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)

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

Part No: 816–3959–10
2002 年 5 月

原典: *System Administration Guide: Naming and Directory Services (FNS and NIS+)*

Part No: 816–2074–10

Revision A

020423@3689

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 *: FNS*、*NIS+* 編*)*』では、Solaris™ 9 オペレーティング環境のネームサービスとディレクトリサービス (FNS と NIS+) を設定、構成、および管理する方法について説明します。このマニュアルは、Solaris 9 System Administrator Collection - Japanese に含まれます。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

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。

このマニュアルでは、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するネットワーク概念について説明します。ただし、ネットワークの基礎および Solaris オペレーティング環境の管理ツールについては説明しません。

このマニュアルの構成

このマニュアルでは各パートで、ネームサービスおよびディレクトリサービスについて個別に説明します。

ネームサービスとディレクトリサービス : パート I

NIS+ の設定と構成 : パート II

NIS+ の管理 : パート III

FNS の設定、構成、および管理 : パート IV

ネームサービス間の移行 : パート V

NIS+ エラーメッセージ : 付録 A

関連書籍

- 『*DNS and Bind*』Cricket Liu および Paul Albitz 共著 (O′ Reilly, 1992)
- 『*Managing FNS and NFS*』Hal Stern 著 (O′ Reilly, 1993)

Sun のオンラインマニュアル

docs.sun.com 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	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。
\	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

ネーミングとディレクトリサービス

次の章では、nsswitch.conf ファイルについて説明します。nsswitch.conf ファイルは、異なるサービスの使用方法を調整するために利用します。

第 1 章

ネームサービススイッチ

この章では、ネームサービススイッチの機能と、これ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。Solaris ネームサービスとディレクトリサービスの DNS、NIS、および LDAP の概要については、『 *Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「ネームサービスとディレクトリサービス (概要)」を参照してください。

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` 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。クライアントアプリケーションは、ネームサービススイッチを使用して、次のような `getXbyY()` インタフェースを呼び出します。

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

各マシンは、それぞれの `/etc` ディレクトリの中にスイッチファイルを持っています。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 1–1 を参照してください。

Solaris オペレーティング環境 はインストールの過程において、各マシンの /etc ディレクトリに nsswitch.conf ファイルを自動的にロードします。スイッチファイルの 4 つの代替ファイル (テンプレート) も、LDAP、NIS、NIS+ またはローカルファイルの /etc にロードされます。40 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 ソフトウェアがマシンに最初にインストールされると、インストーラ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 を使用するマシンクライアントの場合、インストーラは nsswitch.ldap を nsswitch.conf にコピーします。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。

マシンの主要なネームサービスをあとから変更する場合は、適切な代替スイッチファイルを nsswitch.conf にコピーします (40 ページの「nsswitch.conf テンプレートファイル」を参照)。また、/etc/nsswitch.conf ファイルの適切な行を編集することによって、クライアントが使用するネットワーク情報のソースを変更することもできます。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、45 ページの「ネームサービススイッチを変更する」 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。順序は必ずしも以下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

次の表では、上記の情報タイプにスイッチファイルの中で指定できるソースについて説明します。

表 1–1 スイッチソースの例

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渡されます。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が、見つからないとき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とき、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。また情報が見つからなかった場合、その理由によって、3 種類のメッセージのうちの 1 つを返します。次の表に、返される可能性のある状態メッセージを示します。

表 1–2 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、/etc ディレクトリのファイルが見つからなかった (アクセスできなかった)

表 **1–2** スイッチ状態メッセージ　(続き)

状態メッセージ	意味
NOTFOUND	エントリなし。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

スイッチ状態メッセージに対して、次の表に示す 2 つのアクションのどちらかで応答するように指示することができます。

表 **1–3** スイッチ状態メッセージへの応答

アクション **(Action)**	説明
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れば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、はっきりと指定する必要はありません。ほかの検索基準を明示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移って検索を続行します。networks など、特定の情報を設定して検索すると、検索は NOTFOUND で中止します。スイッチファイルの networks の行を、以下のように編集してください。

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

networks: nis [NOTFOUND=return] files は、NOTFOUND 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、検索関数は以下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、デフォルトにより適当な /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、デフォルトで適当な /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含まれています。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。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、「#」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ルーチン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 **1–4** スイッチファイルのコメント例

行の種類	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 を変更した場合、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そのこと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、4 つの nsswitch.conf テンプレートを用意して、異なるネームサービスに対応しています。各テンプレートでは、デフォルトの情報ソース (一次ソース、およびそれに続くソース) 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について説明します。

- 「LDAP テンプレートファイル」(nsswitch.ldapファイル)。このテンプレートファイルでは、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、『 *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「クライアントの設定 (手順)」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)。このテンプレートファイルで passwd、group、automount、aliases を除くすべての情報の一次ソースとして指定されているのは、NIS+ です。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)。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、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 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)。このテンプレートファイルでは、ローカルの /etc ディレクトリのファイルだけがマシンの情報ソースとして指定されています。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には、以下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の 4 つのスイッチファイルを、以下に示します。

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

```
#
# /etc/nsswitch.nisplus:
#
```

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

```
# An example file that could be copied over to /etc/nsswitch.conf;
# 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 **1–2** 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
```

例 **1–2** NIS スイッチファイルテンプレート　*(続き)*

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

例 **1–3** 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 **1–4** 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.
```

例 **1–4** LDAP スイッチファイルテンプレート　(続き)

```
passwd:     files ldap
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 の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 つ以上の情報源 (NIS+ テーブル、NIS マップ、DNS ホストテーブル、同一ワークステーション上の /etc など) を対応させています。ネームサービスが選択されると、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 + が選択された場合は、nsswitch.nisplus ファイルがコピーされて nsswitch.conf ファイルが作成されます。

/etc/nsswitch.conf ファイルは、Solaris 9 ソフトウェアにより各マシンの /etc ディレクトリに自動的にロードされます。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源として 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。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、マシンのデフォルトのネームサービスが NIS+、NIS、ローカルファイル、

または LDAP から選択されます。インストールの途中、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+ を使用するマシンクライアントの場合、インストーラは nsswitch.nisplus を nsswitch.conf にコピーします。

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

特定のマシンのネームサービスを変更した場合は、そのマシンのスイッチファイルも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源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を変更する

スイッチファイルの変更は次の手順で行います。

1. クライアントにスーパーユーザーとしてログインします。
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 /etc ファイル用」

```
client1# cd /etc
client1# 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はスイッチ情報をキャッシュに書き込みます。また、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。このため、マシンを再起動して、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。

NIS+ クライアントで IPv6 を使用できるようにする方法

1. スーパーユーザーとしてログインします。
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。files を使用しなかった場合は、アドレスの解決に時間がかかることがあります。

4. ファイルを保存してマシンを再起動します。

   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。

+/- 構文との互換性を追加する

/etc/passwd、/etc/shadow 、および/etc/group ファイルに +/- が使用される場合は、互換性を確保するために、nsswitch.conf ファイルを変更する必要があります。

- 「NIS+」。「NIS+ の場合」に 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
```

この指定により、クライアント関数のネットワーク情報獲得先は、ファイル中に +/- エントリを指定したのと同じように、/etc ディレクトリのファイルと NIS+ テーブルになります。

- 「NIS」。SunOS™ 4 と同じ構文を使用するには、 passwd ソースと groups ソースを compat に変更します。

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

この指定により、クライアント関数のネットワーク情報獲得先は、ファイル中に +/- エントリを指定したのと同じように、/etc ファイルと NIS マップになります。

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 行は以下のようになります。

```
passwd: files nisplus
```

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

```
passwd: files nis
```

パート II　NIS+ の設定と構成

このパートでは、Solaris オペレーティング環境上で NIS+ ネームサービスを設定および構成する方法について説明します。

第 2 章

NIS+ の紹介

この章では、NIS+ の概要を述べます。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ について

NIS+ はネットワークネームサービスの一種です。NIS+ は NIS を機能拡張したものではなく、新しいソフトウェアプログラムとなっています。

NIS+ ネームサービスは、ネットワークがどのような構造であっても、その周囲を取り巻くことにより、サービスを設置した組織の形態に適合するように設計されています。

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。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。各ドメインはいくつかのサーバーでサポートされています。主となるサーバーを「マスター」サーバー、補助用のサーバーを「複製」サーバーと呼びます。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。マスターサーバーと複製サーバーの両方で 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維持しています。マスターサーバー上の NIS+ データの変更は、複製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、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。

Solaris のクライアントは、ネームサービススイッチ (`/etc/nsswitch.conf` ファイル) を使用して、ワークステーションがどこからネットワーク情報を取り出すかを決定します。ネットワーク情報が保存されているのは、ローカルの `/etc` ディレクトリのファイル、NIS、DNS、NIS+ です。ネームサービススイッチには、情報の種類ごとに異なる情報源を指定することもできます。このスイッチの詳細は、第 1 章を参照してください。

NIS+ のメリット

NIS と比較すると、NIS+ には次のようなメリットがあります。

- 安全性の高いデータアクセス
- 階層構造を使用した分散型のネットワーク管理
- 非常に大きな名前空間の管理
- 異なるドメインにあるリソースへのアクセス
- 増分 (incremental) 更新

65 ページの「NIS+ のセキュリティ」で説明したセキュリティシステムを使用すれば、特定のテーブルの個々のエントリに対する特定のユーザーのアクセスを制御できます。このようなセキュリティ方式では、システムを安全に維持できます。また、NIS+ 名前空間全体またはテーブル全体を損傷する危険をおかすことなく、管理業務を広く分散させることができます。

NIS+ の階層構造により、名前空間に複数のドメインを置くことができます。ドメインに分割することにより、管理が容易になります。個々のドメインは完全に独立して管理できるため、システム管理者の負担が軽減され、非常に大きな名前空間の管理責任から解放されます。このように、セキュリティシステムを分散型のネットワーク管理と組み合わせることによって、管理作業の負担を分担することができます。

ドメインが個別に管理されているとしても、すべてのクライアントに名前空間内のすべてのドメインの情報を参照するアクセス権を与えることができます。クライアントは自分のドメイン内のテーブルしか見ることができないため、クライアントがほかのドメイン内のテーブルにアクセスするには、そのテーブルについて明示的にアドレスを指定しなければなりません。

増分更新とは、名前空間内での情報の高速の更新を意味します。ドメインは独立して管理されるため、マスターサーバーテーブルへの変更は、名前空間全体にではなく、その複製サーバーだけに伝達しますが、伝達が行われると、これらの変更はすぐに名前空間全体で有効になります。

NIS+ と NIS の違い

ネットワーク情報サービスプラス (NIS+) は、ネットワーク情報サービス (NIS) といくつかの点で異なります。NIS+ には、多くの新しい機能が追加されています。また、NIS と似た概念に対して、使用する用語が異なることがあります。わからない用語がある場合は、用語集を参照してください。次の表に、NIS と NIS+ の主な相違点を示します。

表 2–1 NIS と NIS+ の相違

NIS	NIS+
フラットなドメイン - 階層なし	階層構造 - データを名前空間内の異なるレベルに格納
データを 2 列のマップに格納	データを複数の列のテーブルに格納
認証を使用しない	DES 認証を使用
ネットワーク情報源は 1 つ	ネームサービススイッチ - クライアントは NIS、NIS+、DNS、またはローカル側の `/etc` ファイルから情報源を選択できる
バッチ伝達のため更新は遅い	すぐに伝達される増分更新

NIS+ は NIS に代わるものとして設計されました。NIS は、1980 年代に普及していたクライアントサーバーコンピューティングネットワークの管理要件に応えるものです。その当時のクライアントサーバーネットワークは、通常、クライアント数が数百を超えることはなく、多目的サーバーの数もわずかでした。これらのネットワークは数カ所のリモートサイトを結んでいるだけであり、しかもユーザーに専門知識があり信頼できたため、セキュリティを必要としませんでした。

しかしクライアントサーバーネットワークは 1980 年代半ば頃から急速な成長を遂げました。現在では、世界中のサイトに配置された 10〜100 台の専用サーバーにサポートされた 100〜10,000 台のマルチベンダークライアントが存在し、複数の公衆網に接

続されています。さらに、ネットワークが格納する情報は、NIS の時代よりもはるかに急速に変化しています。このようなネットワークの規模と複雑性に対処するため、新しい管理方式が必要になりました。NIS+ はこれらの必要性に焦点をあてて設計されました。

NIS の名前空間は、フラットな状態で管理機能を集中管理しています。1990 年代に入りネットワークはスケーラビリティと管理の分散化を求めるようになったため、NIS+ の名前空間は DNS の場合のように階層ドメインをベースに設計されました。

たとえば、図 2–1 は `doc` という名前の親ドメインと、`sales` と `manf` という 2 つのサブドメインを持つ会社の例を示しています。

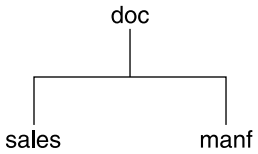


図 2–1 階層型ドメインの例

これによって NIS+ は、小規模から大規模まで、広い範囲のネットワークで使用できます。また、NIS+ のサービスを組織の成長に適合させることもできます。たとえば、ある会社が 2 つの部門に分かれた場合、それに対する NIS+ の名前空間を 2 つのドメインに分割し、これらを個別に管理できます。インターネットがドメインの管理を下のレベルに委譲するように、NIS+ のドメインも程度の差はあっても互いに独立して管理できます。

NIS+ は DNS と似たドメイン階層を使用しますが、NIS+ のドメインは DNS のドメインよりずっと多くの情報を持っています。DNS のドメインは、そのクライアントの名前とアドレスの情報を格納するだけです。一方 NIS+ のドメインには、組織の一部の中でのマシン、ユーザー、およびネットワークサービスについての「情報」が集められています。

ドメインをこのように分割することで、より個別に管理できるようになり、規模が拡大した場合にもうまく対処できますが、情報へのアクセスが以前より困難になることもありません。クライアントは他のドメインの情報にも、同じドメインと同じようにアクセスできます。あるドメインを別のドメインの内部から管理することもできます。

主サーバーは「マスター」サーバーと呼ばれ、バックアップサーバーは「複製」サーバーと呼ばれます。マスターサーバーと複製サーバーの両方で 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ともに維持しています。NIS の情報はマップに格納されますが、NIS+ の情報はテーブルに格納されます。主サーバーはオリジナルのテーブルを、バックアップサーバーはコピーを、それぞれ格納します。

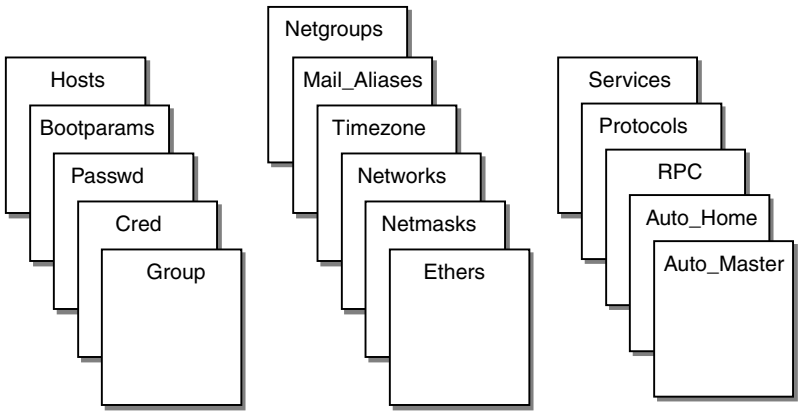
しかし NIS+ は、NIS とはまったく異なる更新方式を使用します。NIS が開発された当時は、格納される情報の型はめったに変化しなかったため、NIS は安定性に重点を置いた更新方式によって開発されました。NIS テーブルの更新は手作業で処理され、大規模な組織では、すべての複製サーバーへの伝達に 1 日以上かかることもあります。この原因の一つは、マップ内の情報が変化するたびにマップ全体を再作成して伝達しなければならなかったからです。

しかし NIS+ は、複製サーバーに対する「増分 (incremental)」更新が可能です。マスターサーバー上での変更は依然として必要ですが、いったん変更すれば、複製サーバーに自動的に伝達され、すぐに名前空間全体から使用できるようになります。マップを「作成」したり、伝達を待つ必要はありません。

NIS+ の ドメイン構造、サーバー、およびクライアントの詳細については、それぞれ 66 ページの「ドメイン」、67 ページの「サーバー」、および 71 ページの「NIS+ 主体とクライアント」 を参照してください。

NIS+ のドメインは、次に示すようなネームサービススイッチを使用して、その NIS+ クライアントを経由してインターネットに接続できます (例 1–1を参照)。このクライアントが DNS のクライアントでもある場合、自分のスイッチ構成ファイルを設定して、NIS+ テーブルだけでなく、DNS ゾーンファイルまたは NIS マップ内の情報を検索できます。

NIS+ は、マップやゾーンファイルではなく、「テーブル」に情報を格納します。NIS + は 16 種類のあらかじめ定義したテーブル (つまり「システム」テーブル) を提供します。



各テーブルにはそれぞれ違う種類の情報が保存されます。たとえば、`hosts` テーブルにはマシンアドレスについての情報が、`passwd` テーブルにはネットワークのユーザーについての情報が格納されています。

NIS+ テーブルには、NIS で使用したマップに比べて大きく改善された機能が 2 つあります。第 1 に、NIS+ のテーブルは、最初の列 (キーと呼ばれることもある) だけではなく、任意の列ごとにアクセスできます。これによって、NIS で使用される `hosts.byname` や `hosts.byaddr` マップなどの二重マップが必要なくなります。第 2 に、NIS+ のテーブル内の情報は、3 つの細分化されたレベルでアクセスし、操作できます。テーブルレベル、エントリレベル、および列レベルです。NIS+ のテーブルとそこに格納されている情報については、第 10 章で説明します。

NIS 管理者は、以下に示す原則および条件下で NIS を NIS+ と共に使用できます。

- 同一ドメインに NIS サーバーと NIS+ サーバーの両方が存在するNIS 管理者は同一ドメインで NIS サーバーと NIS+ サーバーの両方を動作させることは可能ですが、このような動作を長時間行わせることは望ましくありません。一般に、同一ドメイン内での NIS サーバーと NIS+ サーバーを両方使用するのは、NIS から NIS+ への短い移行期間だけに制限するべきです。
- サブドメインNIS 管理者のルートドメインのマスターサーバーで NIS+ が動作している場合は、NIS 管理者は、すべてのサーバーで NIS が動作しているサブドメインを設定できます。NIS 管理者のルートドメインのマスターサーバーで NIS が動作している場合は、NIS 管理者はサブドメインの設定はできません。
- 同一ドメインに複数のマシンが存在する
  - 同一ドメイン内のサーバーで NIS+ が動作している場合は、NIS+、NIS、または `/etc` ファイルを使ってネームサービス情報を取得するように、ドメイン内の各マシンを設定できます。NIS+ サーバーが NIS クライアントのニーズを満たすには、この NIS+ サーバーが NIS 互換モードで動作している必要があります。
  - 同一ドメイン内のサーバーで NIS が動作している場合は、NIS または `/etc` ファイルを使ってネームサービス情報を取得するように、このドメイン内の各マシンを設定できます (各マシンは NIS+ を使用することはできません) 。

  ネームサービスに使用されるサービスは、そのマシンの `nsswitch.conf` ファイルで制御されます。このファイルは、「スイッチ」ファイルと呼ばれています。詳細は、第 1 章を参照してください。

NIS+ のセキュリティ

NIS+ は、「承認と認証」という相補的なプロセスによって、名前空間の構造と格納する情報を保護します。

- 「承認 (authorization)」。「承認」は、名前空間の構成要素に対して、「誰がどのような操作を行えるのか」ということを指定することです。すべての構成要素について行われます。
- 「認証 (authentication)」。名前空間に対するアクセス要求はすべて、NIS+ によって「認証」されます。アクセス要求は、 NIS+ 主体から発行されます。NIS+ 主体には、プロセス、マシン、ルート、またはユーザーがなることができます。「正当

な NIS+ 主体」とは、NIS+ 資格を持っているものを指します。名前空間へのアクセスが行われると、必ずその要求者 (主体) の認証が行われます。

NIS+ が要求どおりの動作をするのは、「主体が認証されていて (正当な資格を持っていて)、要求された操作が、該当する構成要素において承認されている」という場合のみです。資格がない (または完全でない)、要求された操作が承認されていないという場合、NIS+ はアクセス要求を拒否します。NIS+ セキュリティシステムの詳細は、第 11 章を参照してください。

