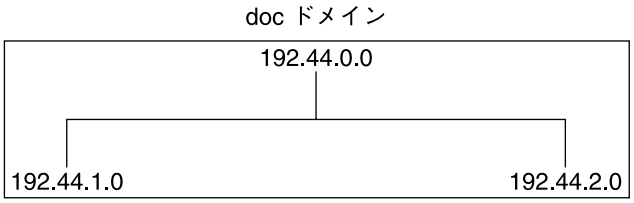
NIS アーキテクチャ

NIS はクライアントサーバー方式を使用します。NIS サーバーが NIS のクライアントへサービスを提供します。主サーバーは「マスター」サーバーと呼ばれ、信頼性を保証するためにバックアップつまり「スレーブ」サーバーを持っています。マスターサーバーとスレーブサーバーは、NIS の情報検索ソフトウェアを使い、NIS のマップを格納します。

NIS はドメインを使用して、マシン、ユーザー、およびネットワークを自分の名前空間に配置します。しかし、ドメイン階層を使用しないため、 NIS の名前空間はフラットになっています。



したがって、上記のような物理ネットワークは、次のように 1 つの NIS ドメインに配置されます。



NIS だけを使っても、NIS ドメインをインターネットに直接接続することはできません。ただし、NIS を使用してインターネットへも接続したいと希望する組織では、NIS と DNS を組み合せることができます。その場合、NIS を使用してすべてのローカル情報を管理し、DNS を使用してインターネットのホストを検索できます。NIS は、NIS マップで情報が見つからない場合にホスト検索の機能を DNS へ転送する転送サービス機能を持っています。Solaris オペレーティング環境では、ホスト検索要求

を DNS だけに転送する、DNS で情報が見つからなければ次に NIS に転送する、あるいは NIS で情報が見つからなければ次に DNS に転送する、という切り替えも `nsswitch.conf` ファイルで設定できます。 詳細については、第 2 章を参照してください。

NIS マシンのタイプ

NIS マシンには、次の 3 つのタイプがあります。

- マスターサーバー
- スレーブサーバー
- NIS サーバーのクライアント

NIS クライアントにはどのマシンでもなれますが、NIS サーバー (マスターまたはスレーブ) となるのはディスクが装備されているマシンだけです。一般にサーバーは、多くの場合はそのサーバー自身のクライアントでもあります。

NIS サーバー

NIS サーバーは、FNS ファイルサーバーと同じマシンである必要はありません。

NIS サーバーには、マスターサーバーとスレーブサーバーがあり、マスターサーバーとして指定されているマシンには、NIS 管理者が必要に応じて作成、更新する一群のマップが保存されます。各 NIS ドメインには、マスターサーバーが 1 つだけ存在している必要があります。マスターサーバーは、パフォーマンスの低下を最小限に抑えながら NIS の更新をスレーブサーバーに伝播できます。

システム管理者は、ドメインに別の NIS サーバーをスレーブサーバーとして指定できます。各スレーブサーバーには、マスターサーバーの NIS マップセットの完全なコピーが存在します。マスターサーバーの NIS マップが更新されると、必ずこれらの更新がスレーブサーバーに伝播されます。スレーブサーバーは、マスターサーバーからの要求のオーバーフローに対処して、「サーバー使用不可」エラーを最小限に抑えることができます。

通常、システム管理者はすべての NIS マップに対してマスターサーバーを 1 つ指定します。 ただし、各 NIS マップ内ではマスターサーバーのマシン名が符合化されているので、異なる複数のマップに対して異なる複数のサーバーを、マスターサーバーやスレーブサーバーとして動作するように指定することもできます。管理の複雑さを最小限に抑えるには、1 つのドメイン内で作成されるすべてのマップに対して、マスターサーバーを 1 つだけ指定します。 この章の例では、1 つのサーバーがドメイン内のすべてのマップのマスターサーバーとなっています。

NIS クライアント

NIS クライアントでは、サーバー上のマップのデータを要求するプロセスが動作します。各 NIS サーバーに保存されている情報は同じであるはずなので、クライアントではマスターサーバーとスレーブサーバーの区別は行われません。

NIS の要素

NIS ネームサービスは、次の要素から構成されています。

- ドメイン (130 ページの「NIS ドメイン」を参照)
- マップ (131 ページの「NIS マップ」を参照)
- デーモン (130 ページの「NIS デーモン」を参照)
- ユーティリティ (131 ページの「NIS ユーティリティ」を参照)
- NIS コマンドセット (135 ページの「NIS 関連コマンド」を参照)

NIS ドメイン

NIS「ドメイン」は、共通の NIS マップセットを共有しているマシンの集合です。各ドメインにはドメイン名が指定されており、共通の NIS マップセットを共有している各マシンがそのドメインに属しています。

どのマシンも指定されたドメインに属することができます。ただしこれは、そのドメインのマップに対するサーバーが同一ネットワーク上に存在する場合に限ります。NIS クライアントマシンは、ブートプロセス中にドメイン名を取得して、NIS サーバーにバインドします。

NIS デーモン

NIS サービスは、表 7–1 に示す 5 つのデーモンで提供されます。

表 **7–1** NIS デーモン

デーモン	機能
`ypserv`	サーバープロセス
`ypbind`	バインドプロセス
`ypxfr`	高速マップ転送

表 7–1 NIS デーモン　(続き)

デーモン	機能
rpc.yppasswdd	NIS パスワード更新デーモン ** 下の注を参照 **
rpc.ypupdated	他のマップ (publickey など) を更新する

注 – rpc.yppasswdd は、r で始まるすべてのシェルを制限付きとみなします。 つまり、r で始まるシェルを持っているユーザーが制約を受けます。たとえば、/bin/rksh で作業しているユーザーはそのシェルを別のシェルに変更できません。r で始まるシェルを持っているが、そのような制約を受けたくない場合は、第 10 章の対処方法を参照してください。

NIS ユーティリティ

NIS サービスは、表 7–2 に示す 9 つのユーティリティでサポートされています。

表 7–2 NIS ユーティリティ

ユーティリティ	機能
makedbm	NIS マップの dbm ファイルを作成する
ypcat	マップのデータを一覧表示する
ypinit	NIS データベースの作成、インストール、および NIS クライアントの ypservers リストの初期化を行う
yppmatch	マップの特定エントリを検索する
yppoll	サーバーからマップ順序番号を取得する
yppush	データを NIS マスターサーバーから NIS スレーブサーバーに伝播する
ypset	特定サーバーにバインドを設定する
ypwhich	NIS サーバー名およびニックネーム変換テーブルを表示する
ypxfr	NIS マスターサーバーから NIS スレーブサーバーにデータを転送する

NIS マップ

NIS マップの情報は、ndbm フォーマットで保存されます。マップファイルのフォーマットについては、ypfiles(4) と ndbm(3C) のマニュアルページに説明されています。

NIS マップは、UNIX の /etc ファイルおよび他の構成ファイルを置換するように設計されているので、名前およびアドレスよりはるかに多くの情報を保存できます。NIS が動作しているネットワーク上では、各 NIS ドメインの NIS マスターサーバーは、照会されるドメイン内の他のマシンの NIS マップセットを保持します。NIS スレーブサーバーは、NIS マスターサーバーのマップのコピーを保持します。NIS クライアントマシンは、マスターサーバーまたはスレーブサーバーから名前空間情報を取得できます。

NIS マップは、本質的には「キー」およびキーに関する情報の 2 列からなるテーブルです。NIS は、キーを検索してクライアントに関する情報を見つけます。各マップでは異なるキーが使われるので、一部の情報はいくつかのマップに保存されます。たとえば、マシン名とアドレスは、hosts.byname と hosts.byaddr という 2 つのマップに保存されます。サーバーがマシンの名前を持っており、そのマシンのアドレスを見つける必要がある場合は、サーバーは hosts.byname マップを調べます。サーバーがマシンのアドレスを持っており、そのマシンの名前を見つける必要がある場合は、サーバーは hosts.byaddr マップを調べます。

NIS Makefile は、インストール時に NIS サーバーとして指定されたマシンの /var/yp ディレクトリに保存されます。このディレクトリで make を実行すると、makedbm が入力ファイルからデフォルトの NIS マップを作成または更新します。

注 – マップは必ずマスターサーバー上で作成してください。スレーブサーバーで作成したマップはマスターサーバーに自動的に格納されません。

デフォルトの NIS マップ

Solaris オペレーティング環境では、NIS マップのデフォルトセットが提供されます。システム管理者は、これらのマップをすべて使用することも一部だけを使用することもできます。また、他のソフトウェア製品のインストール時にシステム管理者が作成または追加したマップもすべて NIS で使用できます。

NIS ドメインのデフォルトのマップは、各サーバーの /var/yp/*domainname* ディレクトリに入っています。たとえば、test.com ドメインに属しているマップは、各サーバーの /var/yp/test.com ディレクトリにあります。

表 7–3 には、デフォルトの NIS マップ、これらの NIS マップに存在する情報、および NIS 動作時にソフトウェアが対応する管理ファイルを調べているか否かが示されています。

表 **7–3** NIS マップに関する説明

マップ名	対応する **NIS** 管理ファイル	説明
bootparams	bootparams	ブート時にクライアントが必要とするファイルのパス名 (ルート、スワップ、その他) を含む
ethers.byaddr	ethers	マシン名と Ethernet アドレスを含む。Ethernet アドレスはマップ内のキー
ethers.byname	ethers	ethers.byaddr と同じ。ただしキーは、Ethernet アドレスではなくマシン名
group.bygid	group	グループセキュリティ情報を含む。キーはグループ ID
group.byname	group	グループセキュリティ情報を含む。キーはグループ名
hosts.byaddr	hosts	マシン名と IP アドレスを含む。キーは IP アドレス
hosts.byname	hosts	マシン名と IP アドレスを含む。キーはマシン (ホスト) 名
mail.aliases	aliases	エイリアスとメールアドレスを含む。キーはエイリアス
mail.byaddr	aliases	メールアドレスとエイリアスを含む。キーはメールアドレス
netgroup.byhost	netgroup	グループ名、ユーザー名、マシン名を含む。キーはマシン名
netgroup.byuser	netgroup	netgroup.byhost と同じ。ただし、キーはユーザー名
netgroup	netgroup	netgroup.byhost と同じ。ただし、キーはグループ名
netid.byname	passwd, hosts group	UNIX スタイルの認証に使用される。マシン名とメールアドレスを含む (ドメイン名も含む)。netid ファイルがある場合には、他のファイルを使用して利用できるデータの他にそれが参照される
netmasks.byaddr	netmasks	IP 送出時に使用するネットワーを含む。キーはアドレス
networks.byaddr	networks	システムに認識されているネットワーク名、および IP アドレスを含む。キーは IP アドレス

表 **7–3** NIS マップに関する説明　(続き)

マップ名	対応する **NIS** 管理ファイル	説明
networks.byname	networks	networks.byaddr と同じ。ただし、キーはネットワーク名
passwd.adjunct. byname	passwd と shadow	C2 クライアント用の監査情報と隠蔽されたパスワード情報を含む
passwd.byname	passwd と shadow	パスワード情報を含む。キーはユーザー名
passwd.byuid	passwd と shadow	passwd.byname と同じ。ただし、キーはユーザー ID
protocols.byname	protocols	システムに認識されているネットワークプロトコルを含む
protocols.bynumber	protocols	protocols.byname と同じ。ただし、キーはプロトコル番号
rpc.bynumber	rpc	システムに認識されている RPC のプログラム番号と名前を含む。キーは RPC のプログラム番号
services.byname	services	ネットワークに認識されているインターネットサービスを一覧表示する。キーはポートまたはプロトコル
services.byservice	services	ネットワークに認識されているインターネットサービスを一覧表示する。キーはサービス名
ypservers	なし	ネットワークに認識されている NIS サーバーを一覧表示する

新しい ipnodes マップ (ipnodes.byaddr および ipnodes.byname) が、NIS に追加されました。このマップには、IPv4 アドレスと IPv6 アドレスの両方が格納されます。 ipnodes(4) のマニュアルページを参照してください。 NIS クライアントとサーバーは、IPv4 または IPv6 のどちらかの RPC トランスポートを使用して通信することができます。

NIS マップの使用

NIS を使うと、/etc ファイルシステムを使った場合に比べ、ネットワークデータベースの更新がはるかに簡単になります。/etc ファイルシステムではネットワーク環境を更新するたびに各マシンの管理 /etc ファイルを変更する必要がありましたが、NISではこのような操作を行う必要はありません。

たとえば、NIS が動作しているネットワークに新しいマシンを追加する場合は、マスターサーバーの入力ファイルを更新し、make を実行するだけです。これで、hosts.byname および hosts.byaddr マップが自動的に更新されます。次に、これらのマップはすべてのスレーブサーバーに転送され、ドメインのすべてのクライアントマシン、およびこれらのクライアントマシンのプログラムはこれらのマップを使用することが可能になります。クライアントマシンまたはアプリケーションがマシン名またはアドレスを要求すると、NIS サーバーは必要に応じて hosts.byname または hosts.byaddr マップを参照し、要求された情報をクライアントに送信します。

ypcat コマンドを使うと、マップの値を表示できます。ypcat の基本な使用形式は、次のとおりです。

% **ypcat** *mapname*

mapname は、調べたいマップ名またはその「ニックネーム」です。ypservers の場合のようにマップがキーだけで構成されている場合は、ypcat -k と入力してください。ypcat -k と入力しない場合は、空白行が出力されます。 ypcat の他のオプションについては、ypcat(1) のマニュアルページに説明されています。

ypwhich コマンドを使うと、どのサーバーが特定マップのマスターサーバーなのかを判断できます。次のように入力します。

% **ypwhich -m** *mapname*

mapname は、検索するマスターサーバーのマップ名またはニックネームです。*mapname* を入力すると、マスターサーバー名が表示されます。詳細については、ypwhich(1) のマニュアルページを参照してください。

NIS マップのニックネーム

「ニックネーム」は、マップのフルネームの別名です。使用可能なマップのニックネーム (たとえば、passwd.byname の場合は passwd) を一覧表示するには、ypcat -x または ypwhich -x と入力してください。

ニックネームは、/var/yp/nicknames ファイルに保存されています。/var/yp/nicknames ファイルには、マップのニックネームとフルネームが 1 つの空白で区切られて入っています。ニックネームのリストは、追加または更新できます。ニックネーム数は現在、500 に制限されています。

NIS 関連コマンド

NIS サービスには、特殊なデーモン、システムプログラム、コマンドが含まれています。これらのコマンドについては次の表にまとめてあります。

表 **7–4** NIS コマンドについてのまとめ

コマンド名	説明
ypserv	NIS クライアントが要求する NIS マップの情報を提供する。ypserv は、完全なマップセットを備えた NIS サーバー上で動作するデーモン。NIS サービスが機能するには、少なくとも 1 つの ypserv デーモンがネットワークに存在する必要がある
ypbind	クライアントに NIS サーバーバインド情報を提供する。ypbind は、要求元クライアントのドメイン内のマップにサービスを提供する ypserv プロセスを見つけてバインドを行う。ypbind はすべてのサーバーとクライアント上で実行される必要がある
ypinit	自動的に入力ファイルから NIS サーバーのマップを作成する。ypinitはまた、クライアント上に /var/yp/binding/*domain*/ypservers 初期ファイルを作成する際にも使用される。NIS マスターサーバーおよび NIS スレーブサーバーを初めて設定する場合は、ypinit を使用する
make	Makefile を読み込みこむことで NIS マップを更新する (make を /var/yp ディレクトリで実行した場合)。make を使うと、入力ファイルに基づいてすべてのマップを更新したり、個々のマップを更新したりできる。NIS の make の機能については、ypmake(1M) のマニュアルページに説明されている
makedbm	makedbm は入力ファイルを取得し、これを dbm.dir および dbm.pag ファイルに変換する (これらのファイルは、NIS がマップとして使用できる有効な dbm ファイル)。また、makedbm -u と入力すると、マップを分解できるため、システム管理者はマップを構成するキーと値のペアを参照できる
ypxfr	NIS 自体をトランスポートメディアとして使い、NIS マップを遠隔サーバーから /var/yp/*domain* ローカルディレクトリに取り込む。システム管理者は ypxfr を対話形式で実行したり、crontab ファイルから定期的に実行したりできる。また、ypxfr が ypserv によって呼び出されると、転送が開始される
ypxfrd	ypxfr 要求 (一般にスレーブサーバーで発生する) に対してマップ転送サービスを提供する。ypxfr は、マスターサーバー上でだけ動作する
yppush	NIS マップの新バージョンを NIS マスターサーバーからそのスレーブサーバーにコピーする。yppush の実行は、NIS マスターサーバー上で行う
ypset	指定された NIS サーバーにバインドするように ypbind プロセスに要求する。 ypset は、セキュリティの関係上、通常のオペレーションで気軽に使用できるようには設計されていない。したがって、ypset はできる限り使用しない。ypbind プロセスの ypset および ypsetme オプションについては、ypset(1M) および ypbind(1M) のマニュアルページを参照

表 7–4 NIS コマンドについてのまとめ　(続き)

コマンド名	説明
yppoll	指定されたサーバー上で NIS マップのどのバージョンが動作しているかを通知する。yppoll はまた、NIS マップのマスターサーバーを一覧表示する
ypcat	NIS マップの内容を表示する。
ypmatch	NIS マップ内の指定された 1 つ以上のキーの値を出力する。システム管理者は、NIS サーバーマップのバージョンを指定することはできない
ypwhich	クライアントが現在どの NIS サーバーを使用して NIS サービスを取得しているかを表示する。また、-m *mapname* オプションを指定してこのコマンドを起動した場合は、どの NIS サーバーが各マップのマスターサーバーであるかが表示される。-m だけを指定した場合は、使用可能なすべてのマップ名、およびこれらのマップのマスターサーバーが表示される

NIS のバインド

NIS クライアントは、バインドプロセスにより NIS サーバーから情報を取得します。バインドプロセスは、サーバーリストおよび同報通信という 2 つのモードのどちらかで動作できます。

- サーバーリストモード。サーバーリストモードでは ypbind プロセスは、/var/yp/binding/*domain*/ypservers リストでドメイン内のすべての NIS サーバー名を調べます。ypbindプロセスは、このファイルに存在するサーバーにだけバインドされます。このファイルは、ypinit -c を実行すると作成されます。
- 同報通信モード。ypbind プロセスはまた、RPC 同報通信を使ってバインドを開始できます。同報通信は、それ以上配信されない唯一のローカルサブネットイベントです。したがって、クライアントと同じサブネット上に少なくとも 1 つのサーバー (マスターまたはスレーブ) が存在しなければなりません。サーバーは、異なる複数のサブネット上に存在できます (マップはサブネット境界を超えて伝播されるため) 。サブネット環境での 1 つの一般的方法は、NIS サーバーとしてサブネットルーターを使用することです。この方法を使用すると、ドメインサーバーはどちらかのサブネットインタフェース上でクライアントにサービスを提供できます。

サーバーリストモード

サーバーリストモードでは、バインドプロセスは次のように動作します。

1. NIS マップで提供された情報を必要とする、NIS クライアントマシン上で動作しているプログラムが、ypbind にサーバー名を要求します。

2. ypbind が、/var/yp/binding/*domainname*/ypservers ファイルを調べてドメインの NIS サーバーリストを見つけます。
3. ypbind が、NIS サーバーリストの先頭サーバーへのバインドを開始します。先頭サーバーが応答しない場合、ypbind はサーバーが見つかるまで、あるいは NIS サーバーリストの最後に達するまで、2 番目以降のサーバーへのバインドを順に試みます。
4. ypbind が、どのサーバーにアクセスすべきかをクライアントプロセスに通知します。次に、クライアントプロセスが直接、サーバーに要求を送信します。
5. NIS サーバー上の ypserv デーモンが、該当するマップを調べて要求を処理します。
6. ypserv デーモンが、要求された情報をクライアントに送り返します。

同報通信モード

同報通信モードでは、バインドプロセスは次のように動作します。

1. 同報通信オプション (broadcast) が設定されている状態で ypbind を起動する必要があります。
2. ypbind が、RPC 同報通信を送出して NIS サーバーを探索します。

注 – このようなクライアントをサポートするには、NIS サービスを要求している各サブネット上に 1 つの NIS サーバーが存在する必要があります。

---

3. ypbind が、同報通信に応答する先頭サーバーへのバインドを開始します。
4. ypbind が、どのサーバーにアクセスすべきかをクライアントプロセスに通知します。次に、クライアントプロセスが直接、サーバーに要求を送信します。
5. NIS サーバー上の ypserv デーモンが、該当するマップを調べて要求を処理します。
6. ypserv デーモンが、要求された情報をクライアントに送り返します。

通常、いったんクライアントがサーバーにバインドされると、何らかの原因でバインドが解除されるまではクライアントはサーバーにバインドされたままになります。たとえば、サーバーがサービスを提供できなくなると、このサーバーがサービスを提供していたクライアントは、新しいサーバーにバインドされます。

どの NIS サーバーが現在、特定クライアントにサービスを提供しているかを調べる場合は、次のコマンドを入力してください。

%**ypwhich** *machinename*

machinename は、クライアント名です。マシン名が指定されていない場合は、ypwhich はデフォルトとしてローカルマシン (コマンドが実行されるマシン) を使用します。

NIS に関する Solaris 9 と旧バージョンとの相違点

Solaris 9 の NIS の特徴は、次のとおりです。

NSKit が存在しない

NIS サービスは、2.6 より前の Solaris リリースには組み込まれていませんでした。NIS サービスは今までは、個別販売される NSKit からインストールしなければなりませんでした。現在、NIS サービスは Solaris 9 に組み込まれているため、Solaris 9 の NSKit はありません。

Solaris 2.6 以降のリリースには NIS サービスが組み込まれているので、`SUNWnsktu` および `SUNWnsktr` パッケージはもうありません。NIS のインストールは、NIS サーバークラスタ (`SUNWypu` および `SUNWypr` パッケージが含まれている) により行われています。

NIS サービスは現在、`/etc/init.d/yp` スクリプトでは起動されません。`/etc/init.d/yp` スクリプトは現在、存在していません。Solaris 9 では、まずマスターサーバーの NIS マップを `ypinit` スクリプトで作成し、次に `ypstart` で NIS を起動します。NIS サービスの停止には、`ypstop` コマンドを使用します。

ypupdated デーモン

`ypupdated` デーモンは、NSKit の 2.6 以前のバージョンには組み込まれていませんでしたが、Solaris 7 以降のリリース には組み込まれています。

/var/yp/securenets

`/var/yp/securenets` ファイルは、以前の NSKit リリースの場合と同様に、NIS サービスへのアクセスを制限するために使用されます。このファイルが存在する NIS サーバーでは、ファイルに収められている IP アドレスのマシンおよびネットワークに対してのみ、問い合わせに答えたり、マップを提供したりします。このファイルのフォーマットについては、`securenets(4)` のマニュアルページを参照してください。

`securenets` ファイルの例を次に示します。

```
255.255.255.10     192.168.0.1
host    13.13.14.1
host    13.13.14.2
```

上記において、255.255.255.10 はネットマスクで、13.13.13.255 はネットワークアドレスです。1 行目の設定では、ypserv はサブネットの 13.13.13.255 の範囲のこれらのアドレスにのみ応答します。

/var/yp/securenets ファイルのエントリを変更したときは、ypserv と ypxfrd のデーモンを終了させて再起動をする必要があります。

マルチホームマシンのサポート

ypserv プロセスは、以前の NSKit リリースの場合と同様に、複数のネットワークアドレスを持つマシンをサポートします。マシンマップが作成されると、Makefile は、複数のアドレスを持つマシンのマップに YP_MULTI_HOSTNAME エントリを作成します。このエントリには、そのマシンのすべてのアドレスがリストされます。マシンアドレスが必要な場合は、このリストに存在するアドレスのなかで、希望するアドレスに最も近いアドレスを使用しようとします。詳細については、ypserv(1) のマニュアルページを参照してください。

希望するアドレスに最も近いアドレスの判断は算術的判断なので、アドレスの妥当性検査は行われません。たとえば、マルチホームマシンが 6 つの IP アドレスを持っているが、このマルチホームマシン上の 5 つのインタフェースだけが正常に動作していると仮定します。このマルチホームマシンに直接接続されていないネットワーク上のマシンは、ypserv からダウンインタフェースの IP アドレスを受け取ることができます。このように、この仮説上のクライアントはマルチホームマシンにアクセスできません。

注 – マルチホームマシンのすべてのアドレスは通常、有効でなければなりません。特定のアドレスまたはマシンでサービスが提供できなくなる恐れがある場合は、そのアドレスまたはマシンは NIS マップから削除してください。

SunOS 4 互換モード

NIS は、パスワード構成ファイルを SunOS 4 (Solaris 1) パスワードフォーマットと Solaris 2 パスワードおよびシャドウファイルフォーマットの両方でサポートしています。

動作モードは、$PWDIR/shadow ファイルが存在するかどうかによって決定されます ($PWDIR は、/var/yp/Makefile ファイルに設定されている Makefile マクロセットです) 。shadow ファイルが存在する場合、NIS は Solaris 2 モードで動作します。shadow ファイルが存在しない場合、NIS は SunOS 4 モードで動作します。

SunOS 4 モードでは、すべてのパスワード情報が passwd ファイルに保存されています。Solaris 2 モードでは、パスワード情報は shadow ファイルに保存され、ユーザーアカウント情報は passwd ファイルに保存されます。

make マクロ PWDIR が /etc ディレクトリに設定されている場合、Solaris 2 の passwd 処理要件の関係上、NIS は Solaris 2 モードでしか動作できません。ただし、PWDIR が /etc 以外のディレクトリに設定されている場合、ユーザーは passwd 構成ファイルを SunOS 4 フォーマットでも Solaris 2 フォーマットでも保存できます。rpc.yppasswdd デーモンはこれら両方のパスワードフォーマットを認識しますが、Solaris リリース 2 フォーマットを使用することをお勧めします。

第 8 章

NIS サービスの設定と構成

この章では、ネットワーク情報サービス (NIS) の初期設定と構成について説明します。

注 – 「マシン」名の代わりに「ホスト」名が使われることがあります。この解説では「マシン」名が使われていますが、一部の画面メッセージや NIS マップ名では「ホスト」名または「マシン」名が使われています。

この章の内容は次のとおりです。

NIS の構成 — 作業マップ

作業	参照先
変換用のソースファイルを準備する	146 ページの「NIS マップへの変換用のソースファイルを準備する」

作業	参照先
ypinit を使用してマスターサーバーを設定する	148 ページの「ypinit を使用してマスターサーバーを設定する」
マスターサーバーで NIS を起動する	150 ページの「マスターサーバーでの NIS サービスの開始」
スレーブサーバーを設定する	152 ページの「スレーブサーバーを設定する」
NIS クライアントを設定する	153 ページの「NIS クライアントの設定」

NIS の構成を始める前に

NIS の名前空間を構成する前に、次の操作を行う必要があります。

- NIS を使用するすべてのマシンで、正しく構成された nsswitch.conf ファイルをインストールする。詳細については、第 2 章を参照してください。
- NIS ドメインを設計する。次の節を参照してください。

NISドメインの設計

NIS サーバーまたはクライアントとしてマシンを構成する前に、NIS ドメインを設計する必要があります。

まず、NIS ドメインに入れるマシンを決めます。NIS ドメインは、ネットワークと同一である必要はありません。ネットワークには複数の NIS ドメインが存在でき、NIS ドメインに属さないマシンもネットワーク上に存在できます。

次に、NIS ドメイン名を選択します。NIS ドメイン名には、最高 256 文字を指定できます。ドメイン名が 32 文字を超えないように制限するとよいでしょう。ドメイン名は大文字と小文字を区別します。便宜上、インターネットのドメイン名に基づいて NIS ドメイン名を指定することもできます。たとえば、インターネットのドメイン名が doc.com の場合、NISドメイン名も doc.com にすることができます。doc.com を 2 つの NIS ドメインに分けて、1 つを営業部門に、もう 1 つを製造部門に使用する場合は、一方を sales.doc.com とし、もう一方を manf.doc.com とできます。

NIS ドメイン名とマシン名を正しく設定しないと、マシンが NIS サービスを使用できるようになりません。マシン名はマシンの /etc/nodename ファイルによって設定され、マシンのドメイン名はマシンの /etc/defaultdomain ファイルによって設定されます。これらファイルはブート時に読み取られ、その内容はそれぞれ uname -S コマンドと domainname コマンドによって使用されます。ディスクレスマシンは、そのブートサーバーからこれらのファイルを読み取ります。

NIS サーバーとクライアントを特定する

マスターサーバーになるマシンを 1 つ選択します。スレーブサーバーを作成する場合は、スレーブサーバー用のマシンを決定します。

NIS クライアントになるマシンを決定します。通常は、ドメイン内のすべてのマシンが NIS クライアントになるように設定されますが、これは必須ではありません。

マスターサーバーの準備

ソースファイルディレクトリ

ソースファイルは、マスターサーバーの `/etc` ディレクトリか、その他のディレクトリにあります。ソースファイルを `/etc` に入れることは望ましくありません。マップの内容がマスターサーバー上のローカルファイルの内容と同じになるからです。これは `passwd` ファイルと `shadow` ファイルに固有の問題です。ユーザー全員がマスターサーバーのマップにアクセスし、`passwd` マップを通じてすべての YP クライアントに root パスワードが渡されるためです。詳細については、145 ページの「`passwd` ファイルと名前空間のセキュリティ」を参照してください。

ただし、ソースファイルを他のディレクトリに入れた場合は、`/var/yp` 内の `Makefile` の `DIR=/etc` 行を `DIR=/`*your-choice* に変更する必要があります。*your-choice* はソースファイルを格納するためのディレクトリの名前です。これによって、サーバー上のローカルファイルをクライアント上のファイルのように扱うことができます。`Makefile` を編集する前に、編集前の `Makefile` のコピーを保存しておくことをお勧めします。

また、`audit_user`、`auth_attr`、`exec_attr`、`prof_attr` がデフォルト以外のディレクトリから取り出される場合は、`RBACDIR=/etc/security` を `RBACDIR=`*your-choice* に変更します。*your-choice* は取り出し元のディレクトリの名前です。

`passwd` ファイルと名前空間のセキュリティ

`passwd` マップは特殊なケースです。この NIS 実装では、NIS パスワードマップを作成するための入力として、Solaris 1 の `passwd` ファイルのフォーマットに加え、Solaris 7 の `/etc/passwd` ファイルと `/etc/shadow` ファイルのフォーマットも使用できます。

セキュリティ上の理由から未承認の root アクセスを防ぐために、NIS のパスワードマップの構築に使用されるファイルには root のエントリを含めないでください。このため、パスワードマップはマスターサーバーの /etc ディレクトリに置かれたファイルから構築しないでください。パスワードマップの構築に使用されるパスワードファイルは、root エントリが削除された上、未承認のアクセスから保護されるディレクトリに置かれている必要があります。

たとえば、マスターサーバーのパスワード入力ファイルは、ファイル自体が別のファイルへのリンクではなく、ファイルの場所が Makefile に指定されている限り、/var/yp/ などのディレクトリに格納されているか、選択したディレクトリに格納されている必要があります。/usr/lib/netsvc/yp/ypstart スクリプトは、Makefile に指定された構成に従って適切なディレクトリオプションを自動的に設定します。

注意 – PWDIR によってディレクトリ内に指定された passwd ファイルに root のエントリが含まれないようにしてください。

ソースファイルが /etc 以外のディレクトリにある場合は、Makefile の PWDIR パスワードマクロが、passwd ファイルと shadow ファイルが入っているディレクトリを参照するように変更します。この操作を行うには、PWDIR=/etc 行を PWDIR=/*your-choice* に変更します。*your-choice* は、passwd マップソースファイルを格納するのに使用するディレクトリの名前です。

NIS マップへの変換用のソースファイルを準備する

NIS マップへの変換用のソースファイルを準備します。

▼ 変換用のソースファイルを準備する方法

1. スーパーユーザーになります。
2. マスターサーバーのソースファイルをチェックして、それらのファイルが環境の最新の状態を反映しているかどうか確認します。

   次のファイルを確認します。

   - auto.home または auto_home
   - auto.master または auto_master
   - bootparams
   - ethers
   - group
   - hosts
   - ipnodes

- netgroup
- netmasks
- networks
- passwd
- protocols
- rpc
- services
- shadow
- user_attr

3. これらの入力ファイル **(**passwd を除く**)** をすべて、選択した DIR ディレクトリにコピーします。

4. passwd ファイルを、選択した PWDIR ディレクトリにコピーします。

5. audit_user、auth_attr、exec_attr、prof_attr を、選択した RBACDIR ディレクトリにコピーします。

6. /etc/mail/aliases ファイルを確認します。

   他の入力ファイルと異なり、/etc/mail/aliases ファイルは別のディレクトリに移動できません。このファイルは /etc/mail ディレクトリに格納されていなければなりません。ドメイン全体で利用するメールエイリアスがすべて /etc/mail/aliases ソースファイルに含まれていることを確認します。詳細については、aliases(4) のマニュアルページを参照してください。

7. ソースファイルからすべてのコメントと、その他の余計な行や情報を取り除きます。

   これらの操作は、sed または awk の各スクリプトを通じて、またはテキストエディタを使用して行えます。Makefile はソースファイルから不要なエントリをある程度自動的に削除しますが、これらのファイルを手動で検証し、クリーンアップしてから実行することをお勧めします。

8. すべてのソースファイルのデータが正しい形式になっていることを確認します。

   ソースファイルのデータは、それぞれのファイルに適した形式で格納されている必要があります。該当するマニュアルページを参照して、各ファイルが正しい形式になっていることを確認します。

Makefile を準備する

ソースファイルをチェックしてソースファイルディレクトリにコピーしたら、NIS サービスが使用する ndbm 形式のマップにそのソースファイルを変換する必要があります。次の148 ページの「ypinit を使用してマスターサーバーを設定する」の節で説明しているように、この処理は、マスターサーバーで ypinit が実行されると自動的に行われます。

ypinit スクリプトはプログラム make を呼び出します。このプログラムは、/var/yp ディレクトリに置かれた Makefile を使用します。/var/yp ディレクトリにはデフォルトの Makefile が用意されており、この中にはソースファイルを要求された ndbm 形式のマップに変換するためのコマンドが入っています。

デフォルトの Makefile は、そのまま使用することも必要に応じて修正することもできます。デフォルトの Makefile を修正するときは、将来必要な場合に備えて、必ず最初に修正前の Makefile をコピーして保存するようにしてください。次に説明する Makefile への修正のうち、必要に応じて 1 つまたは複数を実行します。

- 「デフォルトではないマップ」

  デフォルトではない自分専用のソースファイルを作成して NIS マップに変換する場合は、そのソースファイルを Makefile に追加する必要があります。

- 「DIR の値」

  145 ページの「ソースファイルディレクトリ」で説明しているように、/etc 以外のディレクトリに格納されたソースファイルを Makefile で使用する場合は、Makefile の DIR の値を使用するディレクトリに変更する必要があります。この値を Makefile で変更するときは行をインデントしないでください。

- 「PWDIR の値」

  /etc 以外のディレクトリに格納された passwd、shadow、または adjunct の各ソースファイルを Makefile で使用する場合は、Makefile の PWDIR の値を、使用するディレクトリに変更する必要があります。この値を Makefile で変更するときは行をインデントしないでください。

- 「ドメインネームリゾルバ」

  現在のドメインにはないマシンに対して NIS サーバーがドメインネームリゾルバを使用するようにする場合は、Makefile の B= 行をコメントにし B= -b 行のコメントを解除します (有効にします) 。

Makefile は、all の下にリストされるデータベースのそれぞれに対して適切な NIS マップを作成します。データは makedbm で処理され、mapname.dir と mapname.pag の 2 つのファイルに保存されます。この両ファイルは、マスターサーバーの /var/yp/*domainname* ディレクトリに置かれます。

Makefile は必要に応じて、/PWDIR/passwd、/PWDIR/shadow、/PWDIR/security/passwd.adjunct の各ファイルから passwd マップを作成します。

ypinit を使用してマスターサーバーを設定する

ypinit スクリプトは、マスターサーバー、スレーブサーバー、クライアントが NIS を使用するように設定します。また最初に make を実行して、マスターサーバー上にマップを作成します。

ypinit を使用して新規に NIS マップセットをマスターサーバーに作成する場合は、次の手順に従います。

▼ ypinit を使用してマスターサーバーを設定する方法

1. マスターサーバーでスーパーユーザーになります。
2. nsswitch.files ファイルの内容を nsswitch.conf ファイルにコピーします。

   # cp /etc/nsswitch.files /etc/nsswitch.conf
3. /etc/hosts ファイルまたは /etc/inet/ipnodes ファイルを編集して、**NIS** サーバーのそれぞれの名前と **IP** アドレスを追加します。
4. 新しいマップをマスターサーバーに作成します。

   # /usr/sbin/ypinit -m
5. ypinit によって **NIS** スレーブサーバーになる他のマシンのリストを求めるプロンプトが表示されたら、作業中のサーバー名と **NIS** スレーブサーバー名を入力します。
6. 致命的でないエラーが発生したときにすぐに処理を終了するか、引き続き処理を継続するかを ypinit が尋ねてきたら、**y** と入力します。

   y を選択すると、ypinit は最初の問題が発生すると終了します。問題を解決して ypinit を再起動します。ypinit を初めて実行する場合はこの手順に従うようにしてください。処理を継続する場合は、発生する問題をすべて手動で解決してから ypinit を再起動します。

   注 – マップファイルの一部が存在しないと、致命的でないエラーが発生することがあります。これは NIS の機能に影響するエラーではありません。マップが自動的に作成されない場合は、必要に応じて手動で追加します。デフォルトの NIS マップの詳細については、132 ページの「デフォルトの NIS マップ」 を参照してください。

   ---
7. /var/yp/*domainname* ディレクトリ内の既存のファイルを破棄してもよいかどうか ypinit が尋ねてきます。

   このメッセージは、NIS を以前に設定したことがある場合にだけ表示されます。
8. ypinit は、サーバーのリストを作成し終わると make を起動します。

   このプログラムは、/var/yp に置かれた Makefile (デフォルトまたは修正されたもの) に含まれている命令を使用します。make コマンドは、指定したファイルにコメント行があればその行を取り除きます。また、指定したファイルに対して makedbm を実行して適切なマップを作成し、各マップにマスターサーバー名を設定します。

   マスターサーバー上で domainname コマンドを実行すると返されるドメイン以外に対するマップの転送を Makefile で行う場合は、ypinit シェルスクリプトの中で make コマンドの変数 DOM に適切なドメイン名を指定して起動すれば、マップを正しいドメインに転送することができます。次のように入力してください。

   # make DOM=*domainname password*

   このコマンドによって、マスターサーバーが属するドメインではなく目的のドメインに password マップが転送されます。

9. 次のように入力してネームサービスとして **NIS** を有効にします。

   # cp /etc/nsswitch.nis /etc/nsswitch.conf

   現在のスイッチファイルが、デフォルトの NIS 用スイッチファイルに置き換えられます。このファイルは必要に応じて編集可能です。

複数の NIS ドメインをサポートするマスターサーバー

NIS マスターサーバーは通常、NIS ドメインだけをサポートします。ただし、マスターサーバーを使用して複数のドメインをサポートする場合は、前の節で説明したように、追加のドメイン用にサーバーを設定するときに手順を若干修正する必要があります。

サーバー上で `domainname` コマンドを実行します。このコマンドによって返されるドメイン名はサーバーのデフォルトドメインです。前の節で説明した手順は、このデフォルトドメインへのサービスを設定する場合は正しく機能します。他のドメインへのサービスを設定する場合は、`ypinit` シェルスクリプトを次のように修正する必要があります。

 # make DOM=*correct-domain* `passwd`

correct-domain はサービスを設定している他のドメインの名前であり、`passwd` は `make` のターゲットです。このコマンドによって、マスターサーバーが属するドメインではなく目的のドメインに `passwd` マップが転送されます。

マスターサーバーでの NIS サービスの開始

マスターサーバーのマップが作成されると、NIS デーモンをマスターサーバーで起動してサービスを開始できます。これを行うには、`ypserv` をサーバー上で起動して、`ypbind` を実行する必要があります。クライアントがサーバーの情報を要求すると、`ypserv` デーモンは NIS マップ内で検索してクライアントからの情報の要求に応答します。

サーバー上で NIS サービスを起動するには 2 つの方法があります。

- ブートプロセス中に `/usr/lib/netsvc/yp/ypstart` スクリプトを自動的に起動する
- コマンド行から `ypstart` を実行する

NIS サービスを自動的に開始する

ypinit を実行して NIS マスターサーバーを構成し終わると、マシンのブート時に ypstart が自動的に起動され、ypserve が開始されます。148 ページの「ypinit を使用してマスターサーバーを設定する」を参照してください。

コマンド行から NIS を開始または停止する

NIS サービスをコマンド行から開始する場合は、ypstart スクリプトを実行します。

```
# /usr/lib/netsvc/yp/ypstart
```

注 – 起動後に ypserv が呼び出しに応答できるようになるまでに若干の遅延があります。プログラムまたはスクリプトの内部から呼び出す場合は、ypstart の実行後に 3 - 5 秒間スリープ状態にしてください。

NIS サービスを停止する場合は、ypstop スクリプトを実行します。

```
# /usr/lib/netsvc/yp/ypstop
```

NIS スレーブサーバーの設定

ネットワークは 1 つ以上のスレーブサーバーを持つことができます。スレーブサーバーを持つことで、マスターサーバーが利用できない場合にも NIS サービスを継続して利用できます。

スレーブサーバーを準備する

ypinit を実際に実行してスレーブサーバーを作成する前に、domainname コマンドを NIS スレーブサーバーごとに実行してドメイン名がマスターサーバーと一致していることを確認します。

注 – ドメイン名は大文字と小文字を区別します。

ネットワークが正しく機能していることを確認してから、NIS スレーブサーバーを構成してください。特に、rcp を使用して NIS マスターサーバーから NIS スレーブサーバーにファイルを送れるかどうかを確認してください。

スレーブサーバーを設定する

次の手順はスレーブサーバーの設定方法を示しています。

▼ スレーブサーバーを設定する方法

1. スーパーユーザーになります。
2. スレーブサーバーで /etc/hosts ファイルまたは /etc/inet/ipnodes ファイルを編集して、他のすべての **NIS** サーバーの名前と **IP** アドレスを追加します。
3. スレーブサーバー上の /var/yp にディレクトリを変更します。

注 – まず、新しいスレーブサーバーを NIS クライアントとして構成して、最初にマスターサーバーから NIS マップを入手できるようにします。詳細については、153 ページの「NIS クライアントの設定」を参照してください。

---

4. 次のように入力して、スレーブサーバーにするマシンをクライアントとして初期設定します。

   # **/usr/sbin/ypinit -c**

   ypinit コマンドによって、NIS サーバーのリストを求めるプロンプトが表示されます。作業中のローカルマシン (スレーブ) の名前を入力してからマスターサーバーを入力し、その後にドメイン内の他の NIS スレーブサーバーをネットワーク的に近いものから遠いものの順に入力します。

5. 次のように入力して、ypbindが実行されているかどうか確認します。

   # **ps -ef | grep ypbind**

   リストが表示されたら、ypbind は実行されています。

6. ypbind が実行中である場合は停止します。

   # **/usr/lib/netsvc/yp/ypstop**

7. 次のように入力して ypbind を再起動します。

   # **/usr/lib/netsvc/yp/ypstart**

8. 次のように入力して、このマシンをスレーブサーバーとして初期設定します。

   # **/usr/sbin/ypinit -s** *master*

   master は、既存の NIS マスターサーバーのマシン名です。

   この節で説明した手順を、NIS スレーブサーバーとして構成するマシンごとに繰り返します。

次の手順は、スレーブサーバーで NIS サービスを開始する方法を示しています。

▼ スレーブサーバーで NIS を開始する方法

1. スーパーユーザーになります。
2. 既存の yp プロセスをすべて停止します。

   # /usr/lib/netsvc/yp/ypstop
3. スレーブサーバーで ypserve を起動して、ypbind を実行します。

   # /usr/lib/netsvc/yp/ypstart

   この方法とは別に、スレーブサーバーをリブートしてデーモンを自動的に開始することもできます。

NIS クライアントの設定

NIS を使用するようにマシンを設定する

ネームサービスとして NIS を使用するようにマシンを設定するには、次の 2 つの方法があります。

- 「ypinit」。NIS を使用するようにクライアントマシンを設定する場合は、マシンに root としてログインして ypinit -c を実行する方法をお勧めします。

  # ypinit –c

  NIS サーバーを指定するように求められます。クライアントは NIS サーバーからネームサービス情報を得ます。必要な数だけマスターサーバーやスレーブサーバーを指定できます。指定するサーバーはドメイン内のどこにあってもかまいません。クライアントにネットワーク的に近いサーバーから遠いサーバーの順に指定することをお勧めします。
- 「ブロードキャスト方式」。NIS を使用するようにクライアントマシンを設定する旧式の方法です。マシンに root としてログインし、domainname コマンドでドメイン名を設定してから、ypbind を実行します。

  # domainname doc.com

  # ypbind -broadcast

  ypbind を実行すると、NIS サーバーがローカルサブネットで検索されます。NIS サーバーが見つかると、ypbind はそのサーバーにバインドします。この検索を「ブロードキャスト」と呼びます。クライアントのローカルサブネットに NIS サーバーがない場合、ypbind によるバインドは失敗し、クライアントマシンは NIS サービスから名前空間データを入手することができません。

第 9 章

NIS の管理 (手順)

この章では、NIS の管理方法について説明します。内容は次のとおりです。

パスワードファイルと名前空間のセキュリティ

セキュリティの関係上、次の点に注意してください。

- マスターサーバーの NIS マップへのアクセスは制限します。
- 未許可アクセスを防止するためには、NIS パスワードマップの作成に使用されたファイルに `root` エントリを含めないでください。したがって、`root` エントリをこのパスワードファイルから削除して、このパスワードファイルをマスターサーバーの `/etc` ディレクトリ以外のディレクトリにおく必要があります。このディレクトリへの未許可アクセスは、防止しなければなりません。

たとえば、マスターサーバーのパスワード入力ファイルは、別のファイルへのリンクではなく `Makefile` に指定されている限り、`/var/yp/` などのディレクトリに存在するか選択されたディレクトリに存在します。`/usr/lib/netsvc/yp/ypstart` スクリプトは、`Makefile` に指定された構成に従って自動的に適切なディレクトリオプションを設定します。

注 – この NIS 実装では、NIS パスワードマップを作成するための入力として、Solaris 1 バージョンの passwd ファイルのフォーマットに加え、Solaris 2 の passwd ファイルと shadow ファイルのフォーマットも使用できます。