Solaris 1.x と NIS 互換モード

Solaris 1.x または 2.x では、NIS+ を NIS が動作しているマシンで使用できます。つまり、NIS+ ドメイン内にあるマシンはそれぞれの `nsswitch.conf` ファイルを `nisplus` ではなく `nis` に設定できます。NIS を実行中のマシン上で NIS+ のサービスにアクセスする場合は、NIS+ のサーバーを「NIS 互換モード」で動作させる必要があります。

NIS 互換モードを使用すれば、Solaris オペレーティング環境を実行している NIS+ サーバーは、NIS+ クライアントからの要求に応答しながら、NIS クライアントからの要求にも応答できます。NIS+ は 2 つのサービスインタフェースを提供することによってこれを実現します。1 つが NIS+ クライアントの要求に応答し、もう 1 つが NIS クライアントの要求に応答します。

このモードでは、NIS クライアントに対してさらに設定や変更を行う必要はありません。実際、NIS クライアントは、応答しているサーバーが NIS サーバーではないことを意識する必要はありません。ただし、NIS 互換モードで動作している NIS+ サーバーは `ypupdate` と `ypxfr` のプロトコルをサポートしないため、複製またはマスターの NIS サーバーとしては使用できません。NIS 互換モードの詳細については、第 26 章 を参照してください。

さらに 2 つの相違を指摘しておく必要があります。1 つは、NIS 互換モードでサーバーを設定する命令が標準の NIS+ サーバーの設定に使用される命令とは少し異なるということです。もう 1 つの相違としては、NIS 互換モードは、NIS+ の名前空間内のテーブルに対するセキュリティにも関係があります。NIS のクライアントソフトウェアは、NIS+ サーバーが NIS+ クライアントに与える資格 (credential) を提供する機能がないため、NIS クライアントからの要求はすべて「未認証」として分類されます。したがって、NIS クライアントが NIS+ テーブル内の情報にアクセスできるように、NIS+ の名前空間内のテーブルは未認証の要求に対してアクセス権を提供しなければなりません。これはパート II で説明するように、サーバーを NIS 互換モードで設定するために使用されるユーティリティによって自動的に処理されます。認証プロセスと NIS 互換モードについては、第 26 章を参照してください。

NIS+ の管理コマンド

NIS+ では、名前空間を管理するのに必要なコマンドがすべて提供されています。次の表に、これらのコマンドについてまとめます。

表 **2–2** NIS+ の名前空間管理コマンド

コマンド	説明
`nisaddcred`	NIS+ 主体用の資格を作成し、これらを `cred` テーブルに格納する
`nisaddent`	`/etc` ファイル、または、NIS マップの情報を NIS+ テーブルに追加する
`nisauthconf`	オプションで Diffie-Hellman キー長を設定する
`nisbackup`	NIS ディレクトリのバックアップコピーを取る
`nis_cachemgr`	NIS+ クライアント上で NIS+ キャッシュマネージャを起動する
`niscat`	NIS+ テーブルの内容を表示する
`nis_checkpoint`	ログには入力されたが、ディスクにチェックポイントが実行されていないデータについて強制的にチェックポイントを実行する
`nischgrp`	NIS+ オブジェクトのグループ所有者を変更する
`nischmod`	オブジェクトのアクセス権を変更する
`nischown`	NIS+ オブジェクトの所有者を変更する
`nischttl`	NIS+ オブジェクトの有効時間を変更する
`nisclient`	NIS+ 主体を初期化する
`nisdefaults`	NIS+ オブジェクトのデフォルト値 (ドメイン名、グループ名、マシン名、NIS+ 主体名、アクセス権、ディレクトリ検索パス、および生存期間) を表示する
`nisgrep`	NIS+ テーブル内のエントリを検索する
`nisgrpadm`	NIS+ グループの作成または削除、あるいはそのメンバーリストを表示する。また、グループにメンバーを追加または削除したり、グループメンバーかどうかのテストを行う
`nisinit`	NIS+ のクライアントまたはサーバーを初期設定する
`nisln`	2 つの NIS+ オブジェクト間でシンボリックリンクを作成する
`nislog`	NIS+ 処理用ログの内容を表示する
`nisls`	NIS+ ディレクトリの内容を表示する

表 **2–2** NIS+ の名前空間管理コマンド　*(続き)*

コマンド	説明
`nismatch`	NIS+ テーブル内のエントリを検索する
`nismkdir`	NIS+ のディレクトリを作成し、そのマスターサーバーと複製サーバーを指定する
`nispasswd`	NIS+ の `passwd` テーブルに格納されているパスワード情報を変更する (`nispasswd` よりは、`passwd` または `passwd -r nisplus` を使用する)
`nis_ping`	複製サーバーのデータをマスターサーバーのデータに強制更新する
`nispopulate`	新しい NIS+ ドメインに NIS+ テーブルを生成する
`nisprefadm`	クライアントが NIS+ サーバーから NIS+ 情報を検索する順序を指定する
`nisrestore`	以前にバックアップを取った NIS+ ディレクトリを復元する。新しい NIS+ 複製サーバーを急速にオンラインにするときにも使用する
`nisrm`	ディレクトリ以外の NIS+ オブジェクトを名前空間から削除する
`nisrmdir`	名前空間から、NIS+ ディレクトリとその複製を削除する
`nisserver`	新しい NIS+ サーバーを設定するときに使うシェルスクリプト
`nissetup`	`org_dir` ディレクトリと `groups_dir` ディレクトリ、および NIS+ ドメイン用の完全セットの (未生成) NIS+ テーブルを作成する
`nisshowcache`	NIS+ キャッシュマネージャによって管理される NIS+ 共有キャッシュの内容を表示する
`nisstat`	NIS+ サーバーに関する統計やその他の情報を報告する
`nistbladm`	NIS+ テーブルの作成または削除と、NIS+ テーブル内のエントリの追加、修正、または削除を行う
`nistest`	NIS+ 名前空間の現在の状態を報告する
`nisupdkeys`	NIS+ オブジェクトに格納されている公開鍵を更新する
`passwd`	NIS+ の `Passwd` テーブルのパスワード情報を変更するまたパスワードの経過時間やその他のパスワード関連のパラメータを管理する

NIS+ の API

NIS+ の API (アプリケーションプログラミングインタフェース) とは、アプリケーションが NIS+ のオブジェクトへのアクセスと修正を行うために呼び出す関数群です。NIS+ の API には 54 個の関数があり、それらは次の 9 つのグループに分類されます。

- オブジェクト操作関数 (`nis_names()`)
- テーブルアクセス関数 (`nis_tables()`)
- ローカル名関数 (`nis_local_names()`)
- グループ操作関数 (`nis_groups()`)
- アプリケーションサブルーチン関数 (`nis_subr()`)
- その他の関数 (`nis_misc()`)
- データベースアクセス関数 (`nis_db()`)
- エラーメッセージ表示関数 (`nis_error()`)
- トランザクションログ関数 (`nis_admin()`)

設定および構成の前に

NIS+ 名前空間を構成する前に、次の作業を行う必要があります。

- NIS+ を使用するすべてのマシンの `nsswitch.conf` ファイルを正しく構成します。詳細は 第 1 章 を参照してください。
- NIS+ 名前空間のレイアウトを設計します。これには次の 2 つの作業があります。
  - 名前空間の設計。どんなドメイン名を付けるか。サブドメインを作るか。サブドメインを作るとしたら、それらをどのように編成するか。どのマシンをどのドメインに入れるか。ドメインを上位のドメインあるいはインターネットに接続するか。
  - サーバー要件の確認。各ドメインに複製サーバーをいくつ置くか。必要なサーバーの種類、処理速度、メモリ容量はどれほどか。サーバーのディスク容量はどれくらい必要か。

    名前空間の設計および推奨値の詳細については、第 26 章を参照してください。
- 名前空間がすでに存在している場合、NIS+ に移行するための準備をします。86 ページの「名前空間がすでに存在する場合の設定」 を参照してください。
- ルートサーバーマシンを選択します。
- ルートマスターサーバーとして使用できるシステムが少なくとも 1 つは稼動していることを確認します。ルートマスターサーバーでは、`/etc/passwd` などのシステム情報ファイルに少なくとも 1 人のユーザー (root) が登録されていなければなりません 。通常のマシンのシステム情報ファイルには root が存在しますので、これが

問題となることはないはずです。

NIS と NIS+

NIS と NIS+ は、いくつかの同じ作業を行います。ただし NIS+では、NIS で提供されない階層ドメイン、名前空間セキュリティ、その他の機能が使用できます。NIS と NIS+ の相違点の詳細は、第 26 章を参照してください。

NIS 管理者は、以下に示す原則および条件下で NIS を NIS+ と共に使用できます。

- 同一ドメインに NIS サーバーと NIS+ サーバーの両方が存在するNIS 管理者は同一ドメインで NIS サーバーと NIS+ サーバーの両方を動作させることは可能ですが、このような動作を長時間行わせることは望ましくありません。一般に、同一ドメイン内での NIS サーバーと NIS+ サーバーを両方使用するのは、NIS から NIS+ への短い移行期間だけに制限するべきです。クライアントから NIS サービスの要求があった場合に、NIS 管理者は、NIS+ を NIS 互換モードで動作させることができます。詳細は、第 26 章を参照してください。
- サブドメインNIS 管理者のルートドメインのマスターサーバーで NIS+ が動作している場合は、NIS 管理者は、すべてのサーバーで NIS が動作しているサブドメインを設定できます。NIS 管理者のルートドメインのマスターサーバーで NIS が動作している場合は、NIS 管理者はサブドメインの設定はできません。この操作は、NIS 管理者が NIS から NIS+ に切り換えるときに有用であるかもしれません。たとえば、互いに独立した複数の NISドメイン (ドメインは地理的に別々のサイトに置かれていることもある) を持った会社の中で、NIS 管理者がそれらのドメインをすべてリンクさせて NIS+ の下に 1 つの階層型マルチドメイン名前空間を構築するケースを考えてみます。この場合、NIS 管理者はまずルートドメインを NIS+ 下に設定し、次に従来の NISドメインをサブドメインとして指定できます。これらのサブドメインでは、NIS+ への切り換えが行われるまで NIS が継続して動作します。
- 同一ドメインに複数のマシンが存在する
  - 同一ドメイン内のサーバーで NIS+ が動作している場合は、NIS+、NIS、または `/etc` ファイルを使ってネームサービス情報を取得するように、ドメイン内の各マシンを設定できます。NIS+ サーバーが NIS クライアントのニーズを満たすには、この NIS+ サーバーが NIS 互換モードで動作している必要があります。第 26 章を参照してください。
  - 同一ドメイン内のサーバーで NIS が動作している場合は、NIS または `/etc` ファイルを使ってネームサービス情報を取得するように、このドメイン内の各マシンを設定できます (各マシンは NIS+ を使用することはできません) 。

NIS+ のファイルとディレクトリ

表 2–3 は、NIS+ ファイルが格納される UNIX ™ ディレクトリの一覧です。

表 **2–3** NIS+ ファイルが存在する場所

ディレクトリ	存在するマシン	内容
`/usr/bin`	すべて	NIS+ ユーザーコマンド
`/usr/lib/nis`	すべて	NIS+ 管理者コマンド
`/usr/sbin`	すべて	NIS+ デーモン
`/usr/lib/`	すべて	NIS+ 共有ライブラリ
`/var/nis/data`	NIS+ サーバー	NIS+ サーバーの使用できるデータファイル
`/var/nis`	NIS+ サーバー	NIS+ ワーキングファイル
`/var/nis`	NIS+ クライアントマシン	NIS+ の使用するマシン固有のデータ

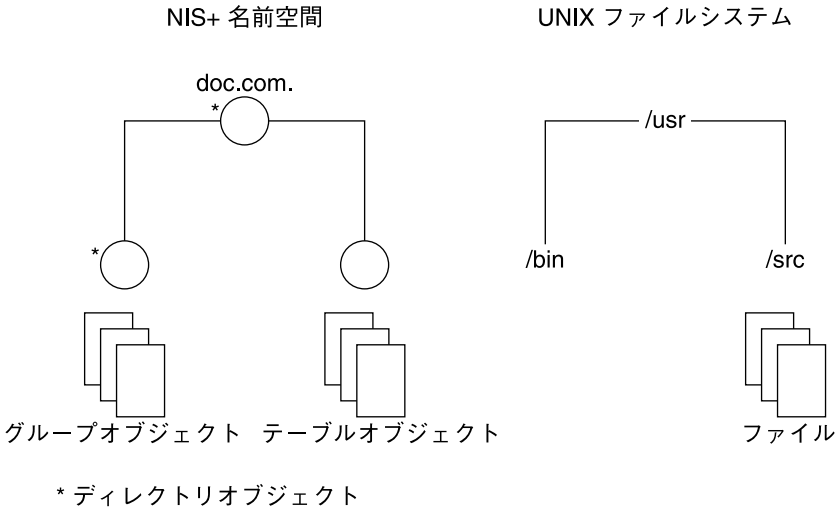
注意 – `nisinit` など NIS+ 設定プロシージャによって作成された `/var/nis`、`/var/nis/data` といったディレクトリ、およびその下のファイルは、名前を変更しないでください。Solaris 2.4 以前では、`/var/nis` ディレクトリに *hostname*`.dict`、*hostname*`.log` という 2 つのファイルが含まれていました。またサブディレクトリ `/var/nis/`*hostname* もありました。Solaris 2.5 を起動すると、2 つのファイル名は `trans.log`、`data.dict` となり、サブディレクトリ名は `/var/nis/data` となります。ファイルの「内容」も変更されており、Solaris 2.4 以前との互換性はなくなっています。したがって、これらのファイルやディレクトリを Solaris 2.4 での名前にしてしまうと、Solaris 2.4、2.5 双方の `rpc.nisd` で機能しなくなりますので名前を変更しないようにしてください。

注 – Solaris オペレーティング環境では、NIS+ データディレクトリ (`/var/nis/data.dict`) は、マシンに依存しません。そのため、NIS+ 名を簡単に変えることができます。また、NIS+ のバックアップコピーを使用したり機能を復元したりして、NIS+ のデータをひとつのサーバーから別のサーバーに転送もできます。第 21 章を参照してください。

NIS+ 名前空間の構造

NIS+ の名前空間は、NIS+ によって格納された情報を配置したものです。この名前空間は、組織のニーズに合わせていろいろな方法で配置することができます。たとえば、3 つの部門を持つ組織の場合、その NIS+ の名前空間も各部門ごとに 1 つずつで 3 つの部分に分割されます。分割した各部分は、その部門のユーザー、マシン、およびネットワークサービスについての情報を格納しますが、互いに通信することも簡単です。このような配置により、ユーザーによる情報へのアクセスや、管理者による情報の管理がさらに容易に行えます。

NIS+ の名前空間の配置はサイトごとに異なりますが、ディレクトリ、テーブル、グループという構成はすべてのサイトが使用します。これらの構成は NIS+「オブジェクト」と呼ばれます。NIS+ オブジェクトは、UNIX のファイルシステムに似た階層形式に配置できます。たとえば、次の図を参照してください。左側の名前空間は、3 つのディレクトリオブジェクト、3 つのグループオブジェクト、および 3 つのテーブルオブジェクトから構成されています。右側の UNIX ファイルシステムは、3 つのディレクトリと 3 つのファイルから構成されています。



NIS+ の名前空間は UNIX ファイルシステムに似ていますが、重要な違いが 5 つあります。

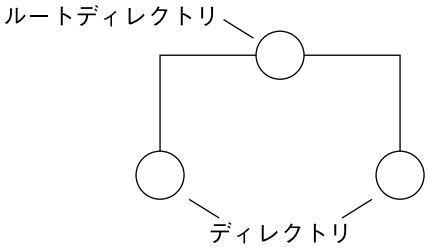
- 両方ともディレクトリを使用するが、NIS+ の名前空間でのオブジェクトはテーブルとグループであり、ファイルではない
- NIS+ の名前空間は、その目的 (システム管理ツール) のために設計された NIS+ の管理コマンド、または Solstice™ AdminSuite™ツールなどのグラフィカルユーザーインタフェース (GUI) を通じてだけ管理され、標準の UNIX ファイルシステ

ムコマンドや GUI では管理できない

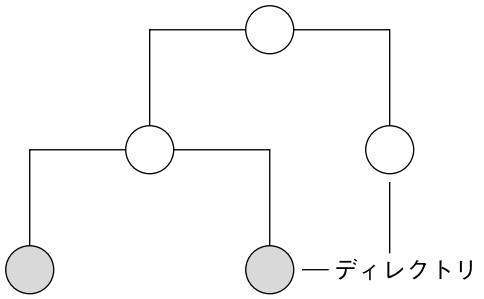
- UNIX ファイルシステム構成要素の名前はスラッシュで区切られる (`/usr/bin`) が、NIS+ の名前空間オブジェクト名はドットで区切られる (`doc.com.`)
- UNIX ファイルシステムの「ルート」へは、ディレクトリを右から左にたどっていけば到達する (`/usr/src/file1`) が、NIS+ の名前空間のルートは左から右にたどって到達する (`sales.doc.com.`)
- NIS+ オブジェクト名は、左から右に構成されているので、完全指定名は常にドットで終わる。また最後に「.」がないものは完全な名前とはみなされず、相対的な名前 (まだ上の階層があるという意味) であるとみなされる

ディレクトリ

ディレクトリオブジェクトは、名前空間の骨格を形成します。ツリー構造に配置した場合、名前空間を別々に分けます。ディレクトリ階層を理解するには、木の根を一番上に置き、逆さまの木として見るとよく分かります。名前空間の一番上のディレクトリが「ルート」ディレクトリです。名前空間が 1 つの層だけの場合、ディレクトリは 1 つしかありませんが、そのディレクトリはルートディレクトリです。ルートディレクトリの下にあるディレクトリオブジェクトは、単に「ディレクトリ」と呼ばれます。



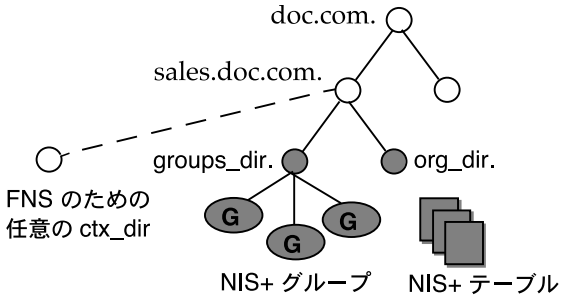
1 つの名前空間には、複数の階層のディレクトリを割り当てることができます。



ディレクトリ間の関係を識別する場合には、下位ディレクトリは「子」ディレクトリと呼ばれ、上位ディレクトリは「親」ディレクトリと呼ばれます。

UNIX のディレクトリは UNIX のファイルを入れるように設計されていますが、NIS+ のディレクトリは NIS+ オブジェクト (他のディレクトリ、テーブル、およびグループ) を入れるように設計されています。各 NIS+ のドメインレベルのディレクトリには、以下のサブディレクトリが含まれています。

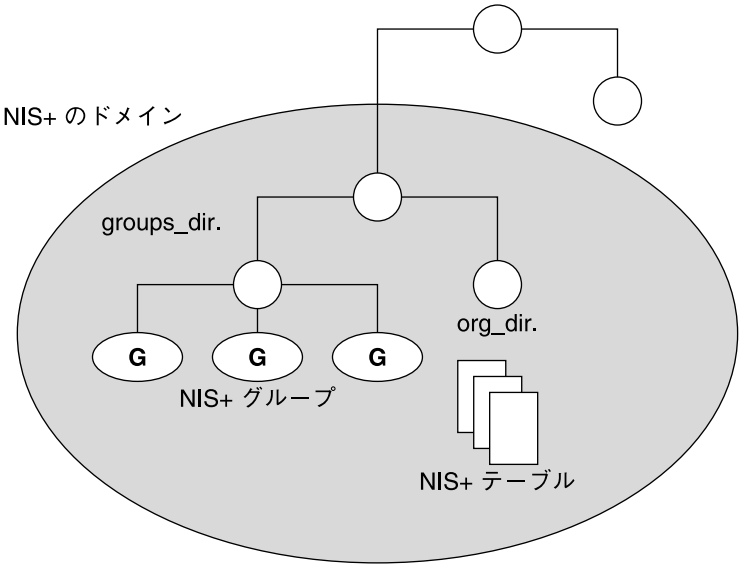
- `groups_dir.` - NIS+ グループ情報を格納する
- `org_dir.` - NIS+ システムテーブルを格納する
- `ctx_dir.` - FNS を使用しているときにだけ存在する