NIS ユーザーの管理

この節では、ユーザーパスワードの設定、NIS ドメインへの新しいユーザーの追加、ネットグループ (netgroups) へのユーザーの割り当てについて説明します。

NIS ドメインに新しいユーザーを追加する

▼ NIS ユーザーを追加する方法

1. **NIS** マスターサーバーでスーパーユーザーになります。
2. useradd コマンドで新しいユーザーのログイン **ID** を作成します。

   # **useradd** *userID*

   userID は新しいユーザーのログイン ID です。このコマンドは、NIS マスターサーバー上の /etc/passwd ファイルと /etc/shadow ファイルにエントリを作成します。
3. 新しいユーザーの初期パスワードを作成します。

   新しいユーザーがログインするための初期パスワードを作成するには、passwd コマンドを実行します。

   # **passwd** *userID*

   userID は新しいユーザーのログイン ID です。このユーザーに割り当てるパスワードを入力するようにプロンプトが表示されます。

   この手順が必要になるのは、useradd コマンドで作成されたパスワードエントリがロックされ、新しいユーザーがログインできないからです。初期パスワードを指定することで、このパスワードエントリのロックが解除されます。
4. 必要に応じて、マスターサーバーの passwd マップ入力ファイルに新しいエントリをコピーします。

   マスターサーバー上のマップソースファイルは、/etc 以外のディレクトリにあります。新しい行を /etc/passwd ファイルと /etc/shadow ファイルから マスターサーバー上の passwd マップ入力ファイルにコピーします。詳細については、155 ページの「パスワードファイルと名前空間のセキュリティ」を参照してください。

たとえば、新しいユーザー brown を追加する場合、/etc/passwd ファイルから passwd 入力ファイルにコピーする行は次のようになります。

```
brown:x:123:10:User brown:/home/brown:/bin/csh:
```

/etc/shadow からコピーされる brown 行は次のようになります。

```
brown:W12345GkHic:6445::::::
```

5. パスワード入力ファイルが格納されているディレクトリが Makefile で正しく指定されていることを確認します。

6. 必要に応じて、/etc/passwd ファイルと /etc/shadow ファイルから新しいユーザーのエントリを削除します。

   セキュリティの関係上、NIS マスターサーバーの /etc/passwd ファイルと /etc/shadow ファイルでユーザーエントリを保持することは望ましくありません。他のディレクトリに存在する NIS マップソースファイルに新しいユーザーのエントリをコピーした後、マスターサーバー上で userdel コマンドを使用して新しいユーザーを削除します。

   たとえば、マスターサーバーの /etc ファイルから新しいユーザー brown を削 除するには次のように入力します。

   # userdel brown

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

7. **NIS** の passwd マップを更新します。

   マスターサーバー上の passwd 入力ファイルを更新した後、ソースファイルが存在するディレクトリで make を実行して passwd マップを更新します。

   #userdel brown

   # cd /var/yp

   # /usr/ccs/bin/make passwd

8. 新しいユーザーのログイン **ID** に割り当てられた初期パスワードを新しいユーザーに通知します。

   ログイン後、新しいユーザーはいつでも passwd を実行して別のパスワードに変更できます。

ユーザーパスワードの設定

ユーザーは passwd を実行してパスワードを変更します。

% **passwd** *username*

username はユーザー名です。ユーザーがパスワードを変更する前に、NIS 管理者はマスターサーバー上で rpc.yppasswdd デーモンを起動してパスワードファイルを更新しなければなりません。rpc.yppasswdd デーモンを起動するコマンドは、/usr/lib/netsvc/yp/ypstart ファイルにすでに存在しています。

rpc.yppasswdd デーモンは、マスターサーバー上の ypstart で自動的に起動されます。rpc.yppasswdd に -m オプションが指定された場合は、ファイルが更新されるとすぐ /var/yp の make が実行されます。passwd ファイルが更新されるたびにこの make が実行されることを回避したい場合は、ypstart スクリプトの rpc.yppasswdd コマンドから -m オプションを削除して、crontab ファイルで passwd マップの転送を制御してください。

注 – rpc.yppasswdd -m コマンドの後に引数を指定しないでください。別の動作のために ypstart スクリプトファイルを編集することは可能ですが、-m オプションを任意に削除すること以外の変更をこのファイルに加えることは望ましくありません。すべてのコマンドおよびデーモンは、適切なコマンド行パラメータのセットが存在するこのファイルで起動されます。このファイルを編集する場合は、rpc.yppasswdd コマンドの編集では特に注意してください。passwd.adjunct ファイルに明示的コールを追加する場合は、パスを $PWDIR/security/passwd.adjunct と正確に指定しなければなりません。正確に指定しない場合は、不適切な処理が行われます。

ネットグループ

NIS ネットグループは、NIS 管理者が管理目的のために定義するユーザーまたはマシンのグループ (集合) です。たとえば、次のようなネットグループを作成できます。

- 特定マシンにアクセスできる一群のユーザーを定義するネットグループ
- 特定のファイルシステムにアクセスできる一群の NFS クライアントマシンを定義するネットグループ
- 特定の NIS ドメインのすべてのマシンに対して管理者権限を持つ一群のユーザーを定義するネットグループ

各ネットグループには 1 つのネットグループ名が与えられます。ネットグループはアクセス権を直接設定しません。代わりに、ユーザー名またはマシン名が一般に使用される場所ではネットグループ名が他の NIS マップで使用されます。たとえば、netadmins というネットワーク管理者ネットグループを作成したと仮定します。netadmins ネットグループのすべてのメンバーに特定マシンへのアクセス権を与えるには、そのマシンの /etc/passwd ファイルに netadmin エントリを追加するだけで、ネットグループ名を /etc/netgroup ファイルに追加して、NIS グループマップに追加することもできます。ネットグループの使い方の詳細については、netgroup (4) のマニュアルページを参照してください。

NIS が使用されているネットワーク上では、NIS マスターサーバー上の netgroup 入力ファイルを使用して、netgroup、netgroup.byuser、netgroup.byhost という 3 つのファイルが生成されます。netgroup マップには、netgroup 入力ファイルの基本情報が入っています。他の 2 つの NIS マップには、マシンまたはユーザーが指定されるとネットグループ情報の検索が迅速に行われるフォーマットで情報が入っています。

netgroup 入力ファイルのエントリのフォーマットは、 *name ID* です。*name* はネットグループ名であり、*ID* は、ネットグループに属しているマシンまたはユーザーを示します。ネットグループの ID (メンバー) は、コンマで区切っていくつでも指定できます。たとえば、3 つのメンバーが存在するネットグループを作成する場合、netgroup 入力ファイルエントリのフォーマットは、*name ID*、 *ID*、*ID* となります。netgroup 入力ファイルエントリのメンバーID のフォーマットは次のようになります。

```
(-|machine, -|user, domain)
```

machine はマシン名、*user* はユーザーID 、*domain* はマシンまたはユーザーの NIS ドメインです。「ドメイン」エレメントは任意指定ですが、他の NIS ドメインのマシンまたはユーザーを示す場合には必ず指定します。各エントリでは「マシン」エレメントと「ユーザー」エレメントは必須ですが、ダッシュ (-) は空であることを示すために使用されます。エントリでは、「マシン」エレメントと「ユーザー」エレメントの関係を示す必要はありません。

netgroup 入力ファイルの 2 つのサンプルエントリを次に示します。これらの各サンプルエントリでは、admins という名前のネットグループが作成されます。これらの各 admins は、遠隔ドメイン sales に存在するユーザー hauri と juanita、およびマシン altair と sirius で構成されます。

```
admins (altair, hauri), (sirius,juanita,sales)

admins (altair,-), (sirius,-), (-,hauri), (-,juanita,sales)
```

さまざまなプログラムでは、ログイン、遠隔マウント、遠隔ログイン、遠隔シェル作成時に NIS ネットグループマップを使用してアクセス権のチェックを行います。さまざまなプログラムとは、mountd、login、rlogin、rsh などです。login コマンドは、passwd データベース内でネットグループ名を見つけた場合に、ネットグループマップでユーザー分類を調べます。mountd デーモンは、/etc/dfs/dfstab ファイル内でネットグループ名を見つけた場合に、ネットグループマップでマシン分類を調べます。rlogin と rsh (インタフェースを使用するプログラムならどれでも) は、/etc/hosts.equiv または .rhosts ファイル内でネットグループ名を見つけた場合に、ネットグループマップでマシン分類とユーザー分類の両方を調べます。

ネットワークに新しい NIS ユーザーまたはマシンを追加する場合は、netgroup 入力ファイルの該当ネットグループに追加してください。次に、make でネットグループマップを作成し、これを yppush コマンドですべての NIS サーバーに転送してください。ネットグループおよびネットグループ入力ファイル構文の使い方の詳細については、netgroup(4) のマニュアルページを参照してください。

NIS マップに関する作業

マップ情報の取得

マップ情報は、ypcat、ypwhich、ypmatch コマンドを使っていつでも取得できます。以下の例では、*mapname* はマップの正式名とニックネーム（存在する場合）の両方を意味します。

マップのすべての値を表示するには、次のように入力します。

% **ypcat** *mapname*

マップのキーと値 (存在する場合) の両方を表示するには、次のように入力します。

% **ypcat -k** *mapname*

マップのすべてのニックネームを表示するには、次のいずれかのコマンドを入力します。

% **ypcat –x**

% **ypwhich –x**

% **ypmatch –x**

使用可能なすべてのマップとそのマスターサーバーを表示するには、次のように入力します。

% **ypwhich –m**

特定のマップのマスターサーバーを表示するには、次のように入力します。

% **ypwhich -m** *mapname*

キーをマップのエントリと照合するには、次のように入力します。

% **ypmatch** *key mapname*

key は照合する文字列です。検索している項目がマップのキーでない場合は、次のように入力します。

% **ypcat** *mapname* | **grep** *item*

item は検索している情報です。他のドメインに関する情報を取得するには、これらのコマンドの -d domainname オプションを指定します。

デフォルト以外のドメインの情報を要求するマシンが、そのドメインに対するバインドを持っていない場合、ypbind は /var/yp/binding/*domainname*/ypservers ファイルを参照して、そのドメインのサーバーリストを検索します。このファイルが存在しない場合、ypbind は RPC 同報通信を送出してサーバーを検索します。この場合、検索先であるドメインのサーバーは要求元マシンと同じサブネットに存在している必要があります。

マップのマスターサーバーの変更

選択されたマップのマスターサーバーを変更するには、まず新しい NIS マスターサーバー上にマップを作成しなければなりません。古いマスターサーバー名は既存のマップにキーと値のペアとして発生するので (このペアは makedbm で自動的に挿入される)、ypxfr でマップを新しいマスターサーバーにコピーしたり、コピーを新しいマスターサーバーに転送するだけでは不十分です。キーと新しいマスターサーバー名との対応づけをし直す必要があります。マップに ASCII ソースファイルが存在する場合は、このファイルを新しいマスターサーバーにコピーします。

▼ マップのマスターサーバーを変更する方法

1. スーパーユーザーになります。
2. 新しいマスターサーバーにスーパユーザーとしてログインして、次のように入力します。

   newmaster# **cd /var/yp**
3. 作成するマップを指定する前に、Makefile にこの新しいマップのエントリが必要です。新しいマップのエントリがない場合は、最初に、sites.byname というマップを使用して Makefile を編集します。
4. マップを更新または再作成するには、次のように入力します。

   newmaster# **make sites.byname**
5. 古いマスターサーバーが **NIS** サーバーとして残っている場合は、古いマスターサーバーに遠隔ログイン **(**rlogin**)** してから、Makefile を編集します。sites.byname を作成した Makefile 内のセクションをコメントアウトして、このセクションで sites.byname が再び作成されないようにします。
6. sites.byname だけが ndbm ファイルとして存在している場合は、新しいマスターサーバー上に作成し直します。まず任意の **NIS** サーバーからコピーを分解し、次に makedbm を使ってそれを実行します。

   newmaster# **cd /var/yp**

   newmaster# **ypcat sites.byname** | **makedbm** -*domain* **/sites.byname**

   新しいマスターサーバー上でマップが作成されたら、そのコピーをこのマスターサーバーのスレーブサーバーに送信します。この場合、yppush を使用しないでください。yppush を使用すると、スレーブサーバーは新しいマスターサーバーからではな

く古いマスターサーバーから新しいコピーを取得します。このような動作を回避するには、一般にマップのコピーを新しいマスターサーバーから古いマスターサーバーに送り返すという方法が用いられます。この操作を行うには、古いマスターサーバーでスーパユーザーとなり次のように入力します。

oldmaster# **/usr/lib/netsvc/yp/ypxfr -h newmaster sites.byname**

これで yppush を使用できます。スレーブサーバーは古いマスターサーバーを現行のマスターサーバーとして認識しているので、マップの現行バージョンを古いマスターサーバーから取得しようとします。クライアントがこの処理を行うときは、新しいマスターサーバーが現行のマスターサーバーとして指定されている新しいマップを取得します。

この方法が失敗した場合は、各 NIS サーバーの root としてログインして上記の `ypxfr` コマンドを実行してください。

構成ファイルの更新

NIS は設定ファイルを正確に構文解析します。このため NIS 管理は容易になりますが、設定ファイルおよび構成ファイルにおける変更により、NIS の動作は影響を受けます。

以下のファイルを変更する場合は、この節の手順を使用します。

- `/var/yp/Makefile`。このファイルは、サポートされているマップを追加または削除するために使用する
- `/etc/resolv.conf`。このファイルを追加または削除することで、DNS 転送が可能または不可になる
- `$PWDIR/security/passwd.adjunct`。このファイルを追加または削除することで、C2 セキュリティが可能または不可になる。`$PWDIR` は、`/var/yp/Makefile` で定義される

▼ 構成ファイルを更新する方法

1. スーパーユーザーになります。
2. **NIS** サーバーを停止します。

   # **/etc/init.d/ypstop**
3. 必要に応じてファイルを変更します。
4. **NIS** サーバーを再起動します。

   # **/etc/init.d/yp start**

   NIS のマップまたはマップソースファイルを更新する場合は、NIS を停止および起動する必要はありません。

   次の点に注意してください。

- NIS マスターサーバーからマップまたはソースファイルを削除しても、スレーブサーバー上の対応するマップまたはソースファイルは自動的には削除されません。スレーブサーバー上の対応するマップまたはソースファイルの削除は、NIS 管理者が手作業で行う必要があります。
- 新しいマップは、自動的には既存のスレーブサーバーに転送されません。新しいマップを既存のスレーブサーバーに転送するには、NIS 管理者がそのスレーブサーバーで `ypxfr` を実行してください。

`Makefile` の更新と使用

`/var/yp` で提供されたデフォルトの `Makefile` を更新することにより、NIS 管理者のニーズを満たすことができます。今後に備えて、必ずこのオリジナルの `Makefile` のコピーを保存しておいてください。`Makefile` を使用すると、マップの追加または削除、および一部のディレクトリ名の変更ができます。

新しい NIS マップを追加するには、マップの `ndbm` ファイルのコピーをドメインに存在する各 NIS サーバーの `/var/yp/`*domainname* ディレクトリに転送する必要があります。この転送は通常、`Makefile` が自動的に行います。どの NIS サーバーがマップのマスターサーバーであるかを決定したら、マップを容易に作成し直せるようにマスターサーバーの `Makefile` を更新してください。異なるサーバーを異なるマップのマスターサーバーとして設定することも可能ですが、このようにするとたいていの場合、管理上の混乱を招きます。したがって、1 つのサーバーだけをすべてのマップのマスターサーバーとして設定するようにしてください。

一般に、人が読めるテキストファイルは、`makedbm` に対する入力として適したものにするために `awk`、`sed`、`grep` でフィルタリングされます。デフォルトの `Makefile` を参照してください。`make` コマンドの概要については、`make(1S)` のマニュアルページを参照してください。

`make` が認識する従属性の作成方法を決定する際には、`Makefile` にすでに備わっているメカニズムを使用してください。`make` では、従属ルール内の行の始まりにタブが存在するか否かが重要であることに注意してください。他の設定が正しくても、タブが存在しないというだけでエントリが無効になることがあります。

`Makefile` にエントリを追加する場合は、次の作業を行ってください。

- データベース名を `all` ルールに追加する
- `time` ルールを作成する
- データベースのルールを追加する

たとえば、`Makefile` をオートマウンタ入力ファイルで動作させるには、`auto_direct.time` および `auto_home.time` マップを NIS データベースに追加してください。

NIS データベースにこれらのマップを追加する場合は、次の手順に従います。

▼ Makefile を更新する方法

1. スーパーユーザーとしてログインします。
2. all という語で始まる行に、追加したいデータベース名 **(1** つまたは複数**)** を追加します。

   ```
   all: passwd group hosts ethers networks rpc services protocols \
   netgroup bootparams aliases netid netmasks \
   auto_direct auto_home auto_direct.time auto_home.time
   ```

   エントリの順序は任意ですが、継続行の始まりの空白はスペースではなくタブにしてください。
3. Makefile の終わりに以下の行を追加します。

   ```
   auto_direct: auto_direct.time
   auto_home: auto_home.time
   ```

4. ファイル中央に auto_direct.time エントリを追加します。

   ```
   auto_direct.time: $(DIR)/auto_direct
    @(while read L; do echo $$L; done < $(DIR)/auto_direct
    $(CHKPIPE)) | \ (sed -e "/^#/d" -e "s/#.*$$//" -e "/^ *$$/d"
    $(CHKPIPE)) | \ $(MAKEDBM) - $(YPDBDIR)/$(DOM)/auto_direct;
    @touch auto_direct.time;
    @echo "updated auto_direct";
    @if [ ! $(NOPUSH) ]; then $(YPPUSH) auto_direct; fi
    @if [ ! $(NOPUSH) ]; then echo "pushed auto_direct"; fi
   ```

   次に、各引数について説明します。

   - CHKPIPE は、次のコマンドに結果を渡す (パイピングする) 前に、パイプ (|) の左側の動作が正しく行われたことを確認します。パイプの左側の動作が正しく行われなかった場合は、「NIS make terminated」というメッセージが表示されてプロセスは終了します。
   - NOPUSH は、Makefile が yppush を呼び出して新しいマップをスレーブサーバーに転送することを防止します。NOPUSH が設定されていない場合は、転送は自動的に行われます。

   継続行の始まりにある while ループは、バックスラッシュで拡張された行を入力ファイルから削除するためのものです。sed スクリプトはコメント行と空行を削除します。

   他のすべてのオートマウンタマップ (auto_home や他のデフォルトでないマップなど) でも、同じ手順が必要となります。
5. make を実行します。

   # **make** *mapname*

   mapname は、作成するマップの名前です。

Makefile に特定データベースのマップを作成しない場合は、Makefile を以下のように編集してください。

1. all ルールからデータベース名を削除します。

2. 削除するデータベースのデータベースルールを削除またはコメントアウトします。
   たとえば、hosts データベースを削除するには、hosts.time エントリを削除します。
3. time ルールを削除します。
   たとえば、hosts データベースを削除するには、hosts: hosts.time エントリを削除します。
4. マスターサーバーとスレーブサーバーからマップを削除します。

Makefile のマクロおよび変数の変更

Makefile の先頭で定義されている変数の設定値を変更するには、等号 (=) の右側の値を変更します。たとえばマップを作成するための入力として、/etc に存在するファイルではなく別のディレクトリに存在するファイル (たとえば /var/etc/*domainname* など) を使用する場合は、DIR を DIR=/etc から DIR=/var/etc/*domainname* に変更します。また、PWDIR を PWDIR=/etc から PWDIR=/var/etc/*domainname* に変更します。

変数は次のとおりです。

- *DIR*=。passwd と shadow を除くすべての NIS 入力ファイルが存在するディレクトリ。デフォルト値は /etc です。マスターサーバーの /etc ディレクトリのファイルを NIS 入力ファイルとして使用することは望ましくないので、この値は変更しなければなりません。
- *PWDIR*=。NIS 入力ファイル passwd と shadow が存在するディレクトリ。マスターサーバーの /etc ディレクトリのファイルを NIS 入力ファイルとして使用することは望ましくないので、この値は変更しなければなりません。
- *DOM*=。NISドメイン名。DOM のデフォルト値は、domainname コマンドで設定されます。ただし、大部分の NIS コマンドでは現在のマシンのドメイン（現在のマシンの /etc/defaultdomain ファイルに設定されている）が使用されます。

既存のマップの更新

NIS のインストール終了後、頻繁に更新しなければならないマップとまったく更新する必要がないマップがあることに気づくかもしれません。たとえば passwd.byname マップは、大企業のネットワークで頻繁に更新されることがありますが、auto_master マップはほとんど更新されません。

132 ページの「デフォルトの NIS マップ」 で説明されているように、デフォルトの NIS マップのデフォルトの位置は、/var/yp/*domainname* のマスターサーバー上です。*domainname* は NIS ドメインの名前です。マップを更新する必要がある場合は、マップがデフォルトのマップか否かによって 2 つの更新手順のどちらかを使用できます。

- デフォルトのマップは、ネットワークデータベースから ypinit で作成されたデフォルトセットに存在するマップです。
- デフォルトでないマップは次のいずれかです。
  - ベンダーから購入したアプリケーションに付属のマップ
  - ユーザーサイト用に特別に作成されたマップ
  - テキスト以外のファイルから作成されたマップ

この節では、さまざまな更新ツールの使用方法について説明します。実際にはこれらの更新ツールはシステム起動後に、デフォルトでないマップを追加する場合または一群の NIS サーバーを変更する場合にだけ使用します。

デフォルトセットに付いているマップの更新

デフォルトセットに付いているマップを更新する場合は、次の手順に従います。

▼ デフォルトセットに付いているマップを更新する方法

1. マスターサーバー上でスーパーユーザーになります。
   必ずマスターサーバー上だけで NIS マップを更新します。
2. 更新するマップのソースファイルを編集します **(**このファイルが /etc に存在しているか、選択された他のディレクトリに存在しているかは問題ではありません**)**。
3. 次のように入力します。

   # cd /var/yp

   # **make** *mapname*

   make コマンドは、対応するファイルに対して NIS 管理者が行った変更に従ってマップを更新します。make コマンドはまた、これらの変更を他のサーバーに伝播します。

NIS マップを伝播する

マップが更新されると、Makefile は yppush を使用して新しいマップをスレーブサーバーに伝播します (ただし、NOPUSH が Makefile に設定されている場合を除きます)。これは、ypserv デーモンに通知してマップ転送要求を送ることで実行されます。次に、スレーブサーバー上の ypserv デーモンが ypxfr プロセスを起動し、

ypxfrd プロセスがマスターサーバー上の ypxfr デーモンに連絡します。いくつかの基本検査 (マップが実際に更新されていることの確認など) が行われてマップが転送されます。そのあと、スレーブサーバー上の ypxfrd が、転送が成功したかどうかを yppush プロセスに通知します。

注 – 上記手順は、新しく作成されたマップがスレーブサーバー上に存在しない場合は動作しません。スレーブサーバー上で ypxfr を実行して、新しいマップをスレーブサーバーに転送する必要があります。

マップの伝播は失敗することがありますが、失敗した場合は ypxfr を使って手動で新しいマップ情報を転送する必要があります。ypxfr は、2 つの方法で使用できます。1 つは root の crontab ファイルを定期的に使用する方法であり、もう 1 つはコマンド行から対話形式で使用する方法です。これらの方法については、以下で説明します。

cron を使ってマップ転送を行う

マップの更新頻度はマップによってそれぞれ異なります。たとえば、デフォルトのマップである protocols.byname やデフォルトでないマップの auto_master など一部のマップは何カ月も更新されないことがありますが、一方、passwd.byname など一部のマップは 1 日に数回更新されることがあります。crontab コマンドでマップ転送をスケジュールすると、個々のマップに対して特定の伝播時間を設定できます。

マップに適切な頻度で ypxfr を定期的に実行するには、各スレーブサーバー上の root の crontab ファイルに、該当する ypxfr エントリを入れる必要があります。ypxfr は、マスターサーバー上のコピーがローカルのコピーより新しい場合に限り、マスターサーバーと連絡をとりマップを転送します。

注 – デフォルトの -m オプションが指定されている rpc.yppasswdd をマスターサーバー上で実行すると、どこかで yp パスワードが変更されるたびに、passwd デーモンが make を実行して passwd マップを作成し直します。

---

cron と ypxfr でシェルスクリプトを使用する

NIS 管理者は、各マップに対する crontab エントリを個々に作成するという方法ではなく、root の crontab コマンドでシェルスクリプトを実行してすべてのマップを定期的に更新するという方法を使用することもできます。マップ更新シェルスクリプトのサンプルは、/usr/lib/netsvc/yp ディレクトリに入っています。スクリプト名は、ypxfr_1perday、ypxfr_1perhour、ypxfr_2perday です。これらのシェルスクリプトを更新または置換することによって、容易にサイト要件に適合させることができます。 例 9–1 は、デフォルトの ypxfr_1perday シェルスクリプトを示しています。

例 **9–1** ypxfr_1perday シェルスクリプト

```
#! /bin/sh
#
# ypxfr_1perday.sh - YP マップの検査・更新を毎日行う
PATH=/bin:/usr/bin:/usr/lib/netsvc/yp:$PATH
export PATH
# set -xv
ypxfr group.byname
ypxfr group.bygid
ypxfr protocols.byname
ypxfr protocols.bynumber
ypxfr networks.byname
ypxfr networks.byaddr
ypxfr services.byname
ypxfr ypservers
```

このシェルスクリプトは、root の crontab が毎日実行されるとマップを 1 日に 1 回更新します。NIS 管理者は、1 週間に 1 回、1 ヶ月に 1 回、または 1 時間に 1 回などといった頻度でマップを更新するスクリプトを使用できますが、マップを頻繁に伝播するとパフォーマンスが低下するので注意してください。

NIS ドメインに対して構成された各スレーブサーバー上で同じシェルスクリプトを root で実行してください。各サーバー上の実行時間を変更して、マスターサーバーが動作不能にならないようにしてください。

特定のスレーブサーバーからマップを転送する場合は、シェルスクリプトの ypxfr の -h *machine* オプションを使用してください。シェルスクリプトに記述するコマンドの構文は、次のとおりです。

/usr/lib/netsvc/yp/ypxfr -h *machine* **[-c]** *mapname*

machine は転送するマップが存在するサーバー名です。*mapname* は要求されたマップ名です。マシンを指定することなく -h オプションを指定すると、ypxfr はマスターサーバーからマップを取得しようとします。ypxfr 実行時に ypserv がローカルに実行されていない場合は、ypxfr がローカル ypserver に現在のマップリクエストの取消しを送信しないよう、-c フラグを使用してください。

-s *domain* オプションを使用すると、別のドメインからローカルドメインにマップを転送できます。これらのマップは、各ドメインにおいて同じでなければなりません。たとえば、2 つのドメインが同じ services.byname マップおよび services.byaddr マップを共有することがあります。また、より細かい制御を行うための rcp または rdist を使用してファイルを複数のドメインに転送することもできます。

ypxfr を直接起動する

2 番目の方法である ypxfr の起動とは、ypxfr をコマンドとして実行することです。一般に、ypxfr をコマンドとして実行するのは例外的状況においてだけです。たとえば、一時的に NIS サーバーを設定して試験環境を作成する場合や、他のサーバーと調和して動作不能になっていた NIS サーバーを迅速に取得しようとする場合などです。

ypxfr のアクティビティを記録する

ypxfr が試みた転送およびその転送結果は、ログファイルに記録されます。/var/yp/ypxfr.log というファイルが存在する場合は、転送結果はこのファイルに記録されます。このログファイルのサイズには制限がありません。このログファイルのサイズが無限に大きくなることを防止するには、ときどき次のように入力してこのログファイルを空にしてください。

```
# cd /var/yp# cp ypxfr.log ypxfr.log.old

# cat /dev/null> /var/yp/ypxfr.log
```

これらのコマンドは、crontab で 1 週間に 1 回実行させることができます。記録を取らないようにするには、ログファイルを削除してください。

デフォルトでないマップの更新

デフォルトでないマップを更新する場合は、次の手順に従います。

1. 対応するテキストファイルを作成または編集します。
2. 新しいマップまたは更新されたマップを作成 (または再作成) します。マップ作成には 2 つの方法があります。
   - Makefile を使用する方法。デフォルトでないマップを作成するには、この方法をお勧めします。Makefile にマップのエントリが存在する場合は、make *name* を実行するだけです (*name* は作成するマップ名)。Makefile にマップのエントリが存在しない場合は、163 ページの「Makefile の更新と使用」を参照してエントリを作成してください。
   - /usr/sbin/makedbm プログラムを使用する方法。このコマンドの詳細については、makedbm(1M) のマニュアルページに説明されています。

デフォルトでないマップを makedbm で更新する

入力ファイルが存在しない場合は、makedbm でマップを更新する方法は 2 つあります。

- makedbm -u の出力先を一時ファイルに変更し、一時ファイルを更新して更新済みの一時ファイルを makedbm の入力として使用します。

- makedbm -u の出力を makedbm に渡されるパイプライン内で動作させます。分解されたマップを awk、sed、または cat で更新できる場合は、この方法をお勧めします。

テキストファイルからマップを新たに作成する

テキストファイル /var/yp/mymap.asc がマスターサーバー上のエディタまたはシェルスクリプトで作成されていると仮定します。この場合、このファイルから NIS マップを作成し、作成された NIS マップを *homedomain* サブディレクトリに入れるには、マスターサーバー上で次のように入力してください。

cd /var/yp # **makedbm** *mymap*.**asc** *homedomain*/ *mymap*

mymap マップは現在、マスターサーバーの *homedomain* ディレクトリに存在しています。この新しいマップをスレーブサーバーに転送するには、ypxfr を実行してください。

ファイルをベースとしたマップにエントリを追加する

mymap にエントリを追加することは簡単です。まず、テキストファイル /var/yp/*mymap*.asc を更新する必要があります。対応するテキストファイルを更新しないで実際の dbm ファイルを更新した場合は、更新内容が失われます。次に、上記のように makedbm を実行してください。

標準入力からマップを作成する

オリジナルのテキストファイルが存在しない場合は、キーボードから makedbm に次のように入力して NIS マップを作成します (最後に Control + D を入力します)。

ypmaster# **cd /var/yp**

ypmaster# **ypmaster# makedbm -** *homedomain*/**mymap** *key1 value1 key2 value2 key3 value3*

標準入力から作成されたマップを更新する

後でマップを更新する必要がある場合は、makedbm でマップを分解して一時ファイルを作成できます。マップを分解して一時ファイルを作成するには、次のように入力します。

% **cd /var/yp**

% **makedbm -u** *homedomain*/ *mymap*> *mymap*.**temp**

作成される一時ファイル *mymap*.temp には、1 行につき 1 つのエントリが存在します。このファイルは、任意のテキストエディタで必要に応じて編集できます。

マップを更新するには、次のように入力して更新後の一時ファイルの名前を makedbm に指定します。

% **makedbm** *mymap*.**temp** *homedomain*/*mymap*

% **rm** *mymap*.**temp**

次に root になり、次のように入力してマップをスレーブサーバーに伝播します。

yppush *mymap*

ここでは makedbm でマップを作成する方法について説明してきましたが、実際に必要なほとんどすべての作業は、ypinit と Makefile で行うことができます。ただし、システム起動後にデフォルトでないマップをデータベースに追加したり一群の NIS サーバーを変更しない場合に限ります。

/var/yp の Makefile を使用しても他の手順を使用しても、最終目的は同じです。正しく作成された dbm ファイルの新しいペアをマスターサーバー上の maps ディレクトリに配置しなければなりません。

スレーブサーバーの追加

NIS の実行後、ypinit に指定された初期リストに含まれていなかった NIS スレーブサーバーを作成する必要がある可能性があります。

NIS スレーブサーバーを追加する場合は、次の手順に従います。

▼ スレーブサーバーを追加する方法

1. スーパーユーザーとしてマスターサーバーにログインします。
2. **NIS** ドメインディレクトリに移動します。
   # **cd /var/yp/***domainname*
3. ypservers ファイルを分解します。
   # **makedbm -u ypservers > /tmp/temp_file**
   makedbm コマンドは、ypservers を ndbm フォーマットから一時 ASCII ファイル、/tmp/temp_file に変換します。

4. テキストエディタで /tmp/temp_file ファイルを編集します。つまり、新しいスレーブサーバー名をサーバーリストに追加します。この後、/tmp/temp_file ファイルを保存し、閉じます。

5. 入力ファイルに /tmp/temp_file を指定し、出力ファイルに ypservers を指定して、makedbm コマンドを実行します。

   # **makedbm /tmp/temp_file ypservers**

   makedbm は、ypservers を変換して ndbm フォーマットに戻します。

6. スレーブサーバーで次のように入力して、ypservers マップが正しいことを確認します **(**ypservers の**ASCII** ファイルは存在しないため**)**。

   slave3# **makedbm -u ypservers**

   makedbm コマンドは、画面に ypservers の各エントリを表示します。

注 **–** ypservers にマシン名が存在しない場合は、ypservers はマップファイルの更新を受信しません。これは、yppush がこのマップでスレーブサーバーリストを調べるからです。

---

7. マスターサーバーから **NIS** マップのセットをコピーして新しいスレーブサーバーの **NIS**ドメインディレクトリを設定します。

   この操作を行うには、新しい NIS スレーブサーバーでスーパーユーザーとなり、ypinit および ypbind コマンドを実行します。

   slave3# **cd /var/yp**

   slave3# **ypinit -c**

   <サーバーのリスト>

   slave3# **/usr/lib/netsvc/yp/ypbind**

8. このマシンをスレーブサーバーとして初期設定します。

   slave3# **/usr/sbin/ypinit -s** *ypmaster*

   ypmaster は、既存の NIS マスターサーバーのマシン名です。

9. ypstop を実行して、**NIS** クライアントとして実行されているマシンを停止します。

   # **/usr/lib/netsvc/yp/ypstop**

10. ypstart を実行して、**NIS** スレーブサーバーのサービスを開始します。

    # **/usr/lib/netsvc/yp/ypstart**

C2 セキュリティが装備されている NIS の使用

`$PWDIR/security/passwd.adjunct` ファイルが存在する場合は、C2 セキュリティが自動的に起動されます。`$PWDIR` は、`/var/yp/Makefile` で定義されます。C2 セキュリティモードでは、`passwd.adjunct` ファイルを使って `passwd.adjunct` NIS マップが作成されます。C2 セキュリティモードでは、`passwd.adjunct` ファイルと `shadow` ファイルの両方を使用してセキュリティを管理できます。`passwd.adjunct` ファイルは、次のように入力した場合にだけ処理されます。

```
# make passwd.adjunct
```

`make passwd` コマンドは、NIS 管理者が C2 セキュリティモードで `make` を手動で実行した場合に `passwd` マップのみを処理します。`passwd.adjunct` マップは処理されません。

マシンの NIS ドメインの変更

マシンの NIS ドメイン名を変更する場合は、次の手順に従います。

▼ マシンの NIS ドメイン名を変更する方法

1. スーパーユーザーになります。
2. マシンの `/etc/defaultdomain` ファイルを編集して、現在のドメイン名をそのマシンの新しいドメイン名に置き換えます。

   たとえば、現在のドメイン名である `sales.doc.com` を `research.doc.com` に変更します。
3. `domainname 'cat /etc/defaultdomain'` を実行します。
4. マシンを **NIS** クライアント、スレーブサーバー、またはマスターサーバーとして設定します。

   詳細については、第 8 章を参照してください。

NIS を DNS と組み合わせて使用する

一般に NIS クライアントは、マシン名とアドレスの検索に NIS だけが使用されるように、nsswitch.conf ファイルで構成されます。このような検索が失敗した場合は、NIS サーバーはこれらの結果を DNS に転送します。

▼ NIS と DNS によるマシン名とアドレスの検索を設定する

1. マシンにログインして、スーパーユーザーになります。
2. hosts.byname と hosts.byaddr という **2** つのマップファイルには、YP_INTERDOMAIN キーが必要です。このキーを検査するために、Makefile を編集し、次の行を変更します。

   ```
   #B=-b
   B=
   ```

   上記の行を次のように変更します。

   ```
   B=-b
   #B=
   ```

   これで、マップの作成時に makedbm が -b フラグで起動され、YP_INTERDOMAIN キーが ndbm ファイルに挿入されます。
3. make を実行してマップを作成し直します。

   # **/usr/ccs/bin/make hosts**
4. **NIS** サーバーのすべての /etc/resolv.conf ファイルが有効なネームサーバーを指していることを確認します。

   注 – Solaris リリース 2 を実行していない NIS サーバーがある場合は、YP_INTERDOMAIN キーがホストマップに存在することを確認してください。

   ---
5. **DNS** 転送を有効にするために、各サーバーを停止します。

   # **/usr/lib/netsvc/yp/ypstop**
6. 各サーバーを再起動します。

   # **/usr/lib/netsvc/yp/ypstart**

   この NIS 実装では、ypstart が -d オプションを使って ypserv を自動的に起動して、DNS に要求を転送します。

混在 NIS ドメインの処理

マスターサーバーとスレーブサーバーのどちらも Solaris リリース 2 を実行していない場合は、次の表を参考にして問題が発生しないように対処してください。「4.0.3+」という表記は、「SunOS のリリース 4.0.3 以降」であることを意味します。`makedm -b` は、`Makefile` の `-B` 変数への参照です。

表 **9–1** 異機種システムが混在する NIS ドメインにおける NIS/DNS

スレーブサーバー		マスターサーバー	
	4.0.3+		Solaris
4.0.3+	マスターサーバー: `makedbm –b`	マスターサーバー: `makedbm –b`	マスターサーバー: `ypserv –d`
	スレーブサーバー: `ypxfr`	スレーブサーバー: `ypxfr`	スレーブサーバー: `ypxrf –b`
Solaris NIS	マスターサーバー: `makedbm –b`	マスターサーバー: `makedbm –b`	マスターサーバー: `ypserv –d`
	スレーブサーバー: `ypxfr`	スレーブサーバー: `ypxfr`	スレーブサーバー: `ypxfr` が存在する `resolve.conf` または `ypxfr -b`

NIS サービスをオフにする

マスターサーバー上の `ypserv` が使用不可になっている場合は、NIS マップを更新できません。ネットワーク上の NIS をオフにする場合は、リブート後に次のように入力して `ypbind` ファイルを `ypbind.orig` に名前を変更すれば NIS を使用不可にできます。

% **mv /usr/lib/netsvc/yp/ypbind /usr/lib/netsvc/yp/ypbind.orig**

リブート後に特定の NIS スレーブサーバーまたはマスターサーバー上の NIS を使用不可にする場合は、その特定の NIS スレーブサーバーまたはマスターサーバー上で次のように入力してください。

% **mv /usr/lib/netsvc/yp/ypserv /usr/lib/netsvc/yp/ypserv.orig**

NIS を停止するには次のように入力してください。

% **/usr/lib/netsvc/yp/ypstop**

NIS サービスは、リブートが行われると自動的に再起動されますが、`ypbind` および `ypserv` ファイルの名前が上記のように変更された場合は起動されません。

第 10 章

NIS の障害追跡

この章では、NIS を実行しているネットワーク上で発生する問題の解決方法について説明します。NIS クライアントで発生する問題と、NIS サーバーで発生する問題を取り上げます。

NIS サーバーやクライアントで問題を解決しようとする前に、NIS 環境について説明している第 7 章を参照してください。その後この節で、各問題を適切に解説している節を参照してください。

NIS のバインドに関する問題

症状

NIS のバインドに関する一般的な問題には、次のようなものがあります。

- `ypbind` がサーバーを見つけられない、あるいはサーバーと通信できないというメッセージが表示される
- サーバーが応答していないというメッセージが表示される
- NIS が使用できないというメッセージが表示される
- クライアントのコマンドがバックグランドモードでゆっくりと処理されているか、通常よりも機能に時間がかかる
- クライアントのコマンドがハングする。システム全体は正常で新しいコマンド を実行できる場合でも、コマンドがハングすることがある
- クライアントのコマンドがあいまいなメッセージとともに、またはまったくメッセージなしでクラッシュする

1 台のクライアントに影響する NIS の問題

NIS のバインドに関する問題を示す症状が、1 台か 2 台のクライアントだけで発生している場合には、問題は多くの場合クライアントにあります。多くの NIS クライアントが、プロパティを正確にバインドできない場合には、問題は多くの場合 1 台以上の NIS サーバーにあります。182 ページの「複数のクライアントに影響する NIS の問題」を参照してください。

ypbind がクライアントで実行されていない

1 台のクライアントに問題があっても、同じサブネット上の他のクライアントは正常に機能しています。問題のあるクライアント上で、`ls -l` を `/usr` のようなディレクトリで実行します。このディレクトリには、多くのユーザーが所有するファイルが含まれ、クライアント `/etc/passwd` ファイルにはないものも含まれます。この結果の表示に、ローカルの `/etc/passwd` に存在しないファイルの所有者が名前ではなく番号で含まれる場合には、NIS サービスがそのクライアント上で機能していないことを示します。

通常これらの症状は、クライアント `ypbind` プロセスが実行されていないことを示します。`ps -e` を実行して、`ypbind` をチェックします。`ypbind` が見つからなければ、スーパーユーザーとしてログインし、次のように入力して、`ypbind` を起動します。

```
client# /usr/lib/netsvc/yp/ypstart
```

ドメイン名が指定されていないか間違っている

あるクライアントに問題があり、他のクライアントは正常に機能していますが、`ypbind` は問題のあるクライアント上で実行されています。クライアントのドメインの設定が間違っている可能性があります。

クライアント上で `domainname` コマンドを実行して、どのドメイン名が設定されているのかを調べます。

```
client7# domainname neverland.com
```

NIS のマスターサーバー上の `/var/yp` 内の実際のドメイン名と、出力を比較します。実際の NIS ドメインは、`/var/yp` ディレクトリ内のサブディレクトリとして表示されます。

```
client7# ls /var/yp...

-rwxr-xr-x 1 root Makefile drwxr-xr-x 2 root
binding drwx------ 2 root doc.com ...
```

マシン上での `domainname` の実行によって得たドメイン名が、`/var/yp` 内のディレクトリとして示されたサーバードメイン名と同じではない場合には、マシンの `/etc/defaultdomain` ファイルで指定されたドメイン名が間違っています。スー

パーユーザーとしてログインして、マシンの /etc/defaultdomain ファイル内のクライアントのドメイン名を修正します。これによって、マシンを起動するたびに、ドメイン名が正しいかどうかが確認されます。ここでマシンをリブートします。

注 – ドメイン名では大文字と小文字を区別します。

クライアントがサーバーにバインドされない

ドメイン名が正しく設定されていて ypbind が実行中でもコマンドがまだハングする場合には、ypbind コマンドを実行してクライアントがサーバーにバインドされていることを確認してください。ypbind を起動したばかりのときは、ypwhich を数回実行します。通常、1 回目ではドメインがバインドされていないことが通知され、2 回目は成功します。

サーバーが使用できない

ドメイン名が正しく設定されていて ypbind が実行中のときに、クライアントがサーバーと通信できないというメッセージを受け取った場合には、いくつかの問題が考えられます。

- バインドするサーバーのリストを含む /var/yp/binding/*domainname*/ypservers ファイルがクライアントにあるかどうかを確認します。ない場合には、ypinit -c を実行して、設定の順番にクライアントのバインド先のサーバーを指定します。
- クライアントに /var/yp/binding/*domainname*/ypservers ファイルがあり、1 台か 2 台のサーバーが使用できない場合には、十分な数のサーバーがあるかどうかを調べます。ない場合には、yppinit -c を実行して、リストにサーバーを追加します。
- クライアントの ypservers ファイルにリストされたサーバーのどれもが使用できない場合には、クライアントはブロードキャストモードで稼働中のサーバーを検索します。稼働中のサーバーがクライアントのサブネットにある場合には、クライアントはそれを見つけます (検索中はパフォーマンスが低下する)。クライアントのサブネットに稼動中のサーバーがない場合には、次のいくつかの方法で問題を解決できます。
  - クライアントがサブネットにサーバーや、それに対する経路を持たない場合には、そのサブネットに新しいスレーブサーバーをインストールできます。
  - ブロードキャストパケットをパスするようにルーターが設定されているかどうかを確認して、クライアントがブロードキャストを使って、別のサブネット上のサーバーを見つけることができるようにします。netstat -r コマンドを使って経路を検証できます。
  - 経路はあるが機能していない場合には、経路デーモン in.routed/in.rdisc が実行中かどうかを確認してください。実行中でない場合には起動します。

注 – セキュリティと管理上の理由から、クライアントにブロードキャストを使ってサーバーを検索させるのではなく、クライアントの ypservers ファイルでクライアントのバインド先のサーバーを指定してください。ブロードキャストは、ネットワークの速度を落とし、クライアントの速度も落とします。また、異なるクライアントに対して異なるサーバーをリストするため、サーバー負荷の均衡がとれなくなります。

- クライアントの ypservers ファイルにリストされたサーバーが、/etc/hosts ファイルにエントリを持っているかどうかを確認します。持っていない場合には、NIS マップホストの入力ファイルにサーバーを追加して、160 ページの「NIS マップに関する作業」で説明しているように、yppinit -c または ypinit -s を実行してマップを再構築します。
- /etc/nsswitch.conf ファイルが設定されていて、NIS の他にマシンのローカルの hosts ファイルを参照できるかどうかを確認します。スイッチの詳細については 第 2 章を参照してください。
- /etc/nsswitch.conf ファイルが設定されていて、services と rpc に対して files を参照できるかどうかを確認します。

ypwhich の表示に一貫性がない

ypwhich を同じクライアントで数回使うと、NIS サーバーが変わるので結果の表示も変わります。これは正常な状態です。NIS クライアントから NIS サーバーへのバインドは、ネットワークや NIS サーバーを使用中の場合は時間の経過に伴って変化します。ネットワークは、すべてのクライアントが受け入れ可能な応答時間を NIS サーバーから得られる点で安定した状態になります。クライアントのマシンが NIS サービスを得ているかぎりは、サービスの供給元は問題にはなりません。たとえば、NIS サーバーマシンがそれ自体の NIS サービスを、ネットワーク上の別の NIS サーバーから受けることもあります。

サーバーのバインドが不可能な場合

サーバーのバインディングが不可能な場合には ypset コマンドを使用すると、別のネットワークまたはサブネットの別のサーバーが使用可能な場合には、そのサーバーへのバインディングが一時的に可能になります。ただし、-ypset オプションを使用するためには、-ypset または -ypsetme オプションのどちらかを指定して ypbind を実行する必要があります。