技術的には、ユーザーはディレクトリ、テーブル、およびグループをどんな構造にでも配置することができます。しかし、名前空間内の NIS+ のディレクトリ、テーブル、およびグループは、「ドメイン」と呼ばれる構成に配置されるのが普通です。ドメインは、名前空間の別々の部分をサポートするよう設計されています。たとえば、あるドメインが会社の営業部門 (Sales Division) をサポートし、別のドメインが製造部門 (Manufacturing Division) をサポートできます。

ドメイン

1 つの NIS+ のドメインは、ディレクトリオブジェクト、その `org_dir` ディレクトリ、その `groups_dir` ディレクトリ、および NIS+ テーブルのセットから構成されます。



NIS+ のドメインは、実際に存在する名前空間の構成要素ではありません。これらは単に、現実の組織をサポートするために使用される名前空間の一部を示すための便利な方法にすぎません。

たとえば、DOC 社に営業部門と製造部門があるとします。これらの部門をサポートするため、DOC 社の NIS+ 名前空間は、以下に示すような構造によって、3 つの主要ディレクトリグループに配置されることになります。

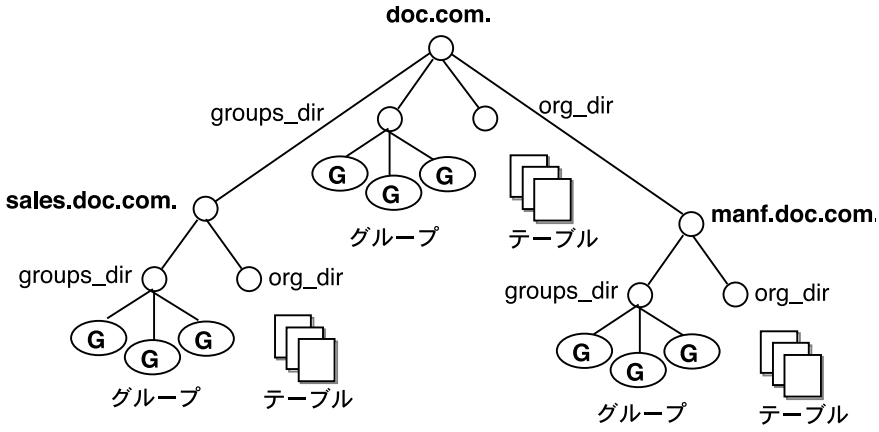


図 **2–2** NIS+ ディレクトリ構造の例

このような構造を、3 つのディレクトリ、6 つのサブディレクトリ、およびいくつかの追加オブジェクトと表現するよりも、3 つのドメインと表現する方が便利です。

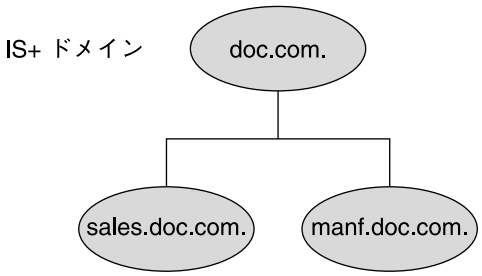
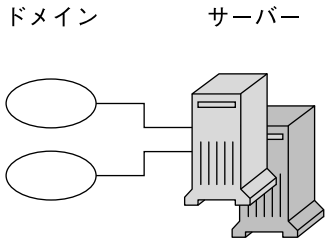


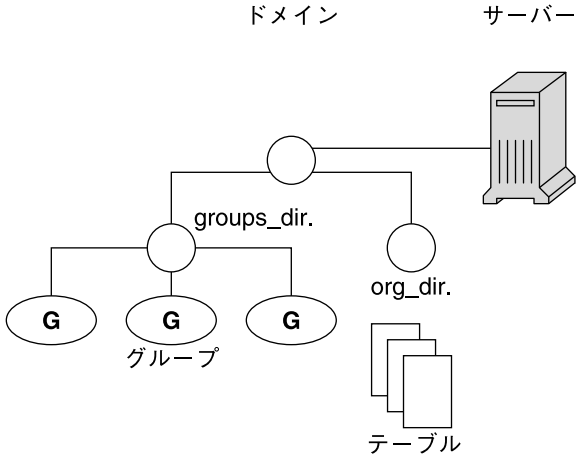
図 **2–3** NIS+ ドメインの例

サーバー

すべての NIS+ ドメインは複数の NIS+ サーバーによってサポートされます。サーバーは、ドメインのディレクトリ、グループ、およびテーブルを格納し、ユーザー、管理者、およびアプリケーションからのアクセス要求に応答します。各ドメインをサポートしているのは、一組のサーバーだけです。ただし、この 1 組のサーバーは複数のドメインをサポートできます。



ドメインはオブジェクトではなく、オブジェクトの集合を意味するものであることを忘れないでください。したがって、ドメインをサポートするサーバーは、実際にはドメインにではなく、ドメインのメイン「ディレクトリ」に関連付けられています。

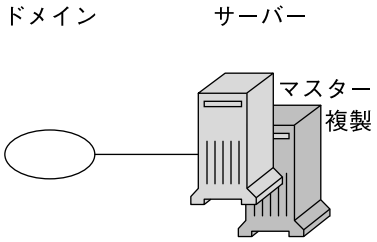


サーバーとディレクトリオブジェクトとのこうした接続は、ドメインを設定するときに行われます。ここで、重要なことを 1 つ指摘しておく必要があります。それは、この接続が確立されたときに、ディレクトリオブジェクトはそのサーバー名と IP アドレスを格納するということです。後述しますが、クライアントはこの情報を使用してサービス要求を送信します。

Solaris オペレーティング環境ベースのマシンは、どれも NIS+ サーバーとなることができます。各 Solaris リリースには、NIS+ のサーバー用とクライアント用の両方のソフトウェアが入っています。したがって、Solaris 2.x がインストールされているマシンであれば、サーバーにもクライアントにも、あるいはその両方にもなることができます。クライアントとサーバーを区別するのは、その果たす役割です。マシンが NIS + サービスを提供している場合、それは NIS+ サーバーとして機能しています。マシンが NIS+ サービスを要求している場合、これは NIS+ クライアントとして機能しています。

多数のクライアント要求にサービスを提供する必要があるため、NIS+ サーバーとして機能するマシンは、平均的なクライアントよりも高いコンピューティング能力と多くのメモリーを装備して構成される可能性があります。また、NIS+ データを格納する必要があるため、より大型のディスクも装備する可能性があります。しかし、性能を高めるためのハードウェアを除いては、サーバーと NIS+ クライアントとの本質的な違いはありません。

NIS+ ドメインをサポートするのは、マスターとその複製の 2 種類のサーバーです。



ルートドメインのマスターサーバーを「ルートマスター」サーバーと呼びます。1 つの名前空間には、 1 つのルートマスターサーバーしか存在しません。他のドメインのマスターサーバーは、単にマスターサーバーと呼びます。同様に、ルート複製サーバーと単なる複製サーバーがあります。

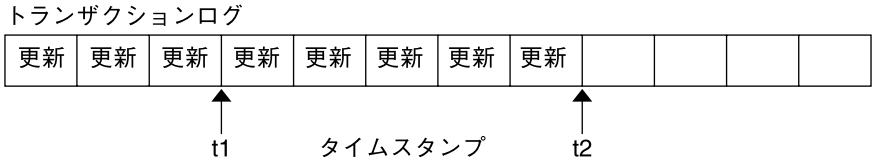
マスターサーバーと複製サーバーは、両方とも NIS+ テーブルを格納し、クライアント要求に応答します。ただし、マスターサーバーはドメインのテーブルのマスターコピーを格納し、複製サーバーは複製だけを格納します。管理者は、情報をマスターサーバー内のテーブルにロードし、マスターサーバーはそれを複製サーバーに伝達します。

この配置には 2 つのメリットがあります。第 1 に、マスターテーブルが 1 組しか存在しないことから、テーブル間の不一致を避けることができます。複製サーバーによって格納されたテーブルは、マスターのコピーにすぎません。第 2 に、これによって NIS+ サービスの「信頼性」が大幅に向上します。マスターまたは複製のどちらかがダウンした場合、他のサーバーがバックアップとして機能し、サービス要求に応えることができます。

サーバーが変更を伝達する方法

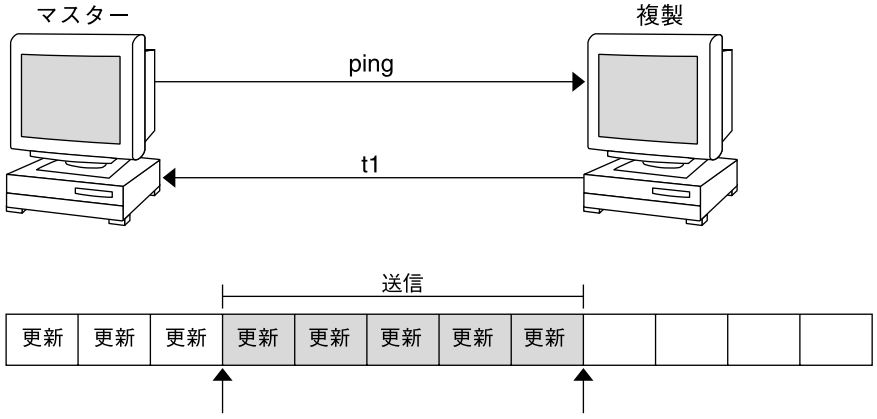
NIS+ のマスターサーバーは、そのオブジェクトをすぐに更新します。しかし、更新内容を複製サーバーに伝達する前に、複数の更新をまとめ (バッチ) ようと試みます。マスターサーバーが、ディレクトリ、グループ、リンク、またはテーブルといったオブジェクトへの更新を受信した場合、約 2 分間ほかの更新が到着しないかどうかを待機して確認します。待機時間が終了すると、これらの更新をディスクと「トランザクションログ」の 2 カ所に格納します (メモリーにはすでに更新を格納)。

マスターサーバーはトランザクションログを使用して、名前空間への変更内容が複製に伝達されるまで、これを格納します。トランザクションログには、更新とタイムスタンプという 2 つの要素が記録されます。



更新とは、変更されたオブジェクトの実際のコピーです。たとえば、ディレクトリが変更された場合、更新はそのディレクトリオブジェクトの完全なコピーです。テーブルエントリが変更された場合、更新は実際のテーブルエントリのコピーです。タイムスタンプは、マスターサーバーによって更新が行われた時間を示します。

その変更内容をトランザクションログに記録したのち、マスターは自分の複製にメッセージを送信し、送信すべき更新があることを知らせます。各複製は、マスターから受信した最後の更新のタイムスタンプでこれに応答します。するとマスターは、各複製のタイムスタンプ以降にログに記録した更新を複製に送信します。



マスターサーバーがその複製サーバーをすべて更新すると、トランザクションログをクリアします。ドメインに新しい複製が追加された場合などには、マスターは、トランザクションログに記録されている最も早い時間のタイムスタンプよりもさらに前のタイムスタンプを複製から受け取ることがあります。この場合、マスターサーバーは全面的な「再同期」、つまり「`resync`」を行います。`resync` では、マスターに格納されているすべてのオブジェクトと情報を該当複製にダウンロードします。`resync` 実行中には、マスターと複製の両方がビジー状態となります。複製は要求に応答できません。マスターは、読み取り要求には応答できますが、更新要求は受け付けできません。両方とも「`Server Busy - Try Again`」というメッセージで応答します。

NIS+ 主体とクライアント

NIS+ 主体とは、NIS+ サービスに要求をするエンティティ (クライアント) のことです。

主体

一般ユーザーであってもスーパーユーザー (root) であってもログインをすれば NIS+ 主体になります。実際の要求は、一般ユーザーの場合はクライアントユーザーから、スーパーユーザーの場合はクライアントマシンから送られます。つまり NIS+ の主体はクライアントユーザーの場合もクライアントマシンの場合もあります。

また NIS+ サーバーから NIS+ サービスを提供するエンティティも、NIS+ 主体になることができます。NIS+ サーバーは、すべて NIS+ クライアントと考えることができるので、ここでの説明のほとんどは NIS+ サーバーにも当てはまります。

クライアント

NIS+ クライアントとは、NIS+ サービスを受けるように設定されたマシンです。NIS+ クライアントの設定では、セキュリティ資格 (credential) を設定し、クライアントを適切な NIS+ グループのメンバーとし、そのホームドメインを確認し、スイッチ構成ファイルを確認し、最後に NIS+ 初期設定スクリプトを実行します (詳細は、パート II 「NIS+ の紹介と概要」を参照)。

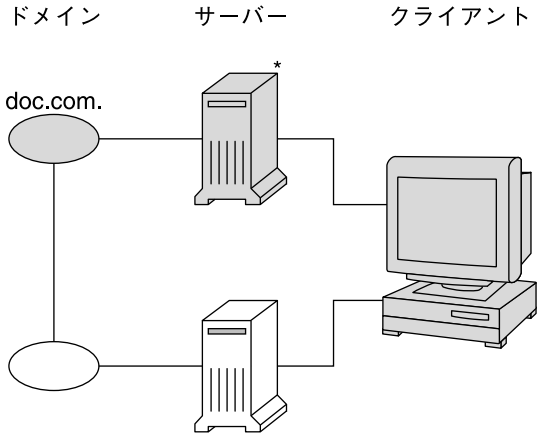
NIS+ クライアントは、セキュリティ上の制約に従って、名前空間のどの部分にもアクセスできます。つまり、認証されており、しかも適切なアクセス権が与えられている場合、名前空間内のどのドメインの情報やオブジェクトにもアクセスできます。

クライアントは名前空間全域にアクセスできますが、1 つのクライアントが「所属する」ドメインは 1 つだけであり、これをその「ホーム」ドメインと呼びます。クライアントのホームドメインは、インストール時に指定されるのが普通ですが、インストール後でも変更または指定できます。クライアントの IP アドレスや資格など、クライアントについてのすべての情報は、そのホームドメインの NIS+ テーブルに格納されています。

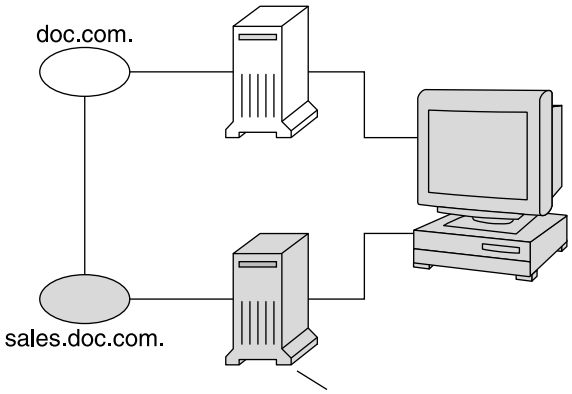
NIS+ クライアントであることと、NIS+ テーブルに登録されていることは、微妙な違いがあります。あるマシンについての情報を NIS+ テーブルに入力しても、そのマシンが自動的に NIS+ クライアントになるわけではありません。この操作は単に、すべての NIS+ クライアントがそのマシンの情報を使用できるようにするだけです。そのマシンを実際に NIS+ クライアントとして設定しない限り、NIS+ サービスを要求することはできません。

逆に、あるマシンを NIS+ クライアントにしても、そのマシンの情報が NIS+ テーブルに入力されるわけではありません。これは単に、そのマシンで NIS+ サービスを受けられるようにするだけです。管理者がそのマシンについての情報を明確に NIS+ テーブルに入力しない限り、他の NIS+ クライアントはその情報を入手できません。

クライアントが名前空間へのアクセスを要求した場合、実際には、クライアントは名前空間内の特定ドメインへのアクセスを要求していることになります。したがって、クライアントは、アクセスしようとしているドメインをサポートするサーバーに自分の要求を送信します。簡単に表現すると次のようになります。



* doc.com ドメイン内のオブジェクトにアクセスする場合、
クライアントはこのサーバーによってサポートされる



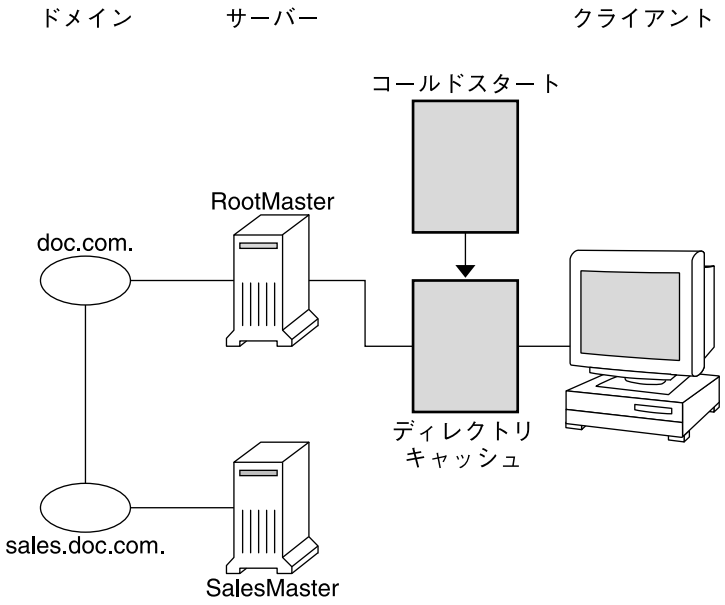
sales.doc.com ドメイン内のオブジェクトにアクセスする場合、
クライアントはこのサーバーによってサポートされる

クライアントはこのサーバーを、単純な試行錯誤によって認識します。クライアントは、自分のホームサーバーから始めて、正しいサーバーが見つかるまでサーバーを 1 台ずつ試していきます。サーバーがクライアントの要求に応じられない場合、サーバーは適切なサーバーの検索に役立つ情報をクライアントに送信します。やがて、クライアントは自分の情報キャッシュを構築し、適切なサーバーをより効率的に検索できるようになります。このプロセスの詳細を次に示します。

コールドスタートファイルとディレクトリキャッシュ

クライアントが初期設定されるとき、クライアントには「コールドスタートファイル」が与えられます。コールドスタートファイルの目的は、名前空間内のサーバーと連絡をとるための起点として使用できるディレクトリオブジェクトのコピーをクライアントに提供することです。ディレクトリオブジェクトには、アドレス、公開鍵、およびそのディレクトリをサポートするマスターサーバーと複製サーバーについての情報が収められています。通常、コールドスタートファイルには、クライアントのホームドメインのディレクトリオブジェクトが収められています。

コールドスタートファイルは、クライアントの「ディレクトリキャッシュ」を初期設定するためにだけ使用されます。ディレクトリキャッシュは、「キャッシュマネージャ」と呼ばれる NIS+ 機能によって管理され、クライアントが自分の要求を適切なサーバーに送信できるようにするためのディレクトリオブジェクトを格納しています。

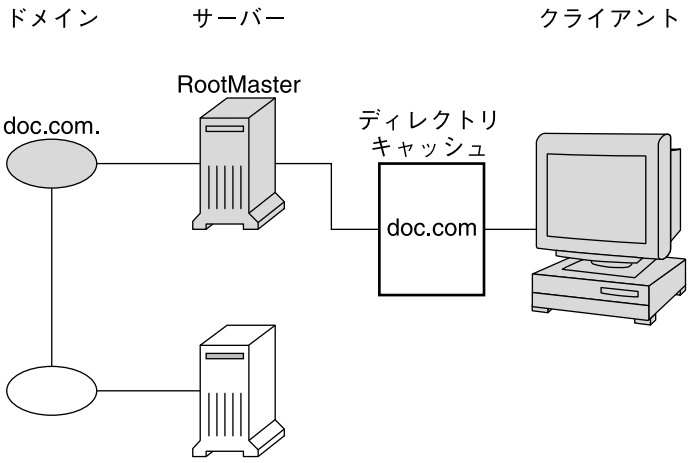


名前空間のディレクトリオブジェクトのコピーを自分のディレクトリキャッシュに格納することによって、クライアントは、どのサーバーがどのドメインをサポートするかを知ることができます。クライアントのキャッシュの内容を表示するには、`nisshowcache` コマンドを使用してください (346 ページの「`nisshowcache` コマンド」を参照)。簡単な例を次に示します。