注 – セキュリティ上の理由から、-ypset と -ypsetme のオプションの使用は、管理された環境のもとでのデバッグだけに限定してください。-ypset と -ypsetme のオプションを使用すると、セキュリティが侵害される恐れがあります。これらのデーモンの実行中はサーバーのバインドをだれでも変更できるため、他のユーザーの作業に問題が生じたり重要なデータへの未承認のアクセスが認められたりするためです。これらのオプションで ypbind を起動する場合は、問題が解決したら ypbind をいったん強制終了し、これらのオプションを指定しないで再起動してください。

ypbind のクラッシュ

ypbind が、起動するたびにすぐにクラッシュする場合には、システムの他の部分で問題を調べます。次のように入力して、rpcbind デーモンが存在するかどうか確認します。

% **ps -ef | grep rpcbind**

rpcbind が存在しない、安定しない、あるいは動作に異常がある場合には、RPC のマニュアルを参照してください。

正常に機能しているマシンから、問題のあるクライアント上の rpcbind と通信ができる場合があります。正常に機能しているマシンから、次のように入力してください。

% **rpcinfo** *client*

問題のあるクライアント上の rpcbind に問題がない場合には、rpcinfo によって次の出力がされます。

```
program version netid address service owner
...
100007 2 udp 0.0.0.0.2.219 ypbind superuser
100007 1 udp 0.0.0.0.2.219 ypbind superuser
100007 1 tcp 0.0.0.0.2.220 ypbind superuser
100007 2 tcp 0.0.0.0.128.4 ypbind superuser
100007 2 ticotsord \000\000\020H ypbind superuser
100007 2 ticots \000\000\020K ypbind superuser ...
```

使用中のマシンには異なる複数のアドレスがあります。それらのアドレスが表示されない場合は、ypbind によってそのサービスが登録できていません。マシンをリブートして再度 rpcinfo を実行します。表示される出力の中に ypbind プロセスがいくつかあり、 /usr/lib/netsvc/yp/ypbind を再起動しようとするたびにそれらのプロセスが変更される場合は、rpcbind デーモンが実行中であってもシステムをリブートをしてください。

複数のクライアントに影響する NIS の問題

1 台か 2 台のクライアントだけで、NIS のバインドに関する問題を示す症状が発生している場合は、そのクライアントに問題があると考えられます。178 ページの「1 台のクライアントに影響する NIS の問題」を参照してください。複数のクライアントが正しくバインドできない場合は、1 台以上の NIS サーバーに問題があると考えられます。

rpc.yppasswdd が r で始まる制限のないシェルを制限付きとみなしている

1. 次のような特殊な文字列が含まれている /etc/default/yppasswdd を作成します。 check_restricted_shell_name=1
2. 「check_restricted_shell_name=1」文字列をコメント扱いにすると、「r」のチェックは行われません。

ネットワークまたはサーバーが過負荷

ネットワークまたは NIS サーバーが過負荷状態で、ypserv が指定時間内にクライアントの ypbind プロセスに応答を戻せない場合は、NIS がハングする可能性があります。

こういった状態では、ネットワーク上のすべてのクライアントで同じまたは類似した問題が発生します。ほとんどの場合、この状態は一時的です。NIS サーバーをリブートして ypserv を再起動するか、NIS サーバーまたはネットワーク自体の負荷が減少すると、メッセージは通常消えます。

サーバーの誤動作

サーバーが起動して実行中であることを確認してください。サーバーが物理的に近くにない場合には、ping コマンドを使ってください。

NIS デーモンが実行されていない

サーバーが起動されていて実行中の場合には、クライアントマシンが正常に動作していることを調べて、ypwhich コマンドを実行します。ypwhich が応答しない場合は、そのコマンドを強制終了します。次に、NIS サーバーで root としてログインし、次のように入力して NIS の ypbind プロセスが実行中かどうかをチェックします。

```
# ps -e | grep yp
```

注 – -f オプションを ps で使用しないでください。このオプションはユーザー ID を名前に変換しようとするため、より多くのネームサービス検索が失敗する可能性があります。

ypbind または ypserv デーモンのどちらかが実行されていない場合は、それらをいったん強制終了してから、次のように入力して再起動します。

/usr/lib/netsvc/yp/ypstop

/usr/lib/netsvc/yp/ypstart

ypbind と ypserv の両プロセスが NIS サーバーで実行中の場合は、ypwhich を使用します。

ypwhich が応答しない場合は、ypserv がハングしたと考えられるため再起動が必要です。サーバーに root としてログインして、ypserv をいったん強制終了し、次のように入力して再起動します。

/usr/lib/netsvc/yp/ypstop # **/usr/lib/netsvc/yp/ypstart**

サーバーに別のバージョンの NIS マップが存在する

NIS はマップをサーバー間で伝播するので、ネットワーク上のさまざまな NIS サーバーに、同じマップの異なるバージョンが存在することがあります。相違点が長時間継続しない場合には、このバージョンの違いは許容可能です。

マップの不一致のもっとも一般的な原因は、マップの正常な伝播を妨げる何かが存在するためです。たとえば、NIS サーバーまたはルーターが、NIS サーバー間でダウンしている場合です。すべての NIS サーバーとそれらの間に設置されたルーターが実行中の場合には、ypxfr は成功します。

サーバーとルーターが正常に機能している場合には、次のことをチェックします。

- ypxfr 出力のログをとる (183 ページの「ypxfr 出力のログ」を参照)
- 制御ファイルをチェックする (184 ページの「crontab ファイルと ypxfr シェルスクリプトをチェックする」を参照)
- マスターサーバー上の ypservers マップをチェックする (184 ページの「ypservers マップをチェックする」を参照)。

ypxfr 出力のログ

特定のスレーブサーバーでマップの更新に問題がある場合には、そのサーバーにログインして ypxfr を対話形式で実行します。ypxfr が失敗すると ypxfr がその理由を通知するので、問題の修正が可能になります。ypxfr が成功するが時々失敗するような場合には、メッセージのログを取るためのログファイルを作成します。ログファイルを作成する場合は、スレーブサーバーで次のように入力します。

ypslave# **cd /var/yp**

ypslave# **touch ypxfr.log**

これによって、ypxfr からのすべての出力を保存する ypxfr.log ファイルが作成されます。

この出力は、ypxfr が対話形式で実行しているときに表示する出力と似ていますが、ログファイルの各行にはタイムスタンプが記録されます。タイムスタンプは通常とは異なる順番になる可能性がありますが、問題はありません。タイムスタンプは ypxfr が実行し始めたことを示します。ypxfr のコピーが同時に実行されても作業時間が異なる場合は、起動時とは異なる順番でサマリーステータス行がログファイルに書き込まれることがあります。断続的に発生するあらゆる種類の障害がログに示されます。

注 – 問題を解決したら、ログファイルを削除してログを停止します。削除しないと、ログは制限なく大きくなります。

---

crontab ファイルと ypxfr シェルスクリプトをチェックする

root の crontab ファイルを調べて、それが起動した ypxfr シェルスクリプトをチェックします。これらファイルにタイプミスがあると、伝播に関する問題が発生します。/var/spool/cron/crontabs/root ファイル内でシェルスクリプトを参照できない場合や、任意のシェルスクリプト内でマップを参照できない場合にも、エラーが発生します。

ypservers マップをチェックする

NIS スレーブサーバーが、そのドメインのマスターサーバー上の ypservers マップにリストされていることも確認してください。リストされていない場合でも、スレーブサーバーはサーバーとして正しく機能しますが、yppush はマップの変更をスレーブサーバーに伝播しません。

対策

NIS スレーブサーバーの問題が明白ではない場合には、rcp または ftp を使ってデバッグし、一貫性のないマップの最新バージョンを問題のない NIS サーバーからコピーして問題を解決できます。以下に問題のあるマップを転送する方法を示します。

ypslave# **rcp ypmaster:/var/yp/***mydomain***/** *map***.* /var/yp/***mydomain*

* の文字はコマンド行でエスケープされて、ypslave でローカルにではなく ypmaster で展開されます。

ypserv のクラッシュ

ypserv プロセスがほとんど即座にクラッシュして、何度再起動しても安定しないときは、デバッグプロセスは、181 ページの「ypbind のクラッシュ」で説明する内容と実質的に同じです。rpcbind デーモンの存在を次のようにチェックしてください。

ypserver% **ps -e | grep rpcbind**

デーモンが見つからない場合は、サーバーをリブートします。デーモンが見つかった場合は、デーモンが実行中であれば次のように入力して同様の出力を検索します。

% **rpcinfo -p ypserver**

```
% program     vers     proto     port     service
100000    4    tcp    111    portmapper
100000    3    tcp    111    portmapper
100068    2    udp    32813    cmsd
...
100007    1    tcp    34900    ypbind
100004    2    udp    731    ypserv
100004    1    udp    731    ypserv
100004    1    tcp    732    ypserv
100004    2    tcp    32772    ypserv
```

使用中のマシンには、異なる複数のポート番号があることがあります。ypserv プロセスを表す 4 つのエントリは、次のとおりです。

```
100004   2   udp   731   ypserv
100004   1   udp   731   ypserv
100004   1   tcp   732   ypserv
100004   2   tcp   32772   ypserv
```

エントリが 1 つもなく、ypserv がそのサービスを rpcbind で登録できない場合にはマシンをリブートしてください。エントリがある場合には、rpcbind からサービスの登録を解除してから ypserv を再起動します。rpcbind からサービ スの登録を解除するには、サーバーで次のように入力します。

rpcinfo -d *number* **1**

rpcinfo -d *number* **2**

number は、rpcinfo によって通知される ID 番号です (前述の例では、100004)。

パート IV　iPlanet Directory Server 5.1 の構成

ここでは、iPlanet Directory Server 5.1 を構成する方法について説明します。

第 11 章

iPlanet Directory Server 5.1 の構成

この章では、iPlanet Directory Server 5.1 の構成方法について説明します。Solaris LDAP クライアントで使用するために iPlanet Directory Server 5.1 を構成する前に、この章に記載された手順を実行する必要があります。

注 – iPlanet Directory Server 5.1 以外のディレクトリサーバーを使用する場合は、この章を読む必要はありません。他のディレクトリサーバーを Solaris LDAP ネーミングサービスクライアントと共に使用する場合の基本要件のリストについては、287 ページの「汎用ディレクトリサーバーの要件」 を参照してください。

---

iPlanet Directory Server 5.1 に関する詳細な情報については、次の iPlanet マニュアルを参照してください。

- 『*iPlanet Directory Server 5.1* スキーマリファレンス』
- 『*iPlanet Directory Server 5.1* 導入ガイド』
- 『*iPlanet Directory Server 5.1* 構成、コマンド、およびファイルのリファレンス』
- 『*iPlanet Directory Server 5.1* 管理者ガイド』

この章の内容は次のとおりです。

構成の準備

iPlanet Directory Server 5.1 の構成を開始する前に、さまざまなコンポーネント、設計および構成上の決定事項について理解しておく必要があります。

iPlanet Directory Server 5.1 を構成する上で、以下の節で説明されている概念をよく理解しておく必要があります。

- 構成コンポーネント
- 構成上の決定事項
- 構成プロセスの概要
- 構成権限

『*iPlanet Directory Server 5.1* 導入ガイド』には、ディレクトリの基本的な概念およびディレクトリサービスの設計および配備を成功させるのに役立つガイドラインが説明されています。

構成コンポーネント

iPlanet Directory Server 5.1 には、次のソフトウェアコンポーネントが含まれます。これらのコンポーネントは、Solaris を entire + OEM メタクラスタでインストールすると、デフォルトでインストールされます。

- iPlanet Console

  iPlanet Console は、すべての iPlanet サーバー製品に共通のユーザーインタフェースを提供します。iPlanet Console から、共通のサーバー管理作業 (サーバーの停止や起動、新規サーバーインスタンスのインストールなど) を実行できます。iPlanet Console は、任意のマシンにスタンドアロンのアプリケーションとしてインストール可能です。また、ネットワーク上にインストールして、遠隔サーバーの管理に使用することもできます。

- Administration Server

  Administration Server は、すべての iPlanet サーバーへの共通のフロントエンドとして機能します。Administration Server は、iPlanet Console から通信を受け取って適切な iPlanet サーバーに渡します。

- iPlanet Directory Server 5.1

  iPlanet Directory Server 5.1 は、オンディスクデータベースを備えた、高性能かつスケーラブルな LDAP サーバーです。iPlanet Directory Server 5.1 は、Solaris 上で `ns-slapd` プロセスとして動作します。この製品は、ディレクトリデータベースを管理し、クライアントからの要求に応答して動作するサーバーです。iPlanet Directory Server 5.1 は必須コンポーネントです。

構成の選択

ディレクトリサーバー (Directory Server) の構成時に、基本情報の指定が求められます。構成作業を開始する前に、これらの基本的なパラメータの構成方法を決定しておく必要があります。実行する構成タイプに応じて、次の情報のすべてあるいはいくつかを指定するよう求められます。

- ポート番号
- サーバーを実行するユーザーおよびグループ
- ディレクトリの接尾辞
- 複数の異なる認証ユーザー ID
- 管理ドメイン

一意のポート番号の選択

ポート番号には、1 ～ 65535 までの任意の番号を指定できます。iPlanet Directory Server 5.1 用のポート番号を選択する際、以下の点を考慮してください。

- iPlanet Directory Server 5.1 (LDAP) の標準ポート番号は 389 です。
- ポート 636 は、SSL 経由での LDAP 用に予約されています。このため、ポート 636 が使用されていない場合でも、標準 LDAP 構成ではポート 636 を使用しないでください。標準 LDAP ポート上で、TLS 経由の LDAP を使用することもできます。
- IANA (Internet Assigned Numbers Authority) により、1 ～ 1024 のポート番号がさまざまなサービスに割り当てられています。389 と 636 を除く1024 未満のポート番号は、他のサービスと競合することがあるため、使用しないでください。また、1024 未満のポート番号にアクセスできるのは、root だけです。
- iPlanet Directory Server 5.1 は、ポート 389 または 636 を使用する場合 root で実行する必要があります。
- 選択するポートが使用されていないことを確認してください。また、LDAP と LDAPS の両方を通信に使用している場合、これら 2 タイプのアクセス用に選択したポート番号が重複していないことを確認してください。

注 – LDAP ネームサービスクライアントが SSL 暗号化を使用している場合、デフォルトのポート番号 389 と 636 を使用する必要があり、そのためサーバーを `root` で実行することになります。Transport Layer Security の詳細については、220 ページの「Transport Layer Security (TLS)」 を参照してください。

---

iPlanet Directory Server 5.1 で SSL 経由の LDAP (LDAPS) を設定する方法については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

ユーザーとグループの選択

セキュリティ上の理由から、通常のユーザー権限で UNIX ベースの本稼働用サーバーを実行するのが常に最善です。このため、root 権限でディレクトリサーバーを実行するのは避けてください。ただし、ディレクトリサーバーのデフォルトポートを使用する場合は、root 権限でディレクトリサーバーを実行する必要があります。ディレクトリサーバーの起動が Administration Server により実行される場合、Administration Server は root または iPlanet Directory Server 5.1 と同じユーザーで実行する必要があります。

このため、以下の目的に対してどのユーザーアカウントを使用するかを決定する必要があります。

- iPlanet Directory Server 5.1 を実行するユーザーとグループ

  iPlanet Directory Server 5.1 を root で実行していない場合は、すべての iPlanet サーバーで使用するユーザーアカウントを作成することを強くお勧めします。既存のオペレーティングシステムのユーザーアカウントや、`nobody` アカウントは使用するべきではありません。また、iPlanet Directory Server 5.1 ファイル用の共通グループを作成する必要もあります。この場合も、`nobody` グループは使用しないでください。

- Administration Server を実行するユーザーとグループ

  デフォルトのポート番号を使用する構成の場合は、root を使用する必要があります。ただし、1024 より大きいポート番号を使用する場合は、すべての iPlanet サーバーで使用するユーザーアカウントを作成し、このアカウントで Administration Server を実行してください。

  セキュリティ上の理由から、Administration Server を root で実行している場合は、使用しないときは Administration Server を停止する必要があります。

すべての iPlanet サーバー用の共通グループ (`gid iPlanet` など) を作成して、必要に応じてファイルをサーバー間で共有できるようにする必要があります。

iPlanet Directory Server 5.1 および Administration Server をインストールする前に、使用するユーザーおよびグループのアカウントがシステムに存在することを確認してください。

認証エンティティの定義

iPlanet Directory Server 5.1 および Administration Server を構成する際、さまざまなユーザー名、識別名 (DN)、およびパスワードの入力が求められます。このログインおよびバインドエンティティのリストは、実行する構成タイプによって異なります。

- Directory Manager DN およびパスワード

  Directory Manager DN は、アクセス制御が適用されない特別なディレクトリエントリです。Directory Manager は、ディレクトリのスーパーユーザーと見なすことができます。iPlanet Directory Server の以前のリリースでは、Directory Manager DN は root DN として知られていました)。

デフォルトの Directory Manager DN は、`cn=Directory Manager` です。Directory Manager DN は特別なエントリであるため、iPlanet Directory Server 5.1 用に構成されたどの接尾辞にも準拠する必要はありません。このため、Directory Manager DN と同じ DN を保持する、iPlanet Directory Server 5.1 の実エントリを手動で作成してはなりません。

Directory Manager のパスワードは、8 文字以上でなければならず、ASCII 文字、数字、および記号に限定されています。

注 – クライアントの `add` および `modify` 操作中にマスターサーバーを参照するように複製を設定する場合は特に、すべての LDAP サーバーで同じ Directory Manager DN およびパスワードを使用することをお勧めします。

- 構成ディレクトリ管理者の ID およびパスワード

  構成ディレクトリ管理者は、iPlanet Console からアクセス可能なすべての iPlanet サーバーを管理するユーザーです。このユーザー ID を使用してログインすると、iPlanet Console のサーバートポロジ領域内に表示可能な iPlanet サーバーをすべて管理できます。

  セキュリティ上の理由から、構成ディレクトリ管理者をディレクトリ管理者と同じにすべきではありません。デフォルトの構成ディレクトリ管理者 ID は `admin` です。

- Administration Server ユーザーおよびパスワード

  カスタム構成時にのみ、この情報の入力が求められます。Administration Server ユーザーは、ローカルの Administration Server へのすべてのアクセス権を保持する特別なユーザーです。このユーザーとして認証されると、このサーバー上に格納された iPlanet サーバーをすべて管理できます。

  Administration Server のユーザー ID およびパスワードが使用されるのは、iPlanet Directory Server 5.1 がダウンし、構成ディレクトリ管理者としてログインすることができない場合だけです。このユーザー ID によって、Administration Server にアクセスして、iPlanet Directory Server 5.1 の起動、ログファイルの表示などの重要な障害からの復旧操作を実行できます。

  Administration Server のユーザーおよびパスワードは通常、構成ディレクトリ管理者の ID およびパスワードと同一にする必要があります。

ディレクトリ接尾辞の選択

ディレクトリ接尾辞は、ディレクトリツリー内の最初のエントリを表すディレクトリエントリです。企業のデータを含むツリー用に、ディレクトリ接尾辞が少なくとも 1 つ必要です。一般に、企業で使用する DNS ホスト名に対応するディレクトリ接尾辞を選択します。たとえば、組織で DNS 名 `example.com` を使用する場合、接尾辞 `dc=example`、`dc=com` を選択します。

ディレクトリサービス用の接尾辞を計画する際の詳細については、『*iPlanet Directory Server 5.1* 導入ガイド』を参照してください。

構成ディレクトリの位置の選択

Directory Server 5.1 を含む多数の iPlanet サーバーは、iPlanet Directory Server 5.1 のインスタンスを使用して構成情報を格納します。この情報は、`o=NetscapeRoot` ディレクトリツリーに格納されます。これは、ユーザーのディレクトリデータの格納先の iPlanet Directory Server 5.1 上に保持されなくてもかまいません。使用する構成ディレクトリは、`o=NetscapeRoot` を含む iPlanet Directory Server 5.1 です。

他の iPlanet サーバーをサポートするためにのみ、iPlanet Directory Server 5.1 をインストールする場合、その iPlanet Directory Server 5.1 が構成ディレクトリになります。一般的なディレクトリサービスの一部として使用するために iPlanet Directory Server 5.1 をインストールする場合、企業内に複数の iPlanet Directory Server 5.1 がインストールされていること、構成ディレクトリツリー `o=NetscapeRoot` を保持する iPlanet Directory Server 5.1 を決定することが必要です。この決定をしてから、iPlanet サーバー (iPlanet Directory Server 5.1 を含む) をインストールする必要があります。

アップグレードを容易にするため、`o=NetscapeRoot` ツリーのサポート専用の iPlanet Directory Server 5.1 インスタンスを使用する必要があります。このサーバーインスタンスでは、企業のディレクトリデータ管理に関するその他の機能を一切実行するべきではありません。また、このサーバーインスタンスでポート 389 を使用しないでください。このサーバーインスタンスでポート 389 を使用すると、企業のディレクトリデータ管理に使用するホストに iPlanet Directory Server 5.1 をインストールできなくなる場合があります。

通常、構成ディレクトリはトラフィックをごくわずかしか使用しないため、このサーバーインスタンスをより負荷の高い別の iPlanet Directory Server 5.1 インスタンスと共に 1 台のマシンに共存させることができます。ただし、大規模なサイトに大量の iPlanet サーバーをインストールする場合、他の本稼働サーバーのパフォーマンスに影響が出ないよう、構成ディレクトリ専用のローエンドマシンを準備することもできます。iPlanet サーバーの構成によって、構成ディレクトリへの書き込み動作が行われます。大規模なサイトの場合、この書き込み動作により他のディレクトリ動作に短時間影響を与えることになります。

また、どのようなディレクトリ構成を行う場合でも、可用性と信頼性を向上させるため、構成ディレクトリの複製を作成することを検討してください。複製と DNS ラウンドロビンを使用してディレクトリの可用性を向上させる方法については、『*iPlanet Directory Server 5.1* 導入ガイド』を参照してください。

注意 – 構成ディレクトリツリーが破壊された場合、構成ディレクトリに登録した他の iPlanet サーバーすべての再インストールが必要になることがあります。構成ディレクトリを扱う場合、次のガイドラインに従って操作を行な ってください。

- iPlanet サーバーを新たにインストールした後は、必ず構成ディレクトリのバックアップを作成する
- 構成ディレクトリの使用するホスト名またはポート番号を決して変更しない
- 構成ディレクトリツリーを決して直接変更しない。さまざまな iPlanet サーバーのセットアッププログラムのみがその構成を修正するべきである

ユーザーディレクトリの位置の選択

構成ディレクトリが iPlanet サーバー管理に使用される iPlanet Directory Server 5.1 であるのと同様、ユーザーディレクトリは企業内のユーザーおよびグループのエントリを含む iPlanet Directory Server 5.1 です。

大半のディレクトリ構成では、ユーザーディレクトリと構成ディレクトリは、2 つの別個のサーバーインスタンスにする必要があります。これらのサーバーインスタンスを同じマシンにインストールすることも可能ですが、最良の結果を得るために、構成ディレクトリを別のマシンに配置することを検討してください。

ユーザーディレクトリと構成ディレクトリとでは、ユーザーディレクトリの方が圧倒的に多くのディレクトリトラフィックを受け取ります。このため、最も多くのコンピュータリソースをユーザーディレクトリに割り当てる必要があります。構成ディレクトリが受け取るトラフィックはごくわずかであるため、構成ディレクトリはローエンドのリソースのマシンにインストールできます。

また、ユーザーディレクトリにデフォルトのディレクトリポート (389 と 636) を使用する必要があります。この目的専用のサーバーインスタンスにより構成ディレクトリを管理する場合、構成ディレクトリ用に非標準ポートを使用してください。

構成ディレクトリをネットワーク上にインストールするまで、ユーザーディレクトリをインストールすることはできません。

管理ドメインの選択

管理ドメインを使用すると、iPlanet サーバーを論理的にグループ化してサーバー管理タスクをより容易に分散できます。一般的なシナリオとして、企業内の 2 つの部門がそれぞれ個別の iPlanet サーバーを制御することを望む場合を考えてみましょう。ただし、企業内のすべてのサーバーを集中的に制御することも望まれています。管理ドメインを使用すると、これらの競合する目標を達成することができます。

管理ドメインには次の特性があります。

- すべてのサーバーが、所属するドメインに関係なく共通の構成ディレクトリを共有する
- 2 つの異なるドメイン内のサーバーは、認証用とユーザー管理用に 2 つの異なるユーザーディレクトリを使用できる
- 構成ディレクトリ管理者は、所属するドメインに関係なくインストール済みの iPlanet サーバーすべてに完全なアクセス権を保持する
- 各管理ドメインは、管理ドメイン所有者が構成できる。管理ドメイン所有者は、ドメイン内のすべてのサーバーに完全なアクセス権を保持するが、他の管理ドメイン内のサーバーへのアクセス権は保持しない
- 管理ドメイン所有者は、サーバーに対する管理アクセス権限をドメイン内のサーバー単位に各ユーザーに付与できる

多くの構成では、管理ドメインを 1 つだけ保持することもできます。この場合、組織を表す名前を選択します。その他の構成の場合、サイトの需要に応じて異なる複数のドメインを保持できます。後者の場合には、そのドメイン内のサーバーを制御する組織に関係した名前を管理ドメインに付けてください。

たとえば、ISP が iPlanet サーバーのインストールおよび管理を求める 3 種類の顧客が存在する場合、3 つの管理ドメインを作成してそれぞれに顧客に関連した名前を付けます。

構成プロセスの概要

iPlanet Directory Server 5.1 をインストールする際、複数存在する構成プロセスの 1 つを使用できます。各構成プロセスを使用して、そのプロセスを理解して、さまざまなコンポーネントを適切な順序で構成することができます。

以下の節では、利用可能な構成プロセスの概要を示します。

構成プロセスの選択

iPlanet Directory Server 5.1 ソフトウェアは、セットアッププログラムの提供する 3 つの異なる構成方法の 1 つを使用して構成できます。

- エクスプレス (Express) 構成

  iPlanet Directory Server 5.1 の評価またはテストを目的としてインストールを行う場合、この構成を選択してください。詳細については、197 ページの「エクスプレス構成の使用」を参照してください。
- 標準 (Typical) 構成

通常の iPlanet Directory Server 5.1 インストールを行う場合、この構成を選択してください。詳細については、198 ページの「標準構成の使用」を参照してください。

- カスタム (Custom) 構成

  iPlanet Directory Server 5.1 では、カスタム構成プロセスは、標準構成プロセスと非常に似ています。主な違いは、カスタム構成プロセスでは LDIF ファイルをインポートして、デフォルトで作成されるユーザーディレクトリデータベースを初期化できることです。

使用する構成プロセスのタイプを決定した後、次の手順で iPlanet Directory Server 5.1 を構成します。

1. ディレクトリサービスを計画します。ディレクトリツリーを事前に計画することにより、組織の成長に合わせて管理可能でスケーラブルなサービスを設計できます。ディレクトリサービスを計画する際の指針については、『*iPlanet Directory Server 5.1* 導入ガイド』を参照してください。
2. この章の説明に従って、iPlanet Directory Server 5.1 を構成します。
3. ディレクトリ接尾辞およびデータベースを作成します。ディレクトリをこの時点で生成する必要はありません。ただし、すべての主要ルートおよびブランチポイントを含む、ツリーの基本構造を作成する必要があります。ディレクトリエントリの作成方法の詳細については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。
4. iPlanet Directory Server 5.1 の追加インスタンスを作成し、iPlanet Directory Server 5.1 のインスタンス間の複製規約 (replication agreement) を設定して、データの可用性を保証します。

エクスプレス構成および標準構成の使用

エクスプレス構成の使用

iPlanet Directory Server 5.1 を製品の評価またはテスト目的でインストールする場合は、エクスプレス構成を使用します。エクスプレス構成では、サーバーのポート番号やディレクトリ接尾辞を選択することはできません。このため、この構成は本稼働用には使用しないでください。エクスプレス構成を実行するには、以下を実行します。

▼ エクスプレス構成を使用した iPlanet Directory Server 5.1 の構成方法

1. スーパーユーザーになります。
2. 以下を入力して、**iPlanet Directory Server 5.1** プログラムを実行します。

```
# /usr/sbin/directoryserver setup
```

3. インストール内容を指定するよう求められたら、**Enter** キーを押してデフォルトの **iPlanet** サーバーを指定します。
4. 構成タイプの入力の指定を求められたら、「**Express**」を選択します。
5. 「**the user and group to run the servers as**」に、このサーバーを実行するユーザーおよびグループの識別情報を入力します。
6. 「**1 Configuration Directory Administrator ID and password**」に、完全な権限でコンソールへの認証を行う場合のログイン時の名前およびパスワードを入力します。これは、**iPlanet Console** 用の **root** またはスーパーユーザーの **ID** となります。

   最小限の手順でサーバーの構成が行われ、サーバーが起動します。Administration Server が使用するホストおよびポート番号が表示されます。

   新規 iPlanet Directory Server 5.1 構成に関して、以下の点に注意してください。

   - iPlanet Directory Server 5.1 は、ポート 389 で待機する
   - サーバーは、次の接尾辞を使用するように構成される

     `dc=your_machine s_DNS_domain_name`

     つまり、マシン名が `test.example.com` の場合、このサーバーの接尾辞は `dc=example`、`dc=com` と構成される

     `o=NetscapeRoot`

   接尾辞 `o=NetscapeRoot` 以下のディレクトリの内容を、変更してはなりません。最初の接尾辞の下にデータを作成するか、この目的で使用する接尾辞を新規に作成します。iPlanet Directory Server 5.1 用の新規接尾辞の作成方法については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

標準構成の使用

iPlanet Directory Server 5.1 の初回時の構成の大半は、セットアッププログラムの 標準オプションを使用して実行できます。

▼ 標準構成を使用した iPlanet Directory Server 5.1 の構成

1. スーパーユーザーになります。
2. **iPlanet Directory Server 5.1** プログラムを実行します。

   ```
   # /usr/sbin/directoryserver setup
   ```

3. インストールする内容を指定するよう求められたら、**Enter** キーを押してデフォルトの **iPlanet** サーバーを指定します。

4. **Directory Suite** および **Administration Service** の指定が求められたら、**Enter** キーを押してすべてデフォルトを選択します。

5. **Enter** キーを押して、すべての **Directory Suite** コンポーネントを選択します。

6. **Enter** キーを押して、すべての **Administration** コンポーネントを選択します。

7. ホスト名の指定が求められたら、デフォルトを選択するか、別の完全指定ドメイン名を入力します。

注意 – インストールプログラムがシステムの DNS 名を検出できない場合、デフォルトのホスト名が不正な名前になる可能性があります。たとえば、システムが NIS を使用する場合、DNS 名が指定されない場合があります。ホスト名は、完全指定のホストおよびドメイン名でなければなりません。デフォルトのホスト名が完全指定のホストおよびドメイン名ではない場合、構成は失敗します。

8. 次に、セットアッププログラムにより、**System User** および **System Group** 名の指定が求められます。その元でサーバーを実行するこれらの識別情報を入力してください。

9. このディレクトリが `o=NetscapeRoot` ツリーを保持する場合、構成ディレクトリについて、デフォルトを選択します。それ以外の場合、「**`Yes`**」と入力します。次に、構成ディレクトリの接続情報を指定するよう求められます。

   インストール中のサーバーが構成ディレクトリではない場合、この構成を続行するには構成ディレクトリが存在している必要があります。

10. セットアッププログラムにより、インストール中のサーバーがユーザーデータ用のサーバーであるかどうかが尋ねられます。たいていの場合、デフォルトを選択できます。ただし、このサーバーインスタンスを構成ディレクトリ専用にする場合は、「**`Yes`**」と入力する必要があります。

11. **iPlanet Directory Server 5.1** ポートとして、デフォルト **(389)** を選択します。ただし、別のアプリケーションがすでにこのポートを使用している場合は除きます。

12. **iPlanet Directory Server 5.1** の識別子として一意の値を入力します **(**通常はデフォルト値でかまいません**)**。

    この値は、iPlanet Directory Server 5.1 インスタンスのインストール先ディレクトリの名前の一部に使用されます。たとえばマシンのホスト名が `phonebook` の場合、この名前がデフォルトになります。この名前を選択すると、iPlanet Directory Server 5.1 インスタンスが `slapd-phonebook` というラベルのディレクトリにインストールされます。

注意 – iPlanet Directory Server 5.1 の識別子に、ピリオドを含めることはできません。たとえば example.server.com は、サーバー識別子の有効な名前ではありません。

13. 「**Configuration Directory Administrator ID and password**」に、コンソールへの認証を完全な権限で行う場合の、ログイン時の名前およびパスワードを入力します。

14. ディレクトリ接尾辞として、企業にとって意味のある識別名を入力します。

    この文字列は全組織のディレクトリエントリ名に使用されます。このため、組織を適切に表す名前を選択してください。インターネット DNS 名に対応する接尾辞を使用することをお勧めします。

    たとえば、組織で DNS 名 example.com を使用している場合、ここには `dc=example`、`dc=com` と入力します。

15. 「**Directory Manager DN**」に、無制限の権限でディレクトリ内容を管理する際に使用する識別名を入力します。

注 – すべての識別名は、UTF-8 文字セット形式で入力する必要があります。ISO-8859-1 など、以前の形式はサポートされていません。

---

    iPlanet Directory Server 5.1 より前のリリースでは、Directory Manager は root DN とされていました。これは、アクセス制御を無視したい場合に、ディレクトリにバインドするエントリです。この識別名は短くてもかまいません。また、ディレクトリ用に構成された接尾辞に準拠する必要もありません。ただし、ディレクトリ内に格納された実エントリに対応させることはできません。

16. **Directory Manager** パスワードに、**8** 文字以上の値を入力します。

17. 「**Administration Domain**」に、このサーバーの所属先ドメインを入力します。

    ドメインの管理を担当する組織を表す、一意の名前を入力する必要があります。

18. 管理ポート番号として、未使用の値を入力します **(**たとえば、**iPlanet Directory Server 5.1** を示す値として **5100** を使用することもできます**)**。この値は記録しておき、後で参照できるようにしておきます。

19. **Administration Server** を実行するユーザーとして、 **`root`** **(**デフォルト**)** を入力します。

    最小限の手順でサーバーの構成が行われ、サーバーが起動します。Administration Server が使用するホストおよびポート番号が表示されます。サーバーは次の接尾辞を使用するように構成されます。

    - 構成した接尾辞
    - `o=NetscapeRoot`

    接尾辞 `o=NetscapeRoot` 以下のディレクトリの内容を、変更してはなりません。最初の接尾辞の下にデータを作成するか、この目的で使用する接尾辞を新規に作成します。iPlanet Directory Server 5.1 用に接尾辞を新たに作成する方法については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

パート V　LDAP ネームサービスの設定と管理

ここでは、LDAP ネームサービスの概要を説明します。 また、Solaris オペレーティング環境での LDAP ネームサービスの設定、構成、管理、および障害追跡について、特に iPlanet Directory Server 5.1 の使用方法に焦点を当てて説明します。

第 12 章

LDAP ネームサービスの紹介 (概要/リファレンス)

LDAP に関する章では、iPlanet Directory Server 5.1 で動作するように Solaris ネームサービスクライアントを設定する方法を説明します。一般的なディレクトリサーバー要件については、第 18 章で簡潔に説明します。

注 – ディレクトリサーバーは LDAP サーバーである必要はありませんが、LDAP に関する章では、「ディレクトリサーバー」という語は「LDAP サーバー」の同義語として使用します。

対象読者

LDAP ネームサービスに関するこれらの章は、LDAP に関する実務上の知識を持つシステム管理者を対象としています。以下のリストは、本書を利用して Solaris ベースの LDAP ネームサービスを配備する前によく理解しておく必要のある概念の一部です。

- LDAP 情報モデル (エントリ、オブジェクトクラス、属性、タイプ、値)
- LDAP ネームモデル (ディレクトリ情報ツリー (DIT) 構造)
- LDAP 機能モデル (検索パラメータ: ベースオブジェクト (DN)、スコープ、サイズ制限、時間制限、フィルタ (iPlanet Directory Server のインデックスを表示する)、属性リスト)
- LDAP セキュリティモデル (認証方式、アクセス制御モデル)
- LDAP ディレクトリサービスの全体計画および設計 (データの計画方法、DIT、トポロジ、複製、およびセキュリティの設計方法を含む)

推奨される前提知識

前述の概念を詳しく知りたい場合、または LDAP や一般的なディレクトリサービスの配備について知りたい場合、以下のドキュメントが役に立ちます。

- 『*Understanding and Deploying LDAP Directory Services* 』Timothy A. Howes, Ph.D、Mark C. Smith 共著

  LDAP ディレクトリサービスの処理をはじめから終わりまで紹介するとともに、大規模な大学、大規模な多国籍企業、およびエクストラネットを備えた企業への LDAP 導入に関する有用なケーススタディが含まれています。

- 『*iPlanet Directory Server 5.1* 導入ガイド』。 このドキュメントは、Solaris 9 Documentation CD に含まれています。

  このドキュメントでは、基本的なディレクトリ計画 (ディレクトリ設計、スキーマ設計、ディレクトリツリー、トポロジ、複製、およびセキュリティを含む) が説明されています。最後の章では、簡単な配備に加え、世界中に存在する何百万ものユーザーをサポートする複雑な配備を行う際の参考になる、サンプルの開発シナリオを紹介しています。

- 『*iPlanet Directory Server 5.1* 管理者ガイド』。 このドキュメントは、Solaris 9 Documentation CD に含まれています。

その他の前提条件

NIS+ から LDAP への移行を行う場合、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: FNS*、*NIS+* 編*)*』の付録「NIS+ から LDAP への移行」を参照して移行を完了してから、本書の各章に進んでください。

iPlanet Directory Server 5.1 のインストールが必要な場合は、『*iPlanet Directory Server 5.1* インストールガイド』を参照してください。

LDAP ネームサービスとその他のネームサービスの比較

以下に、FNS、DNS、NIS、NIS+、および LDAP ネームサービスの比較一覧を示します。

	DNS	NIS	NIS+	FNS	LDAP
名前空間	階層	フラット	階層	階層	階層

	DNS	NIS	NIS+	FNS	LDAP
データ記憶領域	ファイル/ リソースレコード	2 列のマップ	複数列のテーブル	マップ	ディレクトリ (可変) をインデックス化したデータベース
サーバー	マスター/スレーブ	マスター /スレーブ	ルートマスター/ 非ルートマスター主/ 副キャッシュ/スタブ	なし	マスター/複製 マルチマスター複製
セキュリティ	なし	なし (root または、なし)	DES 認証	なし (root または、なし)	SSL、可変
トランスポート	TCP/IP	RPC	RPC	RPC	TCP/IP
スケール	グローバル	LAN	LAN	グローバル (DNS 付) /LAN	グローバル

完全指定ドメイン名の使用

LDAP クライアントと NIS や NIS+ クライアントとの 1 つの重要な相違点は、LDAP クライアントが DNS の場合と同様、ホスト名として常に完全指定ドメイン名 (FQDN) を返すことです。たとえば、次のドメイン名を考えてみましょう。

```
west.example.net
```

この場合、ホスト名 *server* を検索する場合、`gethostbyname()` および `getipnodebyname()` はホスト名を FQDN で返します。

```
server.west.example.net
```

また、`server-#` のようなインタフェース固有の別名を使用した場合も、完全指定ホスト名の長いリストが返されます。ホスト名を使用してファイルシステムの共有や他の検査を実行する場合、この点に留意する必要があります。ローカルホストには非 FQDN を想定し、DNS 解決済み遠隔ホストにのみ FQDN を想定している場合は特に注意が必要です。DNS と異なるドメイン名を使用して LDAP を設定すると、同じホストでも検索元によって FQDN が異なることがあります。

LDAP ネームサービスの利点

- LDAP を使用すると、アプリケーション固有の情報を置き換えて情報の整理統合を実行し、管理するデータベースの数を減らすことができる

- LDAP を使用すると、マスターと複製との間でより頻繁にデータの同期を取ることができる
- LDAP では、プラットフォーム間およびベンダー間の互換性が維持されている

LDAP ネームサービスの欠点

以下に、その他のネームサービスと比較して LDAP の欠点を示します。

- Solaris 8 より前のクライアントがサポートされない
- LDAP サーバーをそのクライアントとして使用することはできない
- LDAP ネームサービスの設定および管理がより複雑なため、注意深い計画が必要である

注 – ディレクトリサーバー (LDAP サーバー) をそのクライアントとして使用することはできません。つまり、ディレクトリサーバーソフトウェアを実行中のマシンを、LDAP ネームサーバークライアントにすることはできません。

Solaris 9 LDAP ネームサービスの新機能

- `idsconfig` の使用による、LDAP ディレクトリサーバー設定の簡略化
- 強力な認証、TLS 暗号化セッションをサポートする、より堅牢なセキュリティモデル。クライアントのプロキシ資格は、ディレクトリサーバー上のクライアントプロファイルには格納されていない
- `ldapaddent` コマンドを使用して、データをサーバー上に生成およびダンプすることができる
- サービス検索記述子および属性マップ
- 新規プロファイルスキーマ

NIS+ から LDAP への移行

注 – NIS+ は、将来のリリースでサポートされない可能性があります。Solaris 9 オペレーティング環境には、NIS+ から LDAP への移行を支援するツールが用意されています。

詳細については、
`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

NIS+ から LDAP への移行の詳細については、第 19 章を参照してください。

LDAP ネームサービスの設定 (作業マップ)

表 **12–1** Java Plug-in の紹介

作業	参照先
ネットワークモデルの計画	230 ページの「ネットワークモデルの計画」
DIT の計画	230 ページの「ディレクトリ情報ツリー (DIT) の計画」
複製サーバーの設定	232 ページの「複製サーバー」
セキュリティモデルの計画	233 ページの「セキュリティモデルの計画」
クライアントプロファイルおよびデフォルト属性値の選択	234 ページの「クライアントプロファイルおよびデフォルト属性値の計画」
データ生成の計画	234 ページの「データ生成の計画」
iPlanet Directory Server 5.1 を構成して、LDAP ネームサービスから使用可能にする	197 ページの「エクスプレス構成および標準構成の使用」
iPlanet Directory Server 5.1 を設定して、LDAP ネームクライアントから使用可能にする	第 15 章

表 **12–1** Java Plug-in の紹介　(続き)

作業	参照先
プリンタエントリの管理	247 ページの「プリンタエントリの管理」
LDAP クライアントの初期化	250 ページの「クライアントの初期設定」
プロファイルを使用したクライアントの初期化	251 ページの「プロファイルを使用してクライアントを初期化する」
手動によるクライアントの初期化	252 ページの「クライアントを手動で初期設定する」
クライアントの初期化解除	253 ページの「クライアントの初期設定を解除する」
サービス検索記述子を使用した、クライアントプロファイルの変更	241 ページの「サービス検索記述子を使用してさまざまなサービスへのクライアントアクセスを変更する」
ネームサービス情報の取得	254 ページの「ネームサービス情報の検出」
クライアント環境のカスタマイズ	256 ページの「クライアント環境のカスタマイズ」

第 13 章

基本コンポーネントおよび概念 (概要)

この章の内容は次のとおりです。

LDAP データ交換フォーマット (LDIF)

LDIF は、ディレクトリサービス情報を記述するために使用されるテキストベースのフォーマットです。LDIF を使用することで、ディレクトリ間で情報の移動が可能となります。以下に、各サービスの LDIF フォーマットの例を示します。この情報を表示するには、 –l オプションを指定して ldaplist を実行してください。

% **ldaplist -l hosts myhost**

```
hosts

dn: cn=myhost+ipHostNumber=7.7.7.115,ou=Hosts,dc=mydc,dc=mycom,dc=com
cn: myhost
iphostnumber: 7.7.7.115
objectclass: top
objectclass: device
objectclass: ipHost
description: host 1 - floor 1 - Lab a - building b
```

% **% ldaplist -l passwd user1**

```
passwd
```

```
dn: uid=user1,ou=People,dc=mydc,dc=mycom,dc=com
uid: user1
cn: user1
userpassword: {crypt}duTx91g7PoNzE
uidnumber: 199995
gidnumber: 20
gecos: Joe Smith [New York]
homedirectory: /home/user1
loginshell: /bin/csh
objectclass: top
objectclass: shadowAccount
objectclass: account
objectclass: posixAccount
```

% **ldaplist –l services name**

```
services

dn: cn=name+ipServiceProtocol=udp,ou=Services,dc=mydc,dc=mycom,dc=com
cn: name
cn: nameserver
ipserviceprotocol: udp
ipserviceport: 42
objectclass: top
objectclass: ipService
```

% **ldaplist –l group mygroup**

```
group

dn: cn=mygroup,ou=Group,dc=mydc,dc=mycom,dc=com
cn: mygroup
gidnumber: 4441
memberuid: user1
memberuid: user2
memberuid: user3
userpassword: {crypt}duTx91g7PoNzE
objectclass: top
objectclass: posixGroup
```

% **ldaplist –l netgroup mynetgroup**

```
netgroup

cn=mynetgroup,ou=netgroup,dc=central,dc=sun,dc=com
objectclass=nisNetgroup
objectclass=top
cn=mynetgroup
nisnetgrouptriple=(user1..mydc.mycom.com,-,)
nisnetgrouptriple=(user1.,-,)
membernisnetgroup=mylab
```

% **ldaplist –l networks 200.20.20.0**

```
networks
```

```
dn: ipNetworkNumber=200.20.20.0,ou=Networks,dc=mydc,dc=mycom,dc=com
cn: mynet-200-20-20
ipnetworknumber: 200.20.20.0
objectclass: top
objectclass: ipNetwork
description: my Lab Network
ipnetmasknumber: 255.255.255.0
```

% **ldaplist –l netmasks 201.20.20.0**

```
netmasks

dn: ipNetworkNumber=201.20.20.0,ou=Networks,dc=mydc,dc=mycom,dc=com
cn: net-201
ipnetworknumber: 201.20.20.0
objectclass: top
objectclass: ipNetwork
description: my net 201
ipnetmasknumber: 255.255.255.0
```

% **ldaplist –l rpc ypserv**

```
rpc

dn: cn=ypserv,ou=Rpc,dc=mydc,dc=mycom,dc=com
cn: ypserv
cn: ypprog
oncrpcnumber: 100004
objectclass: top
objectclass: oncRpc
```

% **ldaplist –l protocols tcp**

```
protocols

dn: cn=tcp,ou=Protocols,dc=mydc,dc=mycom,dc=com
cn: tcp
ipprotocolnumber: 6
description: transmission control protocol
objectclass: top
objectclass: ipProtocol
```