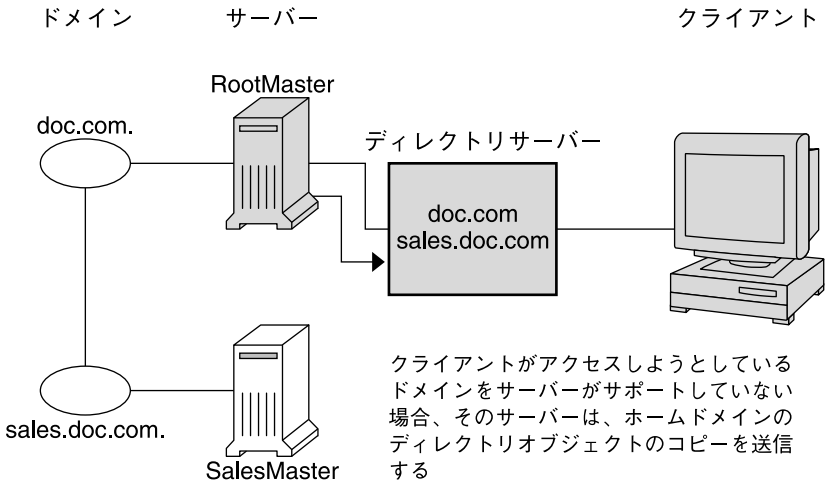
ドメイン名とディレクトリ名が同じ	サポートするサーバー	**IP** アドレス
`doc.com.`	rootmaster	123.45.6.77
`sales.doc.com.`	salesmaster	123.45.6.66
`manf.doc.com.`	manfmaster	123.45.6.37
`int.sales.doc.com.`	Intlsalesmaster	111.22.3.7

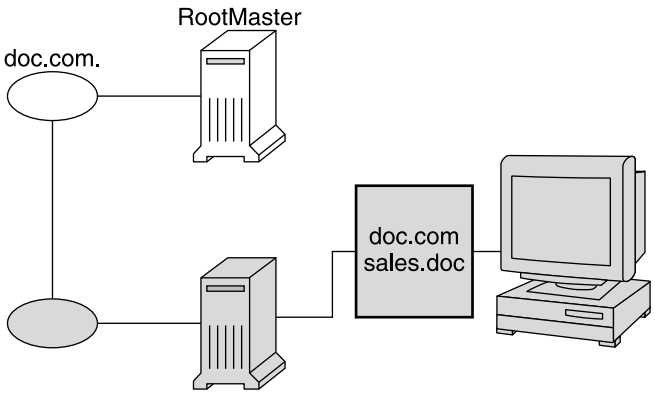
これらのコピーを最新の状態に保つため、各ディレクトリオブジェクトには「生存期間」フィールドがあります。このデフォルト値は 12 時間です。クライアントがディレクトリオブジェクトを求めてディレクトリキャッシュを調べ、最近の 12 時間以内にこれが更新されていないことを発見した場合、キャッシュマネージャはオブジェクトの新しいコピーを獲得します。351 ページの「`nischttl` コマンド」で説明するように、ディレクトリオブジェクトの生存期間の値は、`nischttl` コマンドで変更できます。ただし、生存期間を長くするほどオブジェクトのコピーが古くなる可能性が高くなり、生存期間を短くするほどネットワークトラフィックとサーバーのロードが増加することに注意してください。

ディレクトリキャッシュは、これらのディレクトリオブジェクトをどうやって蓄積するのでしょうか。前に説明したように、コールドスタートファイルはキャッシュ内の最初のエントリを提供します。したがって、クライアントが最初の要求を送信するとき、コールドスタートファイルによって指定されたサーバーに要求を送信します。その要求がそのサーバーによってサポートされるドメインへのアクセスである場合、そのサーバーは要求に応答します。



その要求が別のドメイン (たとえば、`sales.doc.com.`) へのアクセスである場合、サーバーは、クライアントが適切なサーバーを探すのを助けようとします。サーバーが、自分のディレクトリキャッシュ内に該当するドメイン用のエントリを保有している場合、そのドメインのディレクトリオブジェクトのコピーをクライアントに送信します。クライアントは、その情報を将来の参照用として自分のディレクトリキャッシュにロードし、そのサーバーに要求を送信します。





ほとんどありえないことですが、クライアントがアクセスしようとしているディレクトリオブジェクトのコピーをサーバーが保有していない場合、そのサーバーは、自分のホームドメインのディレクトリオブジェクトのコピーをクライアントに送信します。ここには、そのサーバーの親のアドレスが登録されています。クライアントは、親サーバーを使ってこのプロセスを繰り返し、適切なサーバーを見つけるか、または名前空間内の全サーバーを試し終わるまで、これを繰り返します。クライアントがドメイン内の全サーバーを試し終わった後でどうするかは、そのネームサービススイッチ構成ファイルの指示によって決まります。

やがてクライアントは、名前空間内のすべてのディレクトリオブジェクトのコピーをキャッシュ内に蓄積します。したがって、これらディレクトリオブジェクトをサポートするサーバーの IP アドレスも蓄積することになります。クライアントが他のドメインへのアクセス要求を送信する必要があるとき、通常は、自分のディレクトリキャッシュの中から自分のサーバー名を見つけ、そのサーバーにアクセス要求を直接送信できます。

NIS+ サーバーはクライアントでもある

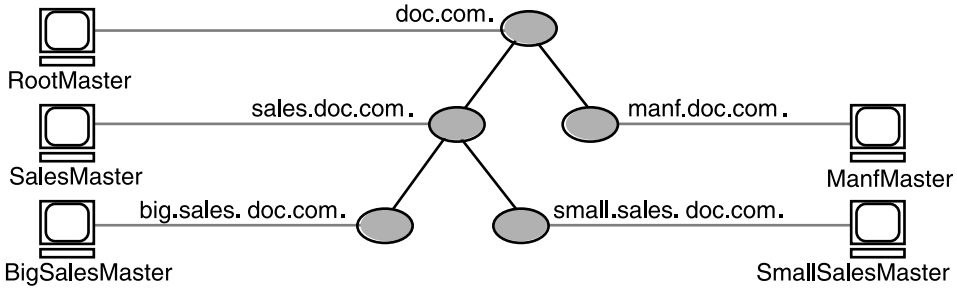
NIS+ サーバーは NIS+ クライアントでもあります。サーバーとして設定するマシンは、まずクライアントとして初期化する必要があります。唯一の例外はルートマスターサーバーであり、このサーバーには独自の設定を行う必要があります。

つまり、サーバーはドメインを「サポートする」だけではなく、ドメインにも「所属する」のです。つまり、クライアントになることによって、サーバーにはホームドメインがあるということです。サーバーのホスト情報は自分のホームドメインの `hosts` テーブルに格納され、その DES 資格は自分のホームドメインの `cred` テーブルに格納されます。他のクライアントと同様、サーバーは、自分のディレクトリキャッシュに記録されているサーバーに対して、サービス要求を送信します。

忘れてはならない重要な点は、ルートドメインを除いて、サーバーのホームドメインは、そのサーバーがサポートするドメインの「親」ドメインであるということです。

つまり、サーバーは 1 つのドメイン内のクライアントをサポートしますが、別のドメインの「クライアント」になるのです。ルートドメインを除いて、サーバーは自分のサポートするドメインのクライアントにはなれません。ルートドメインをサポートするサーバーには親ドメインがないため、これらはルートドメイン自体に所属します。

たとえば、次のような名前空間を考えてみます。



各サーバーがどのドメインをサポートし、どのドメインに所属するかを次に示します。

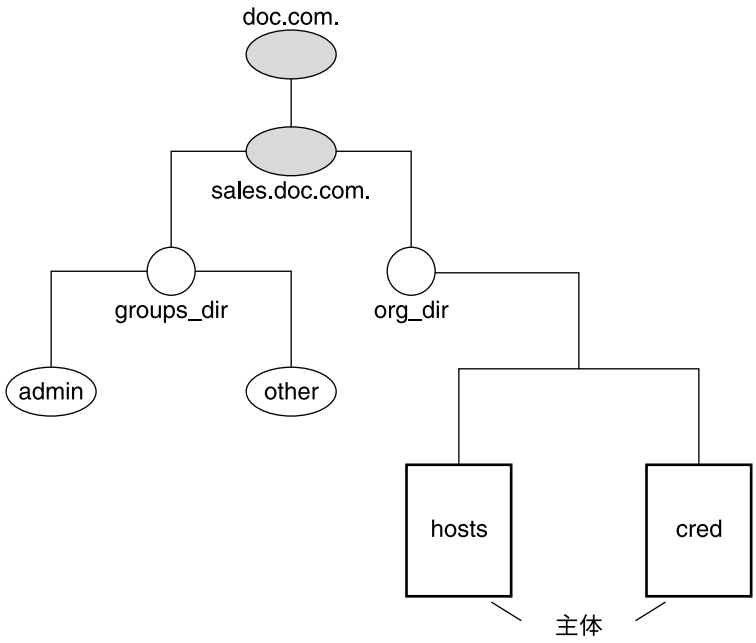
サーバー	サポートするドメイン	所属するドメイン
RootMaster	doc.com.	doc.com.
SalesMaster	sales.doc.com.	doc.com.
IntlSalesMaster	intl.sales.doc.com.	sales.doc.com.
ManfMaster	manf.doc.com.	doc.com.

命名規約

NIS+ 名前空間内のオブジェクトは、「部分指定」および「完全指定」の 2 種類の名前によって識別できます。部分指定名は、「単純」名とも呼ばれ、オブジェクト名だけまたは完全指定名の一部です。管理操作中にオブジェクトまたは主体の部分指定名を入力すると、NIS+ によってその名前は完全指定名に展開されます。詳細については、78 ページの「命名規約」 を参照してください。

完全指定名とは、名前空間内でオブジェクトを探すために必要なすべての情報を含んだオブジェクトの完全な名前です。必要なすべての情報とはオブジェクトの親ディレクトリ (もしあれば)、最後のドットも含んだ完全なドメイン名などです。

これはオブジェクトの種類によって異なるため、各タイプと NIS+ の主体ごとの規則を別個に説明します。次の名前空間を例として使用します。



NIS+ の主体を含めて、この名前空間内の全オブジェクトの完全指定名を次の図に要約します。

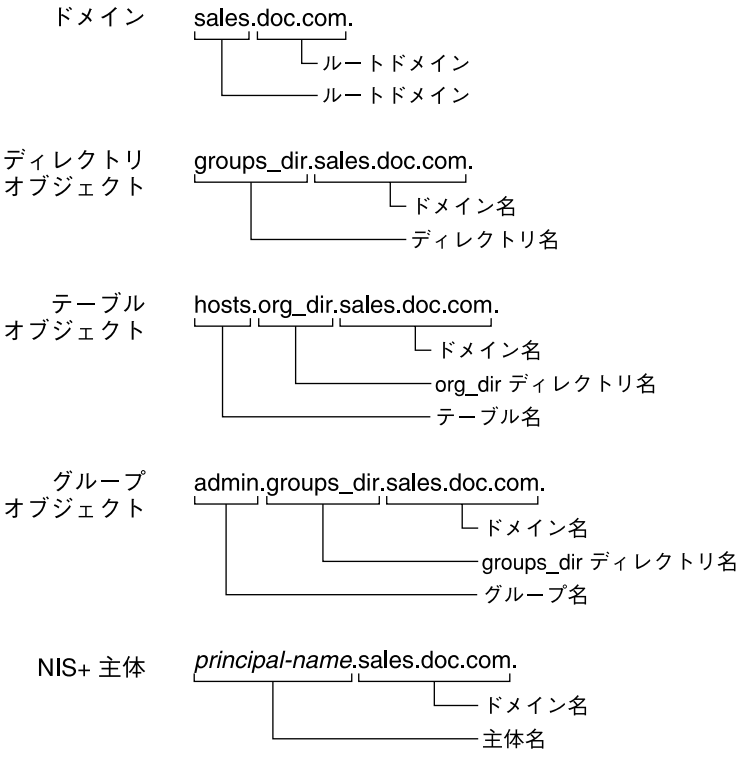


図 **2–4** 名前空間の構成要素の完全指定名

NIS+ ドメイン名

完全指定ドメイン名は左から右に作成され、ローカルドメインで始まってルートドメインで終わります。

`doc.com.` (ルートドメイン)

`sales.doc.com.` (サブドメイン)

`intl.sales.doc.com.` (サブドメイン)

最初の行はルートドメインの名前を示します。ルートドメインは、少なくとも 2 つのラベルを持ち、ドットで終わらなければなりません。最後 (右端) のラベルは自由につけられますが、インターネット上の互換性を維持するためには、以下に示すインターネットの組織名、または 2、3 文字の地域識別子 (日本であれば `.jp`) をつける必要があります。次の表を参照してください。

表 2–4 インターネットの組織ドメイン

ドメイン	種類
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

上の 2 行目と 3 行目は下位レベルドメインの名前です。

ディレクトリオブジェクト名

ディレクトリの単純名は、単にディレクトリオブジェクトの名前です。この完全指定名は、単純名にそのドメインの完全指定名を加えたものです (常に最後のドットを含む)。

groups_dir (単純名)

groups_dir.manf.doc.com. (完全指定名)

複数のディレクトリ層が 1 つのドメインを形成しないような変則的な階層を設定する場合、中間ディレクトリ名を含めるようにしてください。たとえば、次のようにします。

lowest_dir.lower_dir.low_dir.mydomain.com.

通常単純名は、同じドメイン内から使用され、完全指定名は、リモートドメインから使用されます。しかし、ドメインの NIS_PATH 環境変数に検索パスを指定することによって、リモートドメインから単純名を使用できます (84 ページの「環境変数 NIS _PATH」 を参照)。

テーブル名およびグループ名

完全指定されたテーブル名とグループ名は、オブジェクト名から始まり、ディレクトリ名をつけ、その後に完全指定されたドメイン名が続いて作られます。すべてのシステムテーブルオブジェクトは org_dir ディレクトリに格納されて、すべてのグループオブジェクトは groups_dir ディレクトリに格納されていることを忘れないでください。独自の NIS+ テーブルを作成した場合は、どこでも好きなところに格納できます。グループ名とテーブル名の例を次に示します。

```
admin.groups_dir.doc.com.
admin.groups_dir.doc.com.
admin.groups_dir.sales.doc.com.
admin.groups_dir.sales.doc.com.
hosts.org_dir.doc.com.
hosts.org_dir.doc.com.
hosts.org_dir.sales.doc.com.
hosts.org_dir.sales.doc.com.
```

テーブルエントリ名

NIS+ テーブル内のエントリを識別するには、その中にあるエントリとテーブルオブジェクトを識別する必要があります。このような名前を「インデックス付き」名と呼びます。この構文を次に示します。

[*column*=value,column=value,...],*tablename*

Column はテーブル列の名前です。*Value* はその列の実際の値です。*tablename* はテーブルオブジェクトの完全指定名です。`hosts` テーブルのエントリ例を次に示します。

```
[addr=129.44.2.1,name=pine],hosts.org_dir.sales.doc.com.
[addr=129.44.2.2,name=elm],hosts.org_dir.sales.doc.com.
[addr=129.44.2.3,name=oak],hosts.org_dir.sales.doc.com.
```

括弧の中には、テーブルエントリを一意に識別するために必要な最小限の列の値のペアを使用できます。

一部の NIS+ 管理コマンドは、この構文のバリエーションを利用できます。詳細は、パート II「NIS+ の紹介と概要」の `nistbladm`、`nismatch`、`nisgrep` の各コマンドの説明を参照してください。

ホスト名

ホスト名の長さは 24 文字までで、使用できるのは文字、数字、ダッシュ (-)、および下線 (_) です。大文字と小文字の区別はありません。また先頭は英文字とします。スペースは使用できません。

注 – ドット (.) はホスト名には使用できません たとえば、ホスト名には `myco.2` などは使用できません。また、引用符で囲んだドットも、ホスト名には使用できません。たとえば、 `'myco.2'` は使用できません。ドメイン構成要素を識別するために、完全指定ホスト名の一部として使用する場合にのみ、ドットを使用します。たとえば、 `myco-2.sales.doc.com.` は、正しい完全指定のホスト名です。

ドメインとホストで同じ名前を使用しないでください。たとえば、sales ドメインがある場合は、ホスト名に sales を使用することはできません。同様に、home という名前のマシンを使用する場合は、home という名前のドメインを作成しないでください。これは、サブドメインについても同様です。たとえば、すでにホスト名として west がある場合には、sales.west.myco.com というサブドメインを作ることはできません。

NIS+ の主体名

NIS+ の主体名は、Secure RPC のネット名とよく混同することがあります。念のためここで 1 つだけ違いを指摘しておきます。NIS+ の主体名は必ずドットで終わりますが、Secure RPC のネット名はドットで終わることは絶対ありません。

表 **2–5** NIS+ の主体名

olivia.sales.doc.com.	NIS+ 主体名
unix.olivia@sales.doc.com	Secure RPC ネット名

また、たとえ主体の資格が cred テーブルに格納されていても、cred テーブル名も org_dir ディレクトリ名も主体名には含まれません。

使用できる記号

名前空間名は、ISO ラテン 1 セットの印刷可能文字を自由に使用して作成できます。ただし、名前の先頭には次に示す文字は使用できません。 @ < > + [ ] - / = . , : ;

文字列を使用するには、二重引用符で囲みます。名前の中に引用符を使用するには、その記号も引用符で囲みます (たとえば、o'henry を使用するには o"'"henry と入力します)。John Smith などのように空白スペースを組み込むには、次のように一重引用符の中で二重引用符を使用します。

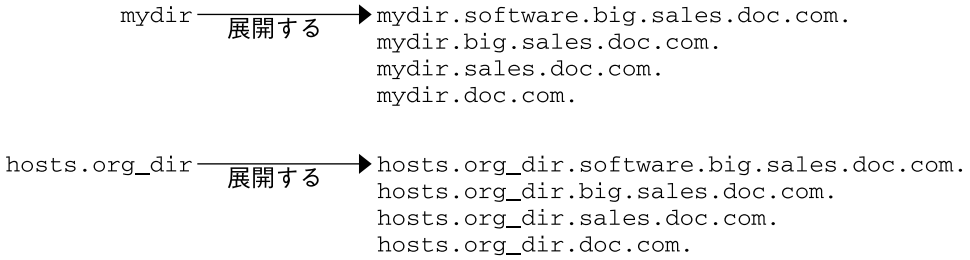
```
'"John Smith"'
```

ホスト名に適用される制限については 82 ページの「ホスト名」を参照してください。

NIS+ の名前展開

NIS+ のコマンドで完全指定名を入力することは手間がかかる作業です。この作業を簡単にするため、NIS+ は名前展開機能を提供します。部分指定名が入力されると、NIS+ は別のディレクトリのもとでこのオブジェクトを見つけようと試みます。最初は、デフォルトドメインから探します。これは、このコマンドを入力したクライ

アントのホームドメインです。デフォルトドメインでオブジェクトが見つからなかった場合、このオブジェクトが見つかるまで、NIS+ はデフォルトドメインの親ディレクトリを下から上へ探していきます。2 つのラベルしかない名前に到達すると、検索を停止します。次にその例を示します。この例で
は、`software.big.sales.doc.com.` ドメインに所属するクライアントにログインしたと仮定します。



環境変数 NIS_PATH

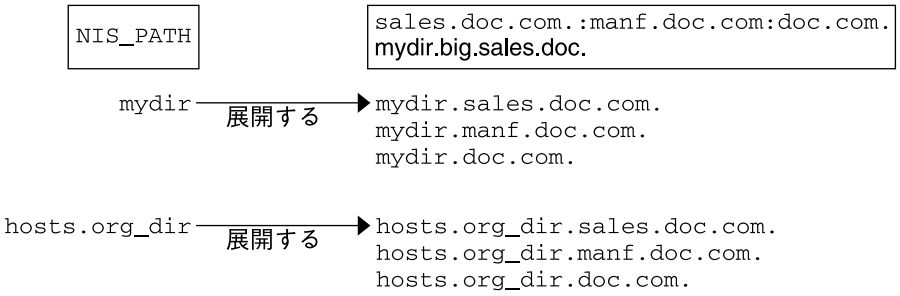
環境変数 NIS_PATH の値を変更することによって、NIS+ が検索するディレクトリのリストを変更または拡張できます。NIS_PATH には、コロンで区切って複数のディレクトリ名を設定できます。次に例を示します。

```
setenv NIS_PATH directory1:     directory2:directory3 ...
```