% **ldaplist –l bootparams myhost**

```
bootparams

dn: cn=myhost,ou=Ethers,dc=mydc,dc=mycom,dc=com
bootparameter: root=boothost:/export/a/b/c/d/e
objectclass: top
objectclass: device
objectclass: bootableDevice
cn: myhost
```

% **ldaplist –l ethers myhost**

```
ethers

dn: cn=myhost,ou=Ethers,dc=mydc,dc=mycom,dc=com
macaddress: 8:1:21:71:31:c1
objectclass: top
objectclass: device
objectclass: ieee802Device
cn: myhost
```

% **ldaplist –l publickey myhost**

```
publickey

dn: cn=myhost+ipHostNumber=200.20.20.99,ou=Hosts,dc=mydc,dc=mycom,dc=com
cn: myhost
iphostnumber: 200.20.20.99
description: Joe Smith
nispublickey: 9cc01614d929848849add28d090acdaa1c78270aeec969c9
nissecretkey: 9999999998769c999c39e7a6ed4e7afd687d4b99908b4de99
objectclass: top
objectclass: NisKeyObject
objectclass: device
objectclass: ipHost
```

% **ldaplist –l aliases myname**

```
aliases

dn: mail=myname,ou=aliases,dc=mydc,dc=mycom,dc=com
cn: myname
mail: myname
objectclass: top
objectclass: mailgroup
mgrprfc822mailmember: my.name
```

デフォルトのディレクトリ情報ツリー (DIT)

Solaris LDAP クライアントはデフォルトで、DIT がある特定の構造を持っていると想定して情報にアクセスします。LDAP サーバーがサポートするドメインごとに、想定された構造を持つ想定されたサブツリーがあります。ただしこのデフォルト構造は、サービス検索記述子 (Service Search Descriptor、SSD) を指定することで上書きできます。指定されたドメインでは、デフォルト DIT が、特定の情報タイプのエントリを含む多数のサブツリーを保持します。これらのサブツリー名については次の表を参照してください。

表 13–1 DIT のデフォルト位置

デフォルトコンテナ	情報タイプ
ou=Ethers	bootparams(4)、ethers(4)
ou=Group	group(4)
ou=Hosts	hosts(4)、ipnodes(4)、publickey (ホスト用)
ou=Aliases	aliases(4)
ou=Netgroup	netgroup(4)
ou=Networks	networks(4)、netmasks(4)
ou=People	passwd(1)、shadow(4)、user_attr(4)、audit_user(4)、publickey (ユーザー用)
ou=printers	printers(4)
ou=Protocols	protocols(4)
ou=Rpc	rpc(4)
ou=Services	services(4)
ou=SolarisAuthAttr	auth_attr(4)
ou=SolarisProfAttr	prof_attr(4)、exec_attr(4)
ou=projects	project
automountMap=auto_ *	auto_*

デフォルトスキーマ

スキーマは、LDAP ディレクトリ内にエントリとして格納可能な情報タイプの定義です。LDAP ネームサービスクライアントをサポートするために、ディレクトリサーバースキーマの拡張が必要な場合があります。IETF および Solaris 固有のスキーマの詳細については第 18 章を参照してください。IETF Web サイト http://www.ietf.org から、さまざまな RFC にアクセスすることもできます。

サービス検索記述子 (SSD) とスキーママッピング

注 – スキーママッピングは、注意深くかつ一貫した方法で使用する必要があります。

前述したように、LDAP ネームサービスはデフォルトで、DIT が特定の方法で構築されているという想定のもとに動作します。必要に応じて、DIT 内のデフォルト以外の場所を検索するように Solaris LDAP ネームサービスに指示することができます。また、デフォルトスキーマで指定された属性やオブジェクトクラスの代わりに、別の属性やオブジェクトクラスを指定して使用することもできます。デフォルトフィルタの詳細については、287 ページの「LDAP ネームサービスで使用されるデフォルトフィルタ」を参照してください。

SSD

`serviceSearchDescriptor` 属性は、LDAP ネームサービスクライアントが特定のサービスに関する情報を検索する方法および場所を定義します。`serviceSearchDescriptor` には、サービス名に続き、1 つ以上のセミコロンで区切られたベース - スコープ - フィルタのセットが含まれます。これらのベース - スコープ - フィルタのセットは特定のサービス専用の検索定義に使用され、指定された順番で検索されます。特定のサービスに対して複数のベース - スコープ - フィルタが指定されている場合、このサービスは、特定のエントリを検索する際、指定されたスコープおよびフィルタを保持する各ベースを検索します。

注 – SSD では、デフォルト位置は SSD に含められていない限り、サービス (データベース) の検索対象にはなりません。 サービスに複数の SSD が指定されている場合、予期しない結果になる場合があります。

次の例では、Solaris LDAP ネームサービスクライアントが、`passwd` サービスに対して、`ou=west,dc=example,dc=com` で 1 レベルの検索を実行し、次に `ou=east,dc=example,dc=com` で 1 レベルの検索を実行します。ユーザーの `username` に対して `passwd` データを検索する場合、各 `BaseDN` に対してデフォルトの LDAP フィルタ `(&(objectClass=posixAccount)(uid=username))` が使用されます。

```
serviceSearchDescriptor: passwd:ou=west,dc=example,dc=com;ou=east,
dc=example,dc=com
```

次の例では、Solaris LDAP ネームサービスクライアントは、ou=west,dc=example,dc=com 内でpasswd サービスのサブツリー検索を実行します。ユーザー username の passwd データを検索する場合、LDAP フィルタ（&(fulltimeEmployee=TRUE)(uid=username)）を使用して、サブツリー ou=west,dc=example,dc=com の検索がされます

```
serviceSearchDescriptor: passwd:ou=west,dc=example,
dc=com?sub?fulltimeEmployee=TRUE
```

特定のサービスタイプに複数のコンテナを関連付けることも可能です。

たとえば、次のサービス検索記述子は、3 つのコンテナ、ou=myuser,dc=example,dc=com、 ou=newuser,dc=example,dc=com、および ou=extuser,dc=example,dc=com でパスワードエントリを検索することを指定しています。末尾に「,」がある場合には、SSD の相対ベースに defaultSearchBase が付加されることを意味します。

```
defaultSearchBase: dc=example,dc=com
serviceSearchDescriptor: \
passwd:ou=myuser;ou=newuser,ou=extuser,dc=example,dc=com
```

スキーママップ

Solaris LDAP ネームサービスを使用すると、1 つ以上の属性名をいずれかのサービスに再マッピングできます(Solaris LDAP クライアントは、第 18 章に記載されているよく知られている属性を使用します)。属性を対応づける場合、その属性が元の属性と同じ意味および構文を必ず保持するようにしてください。userPassword 属性を対応づけると、問題が発生する場合があります。

スキーママッピングを使用する理由として、次の 2 つが挙げられます。

- 既存のディレクトリサーバー内の属性を対応づけしたい
- 大文字小文字のみが異なるユーザー名を使用する場合、大文字小文字を無視する uid 属性を、大文字小文字を無視しない属性に対応づける必要があります。

書式は、service:attribute-name=mapped-attribute-name です。

指定されたサービスに対して複数の属性を対応づける場合は、複数の attributeMap 属性を定義できます。

次の例では、uid および homeDirectory 属性を passwd サービスで利用する場合、常に employeeName および home 属性が使用されます。

```
attributeMap: passwd:uid=employeeName
attributeMap: passwd:homeDirectory=home
```

passwd サービスの gecos 属性を複数の属性に対応づける場合、特殊なケースが 1 つ存在します。次に例を示します。

```
attributemap: gecos=cn sn title
```

上の例では、gecos 値が、空白で区切られた cn、sn、および title 属性値のリストに対応づけられます。

objectClass マップ

Solaris LDAP ネームサービスを使用すると、オブジェクトクラスを任意のサービス用に対応づけしなおすことができます。特定のサービス用に複数のオブジェクトクラスを対応づける場合、複数の objectclassMap 属性を定義できます。次の例では、posixAccount オブジェクトクラスを使用する場合、常に myUnixAccount オブジェクトクラスが使用されます。

```
objectclassMap: passwd:posixAccount=myUnixAccount
```

クライアントプロファイル

Solaris クライアントのセットアップを容易にし、各クライアントに同じ情報を再入力する手間を省くために、ディレクトリサーバー上に単一のクライアントプロファイルを作成します。この単一のプロファイルに、使用するすべてのクライアントの構成を定義します。プロファイル属性への以降の変更はすべて、定義されたリフレッシュ頻度で送信されます。

これらのクライアントプロファイルは、LDAP サーバー上のよく知られた位置に格納されます。指定されたドメインのルート DN は、 nisDomainObject のオブジェクトクラス、およびクライアントのドメインを含む nisDomain 属性を保持する必要があります。すべてのプロファイルは、このコンテナと相対的な関係にある ou=profile コンテナ内に配置されます。これらのプロファイルは、匿名で読み取り可能にする必要があります。

クライアントのプロファイル属性

次のリストに、Solaris LDAP クライアントのプロファイル属性を示します。これらのプロファイル属性は、idsconfig の実行時に自動的に設定できます。クライアントプロファイルを手動で設定する方法については、252 ページの「クライアントを手動で初期設定する」 を参照してください。

表 **13–2** クライアントのプロファイル属性

属性	説明
cn	プロファイル名。デフォルト値なし。必須

表 13–2 クライアントのプロファイル属性　(続き)

属性	説明
preferredServerList	優先使用されるサーバーのホストアドレスの、空白で区切られたリスト。(ホスト名は使用しない)。defaultServerList 内のサーバーより前に、接続が成功するまで、このリスト内のサーバーへの接続が順番に試みられる。デフォルト値なし。preferredServerList または defaultServerList に 1 つ以上のサーバーを指定する必要がある。
defaultServerList	デフォルトサーバーのホストアドレスの、空白で区切られたリスト(ホスト名は使用しない)。preferredServerlist 内のサーバーへの接続試行後に、接続が確立されるまで、クライアントのサブネット上のデフォルトサーバーへの接続、続いて残りのデフォルトサーバーへの接続が試みられる。preferredServerList または defaultServerList に 1 つ以上のサーバーを指定する必要がある。このリスト内のサーバーへの接続は、優先サーバーリストのサーバーへの接続試行後に試みられる。デフォルト値なし
defaultSearchBase	よく知られたコンテナの検索に使用する相対識別名。デフォルト値なし。ただしこの値は、serviceSearchDescriptor 属性で指定されたサービスで置き換えることが可能
defaultSearchScope	クライアントによるデータベース検索の適用範囲を定義する。この値は、serviceSearchDescriptor 属性で置き換えることが可能。指定可能な値は one または sub。デフォルト値は 1 レベルの検索 (値は one)
authenticationMethod	クライアントが使用する認証方式を示す。デフォルト値は none (匿名)。詳細については、223 ページの「認証方式の選択」を参照
credentialLevel	クライアントが認証に使用する証明書タイプを示す。匿名またはプロキシを選択可能。デフォルトは匿名
serviceSearchDescriptor	クライアントがネームデータベースを検索する方法および場所を定義する (例、クライアントが DIT 内の 1 つ以上の場所を検索する)。デフォルトでは、SSD は定義されていない

表 **13–2** クライアントのプロファイル属性　*(続き)*

属性	説明
serviceAuthenticationMethod	クライアントが特定のサービスで使用する認証メソッド。デフォルトでは、サービス認証メソッドは定義されていない。サービスで serviceAuthenticationMethod が定義されていない場合、authenticationMethod の値がデフォルトになる
attributeMap	クライアントが使用する属性マッピング。デフォルトでは、attributeMap は定義されていない
objectclassMap	クライアントが使用するオブジェクトクラスマッピング。デフォルトでは、objectclassMap は定義されていない
searchTimeLimit	クライアントが許可する、タイムアウトまでの最長検索時間 (秒)。この値は、LDAP サーバーが許可する、検索完了までの時間に影響を及ぼさないデフォルト値は 30 秒
bindTimeLimit	クライアントがサーバーとのバインドに許可する最長時間 (秒)。デフォルト値は 30 秒
followReferrals	クライアントが LDAP 参照に準拠するかどうかを指定する。指定可能な値は TRUE または FALSE。デフォルト値は TRUE
profileTTL	ldap_cachemgr(1M) により実行される、LDAP サーバーからのクライアントプロファイルの更新間隔。デフォルト値は 43200 秒 (12 時間)。値が 0 の場合、プロファイルは更新されない

ローカルのクライアント属性

次の表に、ldapclient(1M) を使用してローカルに設定可能なクライアント属性を示します。

表 **13–3** ローカルのクライアント属性

属性	説明
domainName	クライアントのドメイン名 (クライアントマシンのデフォルトドメインになる) を指定します。デフォルト値なし。必須

表 13–3 ローカルのクライアント属性　(続き)

属性	説明
proxyDN	プロキシの識別名。プロキシの credentialLevel を使用してクライアントマシンを構成する場合、 proxyDN を指定する必要がある
proxyPassword	プロキシのパスワード。プロキシの credentialLevel を使用してクライアントマシンを構成する場合、proxyPassword を定義する必要がある
certificatePath	証明書データベースを含む、ローカルファイルシステム上のディレクトリ。TLS を使用し、authenticationMethod または serviceAuthenticationMethod を指定してクライアントマシンを構成する場合、この属性が使用される。 デフォルト値は /var/ldap

注 – SSD 内の BaseDN に末尾のコンマが含まれる場合、 defaultSearchBase の相対値として処理される。検索実行前に、defaultSearchBase の値が BaseDN に付加される。

ldap_cachemgr デーモン

ldap_cachemgr(1M) は、LDAP クライアントマシン上で稼働するデーモンです。このデーモンは、次の主要機能を実行します。

- root として稼働し、構成ファイルへのアクセスを取得する
- サーバー上のプロファイルに格納されたクライアント構成情報を更新して、クライアントからこのデータを引き出す
- 使用可能な LDAP サーバーのソート済みリストを管理
- さまざまなクライアントから送信される一般的な検索要求をキャッシュして、検索効率を向上させる
- ホスト検索の効率を向上させる

注 – LDAP ネームサービスを機能させるには、ldap_cachemgr が常に実行されている必要があります。

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

LDAP ネームサービスのセキュリティモデル

はじめに

Solaris LDAP ネームサービスは、LDAP リポジトリをネームサービスと認証サービスの両方のソースとして使用します。この節では、クライアント認証、認証方式、pam_ldap と pam_unix モジュール、およびパスワード管理の概念について説明します。

LDAP リポジトリに格納された情報にアクセスする場合、クライアントは最初にディレクトリサーバーで識別情報を確立できます。この識別情報は、匿名にも、LDAP サーバーの認識するオブジェクトにもできます。クライアントの識別情報およびサーバーのアクセス制御情報 (ACI) に基づいて、LDAP サーバーはクライアントに対してディレクトリ情報の読み取りまたは書き込みを許可します。ACI の詳細については、『iPlanet Directory Server 5.1 管理者ガイド』を参照してください。

特定の要求に関して匿名以外で接続している場合、クライアントは、クライアントとサーバーの両方がサポートする認証方式でサーバーに識別情報を証明する必要があります。クライアントは識別情報を確立後に、さまざまな LDAP 要求を実行できます。

ネームサービスと認証サービス (pam_ldap) がディレクトリへの認証を行う方法には違いがあることに留意してください。ネームサービスは、事前定義された識別情報に基づくディレクトリから、さまざまなエントリおよびその属性を読み取ります。認証サービス (pam_ldap) は、ユーザーの名前とパスワードを使用して LDAP サーバーへの認証を行い、ユーザーが適正なパスワードを入力したかどうかを確認します。

Transport Layer Security (TLS)

TLS を LDAP クライアントとディレクトリサーバー間の通信のセキュリティ保護に使用すると、機密性とデータの完全性を保証することができます。TLS プロトコルは、Secure Sockets Layer (SSL) のスーパーセットです。Solaris LDAP ネームサービスは、TLS 接続をサポートします。SSL を使用すると、ディレクトリサーバーおよびクライアントに負荷がかかることに留意してください。

SSL 対応のディレクトリサーバーを設定する必要があります。SSL 対応の iPlanet Directory Server 5.1 の設定方法の詳細については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。SSL 対応の LDAP クライアントも設定する必要があります。

注 – Solaris LDAP ネームサービス用の TLS を使用する場合、ディレクトリサーバーは、LDAP および SSL 用にデフォルトポート 389 および 636 をそれぞれ使用する必要があります。ディレクトリサーバーがこれらのポートを使用しない場合、TLS を使用することはできません。

詳細については、253 ページの「TLS セキュリティの設定」を参照してください。

クライアント資格レベルの割り当て

LDAP ネームサービスクライアントは、資格レベルに従って LDAP サーバーを認証します。LDAP クライアントには、次の 3 つの資格レベルの 1 つを割り当てて、ディレクトリサーバーへの認証を行うことができます。

- `anonymous`
- `proxy`
- `proxy anonymous`

Anonymous

匿名でのアクセスを利用する場合、すべてのユーザーが使用可能なデータだけにアクセスできます。また、セキュリティの問題も考慮する必要があります。ディレクトリの特定部分に匿名アクセスを許可する場合、そのディレクトリへのアクセス権を保持するユーザーはすべて、その操作を実行できることになります。資格レベルとして anonymous を使用する場合、すべての LDAP ネームエントリおよび属性に読み取りアクセスを許可する必要があります。

注意 – 匿名でのディレクトリへの書き込みは、決して許可してはなりません。この書き込みを許可すると、どのユーザーでも書き込みアクセス権のある DIT 内の情報 (別のユーザーのパスワードや自分自身の識別情報など) を変更することが可能になります。

注 – iPlanet Directory Server 5.1 を使用すると、IP アドレス、DNS 名、認証方式、および時刻に基づいてアクセスを制限できます。さらに指定を加えて、アクセスを制限することもできます。詳細については、『*iPlanet Directory Server 5.1* 管理者ガイド』のアクセス権の管理に関する章を参照してください。

プロキシ

クライアントは、プロキシアカウントを使用して、ディレクトリへの認証またはバインドを行います。このプロキシアカウントには、ディレクトリへのバインドを許可されるエントリを設定できます。このプロキシアカウントは、LDAP サーバー上でネームサービス機能を実行するのに十分のアクセス権を必要とします。プロキシ資格レベルを使用して、すべてのクライアントで `proxyDN` および `proxyPassword` を構成する必要があります。暗号化された `proxyPassword` はローカルのクライアントに格納されます。別のクライアントグループに対しては別のプロキシを設定できます。たとえば全営業クライアント用のプロキシを構成する場合、企業全体からアクセス可能なディレクトリと営業ディレクトリの両方へのアクセスを許可しつつ、給与情報を保持する人事ディレクトリへのアクセスを許可しない、という方法が可能です。最も極端な例として、各クライアントに別個のプロキシを割り当てることや、すべてのクライアントに同じプロキシを割り当てることも可能です。一般的な LDAP 配備はこの両極端の中間に位置します。選択は慎重に行なってください。プロキシエージェントが不足していると、リソースへのユーザーアクセスを制御する能力が制限されます。ただし、プロキシが多過ぎる場合、システムの設定および保守が困難になります。適切な権限をプロキシユーザーに付与する必要がありますが、その程度は環境によって大きく異なります。使用する構成に最適の認証方式を決定するための情報については、222 ページの「資格の保存」を参照してください。

プロキシユーザーのパスワードを変更した場合、そのプロキシユーザーを使用するすべてのクライアントで情報を更新する必要があります。LDAP アカウントのパスワード有効期間を設定する場合、プロキシユーザーに関してはこの設定を解除してください。

注 – プロキシ資格レベルは、指定されたマシンのすべてのユーザーおよびプロセスに適用されます。2 人のユーザーが異なるネーミングポリシーを使用する場合は、別個のマシンを使用する必要があります。

また、クライアントが認証にプロキシ資格を使用する場合、`proxyDN` はすべてのサーバーで同じ `proxyPassword` を保持する必要があります。

匿名プロキシ

匿名プロキシは複数値のエントリで、複数の資格レベルが内部に定義されています。匿名プロキシレベルを割り当てられたクライアントは、最初にそのプロキシ識別情報を使用して認証を試みます。ユーザーのロックアウト、パスワードの有効期限切れなどの何らかの理由でクライアントがプロキシユーザーとしての認証ができなかった場合、クライアントは匿名アクセスを使用します。この場合、ディレクトリの構成に応じて、別のサービスレベルに移行する可能性があります。

資格の保存

プロキシ識別情報を使用するようクライアントを設定する場合、クライアントは `proxyDN` および `proxyPassword` を `/var/ldap/ldap_client_cred` 内に保存します。セキュリティ保護のため、このファイルへのアクセスは root のみに許可され、`proxyPassword` の値は暗号化されます。過去の LDAP 実装ではプロキシ資格はクラ

イアントのプロファイル内に格納されましたが、Solaris 9 LDAP ネーミングサービスではこれは行われません。初期化時に `ldapclient` を使用して設定されたプロキシ資格は、すべてローカルに保存されます。このため、プロキシの DN およびパスワード情報に関するセキュリティが向上します。クライアントプロファイルの設定方法の詳細については、第 16 章を参照してください。

認証方式の選択

プロキシ資格または匿名プロキシ資格を割り当てる場合、プロキシによるディレクトリサーバーへの認証方式も選択する必要があります。デフォルトの認証方式は `none` (匿名によるアクセス) です。認証方式には、関連するトランスポートセキュリティオプションも含まれます。

認証方式には資格レベルと同様、複数値を指定できます。たとえば、クライアントプロファイルを設定することにより、クライアントが TLS でセキュリティ保護された `simple` メソッドを最初に使用してバインドを試みるようにできます。これが成功しない場合、クライアントは `sasl/digest-MD5` メソッドを使用してバインドを試みます。 その後、`authenticationMethod` は `tls:simple;sasl/digest-MD5` になります。

LDAP ネームサービスは、いくつかの Simple Authentication and Security Layer (SASL) 機構をサポートします。これらの機構を使用すると、TLS なしでセキュリティ保護されたパスワードを交換できます。ただし、これらの機構はデータの完全性や機密性を保証するものではありません。SASL の詳細については、RFC 2222 を参照してください。

次の認証機構がサポートされています。

- `none`

  クライアントは、ディレクトリへの認証を行いません。これは、`anonymous` 資格レベルと等価です。

- `simple`

  認証方式 `simple` を使用する場合、クライアントマシンはユーザーのパスワードを平文で送信してサーバーへのバインドを実行します。このため、パスワードが漏洩しやすくなります。この認証方式を使用する主な利点は、すべてのディレクトリサーバーがこの方式をサポートしていること、および設定が容易であるという点です。

- `sasl/digest-MD5`

  認証時にクライアントのパスワードは保護されますが、セッションは暗号化されません。iPlanet Directory Server 5.1 を含むいくつかのディレクトリサーバーは、`sasl/digest-MD5` 認証方式もサポートします。`digest-MD5` の主な利点は、認証時にパスワードが平文のままネットワーク上を流れないため、`simple` よりも安全であるという点です。`digest-MD5` の詳細については、RFC 2831 を参照してください。`digest-MD5` は、`cram-MD5` のセキュリティが改善されたものと見なされます。

sasl/digest-MD5 を使用する場合、認証はセキュリティ保護されますがセッションは保護されません。

- sasl/cram-MD5

  この場合、LDAP セッションは暗号化されませんが、 sasl/cram-MD5 を使用して認証が行われるため、認証時にクライアントのパスワードが保護されます。

  cram-MD5 認証方式の詳細については、RFC 2195 を参照してください。すべてのディレクトリサーバーがこの認証方式をサポートしているわけではありません。たとえば、iPlanet Directory Server 5.1 は cram-MD5 をサポートしません。

- tls:simple

  クライアントは、simple を使用してバインドを行い、セッションは暗号化されます。パスワードは保護されます。

- tls: sasl/cram-MD5

  sasl/cram-MD5 を使用して、LDAP セッションの暗号化およびクライアントによるディレクトリサーバーへの認証が行われます。

- tls:sasl/digest-MD5

  sasl/digest-MD5 を使用して、LDAP セッションの暗号化およびクライアントによるディレクトリサーバーへの認証が行われます。

注意 – iPlanet Directory Server 5.1 で digest-MD5 を使用する場合、パスワードを平文で格納する必要があります。認証方式を sasl/digest-MD5 または tls:sasl/digest-MD5 に設定する場合、プロキシユーザーのパスワードを平文で格納する必要があります。平文で格納する場合には、userPassword 属性が適切な ACI を保持するようにして、読み取り不可にする必要があります。

認証とサービス

認証メソッドを特定のサービスに対して serviceAuthenticationMethod 属性に指定できます。現在この機能をサポートしているサービスを次に示します。

- passwd-cmd

  このサービスは、passwd(1) により、ログインパスワードおよびパスワード属性の変更に使用されます。

- keyserv

  このサービスは、chkey(1) および newkey(1M) ユーティリティにより、ユーザーの Diffie-Hellman 鍵ペアの作成および変更に使用されます。

- pam_ldap

  このサービスは、pam_ldap を使用したユーザーの認証に使用されます。

注 – サービスが serviceAuthenticationMethod セットを保持しない場合、authenticationMethod 属性の値がデフォルトになります。

次に示す例は、クライアントプロファイルの 1 セクションです。ここで、ユーザーはディレクトリサーバーへの認証に sasl/digest-MD5 を使用しますが、パスワードの変更には SSL セッションを使用します。

```
serviceAuthenticationMethod=pam_ldap:sasl/digest-MD5
serviceAuthenticationMethod=passwd-cmd:tls:simple
```

プラグイン可能な認証方式

PAM フレームワークを使用することにより、いくつかの認証サービスの中から選択できます。LDAP とともに pam_unix または pam_ldap を使用できます。

より高い柔軟性とより強力な認証方式をサポートしていることから、pam_ldap の使用をお勧めします。

pam_unix

pam.conf(4) ファイルを変更していない場合、デフォルトディレクトリ pam_unix が有効になっています。pam_unix は従来の UNIX 認証モデルに従い、次のように動作します。

1. クライアントは、ネームサービスからユーザーの暗号化されたパスワードを取得します。
2. ユーザーは、パスワードの入力を求められます。
3. ユーザーのパスワードが暗号化されます。
4. クライアントは、暗号化された 2 つのパスワードを比較して、ユーザーを認証するかどうかを決定します。 pam_unix を使用する場合、次の 2 つの制限が存在します。

- パスワードは、平文を含む他の暗号化方式ではなく、UNIX crypt 形式で格納する必要がある
- userPassword 属性は、ネームサービスから読み取り可能でなければならない

  たとえば、資格レベルを匿名に設定する場合、すべてのユーザーに対してuserPassword 属性を読み取り可能にする必要があります。同様に、資格レベルを proxy に設定する場合、userPassword 属性の読み取りをプロキシユーザーに許可する必要があります。

注 – pam_unix は、sasl 認証方式 digest-MD5 と互換性があります。これは、iPlanet Directory Server 5.1 では digest-MD5 を使用するためにパスワードを平文で格納する必要があるのに対し、pam_unix ではパスワードを crypt 形式で格納する必要があるためです。

pam_ldap

pam_ldap を使用する場合、ユーザーは LDAP サーバーにバインドします。認証方式は、pam_ldap の serviceAuthenticationMethod パラメタで定義されます (このパラメタが存在する場合)。それ以外の場合、authenticationMethod がデフォルトで使用されます。

pam_ldap を使用して、ユーザーの識別情報および指定されたパスワードをサーバーにバインドする場合、ユーザーが認証されます。

pam_ldap は、userPassword 属性を読み取りません。このため、pam_unix を使用する他のクライアントが存在しない限り、userPassword 属性の読み取りアクセス権を付与する必要はありません。pam_ldap は、認証方式 none をサポートしません。このため、クライアントが pam_ldap を使用できるように、serviceAuthenticationMethod または authenticationMethod attributes を定義する必要があります。

注意 – 認証方式 simple を使用する場合、第三者がネットワーク上で userPassword 属性を読み取ることができます。

詳細については、266 ページの「pam_ldap に対応した pam.conf ファイルの例 」を参照してください。

PAM およびパスワードの変更

パスワードの変更には、passwd(1) を使用します。パスワードを変更するには、userPassword 属性をユーザーから書き込み可能にする必要があります。passwd-cmd 用の serviceAuthenticationMethod が、この操作の authenticationMethod を無効にすることに留意してください。使用する認証によっては、現行のパスワードの暗号化解除がネットワーク上で行われる場合があります。

pam_unix の場合、UNIX crypt を使用して新規 userPassword 属性が暗号化されタグ付けされてから、LDAP への書き込みが行われます。このため、新規パスワードは、サーバーへのバインドに使用される認証方式に関係なく、ネットワーク上で暗号化されます。

pam_ldap の場合、パスワードの変更時に新規パスワードの暗号化解除が行われます。このため、機密性を保つために TLS を使用する必要があります。TLSを使用しない場合、userPassword が漏洩する危険性があります。

iPlanet Directory Server 5.1 で、pam_ldap を使用してパスワードを設定する場合、パスワードは serverStrorageScheme (タグ付けされていない状態) を使用して暗号化されます。passwordStorageScheme 属性の詳細については、『*iPlanet Directory Server 5.1* 管理者ガイド』のユーザーアカウントの管理に関する章を参照してください。

注 – passwordStorageScheme 属性を設定する際、以下の点を考慮する必要があります。pam_unix を使用する NIS、NIS+、または他のクライアントがリポジトリとして LDAP を使用する場合、 passwordStorageScheme に対して crypt を実行する必要があります。また、iPlanet Directory Server 5.1 で sasl/digest-MD5 に対して pam_ldap を使用する場合、passwrodStorageScheme を平文に設定する必要があります。

パスワード管理

Solaris LDAP ネームサービスは現在、iPlanet Directory Server 5.1 のパスワード管理機能をサポートしません。

第 14 章

LDAP ネームサービスの計画

この章では、サーバーとクライアントの設定およびインストール処理を開始する前に実行する必要のある高度な計画について説明します。

この章の内容は次のとおりです。

概要

LDAP クライアントプロファイルは、LDAP クライアントが使用する構成情報の集合体です。LDAP クライアントは、このプロファイルを使用して、サポートする LDAP サーバー上の LDAP ネームサービス情報にアクセスして、LDAP ネームサービスを提供しますこの章では、LDAP 構成の主要部分を使用して、LDAP ネームサービスのさまざまな分野の計画方法を説明します。その中には、ネットワークモデル、ディレクトリ情報ツリー、セキュリティモデル、さまざまなプロファイル属性のデフォルト値、およびデータ生成の準備が含まれます。

ネットワークモデルの計画

可用性およびパフォーマンスを考慮すると、企業規模のネットワークの各サブネットが LDAP サーバーを独自に保持して、サブネット内のすべての LDAP クライアントにサービスを提供する方法が最善です。これらのサーバーの 1 つだけをマスター LDAP サーバーにする必要があります。残りはすべてマスターサーバーの複製にできます。

ネットワーク構成を計画する前に、使用可能なサーバーの数、クライアントがサーバーにアクセスする方法、複数のサーバーへのアクセス順序について考慮する必要があります。サブネットごとに 1 つのサーバーが存在する場合、`defaultServerList` 属性を使用してすべてのサーバーのリストを作成し、LDAP クライアントからアクセス順序をソートおよび操作できます。速度またはデータ管理上の理由から、特定の順序でサーバーにアクセスする必要がある場合は、`preferredServerList` 属性を使用してサーバーへの固定されたアクセス順序を定義します。マスターサーバーをこれらのリストに配置しないことで、マスターサーバーへの負荷を軽減できます。

さらに、サーバーおよびネットワーク構成を計画する際に考慮するに値する 3 つの属性があります。`bindTimeLimit` 属性は TCP 接続要求のタイムアウト値の設定に、`searchTimeLimit` 属性は LDAP 検索操作のタイムアウト値の設定に、`profileTTL` 属性は LDAP クライアントによるサーバーからのプロファイルのダウンロード頻度の制御に、それぞれ使用できます。速度が遅いか不安定なネットワークの場合、 `bindTimeLimit` および `searchTimeLimit` 属性にデフォルト値より大きい値を設定することが必要な場合があります。配備の初期テスト段階で、`profileTTL` 属性値を引き下げて、頻繁に行われる LDAP サーバー内のプロファイルの変更をクライアントが取得するようにしてもよいでしょう。

ディレクトリ情報ツリー (DIT) の計画

前の章で説明したように、LDAP ネームサービスはデフォルトのディレクトリ情報ツリー (DIT) および関連するデフォルトスキーマを保持します。たとえば、`ou=people` コンテナには、ユーザーアカウント、パスワード、およびシャドウ情報が含まれます。`ou=hosts` コンテナには、ネットワーク内のシステムに関する情報が含まれます。`ou=people` コンテナ内の各エントリは、 `objectclass posixAccount` および `shadowAccount` のエントリになります。デフォルト DIT は、巧みに設計されたディレクトリ構造であり、オープンな標準に基づいています。これは、ほとんどのネームサービスのニーズに応えるもので、変更せずに使用することが推奨されています。デフォルト DIT の使用を選択する場合、決定する必要があるのは、ディレクトリツリー内のどのノード (起点識別名) から、指定されたドメインのネームサービス情報を検索するかという点だけです。このノードは、`defaultSearchBase` 属性を使用して指定

されます。さらに、defaultSearchScope 属性を設定して、ネームサービスが実行する検索範囲をクライアントに指定することもできます。検索範囲には、識別名 (DN) 内の 1 レベルだけを検索するか (one)、DN内のサブツリー全体を選択するか (sub) を指定できます。

ただし、既存の DIT を利用する場合でも、ディレクトリツリー内に散在するネームサービスデータを使用してより複雑な DIT を処理する場合でも、LDAP ネームサービスにより高度な柔軟性が求められる場合があります。たとえば、ユーザーアカウントエントリがツリーの別の場所に存在する場合があります。クライアントプロファイル内で serviceSearchDescriptor、attributeMap、および objectclassMap 属性を設定して、これらの状況に対応します。

サービス検索記述子を使用して、特定のサービスのデフォルト検索ベース、検索範囲、および検索フィルタを置き換えることができます。詳細については、214 ページの「サービス検索記述子 (SSD) とスキーママッピング」を参照してください。

AttributeMap および ObjectclassMap 属性を使用して、スキーママッピングを行うことができます。これらの属性を使用すると、既存の DIT で LDAP ネームサービスを動作させることができます。posixAccount オブジェクトクラスの既存のオブジェクトクラス (myAccount など) への対応づけ、および posixAccount オブジェクトクラス内の属性の myAccount オブジェクトクラス内の属性への対応づけを実行できます。

複数のディレクトリサーバー

複数の LDAP サーバーで 1 つの DIT を構成することも可能です。たとえば、DIT のいくつかのサブツリーを、他の LDAP サーバー上に配置できます。この場合、LDAP サーバーは、既知ではあるが自身のデータベース内に存在しないネームデータを求める LDAP クライアントを、別のサーバーに委ねることができます。この種の DIT 構成を計画する場合、LDAP ネームサービスがサーバー参照に従ってネームサービス検索を続行することを示すように、クライアントのプロファイル属性 followReferrals を設定する必要があります。ただし可能であれば、指定されたドメインのネームデータすべてを単独のディレクトリサーバー上に配置するのが最善です。

クライアントが通常は読み取り専用の複製にアクセスし、必要な場合にのみ読み取り/書き込み可能なマスターサーバーへの参照を利用する場合、参照が役に立ちます。この方法では、要求が複製により処理されるため、マスターサーバーに過度の負荷がかかることはありません。

他のアプリケーションとのデータ共有

LDAP を最大限に活用するには、論理エントリごとに 1 つの LDAP エントリが存在する必要があります。たとえば、ユーザーは、企業白書に関する情報だけでなく、Solaris アカウント情報やアプリケーション固有のデータも保持できます。`posixAccount` および `shadowAccount` は補助オブジェクトクラスであるため、これらのデータをディレクトリ内のエントリに追加できます。このため、注意深い計画、設定、および管理が必要になります。

ディレクトリ接尾辞の選択

適切なディレクトリ接尾辞の選択方法については、第 11 章を参照してください。

複製サーバー

複製サーバーを設定する場合、次の 3 つの方法が存在します。

- 単一マスター複製 (Single-master replication)
- 浮動マスター複製 (Floating-master replication)
- 複数マスター複製 (Multi-master replication)

「単一マスター」

単一マスター複製では、指定されたパーティションまたはパーティション化されていないネットワークに対して、1 つのマスターサーバーだけが、ディレクトリエントリの書き込み可能なコピーを保持します。複製サーバーは、ディレクトリエントリの読み込み専用コピーを保持します。複製とマスターの両方が検索、比較、およびバインド操作を実行できますが、書き込み操作を実行できるのはマスターサーバだけです。

単一マスター複製の不利な点は、マスターサーバーでシングルポイント障害が発生した場合です。マスターサーバーがダウンした場合、どの複製サーバーからも書き込み操作を実行できません。

「浮動マスター」

浮動マスターは、指定されたパーティション化されたネットワークまたはパーティション化されていないネットワークに対し、書き込み権限を保持するマスターサーバーは常に 1 つだけである点で、単一マスターを使用する場合と似ています。ただし浮動マスターを使用すると、マスターサーバーがダウンした場合、アルゴリズムにより複製が自動的にマスターサーバーに変化します。

浮動マスター複製の不利な点は、ネットワークがパーティション化され、どちらの側のパーティション上の複製もマスターになった場合、ネットワークを再結合する際、新規マスター間の調整が非常に複雑になり得ることです。

「複数マスター」

複数マスター複製では、ディレクトリエントリデータの独自の読み取り/書き込み複製を保持する、複数のマスターサーバーが存在します。複数マスターを使用すると、シングルポイント障害を防ぐことができますが、サーバー間で更新による競合が発生する可能性があります。つまり、2 つのマスター上でエントリの属性が同時に変更される場合、競合による障害の解決ポリシー (最後の書き込みを優先するなど) の適用が必要になります。

複製サーバーの設定方法については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

セキュリティモデルの計画

セキュリティモデルを計画する場合、最初に、LDAP クライアントが LDAP サーバーとの通信に使用する識別情報を考慮する必要があります。たとえば、強力な認証を使用してネットワーク上を流れるユーザーパスワードを保護するかどうか、また LDAP クライアントと LDAP サーバー間のセッションを暗号化して送信される LDAP データを保護する必要があるかなどを決定する必要があります。

セキュリティモデルの計画には、プロファイル内の `credentialLevel` および `authenticationMethod` 属性が使用されます。`credentialLevel` には、匿名、プロキシ、および匿名プロキシの 3 つの資格レベルのうちの 1 つを指定できます。LDAP ネームサービスのセキュリティ概念については、220 ページの「LDAP ネームサービスのセキュリティモデル」 を参照してください。

セキュリティモデルを計画する際の主要な決定事項を次に示します。

- LDAP クライアントは、どの資格レベルおよび認証方式を使用するか
- TLS を使用するか
- NIS や NIS+ との下位互換性を必要とするか。言い換えれば、クライアントは `pam_unix` または `pam_ldap` を使用するか
- サーバーの `passwordStorageScheme` 属性をどのように設定するか
- アクセス制御情報をどのように設定するかACI の詳細については、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

クライアントプロファイルおよびデフォルト属性値の計画

前述の計画手順 (ネットワークモデル、DIT、およびセキュリティモデル) を理解することにより、次のプロファイル属性の値についてアイデアを得ることができるでしょう。

- cn
- defaultServerList
- preferredServerList
- bindTimeLimit
- searchTimeLimit
- profileTTL
- defaultSearchBase
- defaultSearchScope
- serviceSearchDescriptor
- attributeMap
- objectclassMap
- followReferrals
- credentialLevel
- authenticationMethod
- serviceCredentialLevel
- serviceAuthenticationMethod

上記の属性の中で、必須属性は cn、 defaultServerList、および defaultSearchBase だけです。これらの属性には、デフォルト値は存在しません。残りの属性はオプションであり、デフォルト値がないオプションも存在します。

LDAP クライアントの設定の詳細については、第 16 章 を参照してください。

データ生成の計画

LDAP ネームサービスデータを使用して、LDAP サーバーを生成する場合、適切な DIT およびスキーマを使用して LDAP サーバーを構成した後、新規 ldapaddent ツールを使用するのが最善です。このツールは、対応する /etc ファイルから LDAP コンテナ内にエントリを作成します。このツールを使用して、次のデータタイプ用のコンテナ内にデータを生成することができます。 aliases、auto_*、bootparams、ethers、group、hosts (IPv6 アドレスを含む)、netgroup、netmasks、networks、passwd、shadow、protocols、publickey、rpc、および services。

デフォルトでは、ldapaddent は標準入力からこのデータを読み取って、コマンド行で指定されたデータベースに関連付けられた LDAP コンテナに追加します。ただし、データを読み取る入力ファイルは、-f オプションを使用して指定できます。

エントリはクライアントの構成に基づき、ディレクトリ内に格納されます。このため、LDAP ネームサービスを使用するようにクライアントを構成する必要があります。

パフォーマンスを向上させるために推奨されているデータベースのロード順序を次に示します。

1. passwd データベースの次に shadow データベース
2. networks データベースの次に netmasks データベース
3. bootparams データベースの次に ethers データベース

オートマウントエントリを追加する場合、データベース名は auto_* (たとえば auto_home) の形式で指定します。

別のホストの /etc ファイルを LDAP サーバーに追加する場合、それらすべてを 1 つの /etc ファイルにマージして、1 台のホスト上で ldapaddent を使用して追加できます。あるいは、各ホストが LDAP クライアントとして構成済みであることを想定して各ホストで ldapaddent を実行することもできます。

使用するネームサービスデータが NIS サーバー上にすでに存在し、データを LDAP ネームサービス用の LDAP サーバーに移動する場合、 ypcat (または niscat) コマンドを使用して NIS マップをファイル内にダンプします。続いて、これらのファイルに対して ldapaddent を実行してデータを LDAP サーバーに追加します。

注 – ldapaddent は LDAP クライアント上でしか実行できません。

以下の作業は、テーブルが yp クライアントから抽出されることを想定しています。

▼ ldapaddent を使用して host エントリを持つサーバーを生成する方法

1. idsconfig を使用し、**iPlanet Directory Server 5.1** が設定されているか確認します。
2. クライアントマシン上でスーパーユーザーになります。
3. そのマシンを **LDAP** クライアントに設定します。

   ```
   # ldapclient init –a profileName=new –a \
   domainName=west.example.com 192.168.0.0
   ```

4. データを指定してサーバーを生成します。

   ```
   # ldapaddent –h 192.168.0.0 –D directory_manager –f /etc/hosts
   hosts
   ```

この例では、現在 ldapaddent が simple 認証方式を使用してバインドしているように、平文で directory_manager パスワードが送られます。

rpc.nisd 内の適切な設定を使用して、NIS+ データを保持するディレクトリサーバーを生成できます。第 19 章 を参照してください。

第 15 章

iPlanet Directory Server 5.1 の設定 (手順)

この章では、iPlanet Directory Server 5.1 を構成して、Solaris LDAP ネームサービスクライアントのネットワークをサポートする方法について説明します。この章の内容は、iPlanet Directory Server 5.1 に固有の情報です。

注 – Solaris LDAP クライアントで動作するよう iPlanet Directory Server 5.1 を構成する前に、第 11 章に記載された手順をすべて実行する必要があります。

注 – ディレクトリサーバー (LDAP サーバー) を、サーバー自身のクライアントにすることはできません。

この章の内容は次のとおりです。

idsconfig を使用した iPlanet Directory Server 5.1 の構成

サーバーのインストール用チェックリストの作成

サーバーのインストール処理時に定義する重要な変数を使用して、以下に示すようなチェックリストを作成してから、idsconfig を起動してください。263 ページの「未記入のチェックリスト」 で提供されるチェックリストを使用できます。

注 – 以下の情報は、以降の LDAP 関連の章に示されるすべての例の基礎になります。サンプルのドメインは、全国規模で店舗を展開する部品会社である Example, Inc. のものです。例の中では、その West Coast Division (ドメインは west.example.com) を対象に説明します。

表 **15–1** サーバーで定義する変数

変数	サンプルネットワークの定義
インストールしたディレクトリサーバーインスタンスのポート番号 (デフォルト = 389)	**default**
サーバー名	**ipdserver** (完全指定ドメイン名 ipdserver.west.example.com または 192.168.0.0)
複製サーバー (IP 番号:ポート番号)	**192.168.0.1** [ipdrep.west.example.com 用]
Directory manager [dn: cn=directory manager]	**default**
サービスされるドメイン名	**west.example.com**
クライアント要求の処理がタイムアウトするまでの時間 (秒)	**–1**
各検索要求で返されるエントリの最大数	**–1**

注 – defaultServerList または preferredServerList の定義でホスト名を使用する場合、ホストの検索に LDAP を使用してはなりません。これは、/etc/nsswitch.conf の hosts 行に ldap を含めることはできないことを意味します。

表 15–2 クライアントプロファイルで定義する変数

変数	サンプルネットワークの定義
プロファイル名	**WestUserProfile**
サーバーリスト (デフォルトはローカルサブネット)	west.example.com または **192.168.0.0**
優先されるサーバーリスト (優先順に記載)	**none**
検索範囲 (検索するディレクトリツリーレベルの数、「One」(デフォルト) または「Sub」)	**one** (デフォルト)
サーバーへのアクセスに使用する資格。デフォルトは匿名	**proxy**
参照に従うか (メインサーバーが使用できない場合の別のサーバーへのポインタ)。デフォルトは no	**Y**
検索時にサーバーが情報を返すまでの待機時間の制限 (デフォルトは 30)	**default**
サーバーとの通信時のバインド時間の制限 (デフォルトは 10 秒)デフォルトは秒	**2**
認証方式。デフォルトは none	**simple**

注 – クライアントプロファイルはドメインごとに定義されます。指定されたドメインで、1 つ以上のプロファイルを定義する必要があります。

属性インデックス

idsconfig が作成する次の属性リストにより 、パフォーマンスが向上します。

```
membernisnetgroup                    pres,eq,sub
nisnetgrouptriple                    pres,eq,sub
```

```
memberuid                       pres,eq
uidNumber                       pres,eq
gidNumber                       pres,eq
ipHostNumber                    pres,eq
ipNetworkNumber                 pres,eq
ipProtocolNumber                pres,eq
oncRpcNumber                    pres,eq
```

スキーマ定義

`idsconfig(1M)` により、必要なスキーマ定義が自動的に追加されます。LDAP 管理に精通しているユーザー以外、サーバースキーマを手動で変更してはなりません。LDAP ネームサービスの使用するスキーマ拡張リストについては、第 18 章を参照してください。