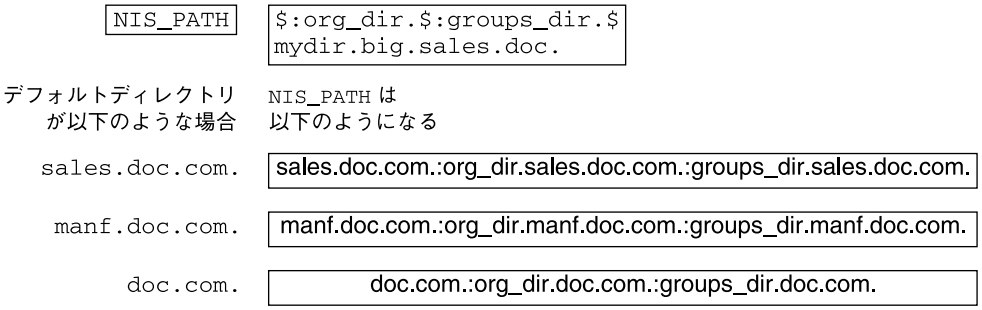
または

```
NIS_PATH=directory1:directory2:directory3 ...;export NIS_PATH
```

NIS+ は、これらのディレクトリを左から右へ検索していきます。たとえば、次のように検索されます。



$PATH や $MANPATH のように、変数 NIS_PATH には特殊記号 $ を使用できます。$ 記号は、ディレクトリ名に追加するか、または単独で追加できます。ディレクトリ名に追加した場合、NIS+ はその名前にデフォルトディレクトリを追加します。たとえば、次のようにします。



$ 記号を単独で使用する (たとえば、org_dir.$:$) 場合、NIS+ は前述のような標準の名前展開を実行します。つまり、デフォルトディレクトリの検索から始め、親ディレクトリへと進めていきます。NIS_PATH のデフォルト値は $ です。

注 – NIS_PATH に追加や変更を加えると、NIS+ がより多くの検索を行わなければならないので、性能が低下するという点に注意してください。

名前空間がすでに存在する場合の設定

NIS ドメインが NIS 名前空間にすでに存在する場合は、そのドメインを NIS+ 名前空間に同じフラット構造で移行することができます。移行したドメインは、あとで階層構造に変更できます。NIS から NIS+ に移行するときは、計画および準備方法について、第 26 章 を参照してください。NIS+ のスクリプトを使用すると、NIS マップのデータを利用して簡単に NIS+ を起動できます。第 4 章 で、NIS+ スクリプトを使用してシステムファイルや NIS マップから NIS+ 名前空間を作成する方法を説明します。

ただし、名前空間がすでに存在している場合、スクリプトをスムーズに実行するため NIS+ への移行用の設定が必要です。移行の準備の詳細については、第 26 章 を参照してください。

準備に関する主な注意事項は次のとおりです。

- 「ドメインとホストには同じ名前を使用しない」。たとえば、ドメインが `sales` の場合、マシンの名前には `sales` を付けないでください。同様に、`home` という名前のマシンを使用している場合は、`home` という名前のドメインは作成しないでください。これはサブドメインの場合も同様です。たとえば、マシンに `west` という名前を付けている場合、`sales.west.myco.com` というサブディレクトリを作成しないでください。
- 「ホスト名にはドットを使用しない」。NIS+ では、マシン名とドメイン名の間や、親とサブドメインの間の区切りにドット (ピリオド) を使用します。このため、マシン名の中ではドットを使用できません。使用している場合は、NIS+ に移行する前に (スクリプトを実行する前に) 必ず変更してください。ホスト名中のドットは、ハイフンに置き換えます。たとえば、`sales.alpha` というマシン名を付けることはできません。このホスト名は、`sales-alpha` に変換できます。
- 「ルートサーバーが動作していることを確認する」。ルートサーバーになるマシンが起動されていることを確認します。また、スーパーユーザーとしてアクセスできることも確認します。
- 「データのロード元のローカル `/etc` ファイル、または NIS マップのエントリを確認する」。不正なエントリがないことを確認します。正しいデータが、所定の場所に正しい書式で記録されていることを確認します。エントリのうち、古いもの、無効なもの、破損しているものは削除します。また、不完全なエントリや一部のみのエントリも削除します。構成完了後は、いつでもエントリを追加できます。必要なエントリをあとから追加する方が、不完全なエントリや破損しているエントリを読み込もうとするよりも簡単です。

注意 – Solaris 2.4 以前では、`/var/nis` ディレクトリに *hostname*`.dict`、*hostname*`.log` という 2 つのファイルと、サブディレクトリ `/var/nis/`*hostname* がありました。またサブディレクトリ `/var/nis/`*hostname* もありました。Solaris 2.5 の NIS+ では、2 つのファイル名は `trans.log`、`data.dict` となり、サブディレクトリ名は `/var/nis/data` となります。Solaris 2.5 ではこれらのファイルの内容も変更されており、Solaris 2.4 以前との互換性はなくなっています。したがって、これらのファイルやディレクトリを Solaris 2.4 での名前にしてしまうと、Solaris 2.4、2.5 双方の `rpc.nisd` で機能しなくなります。ディレクトリ名もファイル名も変更しないでください。

2 通りの構成方法

ここでは、NIS+ 名前空間の 2 通りの構成方法を紹介します。

- 「設定 (構成) 用のスクリプトを使う方法」`nisserver`、 `nispopulate`、および`nisclient` という 3 つの NIS+ スクリプトを使用して NIS+ を構成する方法については、この章と 第 4 章で説明しています。最も簡単な方法であるため、お勧めします。
- 「NIS+ コマンドを使う方法」NIS+ コマンドを使用して NIS+ を構成する方法については、第 6 章、第 7 章、および 第 8 章 で説明します。NIS+ コマンドを使用すれば、NIS+ スクリプトより柔軟に構成できますが、複雑になります。NIS+ コマンドは、NIS+ スクリプトによって生成される名前空間と特性が大きく異なる名前空間を構成する必要があるときに、経験豊富な NIS+ 管理者だけが使用してください。

注 – NIS+ コマンドセットを使用する場合は、ネームサービスに NIS+ を使用する各マシンの `/etc` ディレクトリに正しい `nsswitch.conf` ファイルが存在することも確認してください (第 1 章 を参照)。この作業は、NIS+ 構成スクリプトを使った場合は自動的に行われます。

NIS+ ディレクトリまたはドメイン、NIS+ サーバー、あるいは NIS+ 名前空間の削除方法については 第 22 章を参照してください。

第 3 章

NIS+ 設定スクリプト

この章では、NIS+ スクリプトとその機能について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ スクリプトについて

注意 – 必ず、次の章の「NIS+ 設定の概要」の手順を実行してから、NIS+ スクリプトを実行してください。

3 つの NIS+ スクリプト `nisserver`、`nispopulate`、`nisclient` を使用すれば、NIS+ 名前空間を設定できます。NIS+ スクリプトは NIS+ コマンド群を実行する Bourne シェルスクリプトであり、NIS+ コマンドを個別に入力する必要はありません。次の表に、各スクリプトの機能を示します。

表 **3–1** NIS+ スクリプト

NIS+ スクリプト	機能
`nisserver`	ルートマスターサーバー、非ルートマスターサーバー、複製サーバーをレベル 2 のセキュリティ (DES) で設定する

表 **3–1** NIS+ スクリプト　*(続き)*

NIS+ スクリプト	機能
`nispopulate`	NIS+ テーブルを、対応するシステムファイルまたは NIS マップから指定されたドメイン内に生成する
`nisclient`	ホストとユーザーの NIS+ 資格を作成し、NIS+ のホストとユーザーを初期設定する

NIS+ スクリプトで実行すること

NIS+ スクリプトといくつかの NIS+ コマンドと組み合わせることで、NIS+ 名前空間の設定に必要な作業を実行できます。NIS+ スクリプトとそのオプションについての詳しい説明は、`nisserver(1M)`、`nispopulate(1M)`、`nisclient(1M)` のマニュアルページを参照してください。第 4 章では、NIS+ スクリプトを使用した NIS+ 名前空間の設定方法について説明しています。

`-x` オプションを使用して、コマンドを実際に実行せずに各スクリプトを実行できます。このオプションを使用すると、スクリプトが呼び出すコマンドとおおよその結果をシステムを変更せずに確認できます。`-x` オプションを使用してスクリプトを実行することで、予想外のエラーの発生を最小限に抑えることができます。

NIS+ スクリプトでは実行しないこと

NIS+ スクリプトを使用すれば NIS+ 名前空間の作成に必要な作業が軽減されるとはいえ、スクリプトですべての NIS+ コマンドを代行できるわけではありません。スクリプトは、NIS+ の一部の機能を提供するだけです。NIS+ にまだ慣れていない場合、NIS+ 名前空間のサンプルを作成してからこの節を読むこともできます。

`nisserver` スクリプトは、標準のテーブルとアクセス権 (承認) で NIS+ サーバーを設定するだけです。このスクリプトでは次のことは実行しません。

- テーブルとディレクトリに特別なアクセス権を設定する
- NIS+ 管理グループへ NIS+ 主体を追加する

  NIS+ スクリプトではなく `nisgrpadm` コマンドを使用して、NIS+ 管理グループに特別な NIS+ 主体を追加する方法については、第 4 章を参照してください。
- 独自のテーブルを作成する
- レベル 2 以外の任意のセキュリティレベルで NIS+ サーバーを実行する
- リモートサーバーで `rpc.nisd` デーモンを起動する。この作業は、サーバーのインストールを完了するために必要

NIS+ スクリプトではなく `rpc.nisd` コマンドを使用して、NIS+ クライアントマシンを非ルートサーバーに変更する方法については、第 4 章を参照してください。

`nisclient` スクリプトは、DNS を使用してホスト名を検索するように NIS+ クライアントを設定することはありません。この設定を必要とするクライアントの場合、DNS を明確に設定しなければなりません。

第 4 章

スクリプトを使用した NIS+ の設定

この章では、nisserver、nispopulate、および nisclient スクリプトといくつかの NIS+ コマンドを組み合わせて、基本的な NIS+ 名前空間を構成する方法について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ 設定の概要

NIS+ 名前空間を設定および構成するときは、NIS+ スクリプトを使用することをお勧めします。NIS+ コマンドより簡単に NIS+ 名前空間を設定できます。第 6 章、第 7 章、および 第 8 章を参照してください。

(スクリプトの詳細は、nisserver(1M)、nispopulate(1M)、nisclient(1M) の各マニュアルページを参照してください。用語や略語の定義については、用語集を参照してください。)

このチュートリアルで説明するサンプルの小さな NIS+ 名前空間を土台にして実際の NIS+ 名前空間を作ることはしないでください。名前空間についてひととおり理解できたら、サンプルの名前空間は削除してください。サンプルの名前空間に実際の名前空間を追加しないでください。あらためて、初めから NIS+ 階層の計画を注意深く立ててから、実際の名前空間を作成してください。

一般に推奨される設定手順を表 4–1 に要約します。左端の列は、ルートドメインの構成やクライアントの作成などの主な設定作業です。中央の列は、作業の説明です。3 番目の列は、各手順に必要なスクリプトまたは NIS+ コマンドです。

表 **4–1** NIS+ の推奨構成手順の概要

作業	説明	スクリプトまたは **NIS+** コマンド
NIS+ 名前空間の計画を立てる	NIS+ 名前空間の計画を立てる。計画の要件と手順の詳細は、609 ページの「NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする」 を参照。(試験的なネットワークで NIS+ チュートリアルを使用している場合、この手順は不要)	
既存の名前空間を設定する	スクリプトを正常に実行するためには、既存の名前空間を適切に設定する必要がある。必要な準備については、 および 609 ページの「NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする」 を参照。(試験的なネットワークで NIS+ チュートリアルを使用している場合、この手順は不要)	
Diffie-Hellman キー長を設定する	DES 認証を使用する場合には、デフォルトの 192 ビットよりも長い Diffie-Hellman キーの使用を考慮する。キーを長くする場合、その長さはドメイン内のすべてのマシンで同一である必要がある。各初期設定スクリプトの実行する前に、目的のキー長を指定する	`nisauthconf`
ルートドメインの構成	ルートドメインを作成する。ルートマスターサーバーの構成と初期設定を行う。ルートドメイン管理グループを作成する	`nisserver`

テーブルの生成	テキストファイルまたは NIS マップからルートドメインの NIS+ テーブルを生成 (populate) する。ルートドメインクライアントの資格 (credential) を作成する。管理者の資格を作成する	`nispopulate`
		`nisgrpadm`
		`nisping`

ルートドメインクライアントの構成	クライアントマシンを構成する (そのうちの何台かは続いてサーバーになる)。ユーザーを NIS+ クライアントとして初期設定する	`nisclient`
サーバーを使用可能にする	ルートドメインの一部のクライアントをサーバーとして使用可能にする。サーバーの一部は後でルート複製サーバーとなり、そのほかは下位レベルのドメインをサポートする	`rpc.nisd`
ルート複製サーバーの構成	構成したばかりのサーバーのうち、1 台または複数をルートドメインの複製として指定する	`rpc.nisd` `nisserver`
非ルートドメインの構成	新しいドメインを作成する。以前の手順で使用可能にしたサーバーを、そのマスターとして指定する。その管理グループと管理資格を作成する	`rpc.nisd` `nisserver`

表 **4–1** NIS+ の推奨構成手順の概要　(続き)

作業	説明	スクリプトまたは **NIS+** コマンド
テーブルの生成	新しいドメインのクライアントの資格を作成する。テキストファイルまたは NIS マップから、新しいドメインの NIS + テーブルを生成する	`nispopulate`
非ルートドメインのクライアントを構成する	新しいドメインのクライアントを構成する (一部のクライアントは、後に下位レベルのドメインのサーバーとなる場合がある)。ユーザーを NIS+ クライアントとして初期設定を行う	`nisclient`

NIS+ スクリプトを使用することによって、上の作業で示される個々の手順の大部分を省略できます。

NIS+ 名前空間サンプルの作成

この章では、サンプルの NIS+ 名前空間を作成する方法を説明します。NIS+ 名前空間のサンプルは、`/etc` 内のファイルと NIS マップから作成されます。このサンプルでは、サイトで NIS を実行している場合と実行していない場合の両方について、スクリプトの使用方法を説明します。サーバーが NIS クライアントにサービスを提供する場合、サーバーを NIS 互換モードに設定できます。NIS 互換モードの詳細については、第 26 章を参照してください。

注 – サイトの実際の NIS+ 名前空間とそのドメイン階層は、名前空間サンプルのものとはおそらく異なり、サーバー、クライアント、およびドメインの「数」も異なります。最終的なドメイン構成と階層は、このサンプルと異なるものと考えてください。この名前空間サンプルは、NIS+ スクリプトの使用法を説明するためだけのものです。この名前空間サンプルを作成すると、自分のサイトでのドメイン、サーバー、およびクライアントの作成方法も理解できるはずです。

名前空間サンプルには次の構成要素があります。

- ルートマスターサーバー 1 台 (名称 `master`、`doc.com` ドメイン用)
- ルートドメイン (`doc.com.`) のクライアント 4 台
  - `client1` は、ルート複製サーバーとなる (`doc.com.` ドメイン用)
  - `client2` は、新しいサブドメインのマスターサーバーとなる (`sub.doc.com.` ドメイン用)
  - `client3` は、新しいサブドメインの非ルート複製サーバーとなる (`sub.doc.com.` ドメイン用)

- client4 は、ルートドメインのクライアントとして残る (doc.com. ドメイン用)

- サブドメインの 2 台のクライアント (名称 subclient1 および subclient2、sub.doc.com. ドメイン用)

ここでは、/etc/hosts などのシステム情報ファイルと NIS マップの両方を使用してネットワークサービス情報を格納するサイトで、NIS+ の設定に使用されるスクリプトを説明します。NIS+ 名前空間サンプルでこのような混合型のサイトを使用するのは、単にサンプルとして示すことが目的です。

NIS+ スクリプトのコマンド行の要約

表 4–2 に、NIS+ ドメインのサンプルを作成するときの、NIS+ スクリプトとコマンドの汎用的な手順を示します。このあとの節ではこれらのコマンド行について詳しく説明します。NIS+ のドメイン、サーバー、およびクライアントの作成に必要な作業に習熟した後、表 4–2 はコマンド行のクイックリファレンスガイドとして使用してください。表 4–2 は、NIS+ 名前空間サンプルを作成するために実際に入力するコマンドと変数をまとめたものです。

表 **4–2** NIS+ ドメインの構成に使うコマンド行のまとめ

アクション	対象マシン	コマンド
/usr/lib/nis をルートのパスに追加する。C シェルまたは Boure シェルを使用	ルートマスターサーバーとクライアントマシン。スーパーユーザーとして行う	setenv PATH $PATH:/usr/lib/nis または PATH=$PATH:/usr/lib/nis; export PATH
オプションで DES 認証を使用する場合には、Diffie-Hellman キー長を選択する	サーバーとクライアントマシン。スーパーユーザーとして行う	nisauthconf -dh*key-length-alg-type* des
NIS (YP) との互換性があるか、または互換性がないルートマスターサーバーを作成する	ルートマスターサーバー。スーパーユーザーとして行う	nisserver -r-d*newdomain*. または nisserver - Y-r-d *newdomain*.
ファイルまたは NIS マップからルートマスターサーバーテーブルを生成する	ルートマスターサーバー。スーパーユーザーとして行う	nispopulate -F-p */files* -d *newdomain*. または nispopulate -Y-d *newdomain*. -h *NISservername* -a *NIS_server_ipaddress* -y *NIS_domain*
NIS+ 管理グループにユーザーを追加する	ルートマスターサーバー。スーパーユーザーとして行う	nisgrpadm -a admin. *domain.name.domain*.

表 **4–2** NIS+ ドメインの構成に使うコマンド行のまとめ　*(続き)*

アクション	対象マシン	コマンド
NIS+ データベースのチェックポイントを実行する	ルートマスターサーバー。スーパーユーザーとして行う	`nisping -C` *domain.*
新しいクライアントマシンを初期設定する	クライアントマシン。スーパーユーザーとして行う	`nisclient - i -d` *domain.* `-h` *master1*
ユーザーを NIS+ クライアントとして初期設定する	クライアントマシン。ユーザーとして行う	`nisclient -u`

`rpc.nisd` デーモンを起動 – NIS (および DNS) との互換性なし、またはありでクライアントをサーバーにするために必要	クライアントマシン。スーパーユーザーとして行う	`rpc.nisd`
		または
		`rpc.nisd -Y`
		または
		`rpc.nisd -Y -B`

サーバーをルート複製サーバーにする	ルートマスターサーバー。スーパーユーザーとして行う	`nisserver -R -d` *domain.* `-h` *clientname*
サーバーを非ルートマスターサーバーにする	ルートマスターサーバー。スーパーユーザーとして行う	`nisserver -M -d` *newsubdomain.domain.* `-h \` *clientmachine*
ファイルまたは NIS マップから新しいマスターサーバーテーブルを生成する	新しいサブドメインマスターサーバー。スーパーユーザーとして行う	`nispopulate -F -p` */subdomaindirectory* `-d \` *newsubdomain.domain.* または `nispopulate -Y -d` *newsubdomain.domain.* `-h \` *NISservername* `-a` *NIS_server_ipaddress* `-y` *NIS _domain*
クライアントをマスターサーバーの複製にする	サブドメインマスターサーバー。スーパーユーザーとして行う	`nisserver -R -d` *subdomain.domain.* `-h \` *clientname*
サブドメインの新しいクライアントを初期設定する。クライアントは、サブドメインの複製またはそのほかのサーバーにできる	新しいサブドメインクライアントマシン。スーパーユーザーとして行う	`nisclient -i -d` *newsubdomain.domain* `. -h \` *subdomainmaster*
ユーザーを NIS+ クライアントとして初期設定する	クライアントマシン。ユーザーとして行う	`nisclient -u`

注 – 実際にコマンドを実行させずに、NIS+ スクリプトがどんなコマンドを呼び出すかを表示するには、-x オプションを使います。-x オプションによって、あたかも実際にスクリプトを実行しているかのように、それらのコマンド名とそれぞれのおおよその出力を画面に表示できます。最初に -x を指定してスクリプトを実行すれば、結果を先に見ることができます。詳細は各スクリプトのマニュアルページを参照してください。

NIS+ ルートサーバーの設定

NIS+ ドメインを設定するための最初の作業が、ルートマスターサーバーの設定です。この節では、nisserver スクリプトを使用してデフォルト設定でルートマスターサーバーを構成する方法を説明します。ルートマスターサーバーは、次のデフォルトを使用します。

- セキュリティレベル 2 (DES) - NIS+ セキュリティの最高レベル
- NIS 互換を OFF に設定 (NIS 互換に設定する方法も記述)
- ネームサービス情報の情報源としてシステム情報ファイル (/etc) または NIS マップ
- NIS+ グループとして admin. *domainname*

注 – nisserver スクリプトは、ルートマスターサーバーを設定するときに、ネームサービススイッチファイルを NIS+ 用に変更します。/etc/nsswitch.conf ファイルは後で変更できます。ネームサービススイッチについては、第 1 章を参照してください。

nisserver を実行するための前提条件

ルートマスターサーバーにしたいマシンの /etc/passwd ファイルに root のエントリがあるかどうかを確認します。

必要な情報

次の情報が必要です。

- ルートマスターサーバーとなるマシンのスーパーユーザーパスワード

- 新しいルートドメインの名前。ルートドメイン名は少なくとも 2 つの要素 (ラベル) で構成され、その末尾にドット (ピリオド) が打たれていなければならない (例: *something*.com.)。最後の要素は実際には何でもよいですが、インターネットとの互換性を保つには、インターネット組織コード (表 4–3) または 2 〜 3 文字の地域コード (日本であれば .jp.) にしなければなりません。

表 **4–3** インターネットの組織ドメイン

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

次の例では、ルートマスターサーバーに指定するマシンは master1 で、doc.com. が新しいルートドメインになります。

注 – ドメインとホストで同じ名前を使用しないでください。たとえば、ルートドメインに doc.com. という名前を付けている場合、ドメイン内で使用するマシンには doc という名前は付けないでください。同様に、home という名前のマシンを使用する場合は、home という名前のドメインを作成しないでください。これはサブドメインの場合も同様です。たとえば、マシンに west という名前を付けている場合、sales.west.myco.com というサブドメインを作成しないでください。

▼ ルートマスターサーバーを作成する方法

1. スーパーユーザーの **PATH** 変数に /usr/lib/nis を追加します。

   このパスを root の .cshrc または .profile ファイルに追加するか、または変数を直接設定します。

2. **DES** 認証を使用する場合には、オプションで、**Diffie-Hellman** キー長を指定します。

   640 ビットの Diffie-Hellman キーとデフォルトの 192 ビットのキーを使用するには、以下を入力します。

   ```
   nisauthconf dh640-0 des
   ```

   640 ビットのキーだけを許可し、192 ビットのキーを拒否するには、以下を入力します。

```
nisauthconf dh640-0
```

3. 次のコマンドをスーパーユーザー **(root)** として入力し、ルートマスターサーバーを構成します。

   `-r` オプションはルートマスターサーバーを構成することを示します。`-d` オプションは、NIS+ ドメイン名を指定します。

```
master1# nisserver -r -d doc.com.
This script sets up this machine “master1” as a NIS+ root master
server for domain doc.com.
Domain name : doc.com.
NIS+ group : admin.doc.com.
NIS (YP) compatibility : OFF
Security level : 2=DES
Is this information correct? (type ’y’ to accept, ’n’ to change)
```

   「NIS+ グループ」は、`doc.com.` ドメイン (ドメイン名は常にピリオドで終了する) の情報を変更する権限を持つユーザーのグループです。NIS+ グループは、ドメインを削除することもできます。NIS+ グループのデフォルト名は、`admin.`*domainname* です。この名前の変更方法については、101 ページの「誤った情報を変更する方法」を参照してください。

   「NIS 互換」とは、NIS+ サーバーが NIS クライアントからの情報要求を受け付けるかどうかを意味します。デフォルト設定の `OFF` に設定されている場合、NIS+ サーバーは NIS クライアントからの要求を処理しません。`ON` に設定されている場合、NIS+ サーバーはこの要求を処理します。このスクリプトでは、NIS 互換設定を変更できます。101 ページの「誤った情報を変更する方法」を参照してください。

注 – このスクリプトは、マシンを NIS+ セキュリティの最高レベルであるセキュリティレベル 2 で設定します。このスクリプトを使用する場合、セキュリティレベルを変更できません。スクリプトが終了した後、適切な NIS+ コマンドでセキュリティレベルを変更できます。セキュリティレベルの変更の詳細については、`rpc.nisd` のマニュアルページを参照してください。

---

4. `y` を入力します **(**画面の情報が正しい場合**)**。

   `n` を入力すると、スクリプトは正しい情報の入力を要求します。`n` を入力した場合の操作については、101 ページの「誤った情報を変更する方法」を参照してください。

```
Is this information correct? (type ’y’ to accept, ’n’’ to change)
y
This script will set up your machine as a root master server for
domain doc.com. without NIS compatibility at security level 2.
Use "nisclient -r" to restore your current network service environment.
Do you want to continue? (type ‘y’ to continue, ‘n’ to exit the script)
```

5. **y** を入力して、**NIS+** の構成を続けます。

   (`n` を入力するとスクリプトは安全に終了します)。`y` を選んだ後でスクリプトの実行中にスクリプトを中断した場合、スクリプトは動作を停止し、それまでに作成された

構成内容はそのまま残されます。スクリプトは自動回復や後始末は行いません。このスクリプトはいつでも再実行できます。

```
Do you want to continue? (type 'y' to continue, 'n' to exit the script
y
setting up domain information "doc.com." ...
setting up switch information ...
running nisinit ...
This machine is in the doc.com. NIS+ domain.
Setting up root server ...
All done.
starting root server at security level 0 to create credentials...
running nissetup ...
(creating standard directories & tables)
org_dir.doc.com. created
Enter login password:
```

`nissetup` コマンドは、各 NIS+ テーブルのディレクトリを作成します。

6. プロンプトに対してマシンの **root** パスワードを入力し、**Return** キーを押します。

   この例の場合、ユーザーは `master1` マシンの root パスワードを入力しています。

```
Wrote secret key into /etc/.rootkey
setting NIS+ group to admin.doc.com. ...
restarting root server at security level 2 ...
This system is now configured as a root server for domain doc.com.
You can now populate the standard NIS+ tables by using the
nispopulate or /usr/lib/nis/nisaddent commands.
```

これでルートマスターサーバーが構成され、NIS+ の標準テーブルを生成する用意が整いました。続けてテーブルを生成する場合は、103 ページの「NIS+ テーブルの生成 (populate)」に進みます。

▼ 誤った情報を変更する方法

上記の手順 4 で返された情報の一部または全部が誤っていたために `n` を入力した場合、次のように表示されます。

```
Is this information correct? (type 'y' to accept, 'n' to change)
 n
Domain name: [doc.com.]
```

1. ドメイン名が正しい場合は **Return** キーを押します。誤っている場合は、正しいドメイン名を入力して **Return** キーを押します。

   この例では、Return キーを押して、目的のドメイン名が `doc.com.` であることを確認しました。次に、このスクリプトは NIS+ グループ名の確認を要求します。

```
Is this information correct? (type 'y' to accept, 'n' to change)
 n
Domain name: [doc.com.]
NIS+ group: [admin.doc.com.]
```

2. **NIS+** グループが正しければ **Return** キーを押します。間違っている場合、正しい **NIS +** グループ名を入力して **Return** キーを押します。

   この例では、名前を変更しました。次に、スクリプトは NIS 互換性の入力を要求します。

```
NIS+ group: [admin.doc.com.] netadmin.doc.com.
NIS (YP) compatibility (0=off, 1=on): [0]
```

3. **NIS** 互換性を必要としない場合は **Return** キーを押します。互換性が必要な場合は `1` を入力して **Return** キーを押します。

   この例では、Return キーを押して、NIS 互換の状態が正しいことを確認しました。スクリプトは、情報が正しいかどうかを再度尋ねてきます。

注 – このサーバーを NIS 互換に選択した場合、この選択を有効にするには、ファイルを編集して、`rpc.nisd` デーモンを再起動する必要があります。詳細は 115 ページの「クライアントを NIS+ サーバーとして構成する方法」を参照してください。

```
NIS (YP) compatibility (0=off, 1=on): [0]
Domain name : doc.com.
NIS+ group : netadmin.doc.com.
NIS (YP) compatibility : OFF
Security level : 2=DES
Is this information correct? (type 'y' to accept, 'n' to change)
```

情報が正しい場合、99 ページの「ルートマスターサーバーを作成する方法」の手順 3 に移ります。正しい情報となるまで、`n` を繰り返し選択できます。

▼ Multihomed NIS+ ルートマスターサーバーの設定方法

Multihomed NIS+ サーバーを設定する手順は、単一インタフェースサーバーの設定手順と同じです。唯一の相違点は、ローカルの `/etc/hosts` ファイル、`/etc/inet/ipnodes` ファイルと NIS+ の `hosts` テーブルおよび `ipnodes` テーブル内に定義する必要があるインタフェースが、単一インタフェースサーバーよりも多いという点です。ホスト情報を定義したら、`nisclient` と `nisserver` スクリプトを使用して Multihomed NIS+ サーバーを設定します。Multihomed 複製サーバーの設定の詳細は、120 ページの「Multihomed NIS+ 複製サーバーの設定方法」を参照してください。

注意 – Multihomed NIS+ サーバーを設定する場合は、サーバーの主体名はシステムのノード名と同じにする必要があります。これは、Secured RPC と `nisclient` を同時に使用する際の必要条件です。

- Secured RPC は、ノード名を使用して認証に必要なネット名を作成する
- `nisclient` は、主体名を使用してクライアントの資格を作成する

ノード名と主体名が異なる場合、Secured RPC 認証は適正に動作できず、その結果 NIS+ で問題が発生します。

次に、NIS+ ルートマスターサーバーの設定手順を示します。

1. ルートマスター上で、サーバーのホスト情報を `/etc/hosts` ファイルまたは `/etc/inet/ipnodes` ファイル内に追加します。

   たとえば、3 つのイーサネットインタフェースを装備した *hostA* システムの場合、`/etc/hosts` ファイルには次のように入力します。

   ```
   127.0.0.1 localhost loghost
   192.168.10.x hostA hostA-10 hostA-le0
   192.168.11.y hostA hostA-11 hostA-le1
   192.168.12.z hostA hostA-12
   ```

2. `nisserver` を使用して、サーバーを **Multihomed NIS+** ルートサーバーとして設定します。

   ```
   hostA# nisserver -r -d sun.com
   ```

   上記の例では、`sun.com` はルートドメイン名を表しています。ルートドメイン名を指定して `nisserver` コマンドを実行してください。

   Multihomed NIS+ ルートサーバーの設定が完了したら、このあとの設定手順は、単一インタフェースサーバーの設定手順とまったく同じです。

NIS+ テーブルの生成 (populate)

ルートマスターサーバーの構成が終了すると、他のネットワークサービスの情報から標準の NIS+ テーブルを生成できます。この節では、`nispopulate` スクリプトをデフォルトの設定で使用して、ファイルまたは NIS マップのデータからルートマスターサーバーのテーブルを生成する方法を説明します。このスクリプトは次のものを使用します。

- 前の例で作成されたドメイン (`doc.com.`)
- ネットワークサービスのソースとしてシステム情報ファイルまたは NIS マップ

- 標準の NIS+ テーブル (auto_master、auto_home、ethers、group、hosts、 networks、passwd、protocols、services、rpc、netmasks、 bootparams、netgroup および aliases)

注 – テーブルの情報源がファイルである場合、shadow ファイルの内容が passwd ファイルの内容とマージされて passwd テーブルが作成されます。shadow テーブルは作成されません。

---

nispopulate を実行するための前提条件

スクリプト nispopulate を実行するには、次の条件が必要です。

- データを読み込むローカルの /etc 内の各ファイルまたは NIS マップを表示します。不正なエントリがないことを確認します。正しいデータが、所定の場所に正しい書式で記録されていることを確認します。エントリのうち、古いもの、無効なもの、破損しているものは削除します。また、不完全なエントリや一部のみのエントリも削除します。構成完了後は、いつでもエントリを追加できます。必要なエントリをあとから追加する方が、不完全なエントリや破損しているエントリを読み込もうとするよりも簡単です。
- ファイル内の情報は、それがロードされるテーブルに適合する書式でなければなりません。対応する NIS+ テーブルに変換するときに使用するテキストファイルの書式については、第 9 章を参照してください。
- ドメイン名とホスト名が同一でないことを確認します。ドメインとホストで同じ名前は使用しないでください。たとえば、sales というドメインを使用している場合、sales という名前の付いたマシンを使用しないでください。同様に、home という名前のマシンを使用している場合は、home という名前のドメインは作成しないでください。これはサブドメインの場合も同様です。たとえば、マシンに west という名前を付けている場合、sales.west.myco.com というサブドメインを作成しないでください。
- ホスト名のドットと下線はすべて削除します。NIS+ では、ドット (ピリオド) をマシン名とドメインの間、および親ドメインとサブドメインの間の区切りに使用するため、ドットをマシン名に使うことはできません。DNS ではホスト名に下線を使うことを認めていませんので、ホスト名に下線を使うことはできません。nispopulate スクリプトを実行する前に、ホスト名で使用されているドットを必ず削除してください。ドットをハイフンに置き換えるのも 1 つの手です。たとえば、sales.alpha というマシン名を付けることはできません。ただし、sales-alpha というマシン名は使用できます。
- 初めてネットワークを設定する場合、十分なネットワーク情報がどこにも格納されていないことがあります。このような場合、まずネットワーク情報を集め、「入力ファイル」に手入力する必要があります。このファイルは、基本的に /etc 内のファイルに対応するものです。
- 安全のため、/etc 内のファイルのコピーを作成します。実際のファイルは使用せずに、作成したコピーを使用してテーブルを生成してください。たとえば、この例では /nisplusfiles というディレクトリ内のファイルを使用します。

- コピーした NIS テーブルファイルのうち、passwd、shadow、aliases、hosts の各ファイルには、名前空間全体に分散させるとセキュリティ上問題がある項目があるので、それらを編集します。たとえば、次に示す行をローカル側の passwd ファイルのコピーから削除して、名前空間からはそれらの情報にアクセスできないようにします。

```
root:x:0:1:0000-Admin(0000):/:/sbin/sh
daemon:x:1:3:0000-Admin(0000):/:
bin:x:3:5:0000-Admin(0000):/usr/bin:
sys:x:3:3:0000-Admin(0000):/:
adm:x:4:4:0000-Admin(0000):/var/adm:
lp:x:78:9:0000-lp(0000):/usr/spool/lp:
smtp:x:0:0:mail daemon user:/:
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp:
nuucp:x:7:8:0000-uucp (0000):/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:22:6:Network Admin:/usr/net/nls
nobody:x:60000:60000:uid no body:/:
noaccess:x:60002:60002:uid no access:/:
```

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのサーバーには、新しいテーブル情報を収容できるだけの十分なディスク空き領域が必要です。
- NIS+ 主体 (適切な資格を持つクライアント) としてログインし、指定されたドメイン内の NIS+ テーブルに対する書き込み権が必要です (この例では、マシン master1 上のユーザーは root) 。

必要な情報

ファイルから生成する場合、次の情報が必要です。

- 新しい NIS+ ドメイン名
- データが転送される、適切に編集されたテキストファイルのパス
- root パスワード

NIS マップから生成する場合、次の情報が必要です。

- 新しい NIS+ ドメイン名
- NIS ドメイン名
- NIS サーバー名
- NIS サーバーの IP アドレス
- root パスワード

注 – NIS ドメインの名前は大文字と小文字を区別しますが、NIS+ ドメインの名前は区別しません。

▼ ルートマスターサーバーのテーブルを生成する方法

1. 「**a**」または「**b**」を実行してルートマスターサーバーテーブルを生成し、手順 **2** に進みます。

   「a」では、ファイルからテーブルを生成する方法を示します。「b」では、NIS マップからテーブルを生成する方法を示します。これらのコマンドはスクロールできるウィンドウ内で実行します。そうしないと、スクリプトの出力がスクロールされて読めないことがあります。

   注 – システム上の `/tmp` 領域が不足する場合、`nispopulate` スクリプトは異常終了することがあります。これを防止するには、環境変数 `TMPDIR` に別のディレクトリを設定します。`TMPDIR` に有効なディレクトリが設定されていない場合、スクリプトは `/tmp` ディレクトリを使用します。

   ---

   a. 次のように入力して、ファイルからテーブルを生成します。

   ```
   master1# nispopulate -F -p /nis+files -d doc.com.
   NIS+ domain name : doc.com.
   Directory Path : /nis+files
   Is this information correct? (type 'y' to accept, 'n' to change)
   ```

   `-F` オプションは、テーブルがデータをファイルから取り出すことを示します。`-p` オプションには、ソースファイルのディレクトリ検索パスを指定します (この例では、`/nis+files` を指定します)。`-d` オプションには、NIS+ ドメインを指定します (この例では、`doc.com.` を指定します)。

   NIS+ 主体のユーザーは root です。この例では、初めてルートマスターサーバーのテーブルを生成することになるため、この作業はスーパーユーザーとして実行しなければなりません。`nispopulate` スクリプトは、NIS+ 管理グループのすべてのメンバーの資格を追加します。

   b. 次のように入力して、**NIS** マップからテーブルを生成します。

   ```
   master1# nispopulate -Y -d doc.com. -h salesmaster -a 130.48.58.111
   -y sales.doc.com.
   NIS+ domain name : doc.com.
   NIS (YP) domain : sales.doc.com.
   NIS (YP) server hostname : salesmaster
   Is this information correct? (type 'y' to accept, 'n' to change)
   ```

   `-Y` オプションは、テーブルがデータを NIS マップから取り出すことを示します。`-d` オプションは、NIS+ ドメイン名を指定します。`-h` オプションは、NIS サーバーのマシン名を指定します。`salesmaster` は NIS サーバーの名前の例です。サンプルドメインを作成する際は、`salesmaster` の代わりに、サイトでの実際の NIS サーバー名を指定してください。`-a` オプションは、NIS サーバーの IP アドレスを指定します。`130.48.58.111` はアドレスの例です。サンプルドメインを作成する際は、サイトでの実際の NIS サーバーの IP アドレスを指定してください。`-y` オプションは、NIS ドメイン名を指定します。`sales.doc.com.` はド

メイン名の例です。サンプルドメインを作成する際は、sales.doc.com. の代わりに、サイトでの実際の NIS ドメインの NIS ドメイン名を指定してください。

NIS+ 主体のユーザーは root です。この例では、初めてルートマスターサーバーのテーブルを生成することになるため、この作業はスーパーユーザーとして実行しなければなりません。nispopulate(1M) スクリプトは、NIS+ 管理グループのすべてのメンバーの資格を追加します。

2. y を入力します **(**画面に表示された情報が正しい場合**)**。

n を入力すると、スクリプトは正しい情報の入力を要求します。情報が誤っている場合の操作については、101 ページの「誤った情報を変更する方法」 を参照してください。

- 手順 1 の「a」を実行すると、次のように表示されます。

```
Is this information correct?
(type 'y' to accept, 'n' to change)
y

This script will populate the following NIS+ tables for domain doc.com. from
the files in /nis+files: auto_master auto_home ethers group hosts networks
passwd protocols services rpc netmasks bootparams netgroup aliases shadow
**WARNING: Interrupting this script after choosing to continue may leave
the tables only partially populated. This script does not do any automatic
recovery or cleanup.
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
```

- 手順 1 の「b」を実行すると、次のように表示されます。

```
Is this information correct? (type 'y' to accept, 'n' to change)
y
This script will populate the following NIS+ tables for domain doc.com. from the
NIS (YP) maps in domain sales: auto_master auto_home ethers group hosts networks
passwd protocols services rpc netmasks bootparams netgroup aliases
**WARNING: Interrupting this script after choosing to continue may leave the
 tables only partially populated. This script does not do any automatic recovery
or cleanup.
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
```

3. y を入力して、テーブルの生成を続けます。

n を入力すると、スクリプトは安全に終了します。y を選択した後で、スクリプトの実行中にスクリプトを中断した場合、スクリプトは動作を停止し、テーブルは一部だけが生成されたままで残されることがあります。スクリプトは自動回復や後始末は行いません。このスクリプトは安全に再実行できますが、テーブルは最新の情報で上書きされます。