インデックス表示の使用

インデックス表示は、iPlanet Directory Server の機能で、仮想リスト表示とも呼ばれます。クライアントは、インデックス表示を使用して、エントリのグループや番号の記載された長いリストを表示して選択を実行できます。このため、各クライアントの検索処理を短縮できます。インデックス表示により最適化かつ事前定義された検索パラメータが提供されるため、Solaris LDAP ネームクライアントは、さまざまなサービスから特定の情報により素早くアクセスできるようになります。インデックス表示を作成しない場合、サーバーが検索時間を制限するために、クライアントが指定されたタイプのエントリすべてを取得できない場合があります。また、エントリの番号を確認できない場合もあります。

インデックスはディレクトリサーバー上に構成されるため、プロキシユーザーはこれらのインデックスに読み取りアクセス権限を保持します。

iPlanet Directory Server 上でのインデックス作成の詳細、およびその使用によるパフォーマンス上のコストについては、『*iPlanet Directory Server 5.1* 管理者ガイド』を参照してください。

次の例では、`-n` オプションでエントリをインデックス化するデータベースの名前を示し、 `-s` オプションでディレクトリサーバーのインスタンスを示します。

注 – idsconfig により、デフォルト VLV インデックスがすべて作成されます。

```
directoryserver -s ipdserver vlvindex -n userRoot -T getgrent
directoryserver -s ipdserver vlvindex -n userRoot -T gethostent
directoryserver -s ipdserver vlvindex -n userRoot -T getnetent
directoryserver -s ipdserver vlvindex -n userRoot -T getpwent
directoryserver -s ipdserver vlvindex -n userRoot -T getrpcent
directoryserver -s ipdserver vlvindex -n userRoot -T getspent
```

サービス検索記述子を使用してさまざまなサービスへのクライアントアクセスを変更する

サービス検索記述子 (SSD) を使用すると、LDAP 内の指定された操作のデフォルト検索要求を、ユーザーが定義した検索に変更できます。たとえば、これまでカスタマイズしたコンテナ定義や別のオペレーティングシステムで LDAP を使用してきたユーザーが、Solaris 9 に移行する場合などに、SSD は特に役に立ちます。SSD を使用すると、既存の LDAP データベースおよびデータを変更せずに、Solaris 9 LDAP ネームサービスを構成できます。

idsconfig を使用して SSD を変更する

前出の Example, Inc. が LDAP を構成済みで、ユーザーを ou=Users コンテナに格納しているものとします。これを Solaris 9 にアップグレードしましょう。定義によって、Solaris 9 LDAP は、ユーザーエントリが ou=People コンテナに格納されているものと見なします。このままでは、LDAP は passwd サービス検索時に DIT の ou=people レベルを検索するため、適切な値を検出できません。

この問題を解決する手のかかる方法の 1 つは、Example, Inc. の既存の DIT を完全に置き換え、Example, Inc. のネットワーク上の既存アプリケーションすべてを書き換えて、新規 LDAP ネームサービスとの互換性を持たせる方法です。別のはるかに望ましい解決策は、SSD を使用して、LDAP に対しユーザー情報をデフォルトの ou=people コンテナ内ではなく ou=Users コンテナ内で検索するよう指示する方法です。

idsconfig を使用して、iPlanet Directory Server 5.1 の構成時に必要な SSD を定義できます。プロンプト行は次のようになります。

```
Do you wish to setup Service Search Descriptors (y/n/h? y
  A  Add a Service Search Descriptor
  D  Delete a SSD
```

```
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's

  Q  Exit menu
Enter menu choice: [Quit] a
Enter the service id: passwd
Enter the base: service ou=user,dc=west,dc=example,dc=com
Enter the scope: one[default]
  A  Add a Service Search Descriptor
  D  Delete a SSD
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's

  Q  Exit menu
Enter menu choice: [Quit] p

Current Service Search Descriptors:
==================================
Passwd:ou=Users,ou=west,ou=example,ou=com?

Hit return to continue.

  A  Add a Service Search Descriptor
  D  Delete a SSD
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's

  Q  Exit menu
Enter menu choice: [Quit] q
```

idsconfig の実行

注 – idsconfig の実行には特別な権限は不要であり、LDAP ネームサービスクライアントになる必要もありません。idsconfig を実行する準備として、238 ページの「サーバーのインストール用チェックリストの作成」 で説明したチェックリストを作成してください。idsconfig を、サーバーまたは LDAP ネームサービスクライアントマシンから 実行する必要はありません。ネットワーク上の任意の Solaris マシンから idsconfig を実行できます。

注意 – `idsconfig` は、Directory Manager のパスワードを平文で送信します。これを防ぐには、 `idsconfig` をクライアント上ではなく ディレクトリサーバー上で実行する必要があります。

▼ idsconfig を使用して iPlanet Directory Server を構成する方法

1. ターゲットの **iPlanet Directory Server 5.1** が起動済みで実行中であることを確認します。
2. `idsconfig` を実行します。

   # **/usr/lib/ldap/idsconfig**
3. 表示される質問に答えます。ユーザー入力のデフォルトは「**no**」です。質問の詳細を表示する場合は次のように入力します。

   `h`
   すると、簡潔なヘルプが表示されます。

   以下の `idsconfig` の実行例を参照する際、この章の冒頭の 238 ページの「サーバーのインストール用チェックリストの作成」 で示したサーバーおよびクライアントのチェックリスト内の定義を使用してください。以下は、デフォルト設定の多くを変更しない単純な設定の例です。クライアントプロファイルを変更する最も複雑な方法は、SSD を作成する方法です。詳細については、241 ページの「サービス検索記述子を使用してさまざまなサービスへのクライアントアクセスを変更する」を参照してください。

   プロンプトの後のキャリッジリターンは、Enter キーを押してデフォルトの設定を受け入れたことを意味します。

   例 **15–1** Example, Inc. ネットワークでの `idsconfig` の実行

   (sysadmin@test) [3:10pm] ns_ldap [31] % **sh idsconfig.sh**

```
It is strongly recommended that you BACKUP the directory server
before running idsconfig.sh.

Hit Ctrl-C at any time before the final confirmation to exit.

Do you wish to continue with server setup (y/n/h)? [n] Y

Enter the iPlanet Directory Server's (iPlanet Directory Server)
hostname to setup: IPDSERVER

Enter the port number for iPlanet Directory Server (h=help): [389]
Enter the directory manager DN: [cn=Directory Manager]
Enter passwd for cn=Directory Manager :
Enter the domainname to be served (h=help): [west.example.com]
```

例 **15–1** Example, Inc. ネットワークでの idsconfig の実行　(続き)

```
Enter LDAP Base DN (h=help): [dc=west,dc=example,dc=com]
Enter the profile name (h=help): [default]
Default server list (h=help): [192.168.0.0]
Preferred server list (h=help):
Choose desired search scope (one, sub, h=help):  [one]
The following are the supported credential levels:
  1  anonymous
  2  proxy
  3  proxy anonymous
Choose Credential level [h=help]: [1] 2

The following are the supported Authentication Methods:
  1  none
  2  simple
  3  sasl/DIGEST-MD5
  4  tls:simple
  5  tls:sasl/DIGEST-MD5
Choose Authentication Method (h=help): [1] 2

Current authenticationMethod: simple


Do you want to add another Authentication Method? N

Do you want the clients to follow referrals (y/n/h)? [n] Y

Do you want to modify the server timelimit value (y/n/h)? [n] Y

Enter the time limit for iPlanet Directory Server (current=3600): [-1]

Do you want to modify the server sizelimit value (y/n/h)? [n] Y

Enter the size limit for iPlanet Directory Server (current=2000): [-1]

Do you want to store passwords in "crypt" format (y/n/h)? [n] Y

Do you want to setup a Service Authentication Methods (y/n/h)? [n]
Client search time limit in seconds (h=help): [30]
Profile Time To Live in seconds (h=help): [43200]

Bind time limit in seconds (h=help): [10] 2

Do you wish to setup Service Search Descriptors (y/n/h)? [n]


              Summary of Configuration


  1  Domain to serve               : west.example.com
  2  Base DN to setup              : dc=west,dc=example,dc=com
  3  Profile name to create        : default
  4  Default Server List           : 192.168.0.0
  5  Preferred Server List         :
  6  Default Search Scope          : one
  7  Credential Level              : proxy
  8  Authentication Method         : simple
  9  Enable Follow Referrals       : TRUE
 10  iPlanet Directory Server Time Limit               : -1
```

例 **15–1** Example, Inc. ネットワークでの idsconfig の実行　(続き)

```
 11  iPlanet Directory Server Size Limit                : -1
 12  Enable crypt password storage : TRUE
 13  Service Auth Method pam_ldap  :
 14  Service Auth Method keyserv   :
 15  Service Auth Method passwd-cmd:
 16  Search Time Limit             : 30
 17  Profile Time to Live          : 43200
 18  Bind Limit                    : 2
 19  Service Search Descriptors Menu

Enter config value to change: (1-19 0=commit changes) [0]
Enter DN for proxy agent:[cn=proxyagent,ou=profile,dc=west,dc=example,dc=com]
Enter passwd for proxyagent:
Re-enter passwd:


WARNING: About to start committing changes. (y=continue, n=EXIT) Y

1. Changed timelimit to -1 in cn=config.
  2. Changed sizelimit to -1 in cn=config.
  3. Changed passwordstoragescheme to "crypt" in cn=config.
  4. Schema attributes have been updated.
  5. Schema objectclass definitions have been added.
  6. Created DN component dc=west.
  7. NisDomainObject added to dc=west,dc=example,dc=com.
  8. Top level "ou" containers complete.
  9. Nis maps: auto_home auto_direct auto_master auto_shared processed.
  10. ACI for dc=west,dc=example,dc=com modified to disable self modify.
  11. Add of VLV Access Control Information (ACI).
  12. Proxy Agent cn=proxyagent,ou=profile,dc=west,dc=example,dc=com added.
  13. Give cn=proxyagent,ou=profile,dc=west,dc=example,dc=com read permission for
password.
  14. Generated client profile and loaded on server.
  15. Processing eq,pres indexes:
      ipHostNumber (eq,pres)   Finished indexing.
      uidNumber (eq,pres)   Finished indexing.
      ipNetworkNumber (eq,pres)   Finished indexing.
      gidnumber (eq,pres)   Finished indexing.
      oncrpcnumber (eq,pres)   Finished indexing.
  16. Processing eq,pres,sub indexes:
      membernisnetgroup (eq,pres,sub)   Finished indexing.
      nisnetgrouptriple (eq,pres,sub)   Finished indexing.
  17. Processing VLV indexes:
      getgrent vlv_index   Entry created
      gethostent vlv_index   Entry created
      getnetent vlv_index   Entry created
      getpwent vlv_index   Entry created
      getrpcent vlv_index   Entry created
      getspent vlv_index   Entry created

idsconfig.sh: Setup of iPlanet Directory Server server ipdserver is complete.
```

例 **15–1** Example, Inc. ネットワークでの `idsconfig` の実行　(続き)

```
Note: idsconfig has created entries for VLV indexes.  Use the
      directoryserver(1m) script on ipdserver to stop
      the server and then enter the following vlvindex
      sub-commands to create the actual VLV indexes:

  directoryserver -s ipdserver vlvindex -n userRoot -T getgrent
  directoryserver -s ipdserver vlvindex -n userRoot -T gethostent
  directoryserver -s ipdserver vlvindex -n userRoot -T getnetent
  directoryserver -s ipdserver vlvindex -n userRoot -T getpwent
  directoryserver -s ipdserver vlvindex -n userRoot -T getrpcent
  directoryserver -s ipdserver vlvindex -n userRoot -T getspent
```

注 – サマリー画面で空欄になっているパラメータは設定されません。

`idsconfig` によるディレクトリの設定が完了したら、サーバー設定を完了してサーバーをクライアント対応にする前に、サーバー上で指定されたコマンドを実行する必要があります。

`ldapaddent` を使用したディレクトリサーバーの生成

注 – `pam_unix` を使用している場合、データを使用してディレクトリサーバーを生成する前に、パスワードを `Unix Crypt` 形式で格納するようにサーバーを構成する必要があります。`pam_ldap` を使用している場合、任意の形式でパスワードを格納できます。UNIX `crypt` 形式でパスワードを設定する方法については、iPlanet Directory Server のドキュメントを参照してください。

注 – `ldapaddent(1M)` は、LDAP ネームサービス用に構成済みのクライアントでのみ実行できます。

`ldapaddent` は、標準入力から (`/etc/filename passwd` など) データを読み取り、このデータをサービスに関連付けられたコンテナに配置します。クライアント構成により、デフォルトのデータ書き込み方法が決定されます。

次に示す例では、 ldapaddent にデータを指定してサーバーを生成します。

例 **15–2** ldapaddent およびユーザーパスワードデータを使用して iPlanet Directory Server 5.1 を生成する方法

1. ldapaddent コマンドを使用して、 /etc/passwd エントリをサーバーに追加します。

   # ldapaddent -D "cn=directory manager" -f /etc/passwd passwd

ldapaddent(1M) のマニュアルページを参照してください。LDAP セキュリティおよび Directory Server への書き込みアクセスの詳細については、第 13 章を参照してください。

プリンタエントリの管理

プリンタの追加

プリンタエントリを LDAP ディレクトリに追加する場合、 printmgr 構成ツールまたは lpset -n ldap コマンド行ユーティリティを使用します。詳細については、lpset(1M) のマニュアルページを参照してください。ディレクトリに追加されるプリンタオブジェクトは、プリンタの印刷システムクライアントが必要とする接続パラメータのみを定義することに留意してください。ローカルのプリントサーバー構成データはファイル内に保持されます。典型的なプリンタエントリは、次のようになります。

```
printer-uri=myprinter,ou=printers,dc=mkg,dc=example,dc=com
objectclass=top
objectclass=printerService
objectclass=printerAbstract
objectclass=sunPrinter
printer-name=myprinter
sun-printer-bsdaddr=printsvr.example.com,myprinter,Solaris
sun-printer-kvp=description=HP LaserJet (PS)
printer-uri=myprinter
```

lpget の使用

lpget(1M) を使用して、LDAP クライアントの LDAP ディレクトリが認識するプリンタエントリすべてをリスト表示できます。LDAP クライアントの LDAP サーバーが複製サーバーの場合、更新複製規約 (update replication agreement) によって、リスト表示されたプリンタはマスター LDAP サーバーのプリンタと同じ場合も同じでない場合もあります。詳細については、lpget(1M) のマニュアルページを参照してください。

たとえば、指定されたベース DN のプリンタすべてを一覧表示するには、次のように入力します。

```
# lpget -n ldap list

myprinter:
    dn=myprinter,ou=printers,dc=mkt,dc=example,dc=com
    bsdaddr=printsvr.example.com,myprinter,Solaris
    description=HP LaserJet (PS)
```

追加プロファイルを使用してサーバーを生成する

ldapclient を genprofile オプションとともに使用すると、指定された属性に基づいて、構成プロファイルの LDIF 表現を作成できます。次に、作成したプロファイルを LDAP サーバーにロードして、クライアントプロファイルとして使用できます。クライアントは、ldapclient init を使用してこのファイルをダウンロードできます。

ldapclient genprofile の使用方法の詳細については、ldapclient(1M) のマニュアルページを参照してください。

以下の例では、ldapclient とともに追加プロファイルを指定して、サーバーを生成します。

例 **15–3** 追加プロファイルを使用してサーバーを生成する方法

1. スーパーユーザーになります。
2. ldapclient コマンドを genprofile コマンドとともに使用します。

   ```
   # ldapclient genprofile -a profileName=myprofile \
   -a defaultSearchBase=dc=west,dc=example,dc=com \
   -a "defaultServerList=192.168.0.0 192.168.0.1:386" \
   > myprofile.ldif
   ```
3. 新規プロファイルをサーバーにアップロードします。

   ```
   # ldapadd –h 192.168.0.0 –D “cn=directory manager” –f
   myprofile.ldif
   ```

第 16 章

クライアントの設定 (手順)

この章では、Solaris LDAP ネームサービスクライアントの設定方法について説明します。

この章の内容は次のとおりです。

前提条件

Solaris クライアントで LDAP をネームサービスとして使用するためには、次の要件が満たされている必要があります。

- クライアントのドメイン名が LDAP サーバーによって処理される
- `nsswitch.conf` ファイルが、必要なサービスの LDAP を指している。`nsswitch.conf` ファイルについては、第 2 章を参照
- クライアントが、その動作を定義するための特定のパラメータをすべて使って構成されている
- `ldap_cachemgr` がクライアントで実行されている
- クライアントが構成されているサーバーが 1 つ以上起動され、実行されている

ldapclient ユーティリティは、サーバーの起動を除き、上記の手順をすべて実行するので、LDAP クライアントを設定するための鍵となります。この章の後半では、ldapclient ユーティリティを使用して LDAP クライアントを設定する方法や、それ以外の各種 LDAP ユーティリティを使用して LDAP クライアントに関する情報を取得したり LDAP クライアントの状態をチェックしたりする方法について、例を挙げて説明します。

クライアントの初期設定

ldapclient(1M) は、Solaris オペレーティング環境で LDAP クライアントを設定するためのユーティリティです。ldapclient ユーティリティでは、サーバーがすでに適切なクライアントプロファイルで構成されていることを前提としています。サーバーをインストールして、適切なプロファイルで構成してからクライアントを設定する必要があります。

ldapclient を使用してクライアントを設定するには、次の 2 つの方法があります。

- 「プロファイル」

  少なくとも、使用するプロファイルとドメインを含むサーバーアドレスを指定する必要があります。プロファイルが指定されていない場合は、デフォルトのプロファイルが使用されます。プロキシと認証データベースの情報を除いて、必要な情報はサーバーから入手できます。クライアントの資格レベルがプロキシまたは匿名プロキシである場合は、プロキシのバインド DN とパスワードを入力する必要があります。詳細については、221 ページの「クライアント資格レベルの割り当て」を参照してください。

- 「手動」

  クライアント自体でプロファイルを設定します。つまり、コマンド行からすべてのパラメータを定義します。このため、プロファイル情報はキャッシュファイルに格納されサーバーによってリフレッシュされることはありません。

注 – クライアントを手動で構成することも可能ですが、お勧めしません。構成用のプロファイルを使用すると、クライアントの管理が容易になりコストも削減できます。

プロファイルを使用してクライアントを初期化する

▼ プロファイルを使用してクライアントを初期化する方法

1. スーパーユーザーになります。
2. init を指定して ldapclient を実行します。

```
# ldapclient init -a profileName=new -a \
domainName=west.example.com 192.168.0.0

System successfully configured
```

プロキシの資格を使用する

▼ プロキシの資格を使用してクライアントを初期化する方法

1. スーパーユーザーになります。
2. ldapclient を実行します (プロキシ値を定義します)。

```
# ldapclient init -a
proxyDn=cn=proxyagent,ou=profile,dc=west,dc=example,dc=com -a
domainname=west.example.com -a profilename=pit1 -a
proxypassword=test1234 192.168.0.0

System successfully configured
```

使用するプロファイルがプロキシ用に設定されている場合は、-a proxyDn と -a proxypassword が必須です。 サーバーに保存されているプロファイルにはこの資格情報が含まれていないため、クライアントを初期設定するときは資格情報を入力する必要があります。この方法は、プロキシの資格情報をサーバーに保存していた従来の方法に比べて安全性が高くなります。

プロキシ情報は /var/ldap/ldap_client_cred の作成に使用され、それ以外の情報は/var/ldap/ldap_client_file に格納されます。

注 – どちらのクライアント構成ファイルも直接編集しないでください。これらのファイルの内容を作成または変更する場合は、ldapclient を使用してください。

クライアントを手動で初期設定する

スーパーユーザーは、クライアントの構成を手動で行うことができます。ただし、この処理では多数のチェックが省略されるため、システムを正しく構成できないことがよくあります。また、プロファイルを使用するときのように一括に設定するのではなく、「マシンごとに」設定を変更する必要があります。

▼ クライアントを手動で初期設定する方法

1. スーパーユーザーになります。
2. ldapclient manual を実行します。

```
# ldapclient manual –a domainName=dc=west.example.com \
–a credentialLevel=proxy –a defaultSearchBase=dc=west,
dc=example, dc=com \
–a proxyDN=cn=proxyagent,ou=profile,dc=west,dc=example,dc=com \
–a proxyPassword=testtest 192.168.0.0
```

3. ldapclient list を使用して確認します。

```
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
NS_LDAP_SERVERS= 192.168.0.0
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_CREDENTIAL_LEVEL= proxy
```

手動によるクライアント構成を変更する

▼ 手動による構成を変更する方法

1. スーパーユーザーになります。
2. ldapclient mod を実行して、認証方法を simple に変更します。

```
# ldapclient mod -a authenticationMethod=simple
```

3. ldapclient list を実行して、更新が行われたことを確認します。

```
# ldapclient list
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
NS_LDAP_SERVERS= 192.168.0.0
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_AUTH= simple
```

```
NS_LDAP_CREDENTIAL_LEVEL= proxy
```

クライアントの初期設定を解除する

▼ クライアントの初期設定を解除する方法

1. スーパーユーザーになります。
2. ldapclient uninit を実行します。

   # ldapclient uninit

   ```
   System successfully recovered
   ```

   ldapclient uninit コマンドは、クライアントのネームサービスを元の状態 (init、modify、または manual の最後の操作が行われる前の状態) に復元します。言い換えれば、最後に行われた手順を「元に戻します」。たとえば、profile1 を使用するようにクライアントを設定した後で profile2 を使用するように変更した場合、ldapclient uninit を実行すると、クライアントで profile1 を使用するように設定が元に戻ります。

TLS セキュリティの設定

注意 – cert7.db と key3.db の各ファイルには、どのユーザーも読み取り権が必要です。key3.db ファイルに非公開鍵を含めないようにしてください。

TLS を使用する場合は、必要なセキュリティデータベースをインストールしなければなりません。特に、cert7.db と key3.db の両ファイルは必須です。cert7.db ファイルには、信頼できる認証のデータベースが入ります。key3.db ファイルには、クライアントの鍵が入ります。LDAP ネームサービスクライアントではクライアントの鍵を使用しませんが、このファイルは必要です。

ldapclient を実行する前に、この節に記述されている必要なセキュリティデータベースファイルを設定およびインストールしておく必要があります。

これらのファイルを作成および管理する方法については、『*iPlanet Directory Server 5.1* 管理者ガイド』のLDAP クライアントの構成に関する節を参照してください。これらのファイルを構成したら、LDAP ネームサービスクライアントで使用できるように所定の場所にそれらを格納する必要があります。この場所を判断するために、属性 certificatePath が使用されます。この属性はデフォルトで /var/ldap です。

たとえば、Netscape Communicator を使用して必要な cert7.db ファイルと key3.db ファイルを設定した後で、それらのファイルをデフォルトの位置にコピーします。

```
# cp $HOME/.netscape/cert7.db /var/ldap

# cp $HOME/.netscape/key3.db /var/ldap
```

次に、すべてのユーザーに読み取り権を付与します。

```
# chmod 444 /var/ldap/cert7.db

# chmod 444 /var/ldap/key3.db
```

注 – Netscape では、`cert7.db` と`key3.db` の各ファイルを `$HOME/.netscape` ディレクトリで管理します。このため、それらのセキュリティデータベースを LDAP ネームサービスクライアントで使用する場合は、そのコピーをローカルファイルシステム上に格納する必要があります。

PAM を構成する

`pam_ldap` を使用する場合は、266 ページの「`pam_ldap` に対応した `pam.conf` ファイルの例 」に説明されている `pam.conf` ファイルの例に従って、`pam_ldap.so.1` が含まれる行をクライアントの `/etc/pam.conf` ファイルに追加します。`pam_ldap.so.1` が含まれる行がすべて必要なわけではありません。`pam_ldap` を必要とするコマンド、ログイン、パスワードなどに関するセクションだけが変更の対象となります。詳細については、`pam.conf(4)` のマニュアルページを参照してください。

ネームサービス情報の検出

`ldaplist` を使用する

`ldaplist` は、LDAP サーバーから取得したネームサービス情報を LDIF フォーマットで表示する LDAP ユーティリティです。このユーティリティは、障害追跡に役立ちます。詳細については、`ldaplist(1)` のマニュアルページを参照してください。

すべての LDAP コンテナを表示する

`ldaplist` は、レコードを空行で区切って出力を表示します。この表示方法は、複数行にまたがる大きなレコードに有効です。

注 – ldaplist の出力は、クライアントの構成によって変わります。たとえば、ns_ldap_search の値が one ではなく sub である場合は、ldaplist によって現在の検索 baseDN の下にあるエントリがすべて表示されます。

以下に ldaplist の出力例を示します。

```
# ldaplist
dn: ou=people,dc=west,dc=example,dc=com

dn: ou=group,dc=west,dc=example,dc=com

dn: ou=rpc,dc=west,dc=example,dc=com

dn: ou=protocols,dc=west,dc=example,dc=com

dn: ou=networks,dc=west,dc=example,dc=com

dn: ou=netgroup,dc=west,dc=example,dc=com

dn: ou=aliases,dc=west,dc=example,dc=com

dn: ou=hosts,dc=west,dc=example,dc=com

dn: ou=services,dc=west,dc=example,dc=com

dn: ou=ethers,dc=west,dc=example,dc=com

dn: ou=profile,dc=west,dc=example,dc=com

dn: automountmap=auto_home,dc=west,dc=example,dc=com

dn: automountmap=auto_direct,dc=west,dc=example,dc=com

dn: automountmap=auto_master,dc=west,dc=example,dc=com

dn: automountmap=auto_shared,dc=west,dc=example,dc=com
```

すべてのユーザーエントリ属性を表示する

ユーザーの passwd エントリなど特定の情報を表示する場合は、次のように getent を使用します。

```
# getent passwd user1
user1::30641:10:Joe Q. User:/home/user1:/bin/csh
```

すべての属性を表示する場合は、-l オプションを指定して ldaplist コマンドを実行します。

```
# ldaplist -l passwd user1

dn: uid=user1,ou=People,dc=west,dc=example,dc=com
        uid: user1
        cn: user1
        uidNumber: 30641
        gidNumber: 10
        gecos: Joe Q. User
        homeDirectory: /home/user1
        loginShell: /bin/csh
        objectClass: top
        objectClass: shadowAccount
        objectClass: account
        objectClass: posixAccount
        shadowLastChange: 6445
        userPassword: {crypt}J6vlYXRU.sW8c
```

クライアント環境のカスタマイズ

クライアント環境では、必要に応じて 2 種類のカスタマイズを行うことができます。

nsswitch.conf ファイルを変更する

/etc/nsswitch.conf ファイルを変更して、各サービスが情報を取得する場所をカスタマイズできます。デフォルトの設定は /etc/nsswitch.ldap に保存されており、クライアントの初期化時に ldapclient がこのファイルを使って /etc/nsswitch.conf ファイルを作成します。

/etc/resolv.conf ファイルを設定して DNS を使用可能にする場合は、次に示すように、DNS をhosts 行に追加します。

```
hosts:      ldap dns [NOTFOUND=return] files
```

どのサービスも変更できますが注意が必要です。変更したサービスのデータがサーバー上に生成されない場合、カスタマイズは無効になります。さらに、ファイルがデフォルトで設定されない場合もあります。

第 17 章

LDAP 構成に関する障害追跡

この章では、LDAP の構成で発生する問題とその解決方法を示します。

クライアントステータスの監視

この節では、LDAP クライアント環境の状態判定に使用するさまざまなコマンドを紹介します。詳細については、一般的な問題とその解決方法について解説した、障害追跡に関する節を参照してください。使用可能なオプションの詳細については、マニュアルページも参照してください。

ldap_cachemgr が実行中であることを確認する

ldap_cachemgr デーモンは、常に実行中で適切に機能している必要があります。このデーモンが機能していない場合、意図した動作はまったく実行されません。ldap_cachemgr の動作を確認するには、次の 2 つの方法があります。

- **ps -ef** を使用する

  # **ps -ef | grep ldap_cachemgr**

- -g オプションを ldap_cachemgr に渡す

  この操作により、次の状態情報がダンプされます。この情報は問題を診断する際に役立ちます。

  # **/usr/lib/ldap/ldap_cachemgr -g**

```
cachemgr configuration:
server debug level          0
server log file "/var/ldap/cachemgr.log"
number of calls to ldapcachemgr         19
```

```
cachemgr cache data statistics:
Configuration refresh information:
  Previous refresh time: 2001/11/16 18:33:28
  Next refresh time:     2001/11/16 18:43:28
Server information:
  Previous refresh time: 2001/11/16 18:33:28
  Next refresh time:     2001/11/16 18:36:08
  server: 192.168.0.0, status: UP
  server: 192.168.0.1, status: ERROR
    error message: Can't connect to the LDAP server
Cache data information:
  Maximum cache entries:          256
  Number of cache entries:          2
```

現在のプロファイル情報の確認

スーパーユーザーになり、`ldapclient` を `list` オプションを付けて実行します。

 # ldapclient list

```
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
NS_LDAP_SERVERS= 192.168.0.0, 192.168.0.1
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_AUTH= simple
NS_LDAP_SEARCH_REF= TRUE
NS_LDAP_SEARCH_SCOPE= one
NS_LDAP_SEARCH_TIME= 30
NS_LDAP_SERVER_PREF= 192.168.0.0
NS_LDAP_PROFILE= pit1
NS_LDAP_CREDENTIAL_LEVEL= proxy
NS_LDAP_SERVICE_SEARCH_DESC= passwd:ou=people,?sub
NS_LDAP_SERVICE_SEARCH_DESC= group:ou=group,dc=west,dc=example,dc=com?one
NS_LDAP_BIND_TIME= 5
```

`/var/ldap` ファイルは現在 ASCII 形式ですが、バイナリに変更される可能性があるため、このファイルに対して cat コマンドを実行すると問題が発生する可能性があります。この情報へのアクセスにサポートされている方式は、`ldapclient list` です。

基本的なクライアント/サーバー間通信の検証

クライアントが LDAP サーバーに対して通信を行なっていることを確認する最善の方法は、`ldaplist` コマンドを使用することです。引数を付けずに `ldaplist` だけを指定して実行すると、サーバー上のすべてのコンテナがダンプされます。この方法はコンテナが存在している限り可能で、コンテナを生成する必要がありません。最初の手

順が成功したら、`ldaplist passwd` *username* または `ldaplist hosts` *hostname* を実行できますが、大量のデータが含まれている場合には、生成量の少ないサービスを選ぶか、head や more コマンドを使用してデータをパイプ処理することもできます。

クライアント以外のマシンからのサーバーデータの確認

前述のコマンドの大半は、LDAP クライアントであることを前提としています。クライアントを作成していない状態でサーバー上のデータをチェックする場合は、`ldapsearch` コマンドを使用します。次の例では、すべてのコンテナをリスト表示します。

```
# ldapsearch -h server1 -b "dc=west,dc=example,dc=com" -s one
"objectclass=*"
```

構成で発生する問題とその解決方法

LDAP の構成で発生する問題とそれらの解決方法について説明します。

未解決のホスト名

Solaris オペレーティング環境 LDAP クライアントのバックエンドは、ホストの検索で、`gethostbyname`(3N) や `getipnodebyname`(3N) の場合と同様、完全指定ホスト名を返します。格納済みの名前が指定されている (1 つ以上のドットが含まれている) 場合、クライアントはその名前をそのまま返します。たとえば、格納されている名前が `hostB.eng` であれば、返される名前も `hostB.eng` です。

LDAP ディレクトリに格納された名前が指定されていない (ドットが 1 つも含まれない) 場合、クライアントのバックエンドは、その名前にドメイン部分を追加します。たとえば、格納されている名前が `hostA` であれば、返される名前は `hostA.domainname` となります。

LDAP ドメイン内のシステムに遠隔アクセスできない

DNS ドメイン名が LDAP ドメイン名とは異なる場合、格納されたホスト名が完全指定でない限り LDAP ネームサービスをホスト名に対して使用することはできません。

ログインできない

LDAP クライアントは、ログイン時に pam(3) モジュールを使用してユーザーを認証します。UNIX™ 標準の PAM モジュールでは、パスワードをサーバーから読み込みクライアント側で検査します。この動作は、次のいずれかの理由で失敗する場合があります。

1. /etc/nsswitch.conf ファイル内の passwd サービスが ldap を使用しない
2. プロキシエージェントが、サーバーリスト上のユーザーの userPassword 属性を読み取ることができない。プロキシエージェントが比較のためにパスワードをクライアントに返すので、少なくともプロキシエージェントはパスワードを読めなければならない。pam_ldap に関しては、パスワードへの読み取りアクセスを必要としない
3. プロキシエージェントが適切なパスワードを保持していない
4. 該当するエントリに shadowAccount オブジェクトクラスが定義されていない
5. ユーザーの定義済みパスワードが存在しない

   ldapaddent を使用する場合、 -p オプションを使用してパスワードをユーザーエントリに確実に追加する必要があります。ldapaddent を -p オプションなしで実行した場合、ldapaddent を使用して /etc/shadow ファイルを追加しない限り、ユーザーのパスワードはディレクトリに格納されません。
6. どの LDAP サーバーにもアクセスできない

   サーバーの状態を確認します。

   # /usr/lib/ldap/ldap_cachemgr –g
7. pam_conf の構成が不正である
8. LDAP 名前空間でユーザーが定義されていない
9. pam_unix で NS_LDAP_CREDENTIAL_LEVEL が anonymous に設定されており、匿名ユーザーが userPassword 属性を使用できない
10. パスワードが crypt 形式で格納されていない

検索が遅い

LDAP データベースは、パフォーマンス向上にインデックスを使用します。インデックスが正しく作成されていない場合、大幅にパフォーマンスが低下することがあります。このマニュアルの中に、インデックス付けが必要な一連の属性をまとめてあります。また、独自のインデックスを追加して、パフォーマンスの向上を図ることができます。

ldapclient がサーバーにバインドできない

init プロファイルオプションを使用しているときに、ldapclient がクライアントの初期化に失敗しました。これには、いくつかの原因が考えられます。

1. コマンド行で不正なドメイン名が指定された
2. 指定されたクライアントドメインのエントリポイントを表す `nisDomain` 属性が DIT (ディレクトリ情報ツリー) 内に設定されていない
3. アクセス制御情報がサーバー上で適正に設定されていないため、LDAP データベース内の匿名検索が許可されない
4. `ldapclient` コマンドに渡されたサーバーアドレスが間違っている。`ldapsearch(1)` を使用してサーバーのアドレスを確認する
5. `ldapclient` コマンドに渡されたプロファイル名が間違っている。`ldapsearch(1)` を使用して、DIT 内のプロファイル名を確認する
6. クライアントのネットワークインタフェースに対して `snoop(1M)` を実行して外向きのトラフィックを検査し、どのサーバーと通信しているかを確認する

デバッグに `ldap_cachemgr` を使用する

`ldap_cachemgr` を `-g` オプションをつけて使用すると、現在のクライアント構成および統計を表示できるため、デバッグするのに便利です。たとえば、次のように指定します。

```
# ldap_cachemgr –g
```

この結果、すでに説明したように、すべての LDAP サーバーの状態を含む現在のクライアント構成および統計が標準出力に出力されます。このコマンドを実行するのに、スーパーユーザーになる必要はありません。

セットアップ中に `ldapclient` がハングアップする

`ldapclient` コマンドがハングアップした場合、Ctrl-C キーを押すと以前の環境を復元した後で終了します。この状況が発生する場合、サーバーが動作していることをサーバー管理者に確認してください。

プロファイルまたはコマンド行に指定されたサーバーリスト属性で、サーバー情報が適正であることを確認してください。

FAQ (よくある質問)

以前の Solaris リリースで LDAP ネームサービスを使用できますか

現在のところ、LDAP は Solaris 8 および Solaris 9 でのみサポートされています。Solaris 8 と Solaris 9 との相違点については、206 ページの「Solaris 9 LDAP ネームサービスの新機能」 を参照してください。

Solaris LDAP ネームサービスでの DIT のデフォルト位置を教えてください

212 ページの「デフォルトのディレクトリ情報ツリー (DIT)」を参照してください。

第 18 章

一般的なリファレンス

未記入のチェックリスト

表 18–1 サーバーで定義する変数

変数	_____ ネットワークの定義
インストールしたディレクトリサーバーインスタンスのポート番号 (デフォルト = 389)	
サーバー名	
複製サーバー (IP 番号 : ポート番号)	
ディレクトリマネージャ [dn: cn=directory manager]	
サービスされるドメイン名	

表 **18–1** サーバーで定義する変数　(続き)

変数	_____ ネットワークの定義
クライアント要求の処理がタイムアウトするまでの時間 (秒)	
各検索要求で返されるエントリの最大数	

表 **18–2** クライアントプロファイルで定義する変数

変数	_____ ネットワークの定義
プロファイル名	
サーバーリスト (デフォルトはローカルサブネット)	
優先されるサーバーリスト (優先順に記載)	
検索範囲 (検索するディレクトリツリーレベルの数、「`One`」または「`Sub`」)	
サーバーへのアクセスに使用する資格。デフォルトは匿名	
参照に従うか。(メインサーバーが利用不可能な場合に使用される、別のサーバーへのポインタ)。デフォルトは `no`	
検索時にサーバーが情報を返すまでの待機時間の制限 (秒、デフォルトは 30)	
サーバーとの通信時のバインド時間の制限 (秒、デフォルトは 30)。デフォルトは秒	
認証方式。デフォルトは `none`	

アップグレード情報

Solaris 9 クライアントは、Solaris 8 クライアント用に設定されたディレクトリサーバーと完全な互換性があります。`ldapclient(1M)` は、この種のプロファイルをダウンロードしてバージョン 1 のプロファイルを使用してクライアントを構成できます。ただし、Solaris 9 の新機能を利用して新規セキュリティモデルを使用するには、バージョン 2 のプロファイルを使用する必要があります。

サーバーは、旧クライアントと新クライアントの混在環境に対応します。このため、スキーママッピングが無効でバージョン 2 のプロファイルが `serviceSearchDescriptors` 内の特殊フィルタを使用しない構成になっている限

り、どちらのクライアントでも同じ結果を得ることができます。サーバーがデフォルトスキーマを使用しない場合、Solaris 8 クライアントはスキーマを任意に対応づけできないため、旧クライアントはそのサーバーを利用できません。

考慮する必要のあるもう 1 つの追加変更は、Solaris 8 では `ldap_cachemgr()` を実行するクライアントは推奨されず、オプションであったのに対し、Solaris 9 では `ldap_cachemgr()` を「常に実行する必要がある」という点です。このデーモンは、クライアントが適正に動作するために「必須」です。

新しい自動マウントスキーマ

Solaris 9 はデフォルトで、Solaris 8 クライアントの使用する汎用の NIS マップスキーマではなく、自動マウントエントリ用の新規スキーマを使用します。このため Solaris 9 ツールを使用してサーバーを設定した場合、Solaris 8 クライアントから自動マウントエントリを表示できなくなります。サイトで Solaris 9 と Solaris 8 クライアントの両方に対応するサーバーを設定する場合、自動マウントエントリを追加する前に、プロファイルを作成してスキーマを以前のスキーマに対応づけてください。この操作により、`ldapaddent(1M)` が、以前のスキーマを使用してエントリを追加することが保証されます。ただし、すべての Solaris 9 クライアントで、自動マウント用スキーマが対応づけされたプロファイルを使用する必要があることに注意してください。

このマッピングを有効にするため、次のマッピング属性をプロファイルに追加する必要があります。

```
attributeMap:        automount:automountMapName=nisMapName
attributeMap:        automount:automountKey=cn
attributeMap:        automount:automountInformation=nisMapEntry
objectclassMap:      automount:automountMap=nisMap
objectclassMap:      automount:automount=nisObject
```

LDAP コマンド

Solaris には、LDAP 関連のコマンドセットが 2 つ存在します。1 つのセットは一般的な LDAP ツールで、LDAP ネームサービスを使用してクライアントを構成する必要はありません。もう 1 つのセットはクライアント上の共通 LDAP 構成を使用するため、クライアントがネームサービスに LDAP を使用する場合にのみ使用できます。

一般的な LDAP ツール

LDAP コマンド行ツールは、認証やバインドパラメータを含む、一般的なオプションセットをサポートします。

これらのコマンドを使用して、ディレクトリエントリを直接操作できます。`ldapsearch(1)`、`ldapmodify(1)`、`ldapadd(1)`、`ldapdelete(1)` の各ツールは、LDAP データ交換フォーマット (LDIF) と呼ばれる共通の書式をサポートしています。LDIF は、ディレクトリ情報を表現するテキストベースの書式です。

LDAP ネームサービスを必要とする LDAP ツール

表 18–3 ツール (セクション 1 のマニュアルページより)

ツール	機能
`ldapaddent(1M)`	LDAP コンテナ内に、`/etc` 内のファイルに対応するエントリを作成する。このツールを使用して、ファイルからディレクトリを生成できる。たとえば、`/etc/passwd` 形式のファイルを読み込んで、ディレクトリ内に `passwd` エントリを生成する
`ldaplist(1)`	ディレクトリから、さまざまなサービスの内容をリスト表示するのに使用する
`idsconfig(1M)`	LDAP ネームサービスクライアント対応の iPlanet Directory Server 5.1 の設定に使用する

pam_ldap に対応した pam.conf ファイルの例

```
#
# Authentication management
#
# login service (explicit because of pam_dial_auth)
#
login   auth required           pam_authtok_get.so.1
login   auth required           pam_dhkeys.so.1
login   auth required           pam_dial_auth.so.1
login   auth sufficient         pam_unix_auth.so.1
login   auth required           pam_ldap.so.1 try_first_pass
#
# rlogin service (explicit because of pam_rhost_auth)
#
rlogin  auth sufficient         pam_rhosts_auth.so.1
rlogin  auth required           pam_authtok_get.so.1
rlogin  auth required           pam_dhkeys.so.1
rlogin  auth sufficient         pam_unix_auth.so.1
rlogin  auth required           pam_ldap.so.1 try_first_pass
```

```
#
# rsh service (explicit because of pam_rhost_auth)
#
rsh     auth sufficient         pam_rhosts_auth.so.1
rsh     auth required           pam_authtok_get.so.1
rsh     auth required           pam_dhkeys.so.1
rsh     auth sufficient         pam_unix_auth.so.1
rsh     auth required           pam_ldap.so.1 try_first_pass
#
# PPP service (explicit because of pam_dial_auth)
#
ppp     auth required           pam_authtok_get.so.1
ppp     auth required           pam_dhkeys.so.1
ppp     auth required           pam_dial_auth.so.1
ppp     auth sufficient         pam_unix_auth.so.1
ppp     auth required           pam_ldap.so.1 try_first_pass
#
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authenctication

#
other   auth required           pam_authtok_get.so.1
other   auth required           pam_dhkeys.so.1
other   auth sufficient         pam_unix_auth.so.1
other   auth required           pam_ldap.so.1 try_first_pass
#
# passwd command (explicit because of a different authentication module)

#
passwd  auth sufficient         pam_passwd_auth.so.1
passwd  auth required           pam_ldap.so.1  try_first_pass
#
# cron service (explicit because of non-usage of pam_roles.so.1)
#
cron    account required        pam_projects.so.1
cron    account required        pam_unix_account.so.1
#
# Default definition for Account management
# Used when service name is not explicitly mentioned for account
management
#
other   account requisite       pam_roles.so.1
other   account required        pam_projects.so.1
other   account required        pam_unix_account.so.1
#
# Default definition for Session management
# Used when service name is not explicitly mentioned for session
management
#
other   session required        pam_unix_session.so.1
#
# Default definition for  Password management
# Used when service name is not explicitly mentioned for password
management
#
```

```
other   password required       pam_dhkeys.so.1
other   password required       pam_authtok_get.so.1
other   password required       pam_authtok_check.so.1
other   password sufficient     pam_authtok_store.so.1
other   password required       pam_ldap.so.1
#
# Support for Kerberos V5 authentication (uncomment to use Kerberos)
#
#rlogin         auth optional           pam_krb5.so.1 try_first_pass
#login          auth optional           pam_krb5.so.1 try_first_pass
#other          auth optional           pam_krb5.so.1 try_first_pass
#cron           account optional        pam_krb5.so.1
#other          account optional        pam_krb5.so.1
#other          session optional        pam_krb5.so.1
#other          password optional       pam_krb5.so.1 try_first_pass
```

IETF スキーマ

スキーマは、サーバーのディレクトリ内にエントリとして格納可能な情報タイプを記述した定義です。

ディレクトリサーバーが Solaris 9 LDAP ネームサービスクライアントをサポートするには、クライアントのスキーママッピング機能を使用してスキーマをマッピングしていない限り、この章で定義されたスキーマをサーバー内で構成する必要があります。

IETF により定義された 4 つの必須 LDAP スキーマ (RFC 2307 ネットワーク情報サービススキーマ、LDAP メールグループインターネットドラフト、および LDAP Internet Printing Protocol (IPP) ドラフトスキーマ) が存在します。ネーム情報サービスをサポートするには、これらのスキーマ定義をディレクトリサーバーに追加する必要があります。IETF Web サイト http://www.ietf.org で、さまざまな RFC にアクセスできます。

注 – インターネットドラフトとは、最長 6 ヶ月間有効なドラフトの文書で、他の文書によっていつでも更新または廃止される可能性があります。

RFC 2307 ネットワーク情報サービススキーマ

LDAP サーバーは改訂版 RFC 2307 をサポートするように構成する必要があります。

`nisSchema` OID は `1.3.6.1.1` です。RFC 2307 属性を次に示します。

```
( nisSchema.1.0 NAME 'uidNumber'
DESC 'An integer uniquely identifying a user in an
        administrative domain'
```

```
EQUALITY integerMatch SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.1 NAME 'gidNumber'
DESC 'An integer uniquely identifying a group in an
       administrative domain'
EQUALITY integerMatch SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.2 NAME 'gecos'
DESC 'The GECOS field; the common name'
EQUALITY caseIgnoreIA5Match
SUBSTRINGS caseIgnoreIA5SubstringsMatch
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.3 NAME 'homeDirectory'
DESC 'The absolute path to the home directory'
EQUALITY caseExactIA5Match
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.4 NAME 'loginShell'
DESC 'The path to the login shell'
EQUALITY caseExactIA5Match
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.5 NAME 'shadowLastChange'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.6 NAME 'shadowMin'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.7 NAME 'shadowMax'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.8 NAME 'shadowWarning'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.9 NAME 'shadowInactive'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.10 NAME 'shadowExpire'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.11 NAME 'shadowFlag'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.12 NAME 'memberUid'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String' )
```

```
( nisSchema.1.13 NAME 'memberNisNetgroup'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String' )

( nisSchema.1.14 NAME 'nisNetgroupTriple'
DESC 'Netgroup triple'
SYNTAX 'nisNetgroupTripleSyntax' )

( nisSchema.1.15 NAME 'ipServicePort'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.16 NAME 'ipServiceProtocol'
SUP name )

( nisSchema.1.17 NAME 'ipProtocolNumber'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.18 NAME 'oncRpcNumber'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.19 NAME 'ipHostNumber'
DESC 'IP address as a dotted decimal, eg. 192.168.1.1
        omitting leading zeros'
SUP name )

( nisSchema.1.20 NAME 'ipNetworkNumber'
DESC 'IP network as a dotted decimal, eg. 192.168,
        omitting leading zeros'
SUP name SINGLE-VALUE )

( nisSchema.1.21 NAME 'ipNetmaskNumber'
DESC 'IP netmask as a dotted decimal, eg. 255.255.255.0,
         omitting leading zeros'
EQUALITY caseIgnoreIA5Match
SYNTAX 'IA5String{128}' SINGLE-VALUE )

( nisSchema.1.22 NAME 'macAddress'
DESC 'MAC address in maximal, colon separated hex
     notation, eg. 00:00:92:90:ee:e2'
EQUALITY caseIgnoreIA5Match
SYNTAX 'IA5String{128}' )

( nisSchema.1.23 NAME 'bootParameter'
DESC 'rpc.bootparamd parameter'
SYNTAX 'bootParameterSyntax' )

( nisSchema.1.24 NAME 'bootFile'
DESC 'Boot image name'
EQUALITY caseExactIA5Match
SYNTAX 'IA5String' )
```

```
( nisSchema.1.26 NAME 'nisMapName'
SUP name )

( nisSchema.1.27 NAME 'nisMapEntry'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String{1024}' SINGLE-VALUE )

( nisSchema.1.28 NAME 'nisPublicKey'
DESC 'NIS public key'
SYNTAX 'nisPublicKeySyntax' )

( nisSchema.1.29 NAME 'nisSecretKey'
DESC 'NIS secret key'
SYNTAX 'nisSecretKeySyntax' )

( nisSchema.1.30 NAME 'nisDomain'
DESC 'NIS domain'
SYNTAX 'IA5String' )

( nisSchema.1.31 NAME 'automountMapName'
DESC 'automount Map Name'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

( nisSchema.1.32 NAME 'automountKey'
DESC 'Automount Key value'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

( nisSchema.1.33 NAME 'automountInformation'
DESC 'Automount information'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

nisSchema OID は `1.3.6.1.1` です。RFC 2307 `objectClasses` を次に示します。

```
( nisSchema.2.0 NAME 'posixAccount' SUP top AUXILIARY
  DESC 'Abstraction of an account with POSIX attributes'
  MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
  MAY ( userPassword $ loginShell $ gecos $ description ) )

( nisSchema.2.1 NAME 'shadowAccount' SUP top AUXILIARY
  DESC 'Additional attributes for shadow passwords'
  MUST uid
  MAY ( userPassword $ shadowLastChange $ shadowMin
        shadowMax $ shadowWarning $ shadowInactive $
        shadowExpire $ shadowFlag $ description ) )

( nisSchema.2.2 NAME 'posixGroup' SUP top STRUCTURAL
  DESC 'Abstraction of a group of accounts'
```

```
  MUST ( cn $ gidNumber )
  MAY ( userPassword $ memberUid $ description ) )

( nisSchema.2.3 NAME 'ipService' SUP top STRUCTURAL
  DESC 'Abstraction an Internet Protocol service.
        Maps an IP port and protocol (such as tcp or udp)
        to one or more names; the distinguished value of
        the cn attribute denotes the service's canonical
        name'
  MUST ( cn $ ipServicePort $ ipServiceProtocol )
  MAY ( description ) )

( nisSchema.2.4 NAME 'ipProtocol' SUP top STRUCTURAL
  DESC 'Abstraction of an IP protocol. Maps a protocol number
        to one or more names. The distinguished value of the cn
        attribute denotes the protocol's canonical name'
  MUST ( cn $ ipProtocolNumber )
  MAY  description )

( nisSchema.2.5 NAME 'oncRpc' SUP top STRUCTURAL
  DESC 'Abstraction of an Open Network Computing (ONC)
        [RFC1057] Remote Procedure Call (RPC) binding.
        This class maps an ONC RPC number to a name.
        The distinguished value of the cn attribute denotes
        the RPC service's canonical name'
  MUST ( cn $ oncRpcNumber $ description )
  MAY  description )

( nisSchema.2.6 NAME 'ipHost' SUP top AUXILIARY
  DESC 'Abstraction of a host, an IP device. The distinguished
        value of the cn attribute denotes the host's canonical
        name. Device SHOULD be used as a structural class'
  MUST ( cn $ ipHostNumber )
  MAY ( l $ description $ manager $ userPassword ) )

( nisSchema.2.7 NAME 'ipNetwork' SUP top STRUCTURAL
  DESC 'Abstraction of a network. The distinguished value of
        the cn attribute denotes the network's canonical name'
  MUST ipNetworkNumber
  MAY ( cn $ ipNetmaskNumber $ l $ description $ manager ) )

( nisSchema.2.8 NAME 'nisNetgroup' SUP top STRUCTURAL
  DESC 'Abstraction of a netgroup. May refer to other netgroups'
  MUST cn
  MAY ( nisNetgroupTriple $ memberNisNetgroup $ description ) )

( nisSchema.2.9 NAME 'nisMap' SUP top STRUCTURAL
  DESC 'A generic abstraction of a NIS map'
  MUST nisMapName
  MAY description )

( nisSchema.2.10 NAME 'nisObject' SUP top STRUCTURAL
  DESC 'An entry in a NIS map'
  MUST ( cn $ nisMapEntry $ nisMapName )
  MAY description )
```

```
( nisSchema.2.11 NAME 'ieee802Device' SUP top AUXILIARY
  DESC 'A device with a MAC address; device SHOULD be
        used as a structural class'
  MAY macAddress )

( nisSchema.2.12 NAME 'bootableDevice' SUP top AUXILIARY
  DESC 'A device with boot parameters; device SHOULD be
  used as a structural class'
  MAY ( bootFile $ bootParameter ) )

( nisSchema.2.14 NAME 'nisKeyObject' SUP top AUXILIARY
  DESC 'An object with a public and secret key'
  MUST ( cn $ nisPublicKey $ nisSecretKey )
  MAY ( uidNumber $ description ) )

( nisSchema.2.15 NAME 'nisDomainObject' SUP top AUXILIARY
  DESC 'Associates a NIS domain with a naming context'
  MUST nisDomain )

( nisSchema.2.16 NAME 'automountMap' SUP top STRUCTURAL
  MUST ( automountMapName )
  MAY description )

( nisSchema.2.17 NAME 'automount' SUP top STRUCTURAL
  DESC 'Automount information'
  MUST ( automountKey $ automountInformation )
  MAY description )
```

メール別名スキーマ

メール別名情報は、LDAP メールグループインターネットドラフト (正式には draft-steinback-ldap-mailgroups として知られる) で定義されたスキーマを使用します。新しいスキーマが使用可能になるまで、Solaris LDAP クライアントは、このメール別名情報のスキーマの使用を続けます。

インターネットドラフトに定義された LDAP メールグループスキーマには、多数の属性とオブジェクトクラスが含まれています。このうち、Solaris クライアントが使用するのは、2 つの属性と 1 つのオブジェクトクラスだけです。以下にその内容を示します。

メール別名属性を次に示します。

```
( 0.9.2342.19200300.100.1.3
  NAME 'mail'
  DESC 'RFC822 email address for this person'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String(256)'
  SINGLE-VALUE )

( 2.16.840.1.113730.3.1.30
```

```
  NAME 'mgrpRFC822MailMember'
  DESC 'RFC822 mail address of email only member of group'
  EQUALITY CaseIgnoreIA5Match
  SYNTAX 'IA5String(256)' )
```

メール別名 objectClass を次に示します。

```
( 2.16.840.1.113730.3.2.4
  NAME 'mailGroup'
  SUP top
  STRUCTURAL
  MUST mail
  MAY ( cn $ mailAlternateAddress $ mailHost $ mailRequireAuth $
   mgrpAddHeader $ mgrpAllowedBroadcaster $ mgrpAllowedDomain $
   mgrpApprovePassword $ mgrpBroadcasterModeration $ mgrpDeliverTo $
   mgrpErrorsTo $ mgrpModerator $ mgrpMsgMaxSize $
   mgrpMsgRejectAction $ mgrpMsgRejectText $ mgrpNoMatchAddrs $
   mgrpRemoveHeader $ mgrpRFC822MailMember ))
```

ディレクトリユーザーエージェントのプロファイル (DUAProfile) スキーマ

DUAConfSchemaOID は、1.3.6.1.4.1.11.1.3.1 です。

```
DESC 'Default LDAP server host address used by a DUA'
            EQUALITY caseIgnoreMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
            SINGLE-VALUE )

          ( DUAConfSchemaOID.1.1 NAME 'defaultSearchBase'
            DESC 'Default LDAP base DN used by a DUA'
            EQUALITY distinguishedNameMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
            SINGLE-VALUE )

          ( DUAConfSchemaOID.1.2 NAME 'preferredServerList'
            DESC 'Preferred LDAP server host addresses to be used by a
            DUA'
            EQUALITY caseIgnoreMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
            SINGLE-VALUE )

          ( DUAConfSchemaOID.1.3 NAME 'searchTimeLimit'
            DESC 'Maximum time in seconds a DUA should allow for a
            search to complete'
            EQUALITY integerMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
            SINGLE-VALUE )
```

```
( DUAConfSchemaOID.1.4 NAME 'bindTimeLimit'
  DESC 'Maximum time in seconds a DUA should allow for the
  bind operation to complete'
  EQUALITY integerMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
  SINGLE-VALUE )

( DUAConfSchemaOID.1.5 NAME 'followReferrals'
  DESC 'Tells DUA if it should follow referrals
  returned by a DSA search result'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
  SINGLE-VALUE )

( DUAConfSchemaOID.1.6 NAME 'authenticationMethod'
  DESC 'A keystring which identifies the type of
  authentication method used to contact the DSA'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( DUAConfSchemaOID.1.7 NAME 'profileTTL'
  DESC 'Time to live, in seconds, before a client DUA
  should re-read this configuration profile'
      'serviceSearchDescriptor'
  DESC 'LDAP search descriptor list used by a DUA'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( DUAConfSchemaOID.1.9 NAME 'attributeMap'
  DESC 'Attribute mappings used by a DUA'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

( DUAConfSchemaOID.1.10 NAME 'credentialLevel'
  DESC 'Identifies type of credentials a DUA should
  use when binding to the LDAP server'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  SINGLE-VALUE )

( DUAConfSchemaOID.1.11 NAME 'objectclassMap'
  DESC 'Objectclass mappings used by a DUA'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

( DUAConfSchemaOID.1.12 NAME 'defaultSearchScope' SINGLE-VALUE )

( DUAConfSchemaOID.1.13 NAME 'serviceCredentialLevel'
  DESC 'Identifies type of credentials a DUA
  should use when binding to the LDAP server for a
  specific service'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
```

```
( DUAConfSchemaOID.1.15 NAME 'serviceAuthenticationMethod'
  DESC 'Authentication Method used by a service of the DUA'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

   ( DUAConfSchemaOID.2.4 NAME 'DUAConfigProfile'
        SUP top STRUCTURAL
     DESC 'Abstraction of a base configuration for a DUA'
     MUST ( cn )
     MAY ( defaultServerList $ preferredServerList $
    defaultSearchBase $ defaultSearchScope $
    searchTimeLimit $ bindTimeLimit $
    credentialLevel $ authenticationMethod $
    followReferrals $ serviceSearchDescriptor $
    serviceCredentialLevel $ serviceAuthenticationMethod $
    objectclassMap $ attributeMap $
    profileTTL ) )
```

Solaris スキーマ

Solaris オペレーティング環境に必要なスキーマを次に示します。

- Solaris プロジェクトスキーマ
- アクセス制御および実行プロファイルスキーマに基づく役割
- プリンタスキーマ

Solaris プロジェクトスキーマ

/etc/project は、プロジェクトと関連のある属性のローカルソースです。詳細については、project(4) を参照してください。

プロジェクト属性を次に示します。

```
( 1.3.6.1.4.1.42.2.27.5.1.1 NAME 'SolarisProjectID'
  DESC 'Unique ID for a Solaris Project entry'
  EQUALITY integerMatch
  SYNTAX INTEGER SINGLE )

( 1.3.6.1.4.1.42.2.27.5.1.2 NAME 'SolarisProjectName'
  DESC 'Name of a Solaris Project entry'
  EQUALITY caseExactIA5Match
  SYNTAX IA5String SINGLE )

( 1.3.6.1.4.1.42.2.27.5.1.3 NAME 'SolarisProjectAttr'
  DESC 'Attributes of a Solaris Project entry'
```

```
  EQUALITY caseExactIA5Match
  SYNTAX IA5String )

( 1.3.6.1.4.1.42.2.27.5.1.30 NAME 'memberGid'
  DESC 'Posix Group Name'
  EQUALITY caseExactIA5Match
  SYNTAX 'IA5String' )
```

プロジェクト objectClass を次に示します。

```
( 1.3.6.1.4.1.42.2.27.5.2.1 NAME 'SolarisProject'
  SUP top STRUCTURAL
  MUST ( SolarisProjectID $ SolarisProjectName )
  MAY ( memberUid $ memberGid $ description $ SolarisProjectAttr ) )
```

役割ベースのアクセス制御と実行プロファイルスキーマ

ユーザーと役割に関する拡張属性のシステムごとの設定は、/etc/user_attr に置かれます。詳細については、user_attr(4) のマニュアルページを参照してください。

役割ベースのアクセス制御属性を次に示します。

```
( 1.3.6.1.4.1.42.2.27.5.1.4 NAME 'SolarisAttrKeyValue'
  DESC 'Semi-colon separated key=value pairs of attributes'
  EQUALITY caseIgnoreIA5Match
  SUBSTRINGS caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.7 NAME 'SolarisAttrShortDesc'
  DESC 'Short description about an entry, used by GUIs'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.8 NAME 'SolarisAttrLongDesc'
  DESC 'Detail description about an entry'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.9 NAME 'SolarisKernelSecurityPolicy'
  DESC 'Solaris  kernel security policy'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.10 NAME 'SolarisProfileType'
  DESC 'Type of object defined in profile'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.11 NAME 'SolarisProfileId'
  DESC 'Identifier of object defined in profile'
```

```
  EQUALITY caseExactIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.12 NAME 'SolarisUserQualifier'
  DESC 'Per-user login attributes'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.13 NAME 'SolarisReserved1'
  DESC 'Reserved for future use'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.14 NAME 'SolarisReserved2'
  DESC 'Reserved for future use'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )
```

役割ベースのアクセス制御 `objectClassses` を次に示します。

```
( 1.3.6.1.4.1.42.2.27.5.2.3 NAME 'SolarisUserAttr' SUP top AUXILIARY
  DESC 'User attributes'
  MAY ( SolarisUserQualifier $ SolarisAttrReserved1 $ \
        SolarisAttrReserved2 $ SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.4 NAME 'SolarisAuthAttr' SUP top STRUCTURAL
  DESC 'Authorizations data'
  MUST cn
  MAY ( SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisAttrShortDesc $ SolarisAttrLongDesc $ \
        SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.5 NAME 'SolarisProfAttr' SUP top STRUCTURAL
  DESC 'Profiles data'
  MUST cn
  MAY ( SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisAttrLongDesc $ SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.6 NAME 'SolarisExecAttr' SUP top AUXILIARY
  DESC 'Profiles execution attributes'
  MAY ( SolarisKernelSecurityPolicy $ SolarisProfileType $ \
        SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisProfileId $ SolarisAttrKeyValue ) )
```

Internet Printing Protocol 情報

Internet Printing Protocol (IPP) 属性

```
( 1.3.18.0.2.4.1140
NAME 'printer-uri'
DESC 'A URI supported by this printer.
This URI SHOULD be used as a relative distinguished name (RDN).
If printer-xri-supported is implemented, then this URI value
MUST be listed in a member value of printer-xri-supported.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

( 1.3.18.0.2.4.1107
NAME 'printer-xri-supported'
DESC 'The unordered list of XRI (extended resource identifiers) supported
by this printer.
Each member of the list consists of a URI (uniform resource identifier)
followed by optional authentication and security metaparameters.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( 1.3.18.0.2.4.1135
NAME 'printer-name'
DESC 'The site-specific administrative name of this printer, more end-user
friendly than a URI.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )

( 1.3.18.0.2.4.1119
NAME 'printer-natural-language-configured'
DESC 'The configured language in which error and status messages will be
generated (by default) by this printer.
Also, a possible language for printer string attributes set by operator,
system administrator, or manufacturer.
Also, the (declared) language of the "printer-name", "printer-location",
"printer-info", and "printer-make-and-model" attributes of this printer.
For example: "en-us" (US English) or "fr-fr" (French in France) Legal values of
language tags conform to [RFC3066] "Tags for the Identification of Languages".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )
```

```
( 1.3.18.0.2.4.1136
NAME 'printer-location'
DESC 'Identifies the location of the printer. This could include
things like: "in Room 123A", "second floor of building XYZ".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} SINGLE-VALUE )

( 1.3.18.0.2.4.1139
NAME 'printer-info'
DESC 'Identifies the descriptive information about this printer.
This could include things like: "This printer can be used for
printing color transparencies for HR presentations", or
"Out of courtesy for others, please print only small (1-5 page)
jobs at this printer", or even "This printer is going away on July 1, 1997,
please find a new printer".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}
SINGLE-VALUE )

( 1.3.18.0.2.4.1134
NAME 'printer-more-info'
DESC 'A URI used to obtain more information about this specific printer.
For example, this could be an HTTP type URI referencing an HTML page
accessible to a Web Browser.
The information obtained from this URI is intended for end user consumption.'
EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

( 1.3.18.0.2.4.1138
NAME 'printer-make-and-model'
DESC 'Identifies the make and model of the device.
The device manufacturer MAY initially populate this attribute.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )

( 1.3.18.0.2.4.1133
NAME 'printer-ipp-versions-supported'
DESC 'Identifies the IPP protocol version(s) that this printer supports,
including major and minor versions,
i.e., the version numbers for which this Printer implementation meets
the conformance requirements.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1132
NAME 'printer-multiple-document-jobs-supported'
DESC 'Indicates whether or not the printer supports more than one
document per job, i.e., more than one Send-Document or Send-Data
operation with document data.'
```

```
EQUALITY booleanMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

( 1.3.18.0.2.4.1109
NAME 'printer-charset-configured'
DESC 'The configured charset in which error and status messages will be
generated (by default) by this printer.
Also, a possible charset for printer string attributes set by operator,
system administrator, or manufacturer.
For example: "utf-8" (ISO 10646/Unicode) or "iso-8859-1" (Latin1).
Legal values are defined by the IANA Registry of Coded Character Sets and
the "(preferred MIME name)" SHALL be used as the tag.
For coherence with IPP Model, charset tags in this attribute SHALL be
lowercase normalized.
This attribute SHOULD be static (time of registration) and SHOULD NOT be
dynamically refreshed attributetypes: (subsequently).'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} SINGLE-VALUE )

( 1.3.18.0.2.4.1131
NAME 'printer-charset-supported'
DESC 'Identifies the set of charsets supported for attribute type values of
type Directory String for this directory entry.
For example: "utf-8" (ISO 10646/Unicode) or "iso-8859-1" (Latin1).
Legal values are defined by the IANA Registry of Coded Character Sets and
the preferred MIME name.'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} )

( 1.3.18.0.2.4.1137
NAME 'printer-generated-natural-language-supported'
DESC 'Identifies the natural language(s) supported for this directory entry.
For example: "en-us" (US English) or "fr-fr" (French in France).
Legal values conform to [RFC3066], Tags for the Identification of Languages.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} )

( 1.3.18.0.2.4.1130
NAME 'printer-document-format-supported'
DESC 'The possible document formats in which data may be interpreted
and printed by this printer.
Legal values are MIME types come from the IANA Registry of Internet Media Types.'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1129
NAME 'printer-color-supported'
DESC 'Indicates whether this printer is capable of any type of color printing
at all, including highlight color.'
EQUALITY booleanMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.7  SINGLE-VALUE )

( 1.3.18.0.2.4.1128
NAME 'printer-compression-supported'
DESC 'Compression algorithms supported by this printer.
```

```
For example: "deflate, gzip".  Legal values include; "none", "deflate"
attributetypes: (public domain ZIP), "gzip" (GNU ZIP), "compress" (UNIX).'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1127
NAME 'printer-pages-per-minute'
DESC 'The nominal number of pages per minute which may be output by this
printer (e.g., a simplex or black-and-white printer).
This attribute is informative, NOT a service guarantee.
Typically, it is the value used in marketing literature to describe this printer.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1126 NAME 'printer-pages-per-minute-color'
DESC 'The nominal number of color pages per minute which may be output by this
printer (e.g., a simplex or color printer).
This attribute is informative, NOT a service guarantee.
Typically, it is the value used in marketing literature to describe this printer.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1125 NAME 'printer-finishings-supported'
DESC 'The possible finishing operations supported by this printer.
Legal values include; "none", "staple", "punch", "cover", "bind", "saddle-stitch",
"edge-stitch", "staple-top-left", "staple-bottom-left", "staple-top-right",
"staple-bottom-right", "edge-stitch-left", "edge-stitch-top", "edge-stitch-right",
"edge-stitch-bottom", "staple-dual-left", "staple-dual-top", "staple-dual-right",
"staple-dual-bottom".'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1124 NAME 'printer-number-up-supported'
DESC 'The possible numbers of print-stream pages to impose upon a single side of
an instance of a selected medium. Legal values include; 1, 2, and 4.
Implementations may support other values.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27 )

( 1.3.18.0.2.4.1123 NAME 'printer-sides-supported'
DESC 'The number of impression sides (one or two) and the two-sided impression
rotations supported by this printer.
Legal values include; "one-sided", "two-sided-long-edge", "two-sided-short-edge".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1122 NAME 'printer-media-supported'
DESC 'The standard names/types/sizes (and optional color suffixes) of the media
supported by this printer.
For example: "iso-a4",  "envelope", or "na-letter-white".
Legal values  conform to ISO 10175, Document Printing Application (DPA), and any
IANA registered extensions.'
```

```
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1117 NAME 'printer-media-local-supported'
DESC 'Site-specific names of media supported by this printer, in the language in
"printer-natural-language-configured".
For example: "purchasing-form" (site-specific name) as opposed to
(in "printer-media-supported"): "na-letter" (standard keyword from ISO 10175).'
EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1121 NAME 'printer-resolution-supported'
DESC 'List of resolutions supported for printing documents by this printer.
Each resolution value is a string with 3 fields:
1) Cross feed direction resolution (positive integer), 2) Feed direction
resolution (positive integer), 3) Resolution unit.
Legal values are "dpi" (dots per inch) and "dpcm" (dots per centimeter).
Each resolution field is delimited by ">".  For example:  "300> 300> dpi>".'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1120 NAME 'printer-print-quality-supported'
DESC 'List of print qualities supported for printing documents on this printer.
For example: "draft, normal".  Legal values include; "unknown", "draft", "normal",
"high".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1110 NAME 'printer-job-priority-supported'
DESC 'Indicates the number of job priority levels supported.
An IPP conformant printer which supports job priority must always support a
full range of priorities from "1" to "100"
(to ensure consistent behavior), therefore this attribute describes the
"granularity".
 Legal values of this attribute are from "1" to "100".'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1118
NAME 'printer-copies-supported'
DESC 'The maximum number of copies of a document that may be printed as a single job.
A value of "0" indicates no maximum limit.
A value of "-1" indicates unknown.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1111
NAME 'printer-job-k-octets-supported'
DESC 'The maximum size in kilobytes (1,024 octets actually) incoming print job that
this printer will accept.
A value of "0" indicates no maximum limit.  A value of "-1" indicates unknown.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
```

```
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1113
NAME 'printer-service-person'
DESC 'The name of the current human service person responsible for servicing this
printer.
It is suggested that this string include information that would enable other humans
to reach the service person, such as a phone number.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}
SINGLE-VALUE )

( 1.3.18.0.2.4.1114
NAME 'printer-delivery-orientation-supported'
DESC 'The possible delivery orientations of pages as they are printed and ejected
from this printer.
Legal values include; "unknown", "face-up", and "face-down".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1115
NAME 'printer-stacking-order-supported'
DESC 'The possible stacking order of pages as they are printed and ejected from
this printer.
Legal values include; "unknown", "first-to-last", "last-to-first".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1116
NAME 'printer-output-features-supported'
DESC 'The possible output features supported by this printer.
Legal values include; "unknown", "bursting", "decollating", "page-collating",
"offset-stacking".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1108
NAME 'printer-aliases'
DESC 'Site-specific administrative names of this printer in addition the printer
name specified for printer-name.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.6.1.4.1.42.2.27.5.1.63
NAME 'sun-printer-bsdaddr'
DESC 'Sets the server, print queue destination name and whether the client generates
protocol extensions.
"Solaris" specifies a Solaris print server extension. The value is represented b the
following value: server "," destination ", Solaris".'
SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.64
NAME 'sun-printer-kvp'
DESC 'This attribute contains a set of key value pairs which may have meaning to the
```

```
print subsystem or may be user defined.
Each value is represented by the following: key "=" value.'
SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )
```

Internet Printing Protocol (IPP) `ObjectClasses`

```
objectclasses: ( 1.3.18.0.2.6.2549
NAME 'slpService'
DESC 'DUMMY definition'
SUP 'top' MUST (objectclass) MAY ())

objectclasses: ( 1.3.18.0.2.6.254
NAME 'slpServicePrinter'
DESC 'Service Location Protocol (SLP) information.'
AUXILIARY SUP 'slpService')

objectclasses: ( 1.3.18.0.2.6.258
NAME 'printerAbstract'
DESC 'Printer related information.'
ABSTRACT SUP 'top' MAY ( printer-name
$ printer-natural-language-configured
$ printer-location
$ printer-info
$ printer-more-info
$ printer-make-and-model
$ printer-multiple-document-jobs-supported
$ printer-charset-configured
$ printer-charset-supported
$ printer-generated-natural-language-supported
$ printer-document-format-supported
$ printer-color-supported
$ printer-compression-supported
$ printer-pages-per-minute
$ printer-pages-per-minute-color
$ printer-finishings-supported
$ printer-number-up-supported
$ printer-sides-supported
$ printer-media-supported
$ printer-media-local-supported
$ printer-resolution-supported
$ printer-print-quality-supported
$ printer-job-priority-supported
$ printer-copies-supported
$ printer-job-k-octets-supported
$ printer-current-operator
$ printer-service-person
$ printer-delivery-orientation-supported
$ printer-stacking-order-supported $ printer! -output-features-supported ))

objectclasses: ( 1.3.18.0.2.6.255
NAME 'printerService'
DESC 'Printer information.'
STRUCTURAL SUP 'printerAbstract' MAY ( printer-uri
```

```
$ printer-xri-supported ))

objectclasses: ( 1.3.18.0.2.6.257
NAME 'printerServiceAuxClass'
DESC 'Printer information.'
AUXILIARY SUP 'printerAbstract' MAY ( printer-uri $ printer-xri-supported ))

objectclasses: ( 1.3.18.0.2.6.256
NAME 'printerIPP'
DESC 'Internet Printing Protocol (IPP) information.'
AUXILIARY SUP 'top' MAY   ( printer-ipp-versions-supported $
printer-multiple-document-jobs-supported ))

objectclasses: ( 1.3.18.0.2.6.253
NAME 'printerLPR'
DESC 'LPR information.'
AUXILIARY SUP 'top' MUST ( printer-name ) MAY ( printer-aliases))

objectclasses: ( 1.3.6.1.4.1.42.2.27.5.2.14
NAME 'sunPrinter'
DESC 'Sun printer information'
SUP 'top' AUXILIARY MUST (objectclass $ printer-name)  MAY
(sun-printer-bsdaddr $ sun-printer-kvp))
```

Sun プリンタ属性

```
ATTRIBUTE ( 1.3.6.1.4.1.42.2.27.5.1.63
NAME sun-printer-bsdaddr
DESC 'Sets the server, print queue destination name and whether the
     client generates protocol extensions. "Solaris" specifies a
     Solaris print server extension.  The value is represented by
     the following value: server "," destination ", Solaris".'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
)


ATTRIBUTE ( 1.3.6.1.4.1.42.2.27.5.1.64
NAME sun-printer-kvp
DESC 'This attribute contains a set of key value pairs which may have
      meaning to the print subsystem or may be user defined.  Each
      value is represented by the following: key "=" value.'
EQUALITY caseIgnoreIA5Match
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15  )
```

Sun プリンタ ObjectClasses

```
OBJECTCLASS ( 1.3.6.1.4.1.42.2.27.5.2.14
NAME sunPrinter
DESC 'Sun printer information'
```

```
SUP  top
AUXILIARY
MUST ( printer-name )
MAY  ( sun-printer-bsdaddr $ sun-printer-kvp ))
```

汎用ディレクトリサーバーの要件

Solaris 9 LDAP クライアントをサポートする場合、サーバーの種類に関係なく、LDAP v3 プロトコル、複合ネーミングおよび補助オブジェクトクラスをサポートする必要があります。また、次の制御を 1 つ以上サポートする必要があります。

- 単純ページモード (RFC 2696)
- 仮想リスト表示制御

  サーバーは、次の認証方式を 1 つ以上サポートする必要があります。

```
anonymous
simple
sasl/cram-MD5
sasl/digest-MD5
```

pam_unix を使用する場合、サーバーは UNIX 暗号形式 (crypt) でのパスワード保管をサポートする必要があります。

TLS を使用する場合、サーバーは SSL または TLS をサポートする必要があります。

LDAP ネームサービスで使用されるデフォルトフィルタ

SSD を使用して個々のサービスにパラメータを手動で指定しないと、デフォルトフィルタが使用されます。特定のサービスのデフォルトフィルタを表示するには、–v オプションを指定して ldaplist を実行してください。

以下の例では、filter=(&(objectclass=iphost)(cn=abcde) によってデフォルトフィルタが定義されます。

```
database=hosts
filter=(&(objectclass=iphost)(cn=abcde)
user data=(&(%s) (cn=abcde))
```

ldaplist は、以下に示す一連のデフォルトフィルタを生成します (%s は文字列を意味し、%d は数値を意味する)。

```
hosts
(&(objectclass=iphost)(cn=%s))
--------------
passwd
(&(objectclass=posixaccount)(uid=%s))
--------------
services
(&(objectclass=ipservice)(cn=%s))
--------------
group
(&(objectclass=posixgroup)(cn=%s))
--------------
netgroup
(&(objectclass=nisnetgroup)(cn=%s))
--------------
networks
(&(objectclass=ipnetwork)(ipnetworknumber=%s))
--------------
netmasks
(&(objectclass=ipnetwork)(ipnetworknumber=%s))
--------------
rpc
(&(objectclass=oncrpc)(cn=%s))
--------------
protocols
(&(objectclass=ipprotocol)(cn=%s))
--------------
bootparams
(&(objectclass=bootableDevice)(cn=%s))
--------------
ethers
(&(objectclass=ieee802Device)(cn=%s))
--------------
publickey
(&(objectclass=niskeyobject)(cn=%s))
or
(&(objectclass=niskeyobject)(uidnumber=%d))
--------------
aliases
(&(objectclass=mailGroup)(cn=%s))
--------------
```

表 **18–4** getXbyY 呼び出しで使用される LDAP フィルタ

フィルタ	定義
bootparamByName	(&(objectClass=bootableDevice)(cn=%s))
etherByHost	(&(objectClass=ieee802Device)(cn=%s))
etherByEther	(&(objectClass=ieee802Device)(macAddress=%s))
groupByName	(&(objectClass=posixGroup)(cn=%s))

表 **18–4** getXbyY 呼び出しで使用される LDAP フィルタ　(続き)

フィルタ	定義
groupByGID	(&(objectClass=posixGroup)(gidNumber=%ld))
groupByMember	(&(objectClass=posixGroup)(memberUid=%s))
hostsByName	(&(objectClass=ipHost)(cn=%s))
hostsByAddr	(&(objectClass=ipHost)(ipHostNumber=%s))
keyByUID	(&(objectClass=nisKeyObject)(uidNumber=%s))
keyByHost	(&(objectClass=nisKeyObject)(cn=%s))
netByName	(&(objectClass=ipNetwork)(cn=%s))
netByAddr	(&(objectClass=ipNetwork)(ipNetworkNumber=%s))
nisgroupMember	(membernisnetgroup=%s)
maskByNet	(&(objectClass=ipNetwork)(ipNetworkNumber=%s))
printerByName	(&(objectClass=sunPrinter)(printer-name=%s))
projectByName	(&(objectClass=SolarisProject) (SolarisProjectName=%s))
projectByID	(&(objectClass=SolarisProject) (SolarisProjectID=%ld))
protoByName	(&(objectClass=ipProtocol)(cn=%s))
protoByNumber	(&(objectClass=ipProtocol) (ipProtocolNumber=%d))
passwordByName	(&(objectClass=posixAccount)(uid=%s))
passwordByNumber	(&(objectClass=posixAccount)(uidNumber=%ld))
rpcByName	(&(objectClass=oncRpc)(cn=%s))
rpcByNumber	(&(objectClass=oncRpc)(oncRpcNumber=%d))
serverByName	(&(objectClass=ipService)(cn=%s))
serverByPort	(&(objectClass=ipService)(ipServicePort=%ld))
serverByNameAndProto	(&(objectClass=ipService)(cn=%s) (ipServiceProtocol=%s))
specialByNameserver	(ipServiceProtocol=%s))
ByPortAndProto	(&(objectClass=shadowAccount)(uid=%s))
netgroupByTriple	(&(objectClass=nisNetGroup)(nisnetgrouptriple= (%s,%s,%s)))

表 **18–4** getXbyY 呼び出しで使用される LDAP フィルタ　*(続き)*

フィルタ	定義
netgroupByMember	(&(objectClass=nisNetGroup)(| (membernisnetgroup=%s)
authName	(&(objectClass=SolarisAuthAttr)(cn=%s))
auditUserByName	(&(objectClass=SolarisAuditUser)(uid=%s))
execByName	(&(objectClass=SolarisExecAttr)(cn=%s) (SolarisKernelSecurityPolicy=%s) (SolarisProfileType=%s))
execByPolicy	(&(objectClass=SolarisExecAttr) (SolarisProfileId=%s) (SolarisKernelSecurityPolicy=%s) (SolarisProfileType=%s))
profileByName	(&(objectClass=SolarisProfAttr)(cn=%s))
userByName	(&(objectClass=SolarisUserAttr)(uid=%s))

次の表に getent 属性フィルタの一覧を示します。

表 **18–5** getent 属性フィルタ

フィルタ	定義
aliases	(objectClass=rfc822MailGroup)
auth_attr	(objectClass=SolarisAuthAttr)
audit_user	(objectClass=SolarisAuditUser)
exec_attr	(objectClass=SolarisExecAttr)
group	(objectClass=posixGroup)
hosts	(objectClass=ipHost)
networks	(objectClass=ipNetwork)
prof_attr	(objectClass=SolarisProfAttr)
protocols	(objectClass=ipProtocol)
passwd	(objectClass=posixAccount)
printers	(objectClass=sunPrinter)
rpc	(objectClass=oncRpc)
services	(objectClass=ipService)
shadow	(objectclass=shadowAccount)

表 **18–5** getent 属性フィルタ　(続き)

フィルタ	定義
project	(objectClass=SolarisProject)
usr_attr	(objectClass=SolarisUserAttr)

第 19 章

NIS+ から LDAP への移行

この章では、NIS+ ネームサービスから LDAP ネームサービスへの移行方法について説明します。

概要

NIS+ サーバーデーモン `rpc.nisd` は、NIS+ データを `/var/nis/data` ディレクトリ内の NIS+ 固有の書式ファイルに格納します。 NIS+ データは、LDAP と同期化することができます。従来は、そのために外部エージェントが必要でした。しかし、新しい NIS+ デーモンでは、LDAP サーバーを NIS+ データのデータリポジトリとして使用できるようになりました。これにより、NIS+ および LDAP クライアントが同一のネームサービス情報を共有できるため、メインネームサービスを NIS+ から LDAP に移行する作業がより簡単になりました。LDAP をネームサービスとして使用する場合の詳細は、第 12 章を参照してください。

デフォルトの `rpc.nisd` デーモンは、従来と同様に機能し、`/var/nis/data` の NIS+ データベースにデータを格納します。システム管理者は、必要に応じて、NIS+ データベースの一部の権限を LDAP サーバーに譲渡し、NIS+ データのリポジトリとして使用することができます。この場合、`/var/nis/data` ファイルは `rpc.nisd` デーモンのキャッシュとして機能するため、LDAP ルックアップトラフィックが減少します。また、LDAP サーバーが一時的に使用できなくなった場合でも、 `rpc.nisd` デーモンは動作を継続できます。NIS+ および LDAP は常に同期化されるだけでなく、NIS+ および LDAP 間でデータをアップロードまたはダウンロードすることができます。

LDAP に対するデータのマッピングは、構成ファイルの柔軟な構文を使用して行います。`client_info.org_dir` および `timezone.org_dir` 以外のすべての標準 NIS+ テーブルは、テンプレートマッピングファイル `/var/nis/NIS+LDAPmapping.template` で対応できます。ほとんどの NIS+ インストールのテーブルは、変更する必要がないか、わずかな変更で済みます

(client_info.org_dir と timezone.org_dir については、320 ページの「client_info および timezone テーブル」 を参照)。 NIS+ データは、LDAP ディレクトリ情報ツリー (DIT) に配置されます。また、マッピングファイルでは、LDAP から入力された NIS+ データに対して生存期間 (TTL) を設定できます。多くの場合、NIS+ 列値および LDAP 属性値は 1 対 1 で対応づけられますが、マッピングファイルはより複雑な関係にも対応できます。

新しい /etc/default/rpc.nisd ファイルは、LDAP サーバーと認証を選択するときに使用し、rpc.nisd の一般的な動作をいくつか制御します。rpc.nisd(4) のマニュアルページを参照してください。マッピングの詳細は、/var/nis/NIS+LDAPmapping ファイルを使用して指定します。詳細については、NIS+LDAPmapping(4) のマニュアルページを参照してください。このファイルの名前を変更するときは、rpc.nisd に -m コマンド行オプションを指定して使用します。詳細については、rpc.nisd(1M) のマニュアルページを参照してください。

この章では、次の用語を使用します。

- コンテナ

  すべての関連エントリが格納される LDAP DIT 内の場所。たとえば、ユーザーアカウント情報は、多くの場合、 ou=People コンテナに格納される。また、ホストアドレス情報は、 ou=Hosts コンテナに格納される

- ネット名

  認証可能な SecureRPC 内のエンティティ (ユーザーまたはマシン)。

- マッピング

  NIS+ オブジェクトと LDAP エントリとの関係。たとえば、 passwd.org_dir NIS+ テーブルの name 列のデータ (アカウントのユーザー名など) が、ou=People コンテナ内の posixAccount オブジェクトクラスの LDAP uid 属性に対応しているとする。構成によって、name 列および uid 属性が対応づけられる。 name 列が uid 属性に対応づけられる (またはその逆) とも表現できる

- 主体

  認証可能な NIS+ のエンティティ (ユーザーまたはマシン) 。通常、ネット名と主体名は 1 対 1 で対応づけられる

構成ファイル

rpc.nisd の処理は、2 つの構成ファイルを使用して制御します。

- /etc/default/rpc.nisd

  LDAP サーバーと認証、NIS+ ベースドメイン、LDAP デフォルト検索ベース、例外処理、および rpc.nisd の一般的な構成に関する情報を含みます。このファイルは、LDAP マッピングが有効であるかどうかにかかわらず適用されます。

- /var/nis/NIS+LDAPmapping

  NIS+ データと LDAP との間のマッピング情報を含みます。テンプレートファイル (/var/nis/NIS+LDAPmapping.template) は、client_info.org_dir と timezone.org_dir 以外のすべての標準 NIS+ オブジェクトに対応しています。

320 ページの「client_info および timezone テーブル」とNIS+LDAPmapping(4) のマニュアルページを参照してください。

構成とは、値を定義済みの属性に割り当てることです。構成ファイル以外に、構成属性を LDAP から読み込むこともできます (328 ページの「構成情報を LDAP に格納する」 を参照)。また、rpc.nisd コマンドの -x オプションに構成属性を指定することもできます。複数の場所で同じ属性が指定されている場合、優先順位 (高から低) は次のとおりです。

1. rpc.nisd -x オプション
2. 構成ファイル
3. LDAP

属性とオブジェクトクラスの作成

NIS+ と LDAP のマッピングの構成によっては、新しい LDAP 属性とオブジェクトクラスをいくつか作成しなければならないことがあります。次の例では、これらの作成方法として、 ldapadd コマンドの入力として使用できる LDIF データを指定する方法を示します。LDIF データを含むファイルを作成してから、ldapadd(1) を起動します。

```
# ldapadd -D bind-DN–f ldif -file
```

この方法は、iPlanet™ Directory Server 5.1 およびほかの LDAP サーバーでも使用します。

注 – ただし、defaultSearchBase、 preferredServerList、および authenticationMethod 属性を除き、この章で使用されているオブジェクト識別子 (OID) は、SYNTAX 仕様と同様に、説明用に挙げているだけです。OID の基準はありません。任意の OID を使用できます。

開始前に必要な処置

NIS+ データを LDAP リポジトリに格納するために必要な構成の概要については、NIS+LDAPmapping(4) のマニュアルページを参照してください。 ここでは、構成ファイルの編成について詳細に説明します。

/etc/default/rpc.nisd

/etc/default/rpc.nisd ファイルに値を割り当てるときは、すべて attributeName=value タイプとします。

一般的な構成

次の属性は、 rpc.nisd の一般的な構成を制御します。また、LDAP マッピングが有効かどうかにかかわらず、アクティブになります。 これらの属性は通常、デフォルトのままにしておきます。詳細については、rpc.nisd(4) のマニュアルページを参照してください。

- nisplusNumberOfServiceThreads
- nisplusThreadCreationErrorAction
- nisplusThreadCreationErrorAttempts
- nisplusThreadCreationErrorTimeout
- nisplusDumpErrorAction
- nisplusDumpErrorAttempts
- nisplusDumpErrorTimeout
- nisplusResyncService
- nisplusUpdateBatching
- nisplusUpdateBatchingTimeout

LDAP からの構成データ

次の属性は、LDAP からのその他の構成属性の読み込みを制御します。これらの属性自体を LDAP に常駐させることはできません。コマンド行または構成ファイルから読み込む必要があります。詳細については、rpc.nisd(4) のマニュアルページを参照してください。

- nisplusLDAPconfigDN
- nisplusLDAPconfigPreferredServerList
- nisplusLDAPconfigAuthenticationMethod
- nisplusLDAPconfigTLS
- nisplusLDAPconfigTLSCertificateDBPath
- nisplusLDAPconfigProxyUser
- nisplusLDAPconfigProxyPassword

サーバーの選択

- preferredServerList

  LDAP サーバーとポート番号を指定します。

```
# LDAP server can be found at port 389
# LDAP server can be found at port 389
on the local machine
# preferredServerList=127.0.0.1
# Could also be written
# preferredServerList=127.0.0.0.1:389
LDAP server on the machine at IP
# address "1.2.3.4", at port 65042
# preferredServerList=1.2.3.4:65042
```

認証とセキュリティ

- `authenticationMethod`
- `nisplusLDAPproxyUser`
- `nisplusLDAPproxyPassword`

認証方式と、その方式に適切なプロキシユーザー (バインド識別名 DN) とパスワード (鍵またはその他の共有された機密情報)。これらは、`rpc.nisd` デーモンと LDAP サーバーの間で使用されます。詳細については、311 ページの「セキュリティと認証」 を参照してください。

- `nisplusLDAPTLS`
- `nisplusLDAPTLSCertificateDBPath`

必要に応じて、SSL を使用し、証明書ファイルの場所を指定します。詳細については、312 ページの「SSL の使用」 を参照してください。

LDAP および NIS+ 内のデフォルトの場所

- `defaultSearchBase`

  LDAP DIT 内で、RFC 2307 に準拠したネームサービスデータのコンテナが配置される場所。コンテナ DN の完全な検索ベースを個別に指定しなかった場合は、この場所がデフォルトで使用されます。詳細については、301 ページの「`nisplusLDAPobjectDN`」 を参照してください。

- `nisplusLDAPbaseDomain`

  NIS+ オブジェクト仕様 (299 ページの「`nisplusLDAPdatabaseIdMapping`」 を参照) を完全指定しなかった場合は、このデフォルト NIS+ ドメイン名が使用されます。

LDAP 通信のタイムアウト制限、サイズ制限、および参照アクション

- `nisplusLDAPbindTimeout`
- `nisplusLDAPmodifyTimeout`
- `nisplusLDAPaddTimeout`
- `nisplusLDAPdeleteTimeout`

上のパラメータ (順番に、`ldap bind`、`modify`、`add`、および `delete` 操作) でタイムアウトを設定します。これらのパラメータは通常、デフォルトのままにしておきます。

- `nisplusLDAPsearchTimeout`
- `nisplusLDAPsearchTimeLimit`

上のパラメータには、LDAP 検索処理のタイムアウトを設定します。下のパラメータでは、サーバー側の検索時間制限を要求します。`nisplusLDAPsearchTimeLimit` では、LDAP サーバーが検索要求に使用する時間を制御します。このため、

nisplusLDAPsearchTimeLimit には nisplusLDAPsearchTimeout 以上の値を設定してください。NIS+ サーバー、LDAP サーバー、および 2 つのサーバー間の接続のパフォーマンスに応じて、検索制限をデフォルト値より大きくしなければならないことがあります。rpc.nisd から送信されたタイムアウトに関するシステムログメッセージを基にして、これらの値を大きくするかどうかを判断します。

- nisplusLDAPsearchSizeLimit

  このパラメータでは、LDAP 検索要求に対して返される LDAP データ量に対する制限を要求します。デフォルトでは、制限は要求しません。この制限は、サーバー側に適用されます。LDAP サーバーでは、最大データ量に対して制限を適用することがあります。データ量の制限は、使用されているプロキシユーザー (バインド DN) に関連づけられることがあります。最も大きいコンテナに対して十分なデータが送信されるように、LDAP サーバーで rpc.nisd を設定してください。ただし、最も大きなコンテナの場所は、環境によって異なります。多くの場合、passwd.org_dir、mail_aliases.org_dir 、または netgroup.org_dir のコンテナが使用されます。詳細については、LDAP サーバーのマニュアルを参照してください。

- nisplusLDAPfollowReferral

  このパラメータには、LDAP の処理中に別の LDAP サーバーへの参照が発生したときに、実行する処理を指定します。デフォルトでは、参照は行いません。参照を希望するか必要な場合は、参照を有効にします。参照は便利ですが、参照要求が発生するたびに rpc.nisd と複数の LDAP サーバーが対話する必要があるため、処理速度が遅くなることがあります。rpc.nisd には通常、発行する可能性のあるすべての LDAP 要求を処理できる LDAP サーバーを直接指定してください。

エラー処理

次のパラメータには、LDAP 処理中にエラーが発生したときに、実行する処理を指定します。これらのパラメータは通常、デフォルトのままにしておきます。詳細については、rpc.nisd(4) のマニュアルページを参照してください。

- nisplusLDAPinitialUpdateAction
- nisplusLDAPinitialUpdateOnly
- nisplusLDAPretrieveErrorAction
- nisplusLDAPretrieveErrorAttempts
- nisplusLDAPretrieveErrorTimeout
- nisplusLDAPstoreErrorAction
- nisplusLDAPstoreErrorAttempts
- nisplusLDAPstoreErrorTimeout
- nisplusLDAPrefreshErrorAction
- nisplusLDAPrefreshErrorAttempts
- nisplusLDAPrefreshErrorTimeout

一般的な LDAP 処理の制御

- nisplusLDAPmatchFetchAction

このパラメータでは、NIS+ 照合処理のために、LDAP データを事前に取得するかどうかを決定します。ほとんどの場合、この値はデフォルトのままにしておきます。詳細については、rpc.nisd(4) のマニュアルページを参照してください。

/var/nis/NIS+LDAPmapping

この構成の名前は、rpc.nisd(1M) の -m オプションを使用して変更できます。NIS+LDAPmapping ファイルは、NIS+ および LDAP マッピングのマスタースイッチとして機能します。

マッピングファイルに対してデフォルト以外の名前を使用する場合は、/etc/init.d/rpc ブートスクリプトを編集して、rpc.nisd 起動行にそのマッピングファイル名を指定する必要があります。

LDAP に対応づける NIS+ オブジェクトごとに、NIS+LDAPmapping ファイルに 2 - 5 個の属性を指定します。指定する属性値は、オブジェクトとデフォルト値によって異なります。

nisplusLDAPdatabaseIdMapping

別名は、オブジェクトがほかのマッピング属性で使用されるときに設定する必要があります。NIS+ オブジェクト名が完全指定されていない場合 (ドットで終わっていない場合) は 、nisplusLDAPbaseDomain の値が付加されます。

たとえば、次のようになります。