- ファイルからテーブルを生成する場合、スクリプトが hosts 情報と passwd 情報に基づいてホストとユーザーの資格を作成することを示す、次のようなメッセージが表示されます。

```
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
y
populating auto_master table from file /nis+files/auto_master
... auto_master table done.
```

```
populating auto_home table from file /nis+files/auto_home
... auto_home table done.
Credentials have been added for the entries in the hosts and passwd table(s).
Each entry was given a default network password (also known as a Secure-
RPC password). This password is: nisplus
Use this password when the nisclient script requests the network password.
Done!
```

Secure RPC パスワード (上の例では nisplus) は必ず控えておき、忘れないようにしてください。ネットワークパスワードまたは Secure RPC パスワードを求められたら、ここで控えておいたパスワードを入力します。

スクリプトは、必要なすべてのファイルを検索し、使用できるファイルからすべてのテーブルを生成します。

- NIS マップからテーブルを生成する場合、スクリプトが hosts と passwd の情報に基づいてホストとユーザーの資格を作成する際に、次の内容が表示されます。

```
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
y
populating auto_master table from sales.doc.com. NIS(YP) domain...
auto_master table done.
populating auto_home table from file sales.doc.com. NIS(YP) domain...
auto_home table done.
....
Credentials have been added for the entries in the hosts and passwd table(s).
Each entry was given a default network password (also known as a Secure-RPC password).
This password is: nisplus
Use this password when the nisclient script requests the network password.
Done!
```

Secure RPC パスワード (上の例では nisplus) は必ず控えておき、忘れないようにしてください。ネットワークパスワードまたは Secure RPC パスワードを求められたら、ここで控えておいたパスワードを入力します。

これですべてのテーブルが生成されました。parse error という警告が表示されるかもしれませんが無視してかまいません。これらのエラーは、NIS+ が特定の NIS マップのフィールドで空の値または予期せぬ値を見つけたことを示します。必要に応じて、スクリプト終了後、データを検証してください。

4. 必要に応じて、ルートドメインの管理グループに自分自身と他の管理者を追加します。

   たとえば、自分自身のログイン ID が topadm で、他の管理者の ID が secondadmin の場合、次のように入力します。

```
master1# nisgrpadm -a admin.doc.com. topadm.doc.com. secondadm.doc.com.
Added "topadm.doc.com." to group "admin.doc.com.".
Added "secondadm.doc.com." to group "admin.doc.com.".
```

上記の nisgrpadm -a コマンドの引数 admin.doc.com. は、グループ名を表し、最初に記述しなければなりません。残りの 2 つの引数は管理者名です。

注 – この時点で管理グループにユーザーを追加したい場合にだけ、この手順が必要です。ルートサーバーに管理者を追加するには、この時点で行います。NIS+ を構成した後でも、管理グループにユーザーを追加できます。

この手順を実行するには、ほかの管理者がデフォルトパスワードを変更するまで待つ必要はありません。しかし、管理グループにほかの管理者を追加するには、その前に `passwd` テーブルに管理者が登録されていることが必要です。管理グループのメンバーは、ドメインに自分自身を追加しなければ、NIS+ 主体として行動できません。ユーザーの初期化については、113 ページの「NIS+ ユーザーを初期設定する方法」を参照してください。また、新規にメンバーが登録されても、管理グループに対する古いキャッシュが破棄された後でなければ有効になりません。

5. 次のコマンドを入力して、ドメインのチェックポイントを実行します。

```
master1# nisping -C doc.com.
Checkpointing replicas serving directory doc.com.
Master server is master1.doc.com.
 Last update occurred at date
Master server is master1.doc.com.
checkpoint scheduled on master1.doc.com.
```

この手順によって、そのドメインをサポートする全サーバーは、初期設定 (`.log`) ファイルから新しい情報をテーブルのディスク上のコピーに転送します。ルートドメインの設定が終了したばかりであるため、このルートドメインにはまだ複製が存在せず、この手順はルートマスターサーバーにだけ影響を与えます。

注意 – スワップ領域またはディスク領域が不足している場合、サーバーは適切にチェックポイントを実行できませんが、その旨の通知は行われません。スワップ領域またはディスク領域が不足していないか確認するには、`niscat` コマンドでテーブルの内容を表示します。たとえば、`rpc` テーブルの内容をチェックするには、次のように入力します。

```
master1# niscat rpc.org_dir
rpcbind rpcbind 100000
rpcbind portmap 100000
rpcbind sunrpc 100000
```

スワップ領域が不足していると「チェックポイントを実行できない」という旨のメッセージではなく、次のエラーメッセージが表示されます。

```
can’t list table: Server busy, Try Again.
```

このメッセージには明示されていませんが、これはスワップ領域の不足を示します。スワップ空間を増やし、このドメインに再びチェックポイントを実行します。

NIS+ クライアントマシンの設定

ルートマスターサーバーのテーブルがファイルまたは NIS マップから生成されると、NIS+ クライアントマシンを初期設定できます。ルートマスターサーバーは自分のドメインの NIS+ クライアントであるため、ルートマスターサーバーのこれ以上の初期化は必要ありません。この節では、`nisclient` スクリプトを使用して、デフォルトの設定で NIS+ クライアントを初期設定する方法を説明します。このスクリプトは以下のものを使用します。

- 前の例で使用したドメイン `doc.com.`
- 前の例で、`nispopulate` スクリプトによって作成された Secure RPC パスワード (ネットワークパスワードとも呼ばれる。デフォルトは `nisplus`)

注 – 111 ページの「新しいクライアントマシンを初期設定する方法」 で使用する `-i` オプションでは、DNS を使用してホスト名を解決するような NIS+ クライアントを構成しません。DNS を使用する場合には、ネームサービススイッチファイルの中でクライアント用に、DNS を明確に指定しなければなりません。DNS によるホスト名の解決については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)* 』の「ドメインネームシステム (概要)」を参照してください。

`nisclient` を実行するための前提条件

`nisclient` スクリプトを使用するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのマスターサーバーのテーブルが生成されている。(少なくとも、`hosts` テーブルまたは `ipnodes` テーブルには新しいクライアントマシンのエントリが必要)
- NIS+ クライアントとなるマシンにスーパーユーザーとしてログインしている。この例では、新しいクライアントマシン名は `client1`

必要な情報

次の情報が必要です。

- ドメイン名
- デフォルトの Secure RPC パスワード (`nisplus`)
- クライアントとなるマシンの root パスワード
- クライアントのホームドメインにある NIS+ サーバーの IP アドレス

- DES 認証を使用する場合には、マスターサーバー上で、Diffie-Hellman キー長が使用される。nisauthconf を使用して、マスターサーバー上で Diffie-Hellman キー長を確認する

▼ 新しいクライアントマシンを初期設定する方法

1. **DES** 認証を使用する場合には、オプションで、**Diffie-Hellman** キー長を指定します。

   マスターサーバー上で以下を入力します。

   ```
   nisauthconf
   ```

   クライアントマシン上で nisauthconf コマンドを実行するときは、上記コマンドの出力を引数として使用します。たとえば、マスターサーバー上で nisauthconf を実行すると

   ```
   dh640dh-0 des
   ```

   と出力される場合には、クライアントマシン上で次のコマンドを入力します。

   ```
   nisauthconf dh640dh-0 des
   ```

2. 次のコマンドを入力して、新しいクライアントマシン上で新しいクライアントを初期設定します。

   -i オプションはクライアントを初期設定します。-d オプションは新しい NIS+ ドメイン名を指定します。ドメイン名が指定されない場合、そのデフォルトは現在のドメイン名となります。-h オプションは NIS+ サーバーのホスト名を指定します。

```
client1# nisclient -i -d doc.com. -h master1
Initializing client client1 for domain "doc.com.".
Once initialization is done, you will need to reboot your machine.
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
```

3. y を入力します。

   n を入力するとスクリプトが終了します。クライアントの /etc/hosts ファイルまたは /etc/inet/ipnodes ファイルにルートサーバーのエントリがない場合、スクリプトはルートサーバーの IP アドレスの入力を指示します。

```
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
y
Type server master1's IP address:
```

4. 正しい **IP** アドレスを入力して、**Return** キーを押します。

   この例では、アドレス 123.123.123.123 を使用します。

```
Type server master1's IP address: 123.123.123.123
setting up the domain information...
setting up the name service switch information...
At the prompt below, type the network password (also known as the
Secure-RPC password) that you obtained either from your administrator or
from running the nispopulate script.
```

```
Please enter the Secure-RPC password for root:
```

5. **Secure RPC** パスワード **(**ネットワークパスワード**)** が **root** のログインパスワードと異なる場合のみ、**Secure RPC** パスワードを入力します。

   この例では、デフォルトの `nisplus` を使用します。

   パスワードは画面に表示されません。入力を誤った場合、正しいパスワードを入力するよう要求されます。2 回目も入力を誤った場合、スクリプトが終了し、前回のネットワークサービスが復元されます。この場合、スクリプトを再度実行してください。

   ```
   Please enter the login password for root:
   ```

6. このクライアントマシンの **root** パスワードを入力します。

   パスワードは画面に表示されません。ネットワークパスワードと root のログインパスワードが同じ場合、root のログインパスワードの入力要求はありません。

   root パスワードを入力すると、このマシンの資格が変更されます。ここで、このマシンでは RPC パスワードと root パスワードが同じになります。

   ```
   Please enter the login password for root:
   Wrote secret key into /etc/.rootkey
   Your network password has been changed to your login one.
   Your network and login passwords are now the same.
   Client initialization completed!!
   Please reboot your machine for changes to take effect.
   ```

7. クライアントマシンを再起動します。

   ここで行なった変更は、マシンを再起動すると有効になります。

   以上の手順により、この NIS+ クライアントマシンのユーザーを NIS+ ドメインに追加できるようになりました。

クライアントマシンの追加作成

これまでに説明したクライアントの初期設定手順を、必要な各マシンで繰り返します。ほかのドメインのクライアントを初期設定するには、ドメイン名とマスターサーバー名を適宜変更して、上記の手順を繰り返します。

この章で説明する NIS+ ドメインのサンプルでは、`doc.com.` ドメイン内の 4 台のクライアントを初期設定すると想定しています。そして、このうち 2 台のクライアントを非ルート (ルート以外の) NIS+ サーバーとして構成し、3 番目のクライアントを `doc.com.` ドメインのルートマスターサーバーのルート複製サーバーとして構成します。

注 **–** システムをサーバーにするには、システムをどの種類のサーバーにするにしても、事前にシステムを親ドメインのクライアントにしておく必要があります。

NIS+ クライアントユーザーの初期設定

マシンが NIS+ クライアントになったら、そのマシンのユーザーは自分自身を NIS+ ドメインに追加しなければなりません。ユーザーをドメインに追加するということは、Secure RPC パスワードをそのユーザーのログインパスワードに変更することを意味します。実際には、ユーザーのパスワードと Secure RPC パスワードが同じになります。この手順では `nisclient` スクリプトを使用します。

`nisclient` を実行するための前提条件

`nisclient` スクリプトを使用してユーザーを初期設定するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのマスターサーバーのテーブルが生成されている。少なくとも、`hosts` テーブルには新しいクライアントマシンのエントリが必要
- ドメイン内のクライアントマシンが初期設定されている
- クライアントマシンに「ユーザー」としてログインしている (この例では、ユーザー名は `user1`)
- DES 認証を使用する場合には、クライアントマシンはマスターサーバーと同じ Diffie-Hellman キー設定を使用する

必要な情報

次の情報が必要です。

- ユーザーのログイン名 (この例では `user1`)
- デフォルトの Secure RPC パスワード (この例では `nisplus`)
- NIS+ クライアントになるユーザーのログインパスワード

▼ NIS+ ユーザーを初期設定する方法

1. **NIS+** クライアントになるユーザーとしてログインして、次のように `nisclient` コマンドを入力します。

```
user1prompt% nisclient -u
At the prompt below, type the network password (also known as the
Secure-RPC password) that you obtained either from your administrator
or from running the nispopulate script.
Please enter the Secure-RPC password for user1:
```

2. **Secure RPC** パスワード **(**この例では `nisplus`**)** を入力します。

パスワードは画面に表示されません。

```
Please enter the login password for user1:
```

3. ユーザーのログインパスワードを入力し、**Return** キーを押します。
   パスワードは画面に表示されません。

```
Your network password has been changed to your login one.
Your network and login passwords are now the same
```

   これで、このユーザーは NIS+ クライアントになりました。全ユーザーを NIS+ クライアントにする必要があります。

NIS+ サーバーの設定

初期設定が終了したら、どのクライアントマシンも次の NIS+ サーバーに変更できます。

- ルート複製サーバー (ルートマスターサーバー上にある NIS+ テーブルのコピーを格納する)
- ルートドメインのサブドメインのマスターサーバー
- ルートドメインのサブドメインのマスターサーバーの複製サーバー

注 – NIS+ マスタールートサーバーは 1 つしか持てません。ルート NIS+ サーバーは特殊な NIS+ サーバーです。この節ではルートマスターサーバーの構成方法は説明しません。詳細は、98 ページの「NIS+ ルートサーバーの設定」を参照してください。

サーバーを構成する場合、次の 3 つの設定があります。

- NIS と互換性なし
- NIS と互換性あり
- NIS との互換性と DNS 転送あり - NIS+ 名前空間に Sun OS 4.x クライアントを配置する場合は、DNS 転送を設定する必要があります。

サーバーとその複製は、NIS 互換についての設定が同じでなければなりません。設定が異なる場合、ネットワーク情報を受信するために NIS 互換の設定を必要とするクライアントは、必要なサーバーまたは複製サーバーを使用できなければ、この情報を受信できません。

この例では、マシン `client1` がサーバーに変更されます。この手順では、NIS+ スクリプトの代わりに NIS+ コマンド `rpc.nisd` を使用します。

rpc.nisd を実行するための前提条件

rpc.nisd を実行するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのマスターサーバーのテーブルが生成されている。少なくとも、hosts テーブルには新しいクライアントマシンのエントリが必要
- ドメイン内のクライアントマシンが初期設定されている
- クライアントマシンに root としてログインしている。(この例では、クライアントマシン名は client1)
- DES 認証を使用する場合には、クライアントマシンはマスターサーバーと同じ Diffie-Hellman キー設定を使用する必要があります。

必要な情報

サーバーに変換するクライアントのスーパーユーザーパスワードが必要です。

クライアントを NIS+ サーバーとして構成する方法

クライアントをサーバーとして構成するには、次の手順のどれかを実行します。この手順では、サーバーと同じ名前のディレクトリを作成し、サーバーの初期設定ファイルを作成します。これらは /var/nis に置かれます。

注 – 同じドメイン内のすべてのサーバーは、NIS 互換の設定が同じでなければなりません。たとえば、マスターサーバーが NIS 互換である場合、その複製も NIS 互換でなければなりません。

▼ NIS と互換性のない NIS+ サーバーを構成する方法

- **NIS** と互換性のないサーバーを構成するには、次のコマンドを入力します。

```
client1# rpc.nisd
```

▼ NIS と互換性のある NIS+ サーバーを構成する方法

1. サーバーの /etc/init.d/rpc ファイルを編集して、文字列 -EMULYP="-Y" を含む行のコメントアウトを解除して有効にします。
   コメントアウトを解除するには、行の先頭から # を削除します。
2. スーパーユーザーとして、次のように入力します。

```
client1# rpc.nisd -Y
```

▼ NIS との互換性と DNS 転送機能を備えた NIS+ サーバーを構成する方法

次の手順を実行して、DNS 転送機能と NIS+ との互換性の両方の機能を備えた NIS+ サーバーを構成します。SunOS 4.x クライアントをサポートするには、両方の機能が必要です。

1. サーバーの `/etc/init.d/rpc` ファイルを編集します。文字列 `-EMULYP="-Y"` を含む行のコメントアウトを解除 **(**行の先頭から `#` を削除**)** して有効にします。

   コメントアウトを解除するには、行の先頭から `#` を削除します。

2. 上の行の引用符で囲まれた中に次のように `-B` を追加します。

   次のようになります。

   ```
   -EMULYP="-Y -B"
   ```

3. 次のコマンドをスーパーユーザーとして入力します。

   ```
   client1# rpc.nisd -Y -B
   ```

   これで、このサーバーはドメインのマスターまたは複製として指定できます。

サーバーの追加作成

これまでのクライアントをサーバーに変更する手順を、必要なクライアントマシンごとに繰り返します。

この章で説明する NIS+ サンプルドメインは、3 台のクライアントをサーバーに変更すると想定しています。そして、サーバーの 1 台をルート複製サーバーに、もう 1 台を新しいサブドメインのマスターに、そして 3 台目を新しいサブドメインのマスターの複製に構成します。

ルート複製サーバーの作成

NIS+ サービスを常に利用可能な状態にしておくため、ルート複製サーバーを少なくとも 1 つ作成しておくことをお勧めします。複製サーバーを作成すると複数のサーバーが存在することになり、要求の処理を分散させることができるため、ネットワーク要求の処理も高速化されます。

パフォーマンス上の理由から、1 つのドメインに多くの複製サーバーを置くことはお勧めできません。ただし、ネットワークが複数のサブネットで構成されている場合、あるいは広域ネットワーク (WAN) でリモートサイトに接続されている場合には、複製サーバーを追加した方がいい場合があります。

- 「サブネット」。複数のサブネットで構成されているドメインの場合、各サブネットに複製サーバーを少なくとも 1 つは作成することをお勧めします。そうしておけば、ネットワーク間の通信が一時的に途絶しても、接続が回復するまでの間、サブネットレベルの機能は維持されます。
- 「リモートサイト」。WAN によりリモートサイトに接続されているドメインの場合、WAN 接続の両側に複製サーバーを少なくとも 1 つは作成することをお勧めします。組織論的な見地からしても、同一の NIS+ ドメインに物理的に離れた 2 つのサイトがあるのは意味のあることです。たとえば、ドメイン内のマスターサーバーとその複製サーバーがすべて一方のサイトに置かれている場合、そのサイトともう一方のサイトとの間の NIS+ ネットワークトラフィックが増大するのは目に見えています。もう一方のサイトにも複製サーバーを置いておけば、ネットワークトラフィックが減るはずです。

最適な複製サーバー数の決定方法については、116 ページの「ルート複製サーバーの作成」 を参照してください。

118 ページの「ルート複製サーバーを作成する方法」では、マシン `client1` を `doc.com.` ドメインのルート複製サーバーとして構成します。この手順では、NIS+ スクリプトの `nisserver` を使用します。172 ページの「NIS+ コマンドを使って複製サーバーを構成する」 で説明しているように NIS+ コマンドを使用して複製サーバーを構成することもできます。

`nisserver` を実行するための前提条件

`nisserver` を実行して複製サーバーを作成するには、次の条件が必要です。

- ドメインがすでに構成され、マスターサーバーが稼動している
- マスターサーバーのテーブルが生成されている。少なくとも、`hosts` テーブルには新しいクライアントマシンのエントリが必要
- 新たに作成した複製サーバーを `doc.com.` ドメインのクライアントマシンとして初期設定している
- 新たに作成した複製サーバー上で `rpc.nisd` を起動している (具体的な手順については、114 ページの「NIS+ サーバーの設定」を参照)
- ルートマスターサーバーに root としてログインしている (この例では、ルートマスターマシン名は `master1`)

必要な情報

次の情報が必要です。

- ドメイン名

- クライアントマシン名 (この例では、client1)
- ルートマスターサーバーのスーパーユーザーのパスワード

▼ ルート複製サーバーを作成する方法

1. **NIS+** ドメインのルートマスターサーバー上でスーパーユーザー **(root)** として次のように入力して、ルート複製サーバーを作成します。

```
master1# nisserver -R -d doc.com. -h client1
This script sets up a NIS+ replica server for domain doc.com.
Domain name: :doc.com.
NIS+ server    : :client1
Is this information correct? (type 'y' to accept, 'n' to change)
```

-R オプションは、複製サーバーを構成することを示します。-d オプションは NIS+ ドメイン名 (このサンプルでは doc.com.) を指定します。-h オプションはルート複製サーバーとなるクライアントマシン (この例では client1) を指定します。

2. y を入力して操作を続けます。

n を入力すると、スクリプトは正しい情報の入力を要求します。n を入力した場合の操作については、101 ページの「誤った情報を変更する方法」を参照してください。

```
Is this information correct? (type 'y' to accept, 'n' to change)
y
This script will set up machine "client1" as an NIS+ replica server for domain
doc.com. without NIS compatibility. The NIS+ server daemon, rpc.nisd, must
be running on client1 with the proper options to serve this domain.
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
```