```
nisplusLDAPdatabaseIdMapping    rpc:rpc.org_dir
```

このパラメータでは、データベース ID rpc を NIS+ rpc.org_dir テーブルの別名として定義しています。

NIS+ テーブルオブジェクトを 2 つの異なるデータベース ID ごとに 2 回定義する場合、テーブルオブジェクト自体 (このオブジェクトを LDAP に対応づける必要がある場合) として定義し、次にテーブルエントリとして定義します。たとえば、次のようになります。

```
nisplusLDAPdatabaseIdMapping    rpc_table:rpc.org_dir
nisplusLDAPdatabaseIdMapping    rpc:rpc.org_dir
```

まず、データベース ID rpc_table と rpc を、rpc.org_dir テーブルの別名として定義します。次に、rpc_table を rpc.org_dir テーブルオブジェクトに使用し、rpc をそのテーブルのエントリに使用することを定義します。

nisplusLDAPentryTtl

rpc.nisd デーモンのローカルデータベースは、メモリ内およびディスク上で LDAP データのキャッシュとして機能します。nisplusLDAPentryTtl 属性を使用すれば、そのキャッシュ内のエントリの生存期間 (TTL) 値を設定できます。各データベー

ス ID には、3 つの TTL があります。最初の 2 つの TTL は、rpc.nisd が、対応する NIS+ オブジェクトデータをディスクから最初に読み込むときの初期 TTL を制御します。3 番目の TTL は、NIS+ オブジェクトデータを LDAP から読み込んだとき (更新したとき) にオブジェクトに割り当てられます。

たとえば、次の場合、rpc.org_dir テーブルオブジェクトは、21600 - 43200 秒の範囲からランダムに選択された初期 TTL を取得します。

```
nisplusLDAPentryTtl     rpc_table:21600:43200:43200
```

初期 TTL の生存期間が切れると、テーブルオブジェクトが LDAP から再度読み込まれ、TTL が 43200 秒に設定されます。

同様に、次の場合は、テーブルオブジェクトが最初に読み込まれたときに、800 - 3600 秒から選択された初期 TTL が、 rpc.org_dir テーブルのエントリに割り当てられます。

```
nisplusLDAPentryTtl     rpc:1800:3600:3600
```

各エントリは、指定された範囲からランダムに選択された TTL を取得します。テーブルエントリが期間切れになり、更新されると、TTL は 3600 秒に設定されます。

TTL 値を選択するときは、パフォーマンスと整合性のバランスを考慮してください。rpc.nisd によって LDAP データがキャッシュされているときは、その TTL が大きい場合、rpc.nisd に LDAP データとの関連付けを設定していない場合とパフォーマンスは同じになります。しかし、rpc.nisd 以外のエンティティによって LDAP データが変更されると、変更が NIS+ に反映されるまでにかなりの時間がかかります。

逆に、小さな値 (またはゼロ) を TTL に設定すると、LDAP データに対する変更が NIS+ にすばやく反映されますが、パフォーマンスが低下する可能性があります。通常、NIS+ 上で LDAP データの読み込みまたは書き込みを行うときは、LDAP 通信を行わない場合と比較して、2 〜 3 倍の時間に加えて LDAP ルックアップの時間がかかります。パフォーマンスはハードウェアリソースによって大きく異なりますが、大きな LDAP コンテナ (数万 〜 数十万のエントリ) を走査して、更新が必要な NIS+ エントリを識別するのは、かなりの時間を必要とします。rpc.nisd デーモンは、この走査をバックグラウンドで実行して、走査の実行中も可能なデータを供給し続けます。しかし、バックグラウンドで走査している場合でも、NIS+ サーバーの CPU とメモリは消費されます。

NIS+ データと LDAP を同期化する重要性を十分に考慮して、適用可能な最も長い TTL を NIS+ オブジェクトごとに選択してください。デフォルト (nisplusLDAPentryTtl を指定しないとき) は 1 時間です。テンプレートマッピングファイル /var/nis/NIS+LDAPmapping.template を適用すると、テーブルエントリ以外のオブジェクトの TTL が 12 時間に変更されます。ただし、テーブルエントリ以外のオブジェクトは自動的に認識されないため、テーブルエントリ以外のオブジェクトのマッピングを追加すると、TTL はデフォルトの 1 時間に設定されます。

注 – 存在しないオブジェクトには 、TTL はありません。このため、NIS+ テーブル内で LDAP に対応づけられたエントリの TTL を有効にしても、NIS+ に存在しないエントリを要求すると、常に LDAP を照会してそのエントリを取得しようとします。

nisplusLDAPobjectDN

nisplusLDAPobjectDN には、対応づけられた NIS+ オブジェクトごとに、オブジェクトデータが常駐する LDAP DIT 内の対応する場所を設定します。また、LDAP エントリが削除されたときに実行する処理も指定できます。nisplusLDAPobjectDN 値は、3 つの部分から構成されます。最初の部分には、LDAP データの読み込み元を指定します。2 番目の部分には、LDAP データの書き込み先を指定します。3 番目の部分には、LDAP データが削除されたときの処理を指定します。次の例を参照してください。

```
nisplusLDAPobjectDN    rpc_table:\
                                    cn=rpc,ou=nisPlus,?base?\
                                              objectClass=nisplusObjectContainer:\
                                    cn=rpc,ou=nisPlus,?base?\
                                              objectClass=nisplusObjectContainer,\
                                                  objectClass=top
```

この例では、rpc.org_dir テーブルオブジェクトが DN cn=rpc,ou=nisPlus から読み込まれます。このとき、DN 値がコンマで終了しているため、defaultSearchBase 属性 (検索範囲) の値として base が付加されます。また、ObjectClass 属性の値が nisplusObjectContainer であるエントリが選択されます。

このテーブルオブジェクトは、読み込み元と同じ場所に書き込まれます。削除については指定されていないため、デフォルトの処理が実行されます。NIS+ テーブルオブジェクトが削除されると、LDAP エントリ全体も削除されます。

データを読み込むだけで LDAP に書き込まない場合は、書き込み部分を省略し、読み込み部分との区切り文字であるコロンも省略します。

```
nisplusLDAPobjectDN    rpc_table:\
                                    cn=rpc,ou=nisPlus,?base?\
                                    objectClass=nisplusObjectContainer
```

nisplusObjectContainer オブジェクトクラスは、RFC 2307 に準拠していません。 このオブジェクトクラスを使用するには、LDAP サーバーを 314 ページの「テーブルエントリ以外の NIS+ オブジェクトのマッピング」 で説明するように構成する必要があります。

rpc.org_dir テーブルエントリには、次の例も使用できます。

```
nisplusLDAPobjectDN rpc:ou=Rpc,?one?objectClass=oncRpc:\
                                        ou=Rpc,?one?objectClass=onRpc,objectClass=top
```

この例では、テーブルエントリの読み込みおよび書き込みが、ベース ou=Rpc に対して行われます。コンマで終了しているため、 defaultSearchBase 値が付加されます。objectClass 属性の値が oncRpc であるエントリを選択してください。LDAP の ou=Rpc コンテナ内にエントリを作成するときは、objectClass の値として top も指定する必要があります。

デフォルト以外の削除を指定する場合は、次の例を参照してください。

```
nisplusLDAPobjectDN    user_attr:\
                                    ou=People,?one?objectClass=SolarisUserAttr,\
                                        solarisAttrKeyValue=*:\
                                    ou=People,?one?objectClass=SolarisUserAttr:\
                                        dbid=user_attr_del
```

user_attr.org_dir データは、ou=People LDAP コンテナに存在します。このコンテナは、ほかのソース (passwd.org_dir NIS+ テーブルなど) のアカウント情報も入ります。

そのコンテナ内のエントリから、solarisAttrKeyValue 属性を持つものが選択されます。user_attr.org_dir データが、これらのエントリにだけ含まれるためです。nisplusLDAPobjectDN の dbid=user_attr_del 部分の定義によって、user_attr.org_dir NIS+ テーブル内のエントリが削除されると、対応する LDAP エントリが存在する場合は、データベース ID が user_attr_del のルールセットのルールに基づいて 削除されます。 詳細については、302 ページの「nisplusLDAPcolumnFromAttribute」 を参照してください。

nisplusLDAPattributeFromColumn

nisplusLDAPattributeFromColumn には、NIS+ データを LDAP に対応づけるときのルールを指定します。LDAP から NIS+ データへのマッピングルールは、nisplusLDAPcolumnFromAttribute に指定します。

nisplusLDAPcolumnFromAttribute

nisplusLDAPcolumnFromAttribute には、LDAP データを NIS+ に対応づけるときのルールを指定します。

エントリマッピングの完全な構文については、 NIS+LDAPmapping(4) のマニュアルページを参照してください。ここでは、わかりやすい例をいくつか挙げます。

NIS+ rpc.org_dir テーブルには、cname、name、numbe、および comment という列が含まれます。たとえば、NIS+ RPC プログラム番号 (100300) に対して、正規名として nisd が指定され、別名として rpc.nisd と nisplusd が指定されているとします。このエントリは、rpc.org_dir の次の NIS+ エントリを使用して表現できます。

```
nisd nisd 100300    NIS+ server
nisd rpc.nisd 100300    NIS+ server
nisd nisplusd 100300    NIS+ server
```

defaultSearchBase の値を dc=some,dc=domain と想定します。対応する LDAP は、ldapsearch(1) を実行すると次のように表示されます。

```
cn=nisd,ou=Ppc,dc=some,dc=domain
cn=nisd
cn=rpc.nsid
cn=nisplusd
oncrocnumber=100300
description=NIS+ server
objectclass=oncRpc
objectclass=top
```

この例は、NIS+ および LDAP が単純に 1 対 1 で対応づけられている場合です。この場合、NIS+ から LDAP へのマッピング属性値は、次のようになります。

```
nisplusLDAPattributeFromColumn \
rpc:        dn=("cn=%s,", name), \
                cn=cname, \
                cn=name, \
                oncRpcNumber=number, \
                description=comment
```

このエントリの DN として、cn=%s が構成されます。cname 列の値が %s に代入されます。

```
cn=nisd,
```

値がコンマで終了しているため、 nisplusObjectDN の読み込みベース値が付加され、次のようになります。

```
cn=nisd,ou=Rpc,dc=some,dc=domain
```

oncRpcNumber 属性および description 属性の値には、対応する NIS+ 列の値が代入されます。rpc.nisd によって、複数の NIS+ エントリが 1 つの LDAP エントリに収集され、異なる name 列値を表す複数の cn 値が生成されます。

同様に、LDAP から NIS+ へのマッピングは、次のようになります。

```
nisplusLDAPcolumnFromAttribute \
                    rpc:        cname=cn, \
                                    (name)=(cn), \
                                number=oncRpcNumber, \
                                comment=description
```

この例では、oncRpcNumber および description の値が、対応する NIS+ 列に割り当てられます。(cn) で示される複数値列 cn は、(name) で示される name 列値にマップされています。name 列は複数値列でないため、rpc.nisd によって cn 値ごとに 1 つの NIS+ エントリが作成されます。

最後に、削除に使用するルールセットの例を、nisplusLDAPattributeFromColumn 値を使って説明します。

```
nisplusLDAPattributeFromColumn \
user_attr_del:    dn=("uid=%s,", name), \
                      SolarisUserQualifier=, \
```

```
        SolarisAttrReserved1=, \
         SolarisAttrReserved2=, \
        SolarisAttrKeyValue=
```

すでに述べたように、user_attr.org_dir データは、 ほかのテーブル (passwd.org_dir など) のアカウント情報と、 ou=People コンテナを共有しています。user_attr.org_dir テーブルのエントリが削除されたときに、ou=People エントリ全体を削除したいとはおそらく考えないでしょう。この例ではエントリ全体を削除する代わりに、user_attr.org_dir エントリが削除されたときに、SolarisUserQualifier 、SolarisAttrReserved1、SolarisAttrReserved2 、および SolarisAttrKeyValue 属性が存在する場合、次のルールに指定されている ou=People エントリから削除されます。

```
dn=("uid=%s,", name)
```

これ以外の LDAP エントリは変更されません。

NIS+ から LDAP への移行シナリオ

NIS+ から LDAP への移行シナリオの例を挙げます。

- すべての *NIS+* クライアントを *1* 回の操作で *LDAP* に変換する場合。rpc.nisd デーモンを使用すれば、LDAP に存在しないすべての NIS+ データをアップロードできます。305 ページの「すべての NIS+ データを 1 回の操作で LDAP に変換する方法」 を参照してください。
- *NIS+* から *LDAP* に段階的に移行する場合。まず、NIS+ データを LDAP に変換します (305 ページの「すべての NIS+ データを 1 回の操作で LDAP に変換する方法」 を参照)。 NIS+ クライアントと LDAP クライアントで、同じネームサービスを共有することができます。NIS+ および LDAP のデータは、rpc.nisd によって自動的に同期化されます。移行の初期段階では場合によって、NIS+ が認証されたサーバーとして機能し、LDAP サーバーは、NIS+ データを複製して、LDAP クライアントに提供します。適切な段階で、LDAP を認証されたネームサービスに移行します。NIS+ サービスは、段階的に処理を停止していき、NIS+ クライアントの移行が完了した時点で完全になくなります。
- LDAP がすでにネームサービスとして使用されている場合。NIS+ データと LDAP データをマージする必要があります。次の 3 つのマージ方法があります。
  - *NIS+* データを *LDAP* に追加する方法。NIS+ に存在するが LDAP に存在しないエントリが、LDAP に追加されます。エントリが NIS+ および LDAP の両方に存在するが、データが異なる場合は、NIS+ データが優先されます。305 ページの「すべての NIS+ データを 1 回の操作で LDAP に変換する方法」 を参照してください。
  - *NIS+* データを *LDAP* データで上書きする方法。NIS+ に存在するが LDAP に存在しないエントリが、NIS+ から削除されます。 NIS+ および LDAP の両方に存在するエントリでは、 LDAP データが優先されます。305 ページの「すべての LDAP データを 1 回の操作で NIS+ に変換する方法」 を参照してください。

- *NIS+* データと *LDAP* データをマージする方法。衝突が発生した場合は、個別に解決します。306 ページの「NIS+ データと LDAP データのマージ」 を参照してください。

▼ すべての NIS+ データを 1 回の操作で LDAP に変換する方法

- `rpc.nisd` を使用して、**LDAP** に存在しない **NIS+** データをすべてアップロードします。

  NIS+ と LDAP のすべてのデータマッピングが、デフォルトの場所 (`/var/nis/NIS+LDAPmapping`) に設定されている場合は、次のコマンドを使用します。

  ```
  # /usr/sbin/rpc.nisd -D \
  –x nisplusLDAPinitialUpdateAction=to_ldap \
  -x nisplusLDAPinitialUpdateOnly=yes
  ```

  上記のコマンドによって、`rpc.nisd` デーモンによりデータが LDAP にアップロードされて、変換が終了します。この処理を実行しても、NIS+ データは変更されません。

  `rpc.nisd(4)` のマニュアルページの `nisplusLDAPinitialUpdateAction` 属性を参照してください。

▼ すべての LDAP データを 1 回の操作で NIS+ に変換する方法

- `rpc.nisd` を使用して、すべての **LDAP** データを **NIS+** にダウンロードし、既存の **NIS+** データを上書きします。

  NIS+ と LDAP のすべてのデータマッピングが、デフォルトの場所 (`/var/nis/NIS+LDAPmapping`) に設定されている場合は、次のコマンドを使用します。

  ```
  # /usr/sbin/rpc.nisd -D \
  -x nisplusLDAPinitialUpdateAction=from_ldap \
  -x nisplusLDAPinitialUpdateOnly=yes
  ```

  上記のコマンドによって、`rpc.nisd` デーモンによりデータが LDAP からダウンロードされて、変換が終了します。この処理を実行しても、LDAP データは変更されません。

  `rpc.nisd(4)` のマニュアルページの `nisplusLDAPinitialUpdateAction` 属性を参照してください。

NIS+ データと LDAP データのマージ

NIS+ および LDAP 間でデータの衝突が発生したときは、NIS+ または LDAP データのどちらかを正式なものとして解決しなければなりません。304 ページの「NIS+ から LDAP への移行シナリオ」 では、NIS+ データと LDAP データを同期化する方法について説明しています。データをマージするときは、ほかの方法と比べて複雑な手順が必要になります。

この節で挙げた手順では、次の点を前提としています。

- NIS+ データのバックアップを `/nisbackup` ディレクトリに作成する
- 有効なマッピング構成が `/etc/default/rpc.nisd` および `/var/nis/tmpmap` (テーブルをマージする場合) にすでに存在する
- マージ前の NIS+ データのフラットファイル表現を `/before` に格納し、マージ後は `/after` に格納する。
- `niscat` は、`nisaddent(1M)` でサポートされないカスタム NIS+ テーブルを、フラットファイル表現でダンプするときに使用する。このようなカスタムテーブルを、NIS+ からまたは NIS+ にダンプして読み込むために、独自のコマンドまたはスクリプトを作成することがある。この場合は、`niscat` に優先して、独自のコマンドまたはスクリプトを使用する。`niscat` コマンドには、フラットファイル表現のデータを NIS+ に戻す便利な方法がないためである

  `niscat(1)` を使用してデータをダンプした場合は、`nistbladm(1)` を使用すれば、エントリ単位に NIS+ に戻すことができる
- コマンドパスに `/usr/lib/nis` (`nisaddent(1M)` が存在する場所) を含める

▼ NIS+ データと LDAP データをマージする方法

警告 – 手順 4 のダウンロードデータと、手順 10 のアップロードデータが一致しない場合は、アップロードデータによって変更が上書きされます。このため、この手順を実行しているときは、LDAP データの変更はできるだけ避けてください。詳細については、LDAP サーバーのマニュアルを参照してください。

1. `nisbackup` コマンドを使用して、すべての **NIS+** データをバックアップします。

   # nisbackup -a /nisbackup

2. **LDAP** とマージするデータが含まれる **NIS+** テーブルを特定します。これらのテーブルの内容をフラットファイルにダンプします。たとえば、次のように `nisaddent` を使用して、`group.org_dir` の内容をダンプします。

   # nisaddent -d group | sort> /before/group

   パイプを使って `nisaddent` の出力を `sort` の入力として渡すと、比較処理が簡単になります。

3. `rpc.nisd` デーモンを停止します。

```
# pkill rpc.nisd
```

4. **LDAP** データを **NIS+** にダウンロードします。

```
# /usr/sbin/rpc.nisd -D -m tmpmap \
-x nisplusLDAPinitialUpdateAction=from_ldap \
-x nisplusLDAPinitialUpdateOnly=yes
```

5. rpc.nisd デーモンを起動します。

```
# /usr/sbin/rpc.nisd
```

rpc.nisd デーモンが、LDAP からダウンロードしたデータを提供するようになります。解決を必要とする衝突を NIS+ クライアント上で発生させないようにする必要がある場合は、これ以降の手順は、ほとんどまたはすべての NIS+ クライアントが動作していないときに実行してください。

6. 影響を受けるテーブルの **NIS+** データをダンプします。

次の例では、 group.org_dir テーブルをダンプします。

```
# nisaddent -d group | sort> /after/group
```

7. 任意のファイルマージ手順を使用して、マージしたテーブルを作成します。diff(1) 以外のツールを使用できない場合は、このコマンドを使用して /before ファイルと /after ファイルとの相違点を収集し、テキストエディタを使用して手動でマージすることができます。

次の例では、マージ後のテーブルが /after に格納されていることを前提としています。

8. マージ後のデータを **NIS+** に読み込みます。次の例では、 group テーブルを読み込みます。

```
# nisaddent -m -f /after/group group
```

9. マージ後のテーブルから、不要な **LDAP** エントリを削除します。

A . マージ後の NIS+ データ内に存在しない LDAP エントリが、アップロード後の LDAP に必要がない場合、これらの LDAP エントリは削除する必要があります。

LDAP サーバーには、コンテナのすべてのエントリを削除する方法など、複数のエントリを削除する便利な方法が提供されていることがあります。提供されていない場合は、ldapsearch(1) を使用して、各コンテナのエントリの一覧を生成することができます。たとえば、ou=Rpc コンテナに含まれるすべてのエントリの一覧を生成するには、ldapsearch(1) を次のように使用します。

```
# ldapsearch -h server-address -D bind-DN -w password \
-b ou=Rpc,search-base 'objectClass=*' dn | \
grep -i ou=Rpc | grep -v -i \^ou=Rpc> \
/tmp/delete-dn
```

メタ引数 (*server-address*、 *bind-DN* など) については、313 ページの「パフォーマンスとインデックス処理」 を参照してください。

B. 結果ファイル (/tmp/delete-dn) を編集して、削除するエントリだけを指定します。コンテナのすべてのエントリを削除する場合は、該当するファイルは操作しない

で、NIS+ アップロードを使用して LDAP データを復元することもできます。どちらの方法を使用する場合でも、LDAP データをバックアップしてから、次の `ldapdelete` 操作を実行してください。

C. `ldapdelete` を使用して、LDAP エントリを削除します。`stdout` (通常は、削除したエントリごとに空白行が 1 行ずつ出力される) は、`/dev/null` にリダイレクトします。

```
# ldapdelete –h server-address –D bind-DN –w password \
/tmp/delete-dn /dev/null
```

D. 削除するエントリが 1 つ以上含まれるコンテナごとに、前述の手順を繰り返します。

10. これで **NIS+** には、マージされたデータが生成されます。このデータは、**LDAP** にアップロードできます。 次の操作を実行します。

`rpc.nisd` デーモンを停止します。

```
# pkill rpc.nisd
```

アップロードを実行します。

```
# /usr/sbin/rpc.nisd -D -m tmpmap \
-x nisplusLDAPinitialUpdateAction=to_ldap \
-x nisplusLDAPinitialUpdateOnly=yes
```

11. `rpc.nisd` デーモンを再起動します。

- `rpc.nisd` デーモンが LDAP リポジトリを使用する場合は、適切なマッピングファイルを指定します。
- `rpc.nisd` デーモンを NIS (YP) エミュレーションに対応させる場合は、`-Y` オプションを指定します。

```
# /usr/sbin/rpc.nisd -m mappingfile [ -Y ]
```

手順 10 のアップロードコマンドから、`-x nisplusLDAPinitialUpdateOnly=yes` を省略することもできます。省略した場合、アップロードが完了すると、`rpc.nisd` デーモンは NIS+ データの提供を開始します。

マスターと複製

NIS+ マスターだけが、データを LDAP に書き込むことができます。NIS+ 複製は、NIS+ マスターから更新を取得するか (LDAP から取得する場合を含む) 、LDAP サーバーから直接データを読み込みます。この 2 つの方法を組み合わせることもできます。つまり、NIS+ 複製を使用するときは、主に 2 つの方法があります。

- *NIS+* 複製は変更せずに使用し、更新データは *NIS+* マスターから取得する

  この方法の場合は、NIS+ マスター以外は、LDAP サーバーと接続する必要がないため、構成が単純になります。従来の複製関係も変更されません。つまり、 新しいデータはまずマスターに反映され、次に複製に反映されます。ネームサービスの

データを従来どおり NIS+ が管理するときは、ほとんどの場合、この方法が最も便利な方法です。ただし、LDAP と NIS+ 複製サーバーとの間のパスが長くなります。

- *NIS+ 複製は、更新データを NIS+ マスターから取得しないで、LDAP から直接取得する*

  この場合、複製のデータの更新は、LDAP から取得したデータのルックアップトラフィックおよび TTL に基づいて、NIS+ マスターの更新前または更新後に行われます。この方法はより複雑ですが、LDAP がネームサービスリポジトリを管理するときは、便利な方法です。NIS+ データに対する直接の更新は、ほとんどまたはまったく発生しません。

複製タイムスタンプ

NIS+ 複製が特定の NIS+ ディレクトリに含まれる 1 つ以上のオブジェクトのデータを LDAP から取得しているときは、`nisping(1M)` によって出力される更新タイムスタンプが NIS+ マスターおよび NIS+ 複製間のデータの整合性を示しているとは限りません。たとえば、NIS+ ディレクトリ `dir1` に `table1` および `table2` テーブルが含まれているとします。 複製が `table1` および `table2` のデータを NIS+ マスターから取得しているときは、次のようなタイムスタンプが出力されます。

 # nisping dir1

```
Master server is "master.some.domain."
Last update occurred at Mon Aug  5 22:11:09 2002

Replica server is "replica.some.domain."
    Last Update seen was Mon Aug  5 22:11:09 2002
```

これらのタイムスタンプは、マスターおよび複製のデータが完全に一致していることを示しています。しかし、複製が `table1` または `table2`、あるいはその両方のデータを LDAP から取得しているとします。この場合、この出力には、複製がマスターから NIS_PING を受け取り、再同期化のタイムスタンプをシステム管理用に更新したことだけが示されます。LDAP に対応づけられているテーブルのデータは、次のどちらかの場合、NIS+ マスター上のデータと異なることがあります。

- LDAPデータが NIS+ マスター上のデータと異なる
- 期限切れではないが、LDAP と同期がとれていない複製のキャッシュに NIS+ データベースのデータが格納されている

このようなデータの不一致を許容できない場合は、NIS+ 複製が常に NIS+ マスターからデータを取得するようにします。NIS+ マスターが常に LDAP からデータを取得するように構成した場合は、複製を変更する必要はありません。

ディレクトリサーバー

rpc.nisd デーモンの LDAP マッピングでは、LDAP プロトコルバージョン 3 を使用して LDAP サーバーと対話します。 デフォルトのマッピング構成 (/var/nis/NIS+LDAPmapping.template) では、LDAP サーバーが RFC 2307 の拡張版 に準拠していることを前提としています。RFC は、http://www.ietf.org/rfc.html から入手できます。 NIS+ データと LDAP データとのマッピングは、NIS+LDAPmapping(4) を使用して変更できます。ただし、LDAP のデータ編成が RFC 2307 の規定に準拠していることを、基本的な前提としています。

たとえば、LDAP クライアントと NIS+ クライアントとの間でアカウント情報を共有するには、UNIX crypt 書式のアカウント (ユーザー) パスワードを LDAP サーバーに格納できるようにする必要があります。LDAP サーバーをこのように構成できない場合でも、アカウントを含む NIS+ データを LDAP に格納することはできます。 その場合、NIS+ ユーザーと LDAP bindDN との間でアカウント情報を完全に共有することはできません。

iPlanet Directory Server 5.1 の構成

iPlanet Directory Server 5.1 のインストール、設定、および管理の詳細については、「iPlanet™ Directory Server 5.1 Collection」を参照してください。

iPlanet Directory Server バージョン 5.1 を構成して、LDAP クライアントが LDAP をネームサービスとして使用できる ようにするには、idsconfig(1M) を使用します。idsconfig(1M) を使用して設定した構成は、NIS+ で LDAP データリポジトリを使用する場合にも適しています。

注 – iPlanet Directory Server 5.1 以外の LDAP サーバーを使用している場合は、RFC 2307 に準拠するように、サーバーを手動で構成する必要があります。

---

idsconfig の詳細については、マニュアルページを参照してください。

サーバーアドレスとポート番号の割り当て

/etc/default/rpc.nisd ファイルは、ローカル LDAP サーバーをポート 389 で使用するように設定されています。この設定が現在の構成に適していない場合は、preferredServerList 属性に新しい値を設定します。たとえば、LDAP サーバーを IP アドレス 192.0.0.1 とポート 65535 で使用するには、次のように指定します。

```
preferredServerList=192.0.0.1:65535
```

セキュリティと認証

NIS+ クライアントおよび NIS+ サーバー間の認証は、NIS+ サーバーが LDAP からデータを取得する場合でも、影響することはありません。ただし、NIS+ データを LDAP に格納するときの整合性を保持するには、`rpc.nisd` デーモンおよび LDAP サーバー間の認証を必要に応じて設定する必要があります。LDAP サーバーの機能に応じて、さまざまなタイプの認証を利用できます。

`rpc.nisd` デーモンでは、次の LDAP 認証を利用できます。

- `none`

  `none` は、デフォルトの認証方式です。`none` には、固有の設定は必要ありません。ただし、セキュリティは保証されません。セキュリティを考慮する必要がない環境だけで使用してください。

  `none` 認証を使用するときは、 `authenticationMethod` 属性に次の値を設定してください。

  ```
  authenticationMethod=none
  ```

この認証方式を利用するときに一定のセキュリティを保証するには、多くの場合、共有された機密情報 (パスワードまたは鍵) と LDAP の DN を関連付ける必要があります。`rpc.nisd` デーモンで使用する DN は一意なものであり、ほかの目的で使用することもできます。予測される LDAP トラフィックに対応するために、DN には適切な権限を割り当てる必要があります。たとえば、`rpc.nisd` デーモンが LDAP にデータを書き込む場合は、NIS+ データに使用されるコンテナ内で LDAP データを追加、更新、および削除する権限を、選択した DN に割り当てる必要があります。また、LDAP サーバーでは、リソースの使用方法がデフォルトで制限されている場合があります (検索時間制限、検索結果のサイズ制限など) 。この制限がある場合は、必要な数の NIS+ データコンテナをサポートできるように、選択した DN に対して必要な設定をする必要があります。

- `simple`

  `simple` 認証方式では、暗号化されていないパスワード文字列が交換されます。パスワードは、LDAP クライアント (`rpc.nisd` デーモン) および LDAP サーバー間をプレーンテキストとして送信されます。このため、`simple` 方式は、NIS+ と LDAP サーバー間の情報交換が別の方式で保護されている場合にだけ使用してください。

  たとえば、LDAP トラフィックのトランポート層を暗号化するときに使用します。また、NIS+ サーバーと LDAP サーバーが同一システム上にあり、NIS+ および LDAP のトラフィックがカーネル内で処理され、認証されていないユーザーから保護されている場合にも使用できます。

  `simple` 認証を使用するときは、`rpc.nisd` デーモンで使用する DN と パスワードの構成を変更してください。たとえば、DN が `cn=nisplusAdmin`、`ou=People`、`dc=some`、`dc=domain` で、パスワードが `aword` の場合は、次のように設定します。

```
authenticationMethod=simple
nisplusLDAPproxyUser=cn=nisplusAdmin,ou=People,dc=some,dc=domain
nisplusLDAPproxyPassword=aword
```

パスワードが格納されている場所は、認証されないアクセスから確実に保護する必要があります。パスワードを rpc.nisd コマンド行で指定した場合は、ps(1) などのコマンドを使ってシステム上の任意のユーザーに見られる可能性があります。

- sasl/digest-md5

  sasl/digest-md5 認証方式では、digest/md5 アルゴリズムを使用して認証が行われます。

  digest-md5 で使用する認証 ID を設定する方法と、/etc/default/rpc.nisd ファイルに認証 ID とその パスワードを指定する方法については、LDAP サーバーのマニュアルを参照してください。

```
authenticationMethod=sasl/digest-md5
nisplusLDAPproxyUser=cn=nisplusAdmin,ou=People,dc=some,dc=domain
nisplusLDAPproxyPassword=aword
```

パスワードが格納されているファイルは、認証されないアクセスから確実に保護する必要があります。

- sasl/cram-md5

  cram/md5 アルゴリズムを使用した認証方式。通常は、現在使用されていない SunDS LDAP サーバー以外では使用されません。

  cram-md5 を使用してバインド DN を設定する方法と、/etc/default/rpc.nisd ファイルにバインド DN とそのパスワードを指定する方法については、LDAP サーバーのマニュアルを参照してください。

```
authenticationMethod=sasl/cram-md5
nisplusLDAPproxyUser=cn=nisplusAdmin,ou=People,dc=some,dc=domain
nisplusLDAPproxyPassword=aword
```

承認されていないアクセスからパスワードを格納するファイルを確実に保護してください。

SSL の使用

rpc.nisd デーモンは、SSL を使用した LDAP トラフィックのトランスポート層の暗号化にも対応しています。LDAP サーバー認証用の SSL 証明書の生成については、LDAP サーバーのマニュアルを参照してください。SSL 証明書は、NIS+ サーバー上のファイル (/var/nis/cert7.db など) に格納します。/etc/default/rpc.nisd は、次のように変更します。

```
nisplusLDAPTLS=ssl
nisplusLDAPTLSCertificateDBPath=/var/nis/cert7.db
```

SSL 証明書は、承認されていないアクセスから確実に保護する必要があります。この例では、セッションの暗号化と LDAP サーバーの認証が rpc.nisd に提供されます。 SSL 証明書では、LDAP サーバーに対する rpc.nisd の認証は提供されませ

ん。この証明書には、この LDAP クライアント (rpc.nisd) の識別情報が含まれていないためです。ただし、rpc.nisd と LDAP サーバーが相互に認証するには、SSL と別の認証方式 (simple、sasl/digest-md5) を組み合わせることができます。

LDAP のセキュリティの詳細については、220 ページの「LDAP ネームサービスのセキュリティモデル」を参照してください。

パフォーマンスとインデックス処理

niscat(1) などを使用して、LDAP に対応づけられた NIS+ テーブルの列挙を rpc.nisd デーモンに 要求すると、テーブル内のエントリの TTL が 1 つでも期限切れになっている場合は、対応する LDAP コンテナが列挙されます。コンテナの列挙はバックグラウンドで実行されるため、LDAP のパフォーマンスはそれほど重要ではありません。ただし、LDAP にインデックスを設定すれば、コンテナが大きい場合でもすばやく列挙することができます。

特定のコンテナの列挙に必要な時間を見積もるには、次のようなコマンドを使用します。

% **/bin/time ldapsearch -h** *server-address* **-D** *bind-DN* **-w** *password* ****

-b *container*, *search-base 'cn=*'* **/dev/null**

ここで

- *server-address*

  /etc/default/rpc.nisd の preferredServerList 値の IP アドレス部分
- *bind-DN*

  /etc/default/rpc.nisd の nisplusLDAPproxyUser 値
- *password*

  /etc/default/rpc.nisd の nisplusLDAPproxyPassword 値
- *container*

  RFC 2307 に準拠したコンテナ名 (ou=Services、 ou=Rpc など)
- *search-base*

  /etc/default/rpc.nisd の defaultSearchBase 値

/bin/time から出力される実際の値は、経過時間です。この値が、対応するテーブルエントリの TTL を 25 パーセント以上占めている場合は (297 ページの「認証とセキュリティ」 を参照)、LDAP コンテナにインデックスを設定すると有効です。

rpc.nisd では、simple page と VLV インデックス方式がサポートされます。ご使用の LDAP サーバーでサポートされているインデックス方式、およびそのインデックスの作成方法については、LDAP サーバーのマニュアルを参照してください。

テーブルエントリ以外の NIS+ オブジェクトのマッピング

テーブルエントリ以外の NIS+ オブジェクトを LDAP に格納できます。ただし、NIS+ 複製が LDAP からこれらの NIS+ オブジェクトを取得しない限り、LDAP に格納しても値は設定されません。次の方法をお勧めします。

- 複製がない場合、または複製が *NIS+* オブジェクトを *NIS+* マスターだけから取得する場合。

  マッピング構成ファイル (`NIS+LDAPmapping(4)` のマニュアルページを参照) を編集して、テーブルエントリ以外のオブジェクトから次の属性値を削除します。

```
nisplusLDAPdatabaseIdMapping
nisplusLDAPentryTtl
nisplusLDAPobjectDN
```

たとえば、`/var/nis/NIS+LDAPmapping.template` ファイルの場合は、次のセクションを削除するか、コメントにして無効にします。

```
# Standard NIS+ directories
nisplusLDAPdatabaseIdMapping    basedir:
.
.
.

nisplusLDAPdatabaseIdMapping    user_attr_table:user_attr.org_dir

nisplusLDAPdatabaseIdMapping     audit_user_table:audit_user.org_dir

# Standard NIS+ directories
nisplusLDAPentryTtl             basedir:21600:43200:43200
.
.
.

nisplusLDAPentryTtl    user_attr_table:21600:43200:43200
nisplusLDAPentryTtl    audit_user_table:21600:43200:43200

# Standard NIS+ directories
nisplusLDAPobjectDN    basedir:cn=basedir,ou=nisPlus,?base?\

                    objectClass=nisplusObjectContainer:\
                                cn=basedir,ou=nisPlus,?base?\
                                    objectClass=nisplusObjectContainer,\
                                    objectClass=top
.
.
.

nisplusLDAPobjectDN    audit_user_table:cn=audit_user,ou=nisPlus,?base?\
                                objectClass=nisplusObjectContainer:\
```

```
                        cn=audit_user,ou=nisPlus,?base?\
                            objectClass=nisplusObjectContainer,\
                            objectClass=top
```

- *NIS+* 複製が *NIS+* オブジェクトを *LDAP* サーバーから取得する場合。

  nisplusObject の属性と nisplusObjectContainer のオブジェクトクラスを以下の例にしたがって作成します。LDIF データは ldapadd(1) に適しています。属性とオブジェクトクラス OID は、例としてあげているだけです。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( 1.3.6.1.4.1.42.2.27.5.42.42.1.0 NAME 'nisplusObject' \
                                DESC 'An opaque representation of an NIS+ object' \
                                SYNTAX 1.3.6.1.4.1.1466.115.121.1.5 SINGLE-VALUE )

dn: cn=schema
changetype: modify
add: objectclasses

objectclasses(1.3.6.1.4.1.42.2.27.5.42.42.2.0 NAME'nisplusObjectContainer'\

SUP top STRUCTURAL DESC 'Abstraction of an NIS+ object
MUST ( cn $ nisplusObject ) )
```

NIS+ オブジェクトのコンテナも作成する必要があります。次の LDIF 構文は、ou=nisPlus,dc=some,dc=domain コンテナの作成方法を示しています。このコンテナは、ldapadd(1) の入力として使用できます。

```
dn: ou=nisPlus,dc=some,dc=domain
ou: nisPlus
objectClass: top
objectClass: organizationalUnit
```

NIS+ エントリの所有者、グループ、アクセス権、および TTL

NIS+ テーブルエントリを LDAP データから作成するときは、そのエントリオブジェクトが存在するテーブルオブジェクトの対応する値を使用して、所有者、グループ、アクセス権、および TTL を初期化する必要があります。環境によっては、これらの NIS+ エントリ属性を個別に設定する必要があります。たとえば、rpc.nispasswdd (1M) デーモンを使用しない環境では、この操作が必要になります。ユーザー自身が NIS+ パスワードを変更して、cred.org_dir テーブルに格納されている Diffie-Hellman キーを再暗号化できるようにするには、passwd.org_dir および cred.org_dir エントリの所有者をそのユーザーに設定し、その所有者に変更権限を割り当てる必要があります。

1 つ以上の NIS+ テーブルエントリの所有者、グループ、アクセス権、または TTL を LDAP に格納するには、次の操作を実行する必要があります。

▼ エントリ属性を LDAP に追加するには

1. **LDAP** サーバーのマニュアルを参照して、次の新しい属性とオブジェクトクラスを作成します。**LDIF** データは、`ldapadd` に適用できます。属性とオブジェクトクラス **OID** は、例として挙げています。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.4.0 NAME 'nisplusEntryOwner' \
                                DESC 'Opaque representation of NIS+ entry owner' \
                                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.4.1 NAME 'nisplusEntryGroup' \
                                DESC 'Opaque representation of NIS+ entry group' \
                                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.4.2 NAME 'nisplusEntryAccess' \
                                DESC 'Opaque representation of NIS+ entry access' \
                                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.4.3 NAME 'nisplusEntryTtl' \
                                DESC 'Opaque representation of NIS+ entry TTL' \
                                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

dn: cn=schema
changetype: modify
add: objectclasses

objectclasses:(1.3.6.1.4.1.42.2.27.5.42.42.5.0 NAME 'nisplusEntryData'\
SUP top STRUCTURAL DESC 'NIS+ entry object non-column data'\

MUST ( cn ) MAY ( nisplusEntryOwner $ nisplusEntryGroup $\
nisplusEntryAccess $ nisplusEntryTtl ) )
```

2. 関連するテーブルの `nisplusLDAPobjectDN` 属性値を変更して、新しく作成した `nisplusEntryData` オブジェクトクラスを書き込み部分に含めます。

   たとえば、`passwd.org_dir` テーブルの場合、`/var/nis/NIS+LDAPmapping.template` をベースにしたテンプレートファイルを使用しているときは、次のように編集します。

```
nisplusLDAPobjectDN    passwd:ou=People,?one?objectClass=shadowAccount,\
                                        objectClass=posixAccount:\
                                ou=People,?one?objectClass=shadowAccount,\
                                        objectClass=posixAccount,\
                                        objectClass=account,objectClass=top
```

   属性値を次のように編集します。

```
nisplusLDAPobjectDN    passwd:ou=People,?one?objectClass=shadowAccount,\
                                        objectClass=posixAccount:\
                                ou=People,?one?objectClass=shadowAccount,\
                                        objectClass=posixAccount,\
```

```
                                        objectClass=nisplusEntryData,\
                                        objectClass=account,objectClass=top
```

3. nisplusLDAPattributeFromColumn 属性値および nisplusLDAPcolumnFromAttribute 属性値を編集して、所有者、グループ、アクセス権、または **TTL** を必要に応じて指定します。

   手順 2 で、これらの値を格納する LDAP 属性を作成しました。NIS+ には、zo_owner 、zo_group、zo_access、および zo_ttl と呼ばれる定義済みの擬似列名が、あらかじめ定義されています。たとえば、passwd.org_dir エントリの所有者、グループ、およびアクセス権を LDAP に格納するには、次の nisplusLDAPattributeFromColumn 値を変更します。

```
nisplusLDAPattributeFromColumn \
        passwd:         dn=("uid=%s,", name), \
                cn=name, \
                uid=name, \
                userPassword=("{crypt$}%s", passwd), \
                uidNumber=uid, \
                gidNumber=gid, \
                gecos=gcos, \
                homeDirectory=home, \
                loginShell=shell, \
                (shadowLastChange,shadowMin,shadowMax, \
                 shadowWarning, shadowInactive,shadowExpire)=\
                    (shadow, ":")
```

   次のように編集します。

```
nisplusLDAPattributeFromColumn \
        passwd:         dn=("uid=%s,", name), \
                cn=name, \
                uid=name, \
                userPassword=("{crypt$}%s", passwd), \
                uidNumber=uid, \
                gidNumber=gid, \
                gecos=gcos, \
                homeDirectory=home, \
                loginShell=shell, \
                (shadowLastChange,shadowMin,shadowMax, \
                 shadowWarning, shadowInactive,shadowExpire)=\
                    (shadow, ":"), \
                nisplusEntryOwner=zo_owner, \
                nisplusEntryGroup=zo_group, \
                nisplusEntryAccess=zo_access
```

   同様に、 NIS+ エントリの所有者、グループ、LDAP データからのアクセス権を passwd.org_dir テーブルに設定するには、次の値を変更します。

```
nisplusLDAPcolumnFromAttribute \
        passwd:         name=uid, \
                ("{crypt$}%s", passwd)=userPassword, \
                uid=uidNumber, \
                gid=gidNumber, \
                gcos=gecos, \
                home=homeDirectory, \
```

```
                shell=loginShell, \
                shadow=("%s:%s:%s:%s:%s:%s", \
                    shadowLastChange, \
                    shadowMin, \
                    shadowMax, \
                    shadowWarning, \
                    shadowInactive, \
                    shadowExpire)
```

次のように編集します。

```
nisplusLDAPcolumnFromAttribute \
        passwd:         name=uid, \
                ("crypt$%s", passwd)=authPassword, \
                uid=uidNumber, \
                gid=gidNumber, \
                gcos=gecos, \
                home=homeDirectory, \
                shell=loginShell, \
                shadow=("%s:%s:%s:%s:%s:%s", \
                    shadowLastChange, \
                    shadowMin, \
                    shadowMax, \
                    shadowWarning, \
                    shadowInactive, \
                    shadowExpire), \
                zo_owner=nisplusEntryOwner, \
                zo_group=nisplusEntryGroup, \
                zo_access=nisplusEntryAccess
```

4. マッピングの変更を有効にするために、`rpc.nisd` デーモンを再起動します。

   まず、所有者、グループ、アクセス権、および TTL エントリデータを LDAP にアップロードする必要があります。アップロード方法については、305 ページの「すべての NIS+ データを 1 回の操作で LDAP に変換する方法」 を参照してください。

主体名とネット名

NIS+ 認証は、主体名 (ドメイン名で修飾されたユーザー名またはホスト名) とネット名 (SecureRPC での主体名) に基づいて認証可能なエンティティ (主体) を一意に識別します。RFC 2307 では、NIS+ 認証に使用する Diffie-Hellman 鍵の格納場所は規定していますが、主体名またはネット名の格納場所は規定していません。

`/var/nis/NIS+LDAPmapping.template` ファイルでは、この問題を回避するために、`cred.org_dir` テーブルの所有者名 (主体名) から主体名およびネット名のドメイン部分を派生します。つまり、NIS+ ドメインが `x.y.z.` で、`cred.org_dir` テーブルの所有者が `aaa.x.y.z.` の場合、LDAP データから作成された NIS+ エントリの主体名は、次の形式になります。

user or system **`.x.y.z.`**

ネット名は次の形式になります。

unix.*uid***@x.y.z.**

unix.*nodename***@x.y.z.**

ほとんどの NIS+ インストールでは、主体名とネット名を作成するときは、この方式でかまいません。ただし、次のような場合は、この方式では成功しません。

- cred.org_dir テーブルの所有者名が属しているドメインが、cred.org_dir テーブル内の主体名およびネット名によって共有されるドメインと一致していない場合。所有者が親ドメインの主体である場合は、サブドメインの cred.org_dir テーブルも同様です。この問題は、次のいずれかの方法で解決できます。
  - cred.org_dir テーブルの所有者を変更して、テーブルエントリのドメインと一致させます。
  - cred.org_dir データベース ID のマッピングルールを変更して、ほかの NIS+ オブジェクトの所有者を使用します。適切なオブジェクトが存在しない場合は、この目的のために NIS+ オブジェクトを作成します。

    たとえば、 sub.dom.ain ドメイン内の cred.org_dir テーブルを master.dom.ain. が所有し、 cred.org_dir.sub.dom.ain. の主体とネット名が sub.dom.ain に属している場合は、次のようなリンクオブジェクトを作成します。

    # **nisln cred.org_dir.sub.dom.ain. **

    credname.sub.dom.ain.

    リンクオブジェクトの所有者を、次のように sub.dom.ain. 内の適切な主体に設定します。

    # **nischown trusted.sub.dom.ain. credname.sub.dom.ain.**

    マッピングファイルを編集します。次の箇所を

```
(nis+:zo_owner[]cred.org_dir, "*.%s")), \
```

    to

```
(nis+:zo_owner[]credname.sub.dom.ain., "*.%s")), \
```

    ここで使用しているリンクオブジェクト credname は、例として挙げています。エントリオブジェクト以外の、任意の有効なオブジェクトタイプとオブジェクト名が使用できます。オブジェクトの所有者に正しいドメイン名を設定することが重要です。
  - 主体およびネット名に使用されるドメインの主体に対して、特別な目的のオブジェクトであってもその所有権を与えたくない場合は、次に示すように nisplusPrincipalName 属性と nisplusNetname 属性を作成します。
- cred.org_dir テーブルに、複数のドメインに属している主体とネット名が設定されている場合。

LDAP サーバーのマニュアルを参照して、 nisplusPrincipalName 属性および nisplusNetname 属性と、nisplusAuthName オブジェクトクラスを作成します。以下のデータは ldapadd への LDIF データになっています。属性とオブジェクトクラス OID は、例として挙げています。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.7.0 NAME 'nisplusPrincipalName' \
          DESC 'NIS+ principal name' \
          SINGLE-VALUE \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.9.0 NAME 'nisplusNetname' \
          DESC 'Secure RPC netname' \
          SINGLE-VALUE \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

dn: cn=schema
changetype: modify
add: objectclasses
objectclasses:    ( 1.3.6.1.4.1.42.2.27.5.42.42.10.0 NAME 'nisplusAuthName' \
          SUP top AUXILLIARY DESC 'NIS+ authentication identifiers' \
          MAY ( nisplusPrincipalName $ nisplusNetname ) )
```

cred.org_dir マッピングで、新しく作成した nisplusNetname 属性および nisplusPrincipalName 属性を使用できるようにする必要があります。テンプレートマッピングファイル /var/nis/NIS+LDAPmapping.template では、この目的に対応した行がコメントになっています。 credlocal、creduser、および crednode データベース ID については、 nisplusObjectDN、nisplusLDAPattributeFromColumn 属性、および nisplusLDAPcolumnFromAttribute 属性の値を参照してください。マッピングファイルの編集が終了したら、 rpc.nisd デーモンを再起動します。マッピングファイルがデフォルトでない場合は、-m オプションを追加してください。また、rpc.nisd デーモンを NIS (YP) エミュレーションに対応させる場合は、 -Y オプションを追加してください。

pkill rpc.nisd

/usr/sbin/rpc.nisd *-m mapping-file* **[-Y]**

client_info および timezone テーブル

RFC 2307 では、NIS+ の client_info.org_dir および timezone.org_dir テーブルに保存する情報は規定していません。このため、これらのテーブルのマッピングは、テンプレートマッピングファイル (/var/nis/NIS+LDAPmapping.template)

ではデフォルトで無効になっています。client_info および timezone の情報を LDAP に保存する場合は、LDAP サーバーのマニュアルを参照しながら、以降の節で説明する新しい属性とオブジェクトクラスを作成します。

client_info 属性とオブジェクトクラス

次のような属性とオブジェクトクラスを作成し、client_info データのコンテナを作成します。推奨コンテナ名は ou=ClientInfo です。LDIF データは ldapadd(1) に適用します。属性とオブジェクトクラス OID は、例として挙げています。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.12.0 \
          NAME 'nisplusClientInfoAttr' \
          DESC 'NIS+ client_info table client column' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.12.1 \
          NAME 'nisplusClientInfoInfo' \
          DESC 'NIS+ client_info table info column' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.12.2 \
          NAME 'nisplusClientInfoFlags' \
          DESC 'NIS+ client_info table flags column' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

dn: cn=schema
changetype: modify
add: objectclasses
objectclasses:    ( 1.3.6.1.4.1.42.2.27.5.42.42.13.0 \
          NAME 'nisplusClientInfoData' \
          DESC 'NIS+ client_info table data' \
          SUP top STRUCTURAL MUST ( cn ) \
          MAY ( nisplusClientInfoAttr $ nisplusClientInfoInfo $ nisplusClientInfoFlags ) )
```

コンテナを作成するには、次の LDIF データをファイルに入力します。実際の検索ベースを *searchBase* に代入します。

```
dn: ou=ClientInfo, searchBase

objectClass: organizationalUnit

ou: ClientInfo

objectClass: top
```

ou=ClientInfo コンテナを作成するために、上記のファイルを ldapadd コマンドの入力として使用します。たとえば、LDAP 管理者の DN が cn=directory manager で、LDIF データが含まれるファイルが cifile の場合は、次のコマンドを実行します。

```
# ldapadd -D cn="directory manager" -f cifile
```

必要な認証によっては、ldapadd コマンドを実行すると、パスワードプロンプトが表示されることがあります。

/var/nis/NIS+LDAPmapping.template ファイルでは、client_info.org_dir テーブルの定義はコメントになっています。これらの定義を実際のマッピングファイルにコピーし、コメント文字「#」を削除して定義を有効にしてから、rpc.nisd デーモンを再起動します。必要に応じて、NIS+ データと LDAP データを同期化します。方法については、304 ページの「NIS+ から LDAP への移行シナリオ」 を参照してください。

timezone 属性とオブジェクトクラス

次のような属性とオブジェクトクラスを作成し、タイムゾーンデータのコンテナを作成します。推奨コンテナ名は ou=Timezone です。LDIF データは ldapadd(1) に適用できます。属性とオブジェクトクラス OID は、例として挙げています。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.15.0 NAME 'nisplusTimeZone' \
          DESC 'tzone column from NIS+ timezone table' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

dn: cn=schema
changetype: modify
add: objectclasses
objectclasses:    ( 1.3.6.1.4.1.42.2.27.5.42.42.16.0 NAME 'nisplusTimeZoneData' \
          DESC 'NIS+ timezone table data' \
          SUP top STRUCTURAL MUST ( cn ) \
          MAY ( nisplusTimeZone $ description ) )
```

ou=Timezone コンテナを作成するには、次の LDIF データをファイルに入力します。実際の検索ベースを *searchBase* に代入します。

```
dn: ou=Timezone,searchBase ou: Timezone objectClass: top

objectClass: organizationalUnit
```

ou=Timezone コンテナを作成するために、上記のファイルを ldapadd(1) の入力として使用します。たとえば、LDAP 管理者の DN が cn=directory manager で、LDIF データが含まれるファイルが tzfile の場合は、次のコマンドを実行します。

```
# ldapadd -D cn="directory manager" -f tzfile
```

必要な認証によっては、ldapadd コマンドを実行すると、パスワードプロンプトが表示されることがあります。

/var/nis/NIS+LDAPmapping.template ファイルでは、timezone.org_dir テーブルの定義はコメントになっています。これらの定義を実際のマッピングファイルにコピーし、コメント文字「#」を削除して定義を有効にしてから、rpc.nisd デーモンを再起動します。必要に応じて、NIS+ データと LDAP データを同期化します。方法については、304 ページの「NIS+ から LDAP への移行シナリオ」 を参照してください。

新しいオブジェクトマッピングの追加

テンプレートマッピングファイル /var/nis/NIS+LDAPmapping.template には、すべての標準 NIS+ オブジェクトのマッピング情報が含まれます。サイトまたはアプリケーション固有のオブジェクトのマッピングをサポートするには、新しいマッピングエントリを追加する必要があります。エントリ以外のオブジェクト (ディレクトリ、グループ、リンク、またはテーブル) の場合は、簡単に追加できます。しかし、エントリオブジェクトの場合、対応するエントリデータの LDAP 編成が NIS+ で使用される編成と大きく異なるときは、エントリの追加が複雑になることがあります。ここでは簡単な例を挙げます。

▼ エントリ以外のオブジェクトを対応づけるには

1. 対応づけるオブジェクトの完全指定名を検索します。

   このオブジェクト名が nisplusLDAPbaseDomain 属性で指定されるドメイン名に存在する場合は、nisplusLDAPbaseDomain 値に等しい部分は省略できます。

   たとえば、nisplusLDAPbaseDomain の値が some.domain. で、マッピング先のオブジェクトが nodeinfo.some.domain. と呼ばれるテーブルの場合、オブジェクト名は nodeinfo に短縮できます。

2. オブジェクトを識別するデータベース **ID** を作成します。

   データベース ID は、使用するマッピング構成に対して一意でなければなりません。一意でない場合は解釈されません。LDAP データには、データベース ID がありません。エントリオブジェクトのマッピングと混同しないように、テーブルエントリではなくテーブルオブジェクト自体を識別するデータベース ID を作成します。ID の末尾には、_table などのわかりやすい文字列を付加します。

   たとえば、データベース ID nodeinfo_table を使用して、データベース ID とオブジェクトの接続を標準のマッピングファイルの場所 (/var/nis/NIS+LDAPmapping) で確立するには、以下を追加します。

```
nisplusLDAPdatabaseIdMapping    nodeinfo_table:nodeinfo.some.domain.
```

   nisplusLDAPbaseDomain が some.domain. の場合は、以下も機能します。

```
nisplusLDAPdatabaseIdMapping    nodeinfo_table:nodeinfo
```

3. オブジェクトの **TTL** を決定します。

TTL とは、 rpc.nisd デーモンがオブジェクトのローカルコピーを有効とみなす期間のことです。TTL が期限切れになると、オブジェクトが次に参照されるときに LDAP ルックアップが初期化され、オブジェクトが更新されます。

2 つの TTL 値があります。1 番目の TTL は、リブートまたは再起動したあとに、rpc.nisd デーモンがディスクからオブジェクトを最初に読み込んだときに設定されます。2 番目の TTL は、LDAP から更新されたときに設定されます。1 番目の TTL は、設定した範囲からランダムに選択されます。たとえば、 nodeinfo_table の生存期間を、最初に読み込まれたときには 1 - 3 時間、次回以降に読み込まれたときは 12 時間に設定する場合は、次のように指定します。

```
nisplusLDAPentryTtl        nodeinfo_table:3600:10800:43200
```

4. オブジェクトデータを **LDAP** のどこに格納するかを決定します。

   テンプレートマッピングファイルでは、エントリ以外のオブジェクトの格納先が ou=nisPlus コンテナに設定されています。

   この設定を使用する場合に、適切な属性、オブジェクトクラス、およびコンテナをまだ作成していないときは、314 ページの「テーブルエントリ以外の NIS+ オブジェクトのマッピング」 を参照してください。

   たとえば、nodeinfo オブジェクトを ou=nisPlus,dc=some,dc=domain コンテナに格納し、その LDAP エントリを cn nodeinfo にするとします。 次の nisplusLDAPobjectDN を作成してください。

```
nisplusLDAPobjectDN    nodeinfo_table:\
                cn=nodeinfo,ou=nisPlus,dc=some,dc=domain?base?\
                objectClass=nisplusObjectContainer:\
                cn=nodeinfo,ou=nisPlus,dc=some,dc=domain?base?\
                    objectClass=nisplusObjectContainer,\
                    objectClass=top
```

   NIS+ 複製は LDAP にデータを書き込まないため、この nisplusLDAPobjectDN はマスターおよび複製の両方に対して使用できます。

5. マッピング先の **NIS+** オブジェクトがまだ **NIS+** に作成されていない場合は、この手順を省略できます。オブジェクトデータを **LDAP** に格納します。この操作には、rpc.nisd デーモンを使用できます。ただし、 nisldapmaptest(1M) ユーティリティを使用すると、より簡単に行うことができます。この場合、rpc.nisd デーモンを停止する必要はありません。

   # nisldapmaptest -m /var/nis/NIS+LDAPmapping -o -t nodeinfo -r

   –o オプションには、テーブルエントリではなく、テーブルオブジェクト自体を指定します。

6. オブジェクトデータが **LDAP** に格納されたことを確認します。この例では、**LDAP** サーバーがローカルマシンのポート 389 で動作していることを前提としています。

   # ldapsearch -b ou=nisPlus,dc=some,dc=domain cn=nodeinfo

   出力は次のようになります。

```
cn=nodeinfo,ou=nisPlus,dc=some,dc=domain
nisplusobject=NOT ASCII
objectclass=nisplusObjectContainer
objectclass=top
```

```
cn=nodeinfo
```

7. rpc.nisd デーモンを再起動すると、新しいマッピング情報が有効になります。マッピングファイルがデフォルト以外の名前の場合には、必ず -m オプションを指定してください。rpc.nisd デーモンを **NIS (YP)** サービスに対応させる場合は、 -Y オプションを追加します。

   # **pkill rpc.nisd**

   # **/usr/sbin/rpc.nisd** *-m mappingfile* **[-Y]**

エントリオブジェクトの追加

NIS+LDAPmapping(4) には、テーブルエントリマッピングの構文および意味論が詳細に指定されています。また、構文要素ごとの使用例も提供されています。ただし、多くの場合、既存のマッピングから目的のマッピングに近いものを選択し、そのマッピングをコピーして変更すれば、最も簡単に行うことができ、エラーも少なくなります。

たとえば、ノードの在庫一覧と所有者情報を格納する nodeinfo という NIS+ テーブルを想定します。NIS+ テーブルは、次のコマンドを使って作成されたとします。

```
# nistbladm -c -D access=og=rmcd,nw=r -s : nodeinfo_tbl \

cname=S inventory=S owner= nodeinfo.`domainname`.
```

cname 列には、ノードの正式名が格納されます。つまり、ノードの hosts.org_dir テーブルの cname 列と同じ値が格納されます。

また、対応する情報が LDAP の ou=Hosts コンテナに格納され、nodeInfo オブジェクトクラスの必須属性が cn で、オプション属性が nodeInventory と nodeOwner であるとします。nodeInfo オブジェクトクラスは、この例のための仮想クラスで、RFC では定義されていません。

既存の nodeinfo データを LDAP にアップロードするときは、別のファイルに新しいマッピング属性を作成すれば、簡単に行うことができます。たとえば、/var/nis/tmpmapping を使用します。

1. マッピング先の NIS+ テーブルを識別するデータベース ID を作成します。

   ```
   nisplusLDAPdatabaseIdMapping    nodeinfo:nodeinfo
   ```

2. nodeinfoテーブルのエントリに TTL を設定します。この情報はほとんど変更されないため、TTL を 12 時間に設定します。rpc.nisd デーモンがディスクから nodeinfo テーブルを最初に読み込むと、テーブルエントリの TTL が 6 - 12 時間からランダムに選択されます。

   ```
   nisplusLDAPentryTtl         nodeinfo:21600:43200:43200
   ```

3. 既存のマッピングから、作成するマッピングに似ているものを選択します。この例では、属性値の割り当ては簡単で、直接割り当てるだけです。ただし、既存のコンテナに LDAP データを格納する処理が複雑です。このため、nodeinfo データの

削除は、慎重に行う必要があります。ou=Hosts エントリ全体を削除せずに、nodeInventory および nodeOwner 属性だけを削除します。このため、特別の削除ルールが必要になります。

つまり、コンテナを共有し削除ルールを持つマッピングを探します。この候補として netmasks マッピングがあります。このマッピングは、ou=Networks コンテナを共有し、削除ルールを持っています。

4. /var/nis/NIS+LDAPmapping.template の netmasks テンプレートマッピングでは、次のマッピングがデフォルトになっています。

```
nisplusLDAPobjectDN    netmasks:ou=Networks,?one?objectClass=ipNetwork,\
                                                    ipNetMaskNumber=*:\
                                          ou=Networks,?one?objectClass=ipNetwork:
                                                      dbid=netmasks_del
```

このテンプレートマッピングを nodeinfo の新しいマッピングにコピーし、データベース ID を nodeinfo、コンテナを ou=Hosts、オブジェクトクラスを nodeInfo に変更します。つまり、nodeinfo マッピングの最初の行は、次のようになります。

```
nisplusLDAPobjectDN    nodeinfo:ou=Hosts,?one?objectClass=nodeInfo,\
```

netmasks マッピングの 2 行目は、検索フィルタ部分になっています。ipNetMaskNumber 属性を含む ou=Networks エントリだけを選択します。この例では、次の nodeInventory 属性を持つ ou=Hosts エントリを選択します。

```
nodeInventory=*:\
```

3、4 行目は、nisplusLDAPobjectDN の書き込み部分になっています。LDAP nodeinfo データの書き込み先と、nodeinfo データを削除するときのルールが指定されています。ここでは、データベース ID が nodeinfo_del の削除ルールを作成します。ou=Hosts の既存のエントリに常に書き込むため、次のように nodeinfo データ自体のオブジェクトクラスを指定するだけです。

```
ou=Hosts,?one?objectClass=nodeInfo:\
                              dbid=nodeinfo_del
```

この結果、nisplusLDAPobjectDN は次のようになります。

```
nisplusLDAPobjectDN    nodeinfo:ou=Hosts,?one?objectClass=nodeInfo,\
                                                              nodeInventory=*:\
                                                    ou=Hosts,?one?
objectClass=nodeInfo:\
                                                                dbid=nodeinfo_del
```

5. nodeinfo データを NIS+ から LDAP に対応づけるマッピングルールを作成します。netmasks を使用するテンプレートは、次のようになります。

```
nisplusLDAPattributeFromColumn \
        netmasks:    dn=("ipNetworkNumber=%s,", addr), \
                        ipNetworkNumber=addr, \
                        ipNetmaskNumber=mask, \
                        description=comment
```

ここでは、ou=Hosts コンテナはより複雑な構成になります。RFC 2307 の規定では、dn に IP アドレスを含める必要があるためです。しかし、IP アドレスは nodeinfo テーブルに格納されないため、別の方法で取得する必要があります。テンプレートファイルの crednode マッピングには、IP アドレスの取得方法が記述されています。

```
nisplusLDAPattributeFromColumn \
        crednode:    dn=("cn=%s+ipHostNumber=%s,", \
                                (cname, "%s.*"), \
            ldap:ipHostNumber:?one?("cn=%s", (cname, "%s.*"))), \
```

crednodeマッピングの部分をコピーできます。ただし、ここでは、cname 列値は主体名ではなく実際のホスト名です。cname の一部を抽出する必要はありません。属性および列名を完全な名前の代入に変更します。nodeinfo マッピングは次のようになります。

```
nisplusLDAPattributeFromColumn \
        nodeinfo:    dn=("cn=%s+ipHostNumber=%s,", cname, \
            ldap:ipHostNumber:?one?("cn=%s", cname)), \
                nodeInventory=inventory, \
                nodeOwner=owner
```

6. LDAP のデータを NIS+ にマッピングするときは、 netmasks エントリのテンプレートは次のようになります。

```
nisplusLDAPcolumnFromAttribute \
        netmasks:    addr=ipNetworkNumber, \
                mask=ipNetmaskNumber, \
                comment=description
```

属性および列名を代入すると、次のようになります。

```
nisplusLDAPcolumnFromAttribute \
        nodeinfo:    cname=cn, \
                inventory=nodeInventory, \
                owner=nodeOwner
```

7. netmasks の削除ルールは、次のようになっています。

```
nisplusLDAPattributeFromColumn \
        netmasks_del:    dn=("ipNetworkNumber=%s,", addr), \
                ipNetmaskNumber=
```

この例では、NIS+ の netmasks エントリが削除されると、対応する ou=Networks LDAP エントリの ipNetmaskNumber 属性が削除されます。ここでは、 nodeInventory および nodeOwner 属性を削除します。つまり、手順 (5) の dn 指定を使用して、次のように編集します。

```
nisplusLDAPattributeFromColumn \
        nodeinfo_del:    dn=("cn=%s+ipHostNumber=%s,", cname, \
            ldap:ipHostNumber:?one?("cn=%s", cname)), \
                nodeInventory=, \
                nodeOwner=
```

8. マッピング情報はこれで完了です。このマッピングを有効にするために、rpc.nisd デーモンを停止してから、再起動します。

```
# pkill rpc.nisd
```

9. NIS+ nodeinfo テーブルにすでにデータが存在する場合は、そのデータを LDAP にアップロードします。新しい nodeinfo マッピング情報を、別のファイル /var/nis/tmpmapping に格納します。

```
# /usr/sbin/rpc.nisd -D -m /var/nis/tmpmapping \
-x nisplusLDAPinitialUpdateAction=to_ldap \
-x nisplusLDAPinitialUpdateOnly=yes
```

10. 一時ファイル /var/nis/tmpmapping のマッピング情報を実際のマッピングファイルに追加します。エディタを使用するか、次の方法でデータを追加します。対象のマッピングファイルは、 /var/nis/NIS+LDAPmapping とします。

```
# cp -p /var/nis/NIS+LDAPmapping \
/var/nis/NIS+LDAPmapping.backup
# cat /var/nis/tmpmapping>> /var/nis/NIS+LDAPmapping
```

注 – 二重矢印 「>>」はリダイレクトを示します。矢印「>」は、対象ファイルの上書きを示します。

---

11. rpc.nisd デーモンを再起動します。 rpc.nisd デーモンが次のように NIS(YP) データも提供する場合は、-Y オプションを追加します。

```
# /usr/sbin/rpc.nisd -m /var/nis/NIS+LDAPmapping
```

構成情報を LDAP に格納する

NIS+ および LDAP の構成情報は、構成ファイルとコマンド行で格納できますが、構成属性は LDAP にも格納できます。構成情報が多くの NIS+ サーバーによって共有され、定期的に変更される場合は、LDAP に格納すると便利です。

構成属性を LDAP に格納するには、LDAP サーバーのマニュアルを参照して、次の新しい属性とオブジェクトクラスを作成します。構成情報は、 rpc.nisd コマンド行または /etc/default/rpc.nisd の nisplusLDAPconfigDN 値に指定された場所に存在することが前提となっています。また、cn が nisplusLDAPbaseDomain 値であることも前提です (LDAP から構成情報を読み込む前に、rpc.nisd デーモンに認識されているため)。

LDIF データは、ldapadd(1) に適用できます。属性とオブジェクトクラス OID は、例として挙げています。

defaultSearchBase、preferredServerList 、およびauthenticationMethod 属性は、「DUA config」スキーマの原案に準拠しています。このスキーマは、IETF 標準となる見込みです。NIS+LDAPmapping(4) で使用する場合は、次の定義で十分です。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.11.1.3.1.1.1 NAME 'defaultSearchBase' \
          DESC 'Default LDAP base DN used by a DUA' \
          EQUALITY distinguishedNameMatch \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.11.1.3.1.1.2 NAME 'preferredServerList' \
          DESC 'Preferred LDAP server host addresses to be used by a DUA' \
          EQUALITY caseIgnoreMatch \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.11.1.3.1.1.6 NAME 'authenticationMethod' \
          DESC 'Identifies the authentication method used to connect to the DSA'\
          EQUALITY caseIgnoreMatch \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
```

NIS+ および LDAP の構成属性は、次のようになっています。

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.0 \
          NAME 'nisplusLDAPTLS' \
          DESC 'Transport Layer Security' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.1 \
          NAME 'nisplusLDAPTLSCertificateDBPath' \
          DESC 'Certificate file' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.2 \
          NAME 'nisplusLDAPproxyUser' \
          DESC 'Proxy user for data store/retrieval' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.3 \
          NAME 'nisplusLDAPproxyPassword' \
          DESC 'Password/key/shared secret for proxy user' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.4 \
          NAME 'nisplusLDAPinitialUpdateAction' \
          DESC 'Type of initial update' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.5 \
          NAME 'nisplusLDAPinitialUpdateOnly' \
          DESC 'Exit after update ?' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.6 \
          NAME 'nisplusLDAPretrieveErrorAction' \
          DESC 'Action following an LDAP search error' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.7 \
```

```
          NAME 'nisplusLDAPretrieveErrorAttempts' \
          DESC 'Number of times to retry an LDAP search' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.8 \
          NAME 'nisplusLDAPretrieveErrorTimeout' \
          DESC 'Timeout between each search attempt' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.9 \
          NAME 'nisplusLDAPstoreErrorAction' \
          DESC 'Action following an LDAP store error' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.10 \
          NAME 'nisplusLDAPstoreErrorAttempts' \
          DESC 'Number of times to retry an LDAP store' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.11 \
          NAME 'nisplusLDAPstoreErrorTimeout' \
          DESC 'Timeout between each store attempt' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.12 \
          NAME 'nisplusLDAPrefreshErrorAction' \
          DESC 'Action when refresh of NIS+ data from LDAP fails' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.13 \
          NAME 'nisplusLDAPrefreshErrorAttempts' \
          DESC 'Number of times to retry an LDAP refresh' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.14 \
          NAME 'nisplusLDAPrefreshErrorTimeout' \
          DESC 'Timeout between each refresh attempt' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.15 \
          NAME 'nisplusNumberOfServiceThreads' \
          DESC 'Max number of RPC service threads' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.16 \
          NAME 'nisplusThreadCreationErrorAction' \
          DESC 'Action when a non-RPC-service thread creation fails' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.17 \
          NAME 'nisplusThreadCreationErrorAttempts' \
          DESC 'Number of times to retry thread creation' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.18 \
          NAME 'nisplusThreadCreationErrorTimeout' \
          DESC 'Timeout between each thread creation attempt' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.19 \
          NAME 'nisplusDumpErrorAction' \
          DESC 'Action when an NIS+ dump fails' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.20 \
          NAME 'nisplusDumpErrorAttempts' \
          DESC 'Number of times to retry a failed dump' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

```
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.21 \
          NAME 'nisplusDumpErrorTimeout' \
          DESC 'Timeout between each dump attempt' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.22 \
          NAME 'nisplusResyncService' \
          DESC 'Service provided during a resync' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.23 \
          NAME 'nisplusUpdateBatching' \
          DESC 'Method for batching updates on master' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.24 \
          NAME 'nisplusUpdateBatchingTimeout' \
          DESC 'Minimum time to wait before pinging replicas' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.25 \
          NAME 'nisplusLDAPmatchFetchAction' \
          DESC 'Should pre-fetch be done ?' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.26 \
          NAME 'nisplusLDAPbaseDomain' \
          DESC 'Default domain name used in NIS+/LDAP mapping' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.27 \
          NAME 'nisplusLDAPdatabaseIdMapping' \
          DESC 'Defines a database id for an NIS+ object' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.28 \
          NAME 'nisplusLDAPentryTtl' \
          DESC 'TTL for cached objects derived from LDAP' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.29 \
          NAME 'nisplusLDAPobjectDN' \
          DESC 'Location in LDAP tree where NIS+ data is stored' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.30 \
          NAME 'nisplusLDAPcolumnFromAttribute' \
          DESC 'Rules for mapping LDAP attributes to NIS+ columns' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
attributetypes:    ( 1.3.6.1.4.1.42.2.27.5.42.42.18.31 \
          NAME 'nisplusLDAPattributeFromColumn' \
          DESC 'Rules for mapping NIS+ columns to LDAP attributes' \
          SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

dn: cn=schema
changetype: modify
add: objectclasses
objectclasses:    ( 1.3.6.1.4.1.42.2.27.5.42.42.19.0 NAME 'nisplusLDAPconfig' \
          DESC 'NIS+/LDAP mapping configuration' \
          SUP top STRUCTURAL MUST ( cn ) \
          MAY ( preferredServerList $ defaultSearchBase $
authenticationMethod $ nisplusLDAPTLS $ nisplusLDAPTLSCertificateDBPath
$ nisplusLDAPproxyUser $ nisplusLDAPproxyPassword $ nisplusLDAPinitialUpdateAction
$ nisplusLDAPinitialUpdateOnly $ nisplusLDAPretrieveErrorAction
```

```
$ nisplusLDAPretrieveErrorAttempts $ nisplusLDAPretrieveErrorTimeout
$ nisplusLDAPstoreErrorAction $ nisplusLDAPstoreErrorAttempts
$ nisplusLDAPstoreErrorTimeout $ nisplusLDAPrefreshErrorAction
$ nisplusLDAPrefreshErrorAttempts $ nisplusLDAPrefreshErrorTimeout
$ nisplusNumberOfServiceThreads $nisplusThreadCreationErrorAction
$ nisplusThreadCreationErrorAttempts $ nisplusThreadCreationErrorTimeout
$ nisplusDumpErrorAction $ nisplusDumpErrorAttempts
$ nisplusDumpErrorTimeout $ nisplusResyncService $ nisplusUpdateBatching
$ nisplusUpdateBatchingTimeout $ nisplusLDAPmatchFetchAction
$ nisplusLDAPbaseDomain $ nisplusLDAPdatabaseIdMapping $ nisplusLDAPentryTtl
$ nisplusLDAPobjectDN $ nisplusLDAPcolumnFromAttribute !
$ nisplusLDAPattributeFromColumn ) )
```

次の LDIF データを含むファイルを作成します。実際の検索ベースを *searchBase* に、完全指定ドメイン名を *domain* に代入します。

dn: cn=*domain*,*searchBase*

cn:*domain*

objectClass: top objectClass: nisplusLDAPconfig

上のファイルを ldapadd(1) の入力として使用し、NIS+ および LDAP の構成エントリを作成します。最初は、エントリは空になっています。ldapmodify(1) を使用して、構成属性を追加します。たとえば、nisplusNumberOfServiceThreads 属性に「32」を設定するには、ldapmodify(1) の入力として次のファイルを作成します。

dn: cn=*domain*, *searchBase* **nisplusNumberOfServiceThreads: 32**

付録 A

『Solaris のシステム管理 (ネーミングとディレクトリサービス : DNS、NIS、LDAP 編)』の変更点

Solaris 9 9/02 アップデート

本書の LDAP セクションに、「NIS+ から LDAP への移行」という章が追加されました。

LDAP 関連の章に、説明と例がいくつか追加されています。

用語集

DES	「データ暗号化規格 (DES)」の項を参照。
DIT	「ディレクトリ情報ツリー」の項を参照。
DNS	「ドメインネームシステム (DNS)」の項を参照。
DNS ゾーン	ネットワークドメイン内の管理境界であり、通常 1 つまたは複数のサブドメインから構成される。
DNS ゾーンファイル	DNS ソフトウェアがドメイン内の全ワークステーションの名前と IP アドレスを格納する一連のマップ。
DNS 転送	NIS サーバーまたは NIS 互換設定の NIS+ サーバーは、自分で応答できない要求を DNS サーバーに転送する。
FNS	「フェデレーテッドネームサービス」の項を参照。
GID	「グループ ID」の項を参照。
IP	インターネットプロトコル。インターネットプロトコル体系の「ネットワーク層」プロトコル。
IP アドレス	ネットワーク内の各ホストを識別する一意な番号。
LDAP	Lightweight Directory Access Protocol。LDAP ネームサービスクライアントとサーバー間の通信に使用される標準の拡張可能なディレクトリアクセスプロトコル。
MIS	経営情報システムまたはサービス。
NIS	ネットワーク上のシステムとユーザーについての重要な情報が収められている分散型ネットワーク情報サービス。NIS データベースは、「マスターサーバー」とすべての「スレーブサーバー」に格納されている。
NIS+	ネットワーク上のシステムとユーザーについての階層情報が収められている分散型ネットワーク情報サービス。NIS+ データベースは、「マスターサーバー」とすべての「複製サーバー」に格納されている。

NIS 互換モード	NIS+ の構成の 1 つであり、このモードでは NIS クライアントは NIS+ テーブルに格納されたデータにアクセスできる。また、NIS+ サーバーは NIS クライアントと NIS+ クライアントからの情報要求に応答できる。
NIS マップ	NIS によって使用されるファイルであり、ネットワーク上の全ユーザーのパスワードエントリやネットワーク上の全ホストマシンの名前など特定種類の情報を格納する。NIS サービスの一部であるプログラムはこれらのマップを参照する。「NIS」の項を参照。
RPC	「遠隔手続き呼び出し (RPC)」の項を参照。
SASL	簡単な認証およびセキュリティ層 (Simple Authentication and Security Layer)。アプリケーション層プロトコルにおける認証およびセキュリティ層の意味上の取り決め。
Secure RPC パスワード	Secure RPC プロトコルにおいて必要となるパスワードのこと。秘密鍵の暗号化に使用されるユーザーのログインパスワードと同じでなければならない。
SSL	Secure Sockets Layer プロトコル。LDAP セキュアなどのアプリケーションプロトコルを作成するためのトランスポート層のセキュリティ機構の総称。
TCP	「Transport Control Protocol (TCP)」の項を参照。
TCP/IP	Transport Control Protocol/Internet Protocol の頭文字。このプロトコル群は、最初はインターネット用に開発された。「インターネット」プロトコル群とも呼ばれる。Solaris のネットワークはデフォルトで、TCP/IP 上で動作する。
Transport Control Protocol (TCP)	インターネットプロトコル群での主要なトランスポートプロトコルであり、高信頼性でコネクション型の全二重ストリームを提供する。配信には IP を使用する。「TCP/IP」の項を参照。
Transport Layer Security (TLS)	LDAP クライアントとディレクトリサーバーとの通信を保護して、機密性とデータ整合性を確保する。TLS プロトコルは、Secure Sockets Layer (SSL) プロトコルのスーパーセットである。
X.500	開放型システム間相互接続 (OSI) 規格で定義されたグローバルディレクトリサービス。LDAP の前身。
アプリケーションレベルのネームサービス	ファイル、メール、印刷などのサービスを提供するアプリケーションに組み込まれているネームサービスのこと。アプリケーションレベルのネームサービスは、エンタープライズレベルのネームサービスの下に位置する。エンタープライズレベルのネームサービスが提供するコンテキストの中に、アプリケーションレベルのネームサービスのコンテキストを組み込むことができる。
暗号化	データの機密性を保護するための手段。
暗号化鍵	「データ暗号化鍵」の項を参照。

インターネットアドレス	「TCP/IP」を使用するホストに割り当てられた 32 ビットのアドレス。「ドット形式の 10 進表記」の項を参照。
インデックス付き名前	テーブル内のエントリの識別に使用されるネーミング形式。
遠隔手続き呼び出し **(RPC)**	分散コンピューティングのクライアントサーバーモデルを実現する簡単で一般的なパラダイム。与えられた引数を使用することによって、要求が遠隔システムに送信され、指定された手順が実行される。その後、その結果が呼び出し側に返される。
エンタープライズレベルのネットワーク	「エンタープライズレベル」のネットワークは、ケーブル、赤外線、ラジオブロードキャストを通して通信する単一のローカルエリアネットワーク (LAN) にできる。また、ケーブルや直接通話接続によってリンクされた複数の LAN にできる。エンタープライズレベルのネットワーク内では、DNS や X.500/LDAP などのグローバルネームサービスを使用せずに、どのマシンからでも任意のマシンにアクセスできる。
エントリ	データベーステーブルの中の一列のデータのこと。
親ドメイン	「ドメイン」の項を参照。
鍵 **(暗号化)**	鍵の管理および配布システムの一部として、他の鍵の暗号化と復号化に使用される鍵。「データ暗号化鍵」の項も参照。
キーサーバー	秘密鍵を格納する Solaris オペレーティング環境のプロセス。
逆解決	DNS ソフトウェアを使用して、ワークステーションの IP アドレスをワークステーション名に変換するプロセス。
キャッシュマネージャ	NIS+ クライアントのローカルキャッシュ (`NIS_SHARED_DIRCACHE`) を管理するプログラム。これらのクライアントによって最も頻繁に使用されるディレクトリをサポートする NIS+ サーバーについての位置情報 (トランスポートアドレス、認証情報、生存期間など) を格納するために使用される。
クライアント	(1) クライアントとは、ネームサーバーに対してネームサービスを要求する主体 (マシンまたはユーザー)。 (2) ファイルシステムのクライアントサーバーモデルでは、クライアントとは、計算パワーや大きな記憶容量などの計算サーバーのリソースに遠隔アクセスするマシン。 (3) ウィンドウシステムのクライアントサーバーモデルでは、クライアントとは「サーバープロセス」からウィンドウサービスにアクセスする「アプリケーション」。このモデルでは、クライアントとサーバーは同じマシン上でも別のマシン上でも動作できる。
クライアントサーバーモデル	ネットワークサービスおよびこれらのモデルユーザープロセス (プログラム) を説明する一般的な方法の 1 つ。たとえば、「ドメインネームシステム (DNS) 」のネームサーバー/ネームリゾルバパラダイムなど。「クライアント」の項も参照。

グループ **ID**	ユーザーのデフォルト「グループ」を識別する番号。
グローバルネームサービス	電話回線、衛星回線、その他の通信システムにより連結された世界中のエンタープライズレベルネットワークの名前を管理するサービスのこと。この世界中のネットワークの集合体がいわゆる「インターネット」である。グローバルネームサービスでは、ネットワーク名だけでなく、任意のネットワーク内の個々のマシンやユーザーも識別できる。
広域ネットワーク **(WAN)**	地理的に離れた複数のローカルエリアネットワーク (LAN) またはシステムを、電話回線、光ファイバ、衛星などを使用して接続したネットワークのこと。
公開鍵	数学的に生成された 1 対の番号の公開構成要素であり、秘密鍵と組み合わせれば DES 鍵が生成される。この DES 鍵を使用すれば、情報の暗号化と復号化を行える。公開鍵は、すべてのユーザーとマシンが使用できる。どのユーザーやマシンにも、固有の公開鍵と秘密鍵が 1 対ある。
子ドメイン	「ドメイン」の項を参照。
サーバー	(1) NIS+ 、NIS 、DNS 、LDAP では、ネットワークに NIS+ サービスを提供するホストマシンのこと。 (2) ファイルシステムの「クライアントサーバーモデル」では、サーバーとは計算リソース (計算サーバーとも呼ばれる) と大きな記憶容量を備えたマシン。クライアントマシンは遠隔アクセスが可能であり、これらのリソースを使用できる。ウィンドウシステムのクライアントサーバーモデルでは、サーバーとはアプリケーションまたは「クライアントプロセス」にウィンドウサービスを提供するプロセス。このモデルでは、クライアントとサーバーは同じマシン上でも別のマシン上でも実行できる。 (3) ファイルの提供を実際に処理する「デーモン」。
サーバーリスト	「優先サーバーリスト」の項を参照。
サブネット	経路指定を簡単にするため、1 つの論理ネットワークを小さな物理ネットワークに分割する方式。
資格	クライアントソフトウェアが各要求とともにネームサーバーに送信する認証情報。この情報によって、ユーザーまたはマシンの ID が検査される。
識別名	X.500 ディレクトリ情報ベース (DIB) のエントリ。ルートから指定エントリまでのパスに沿ったツリーの各エントリから選択した属性で構成される。
スレーブサーバー	(1) ネットワーク情報サービス (NIS) データベースのコピーを管理するサーバーシステム。このシステムには、ディスクと動作環境の完全なコピーが存在する。

(2) スレーブサーバーは、NIS+ では「複製サーバー」と呼ばれる。

ディレクトリ — (1) LDAP ディレクトリは LDAP オブジェクトのコンテナのこと。(2) UNIX では、ファイルまたはサブディレクトリのコンテナのこと。

ディレクトリキャッシュ — ディレクトリオブジェクトに関連付けられたデータの格納に使用されるローカルファイル。

ディレクトリ情報ツリー **(DIT)** — ある特定のネットワークの分散型ディレクトリ構造のこと。Solaris LDAP クライアントはデフォルトで、DIT がある特定の構造を持っていると想定して情報にアクセスする。LDAP サーバーがサポートするドメインごとに、想定された構造を持つ想定されたサブツリーがある。

データ暗号化鍵 — 暗号化を行うプログラムに使用されるデータを暗号化および復号化するための鍵。「鍵 (暗号化)」の項も参照。

データ暗号化規格 **(DES)** — アメリカ商務省標準局によって開発された、データの暗号化と復号化のために一般的に使用される高度なアルゴリズム。「SUN-DES-1」の項も参照。

テーブル — NIS+ においては、NIS+ データを行および列の中に持つ 2 次元的な (リレーショナルでない) データベースオブジェクトのこと (NIS における「NIS マップ」は、列を 2 つ持つ NIS+ テーブルに似ている)。NIS+ データはテーブルの形で保存される。NIS+ では定義済み (システム) テーブルが 16 個提供される。保存される情報のタイプはテーブルごとに異なる。

ドット形式の **10** 進表記 — 32 ビット整数用の構文表現であり、10 進表記された 4 つの 8 ビット数が小数点 (ドット) で区切って表現される。192.67.67.20 のように、インターネットでの IP アドレスを表現するために使用される。

ドメイン — (1) NIS+ では、NIS+ によって管理されるオブジェクト (階層構造になっている) のグループ。最上位のドメイン (ルートドメイン) 1 つと、サブドメイン 0 個以上からなる。ドメインおよびサブドメインは、地理的、組織的、機能的な基準によって編成される。

- 「親ドメイン」階層構造の中で、現在のドメインのすぐ上のドメインを表す相対的な名称。
- 「子ドメイン」階層構造の中で、現在のドメインのすぐ下のドメインを表す相対的な名称。
- 「ルートドメイン」現在の NIS+ 階層の最上位のドメイン。

(2) インターネットではネーミング階層の一部で、通常、Local Area Network (LAN)、Wide Area Network (WAN)、またはその一部に相当する。構文上、インターネットドメイン名は小数点 (ドット) によって区切られた一連の名前 (ラベル) から構成される。たとえば、`sales.doc.com`。

	(3) ISO の開放型システム間相互接続 (OSI) では、「ドメイン」は、MHS プライベート管理ドメイン (PRMD) やディレクトリ管理ドメイン (DMD) などのように、複雑な分散システムの管理パーティションとして使用されるのが普通。
ドメインネームサービス **(DNS)**	インターネットで使用されるネットワーク情報サービスのこと。DNS は、ドメイン名とマシン名をインターネットなどのエンタープライズ外部のアドレスにマッピングする場合のネーミングの方針と機構を提供する。
ドメイン名	ローカルネットワーク上のシステムグループに割り当てられた名前であり、管理ファイルを共有する。ネットワーク情報サービスのデータベースが正常に動作するためにはドメイン名が必要。「ドメイン」の項も参照。
名前解決	ワークステーションやユーザーの名前をアドレスに変換するプロセス。
名前空間	(1) 名前空間はユーザー、ワークステーション、およびアプリケーションがネットワーク上で必ず必要とする情報を格納している。 (2) ネーミングシステムで使用される名前セット。
認証	サーバーがクライアントの ID を確認するための手段。
ネットワークパスワード	「Secure RPC パスワード」を参照。
ネットワークマスク	ソフトウェアが、ローカルサブネットアドレスをそれ以外のインターネットプロトコルアドレスから分離するために使用する番号。
ネームサーバー	1 つ以上のネットワークネームサービスを実行するサーバー。
ネームサービス	マシン、ユーザー、プリンタ、ドメイン、ルーターなどの、ネットワーク上の名前とアドレスを管理するネットワークサービスのこと。
ネームサービススイッチ	ネームサービスクライアントがそのネットワーク情報を獲得できるソースを定義する構成ファイル (`/etc/nsswitch.conf`)。
秘密鍵	数学的に生成された 1 対の番号の非公開構成要素であり、公開鍵と組み合わせれば DES 鍵が生成される。この DES 鍵を使用すれば、情報の暗号化と復号化を行える。送信側の非公開鍵は、その鍵の所有者だけが使用できる。どのユーザーやマシンにも、固有の公開鍵と秘密鍵が 1 対ある。
フェデレートされた名前空間	FNS (XFN) の用語。メンバーであるネームシステムと名前空間との関係に関する管理方針に従って生成され得るすべての名前を集めたものの総称。
マスターサーバー	ドメイン内の NIS データベースのマスターコピーを保持しているサーバーのこと。名前空間に対する変更は、必ずマスターサーバーのネームサービスデータベース上で行う。ドメイン中に複数のマスターサーバーを作成できない。

メール交換レコード	DNS ドメイン名、およびこれらに対応するメールホストのリストが収められているファイル。
メールホスト	サイトの電子メールのルーターおよび受信側として機能するワークステーション。
優先サーバーリスト	`client_info` テーブルまたは `client_info` ファイルのこと。優先サーバーリストには、あるクライアントマシンまたはドメインから見た優先サーバーが指定される。
ルートドメイン	「ドメイン」の項を参照。
レコード	「エントリ」の項を参照。
ローカルエリアネットワーク **(LAN)**	1 つの地理的なサイトの中にある複数のシステムをデータやソフトウェアの共有や交換の目的で接続したもの。

索引

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

X

Y

Z

あ

い

え

お

か

き