3. y を入力して操作を続けます。

n を入力するとスクリプトは問題なく終了します。クライアントマシン上で rpc.nisd が動作していない場合、スクリプトは自動的に終了します。

```
Is this information correct? (type 'y' to continue, 'n' to exit this script)
y
The system client1 is now configured as a replica server for domain doc.com..
The NIS+ server daemon, rpc.nisd, must be running on client1 with the proper
options to serve this domain. If you want to run this replica in NIS (YP)
compatibility mode, edit the /etc/init.d/rpc file on the replica server '
to uncomment the line which sets EMULYP to "-Y". This will ensure that
rpc.nisd will boot in NIS-compatibility mode. Then, restart rpc.nisd with
the "-Y" option. These actions should be taken after this script completes.
```

注 – 上記のメッセージはオプションの手順に関するものです。NIS 互換ではないルート複製サーバーを NIS 互換にする場合にだけ、/etc/init.d/rpc ファイルを変更します。つまり、NIS 互換のサーバーとして構成されていないルート複製サーバーで NIS クライアント要求を実行する場合にだけ、このファイルの変更が必要になります。NIS 互換サーバーの作成については、115 ページの「クライアントを NIS+ サーバーとして構成する方法」を参照してください。

---

4. 必要に応じて、複製サーバーを **NIS (YP)** 互換モードで稼動させることができるように構成します。

   複製サーバーを NIS 互換モードで稼動させるには、次の手順に従います。

   a. rpc.nisd を終了します。

   b. サーバーの /etc/init.d/rpc ファイルをオープンして、EMULYP が -Y に設定されている行のコメントアウトを解除します。

      具体的には、EMULYP 行の先頭の # を削除します。

   c. rpc.nisd を再起動します。

5. 新たに作成した複製サーバーに名前空間データをロードします。

   NIS+ データセットのロードには、2 通りの方法があります。

   - NIS+ の保存と復元の機能を使用して、マスターサーバーのデータを保存し、作成した複製サーバーに復元します (こちらの方法をお勧めします)。具体的な手順については、175 ページの「名前空間データをロードする方法—nisrestore による方法」 を参照してください。
   - nisping を実行します。nisping を実行すると、マスターサーバーのすべての NIS+ データを複製サーバーに完全に再同期させることができます。しかし、大きな名前空間の場合、処理に長時間かかることがあります。その間、マスターサーバーはビジーになって応答が遅く、また作成した複製サーバーはまだ NIS+ 要求に応答できない状態になることがあります。具体的な手順については、177 ページの「名前空間データをロードする方法—nisping による方法」を参照してください。

   名前空間データのロードが終了すれば、マシン client1 は NIS+ ルート複製サーバーとなります。この新しいルート複製サーバーは、ルートドメインのクライアントからの要求を処理できます。これでドメインで使用可能なサーバーが 2 台となるため、要求を速く処理できます。

   これらの手順に従って、複製サーバーを必要なだけ作ることができます。同じ手順を使ってサブドメインに複製サーバーを作ることもできます。

▼ Multihomed NIS+ 複製サーバーの設定方法

Multihomed NIS+ サーバーを設定する手順は、単一インタフェースサーバーの設定手順と同じです。唯一の相違点は、ローカルの `/etc/hosts` ファイル、`/etc/inet/ipnodes`ファイルと NIS+ の `hosts` テーブルおよび `ipnodes` テーブル内に定義する必要があるインタフェースが、単一インタフェースサーバーよりも多いという点です。ホスト情報を定義したら、`nisclient` と `nisserver` スクリプトを使用して Multihomed NIS+ サーバーを設定します。

注意 – Multihomed NIS+ サーバーを設定する場合は、サーバーの主体名はシステムのノード名と同じにする必要があります。これは、Secured RPC と `nisclient` を同時に使用する際の必要条件です。

- Secured RPC は、ノード名を使用して認証に必要なネット名を作成する
- `nisclient` は、主体名を使用してクライアントの資格を作成する

ノード名と主体名が異なる場合、Secured RPC 認証は適正に動作できず、その結果 NIS+ で問題が発生します。

次に、NIS+ 非ルートマスターサーバーの設定手順を示します。以下の例では、ルートドメインの複製を作成しています。Muitihomed ルートサーバーの設定方法については、102 ページの「Multihomed NIS+ ルートマスターサーバーの設定方法」を参照してください。

1. サーバーのホスト情報を `hosts` ファイルまたは `ipnodes` ファイル内に追加します。

   たとえば、3 つのインタフェースを装備した `hostB` システムの場合は、次のように入力します。

   ```
   192.168.11.y hostB hostB-11
   192.168.12.x hostB hostB-12
   192.168.14.z hostB hostB-14
   ```

2. ルートマスターサーバー上で、`nispopulate` または `nisaddent` のどちらかを使用して、新しいホスト情報を `hosts` テーブルまたは `ipnodes` テーブル内にロードします。

   たとえば:

   ```
   hostA# nispopulate -F -d sun.com hosts
   ```

   この例では、`sun.com` は NIS+ ルートドメイン名を表しています。自分の NIS+ ルートドメイン名を指定して `nispopulate` コマンドを実行してください。

3. ルートマスターサーバー上で、`nisclient` スクリプトを使用して新しいクライアントの資格を作成します。

   たとえば:

   ```
   hostA# nisclient -c -d sun.com hostB
   ```

この例では、sun.com は ルートドメイン名を表しています。自分のルートドメイン名を指定して nisclient コマンドを実行してください。

4. 非ルートマスターサーバー上で、新しいサーバーがまだ稼動していない場合は、nisclient を使用して新しいサーバーを起動し、このマシンを **NIS+** クライアントとして初期化します。

   たとえば:

   ```
   hostB# nisclient -i -d sun.com
   ```

   この例では、sun.com は ルートドメイン名を表しています。自分のルートドメイン名を指定して nisclient コマンドを実行してください。

5. ルートマスター上で、nisserver を使用して非ルートマスターサーバーを設定します。

   たとえば:

   ```
   hostA# nisserver -M -d eng.sun.com -h hostB.sun.com.
   ```

   この例では、eng.sun.com は NIS+ ドメイン名を表し、hostB.sun.com は NIS+ サーバーの完全指定ホスト名を表しています。nisserver コマンドは、自分の NIS+ ドメイン名と NIS+ サーバーの完全指定ホスト名を指定して実行してください。

6. ルートマスターサーバー上で、nisserver を使用して複製サーバーを設定します。

   たとえば:

   ```
   hostA# nisserver -R -d sun.com -h hostB.sun.com.
   ```

   この例では、sun.com は 複製サーバーを表し、hostB.sun.com は NIS+ サーバーの完全指定ホスト名を表しています。nisserver コマンドは、自分の複製サーバー名と NIS+ サーバーの完全指定ホスト名を指定して実行してください。

   Multihomed NIS+ 複製サーバーの設定が完了したら、このあとの設定手順は、単一インタフェースのサーバーの設定手順とまったく同じです。

サブドメインの作成

この節では、ルート以外の新しいドメイン (サブドメイン) のマスターサーバーの作成方法を説明します。この新しいドメインは doc.com. ドメインのサブドメインとなります。NIS+ の階層構造によって、組織構造に合わせたドメイン構造を作成できます。

この例では、マシン client2 が sub.doc.com. という新しいドメインのマスターサーバーに変更されます。ここでは NIS+ スクリプトの nisserver を使用します。

nisserver を実行するための前提条件

nisserver を実行して、新しいサブドメインのマスターサーバーを作成するには、次の条件が必要です。

- 親ドメインがすでに構成され、そのマスターサーバーが実行されている
- 親ドメインのテーブルが生成されている。少なくとも、hosts テーブルには新しいクライアントマシンのエントリが必要
- 親ドメイン内の新しいクライアントマシンが初期設定されている
- クライアント上で rpc.nisd が起動されている
- 新しいドメインを追加するための適切なアクセス権がある。この例では、親マスターサーバーに root としてログインする。この例では、親マスターマシン名は master1

必要な情報

次の情報が必要です。

- 新しいサブドメインの名前 – 親ドメインの名前を含んだ新しいドメイン名。*newdomain.rootdomain* という構文に従うこと
- クライアントマシンの名前 (この例では、client2)
- 親マスターサーバーのスーパーユーザーパスワード

123 ページの「新しい非ルートドメインを作成する方法」では、新しいサブドメインを sub.doc.com. とします。

注 – Solaris リリース 2.6 およびこれ以前のリリースでは、どんな NIS+ クライアントでも、そのサービスの提供先となるドメインの上位ドメインに置かれている限り、NIS+ マスターサーバーに変更できます。たとえば、ドメイン sales.doc.com. の NIS+ クライアントは、 west.sales.doc.com. や alameda.west.sales.doc.com. などの下位ドメインにサービスを提供することができます。ただし、このクライアントは、ドメイン doc.com. にサービスを提供することはできません。doc.com. は sales.doc.com. の上位ドメインであるためです。ただし、ルート複製サーバーはこのルールの唯一の例外であり、サービス提供先のドメインのクライアントです。

注 – Solaris リリース 7 では、非ルート NIS+ サーバーのドメイン名を、そのサーバーがサービスしているドメインに設定することができます。非ルートのサーバーは、設定されたドメイン内でサーバーとして動作します。つまり、非ルートのサーバー上のアプリケーションを構成して、上位ドメインから渡される情報を使用することができます。

ただし、非ルートのサーバーの資格は、上位ドメインになければなりません。非ルートのサーバーの構成方法については、123 ページの「新しい非ルートドメインを作成する方法」 を参照してください。サーバーを正しく構成したあとで、ドメイン名をサーバーとして機能するドメインの名前に変更できます。`nisinit` の `-k` オプションおよび `nisserver` の `-d` オプションを参照してください。

▼ 新しい非ルートドメインを作成する方法

1. **NIS+** ドメインのルートマスターサーバー上のスーパーユーザー **(root)** として次のように入力し、新しい非ルートドメインマスターサーバーを作成します。

   `-M` オプションは、新しい非ルートドメインのマスターサーバーを作成することを示します。`-d` オプションは、ドメイン名 (この例では、`sales.doc.com.`) を指定します。`-h` オプションは、新しいドメインのマスターサーバーになるクライアントマシン (この例では、`client2`) を指定します。

   ```
   master1# nisserver -M -d sales.doc.com. -h client2
   This script sets up a non-root NIS+ master server for domain sales.doc.com.
   Domain name : sales.doc.com.
   NIS+ server : client2
   NIS+ group : admin.sales.doc.com.
   NIS (YP) compatibility : OFF
   Security level : 2=DES
   Is this information correct? (type 'y' to accept, 'n' to change)
   ```

   新しい非ルートドメインのマスターサーバーは、ルートサーバーと同じデフォルト値によって作成されます。NIS+ グループ、NIS 互換性、およびセキュリティレベルの詳細は、99 ページの「ルートマスターサーバーを作成する方法」を参照してください。

2. `y` を入力して操作を続けます。

   `n` を入力すると、スクリプトは正しい情報の入力を要求します。`n` を入力した場合の操作については、101 ページの「誤った情報を変更する方法」を参照してください。

```
Is this information correct?
(type 'y' to accept, 'n' to change) y
This script sets up machine "client2" as an NIS+ non-root master
server for domain sales.doc.com.
Do you want to continue? (type 'y' to continue, 'n' to exit this script)
```

3. `y` を入力して操作を続けます。

   `n` を入力すると、スクリプトは安全に終了します。クライアントマシン上で `rpc.nisd` が動作していない場合、スクリプトは自動的に終了します。

```
Do you want to continue? (type 'y' to continue, 'n'
to exit this script)
y
running nissetup ...
org_dir.sales.doc.com. created
groups_dir.sales.doc.com. created
...
...
setting NIS+ group admin.sales.doc.com. ...
The system client2 is now configured as a non-root server for
domain sales.doc.com.
You can now populate the standard NIS+ tables by using the
nispopulate or /usr/lib/nis/nisaddent commands.
```

これで、マシン client2 は sales.doc.com. ドメインのマスターサーバーになりました。sales.doc.com. ドメインは、 doc.com. ドメインのサブドメインです。マシン client2 は、依然としてルートドメイン doc.com. のクライアントであり、しかも sales.doc.com. ドメインのマスターサーバーでもあります。

これで、sales.doc.com. ドメインの新しいマスターサーバーに標準の NIS+ テーブルを生成できます。

ドメインの追加作成

サーバーを新しい非ルートドメインのマスターサーバーに変更する前述の手順を、必要な各サーバーマシンに対し繰り返します。新しいマスターサーバーはすべて新しいドメインです。ドメイン構造の計画を立ててから、NIS+ 名前空間の作成を始めます。NIS+ 階層の計画については、63 ページの「NIS+ 名前空間の構造」を参照してください。

新しいサブドメインのテーブルの生成

新しいドメインを作成したら、そのマスターサーバーの標準の NIS+ テーブルを生成しなければなりません。新しいマスターサーバーのテーブルを生成するには、ルートマスターサーバーのテーブルを生成するときと同じ手順を使用します。大きな違いは、nispopulate スクリプトが、ルートマスターサーバー上ではなく、新しいマスターサーバー上で実行されることです。また、ドメイン名、およびファイルパス、または NIS サーバー名も変わることがあります。

この例では、新しいドメイン (sales.doc.com.) のテーブルを生成します。

nispopulate を実行するための前提条件

nispopulate スクリプトを実行して新しいマスターサーバーのテーブルを生成するには、次の条件が必要です。

- ファイル内の情報は、それがロードされるテーブルに合わせて書式設定されていなければなりません。
  - スクリプトを実行する前に、データを読み込むローカルの /etc 内の各ファイルまたは NIS マップを調べます。不正なエントリがないことを確認します。正しいデータが、所定の場所に正しい書式で記録されていることを確認します。エントリのうち、古いもの、無効なもの、破損しているものは削除します。また、不完全なエントリや一部のみのエントリも削除します。構成完了後は、いつでもエントリを追加できます。必要なエントリをあとから追加する方が、不完全なエントリや破損しているエントリを読み込もうとするよりも簡単です。
  - 初めてネットワークを設定する場合、十分なネットワーク情報がどこにも格納されていないことがあります。このような場合、まずネットワーク情報を集め、「入力ファイル」に手入力する必要があります。このファイルは、基本的に /etc 内のファイルに対応するものです。
- 安全のため、/etc 内のファイルのコピーを作成します。実際のファイルは使用せずに、作成したコピーを使用してテーブルを生成してください。たとえば、この例では /nis+files というディレクトリ内のファイルを使用します。
- コピーした NIS テーブルファイルのうち、passwd、shadow、aliases、hosts の各ファイルには、名前空間全体に分散させるとセキュリティ上問題がある項目があるので、それらを編集します。たとえば、次に示す行をローカル側の passwd ファイルのコピーから削除して、名前空間からはそれらの情報にアクセスできないようにします。

```
root:x:0:1:0000-Admin(0000):/:/sbin/sh
daemon:x:1:3:0000-Admin(0000):/:
bin:x:3:5:0000-Admin(0000):/usr/bin:
sys:x:3:3:0000-Admin(0000):/:
adm:x:4:4:0000-Admin(0000):/var/adm:
lp:x:78:9:0000-lp(0000):/usr/spool/lp:
smtp:x:0:0:mail daemon user:/:
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp:
nuucp:x:7:8:0000-
uucp (0000):/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:22:6:Network Admin:/usr/net/nls:
nobody:x:60000:60000:uid no body:/:
noaccess:x:60002:60002:uid no access:/:
```

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのサーバーには、新しいテーブル情報を収容できるだけの十分なディスク空き領域が必要です。
- NIS+ 主体としてログインしていて、しかも指定されたドメイン内の NIS+ テーブルに対する書き込み権が必要です。この例では、マシン client2 上の root でなければなりません。

注 – システム上の /tmp 領域が不足する場合、nispopulate スクリプトは異常終了することがあります。これを防止するには、環境変数 TMPDIR に別のディレクトリを設定します。TMPDIR に有効なディレクトリが設定されていない場合、スクリプトは代わりに /tmp ディレクトリを使用します。

必要な情報

ファイルから生成する場合、次の情報が必要です。

- 新しい NIS+ ドメイン名
- データが転送される、適切に編集されたテキストファイルのパス
- NIS+ マスターサーバーの root パスワード

NIS マップから生成する場合、次の情報が必要です。

- 新しい NIS+ ドメイン名
- NIS ドメイン名
- NIS サーバー名
- NIS サーバーの IP アドレス
- NIS+ マスターサーバーの root パスワード

注 – NIS ドメインの名前は大文字と小文字を区別しますが、NIS+ ドメインの名前は区別しません。

マスターサーバーテーブルを生成する方法

この手順は、106 ページの「ルートマスターサーバーのテーブルを生成する方法」の手順と実質的に同じなので、この例では、新しいドメイン (sales.doc.com.) のテーブルを生成するための入力についてだけ説明します。この手順の詳細は、106 ページの「ルートマスターサーバーのテーブルを生成する方法」を参照してください。

注 – このスクリプトは、ルートマスターサーバーではなく、新しいドメインのマスターサーバー上で実行しなければなりません。

次のいずれかの方法で、新しいマスターサーバー上にマスターサーバーテーブルを生成します。

- ファイルからマスターサーバーテーブルを生成する
- NIS マップからマスターサーバーテーブルを生成する

実行結果が一度に表示しきれないことがあるので、どちらの方法で行うにしてもスクロール可能なウィンドウを使用してください。

▼ テーブルをファイルから生成する方法

ファイルからマスターサーバーテーブルを生成するには、次のコマンドを入力します。

```
client2# nispopulate -F -p /nis+files -d sales.doc.com.
NIS+ domain name : sales.doc.com.
Directory Path : /nis+files
Is this information correct? (type 'y' to accept, 'n' to change
```

▼ テーブルを NIS マップから生成する方法

NIS マップからマスターサーバーテーブルを生成するには、次のコマンドを入力します。

```
client2# nispopulate -Y -d sales.doc.com. -h businessmachine -a
 IP_addr_of_NIS_server  -y business.doc.com.
NIS+ Domain name : sales.doc.com.
NIS (YP) domain : business.doc.com.
NIS (YP) server hostname : businessmachine
Is this information correct? (type 'y' to accept, 'n' to change)
```

このスクリプトの残りの出力については、106 ページの「ルートマスターサーバーのテーブルを生成する方法」を参照してください。

サブドメイン複製サーバーの作成

ルートドメイン複製サーバーに対して適用される規則は、サブドメイン複製サーバーに対しても適用されます (116 ページの「ルート複製サーバーの作成」を参照)。

サブドメインの複製サーバーの作成には、ルート複製サーバーの作成と同じ手順を使用します。このルート複製サーバーの作成とサブドメイン複製サーバーの作成との大きな違いは、サブドメイン複製サーバーに変更しようとしているマシンが、複製サーバーとしてサービスを提供することになるドメインより上位のドメインのクライアントのままであることです。この例では、新しいドメインの複製サーバーを作成するための入力だけを示します。スクリプトの残りの出力については、118 ページの「ルート複製サーバーを作成する方法」を参照してください。

nisserver を実行するための前提条件

nisserver を実行して複製サーバーを作成するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのテーブルが生成されている。少なくとも、hosts テーブルには新しいクライアントマシンのエントリが必要
- 親ドメイン内のクライアントマシンが初期設定されている
- クライアント上で rpc.nisd が起動されている
- マスターサーバーにルートとしてログインしている (この例では、マスターマシン名は client2)

必要な情報

- ドメイン名
- クライアントマシン名 (この例では、client3)
- ルートマスターサーバーのスーパーユーザーのパスワード

▼ 複製サーバーを作成する方法

- 複製サーバーを作成するには、**NIS+** ドメインのマスターサーバー上でスーパーユーザー **(root)** として、nisserver -R コマンドを実行します。

```
client2# nisserver -R -d sales.doc.com. -h client3
This script sets up a NIS+ replica server for domain sales.doc.com.
Domain name     ::sales.doc.com.
NIS+ server :client
Is this information correct? (type 'y' to accept, 'n' to change)
```

この例では、client2 がマスターサーバーです。-R オプションは、複製サーバーを構成することを示します。-d オプションは NIS+ ドメイン名 (ここでは、sales.doc.com.) を指定します。-h オプションは複製サーバーとなるクライアントマシン (ここでは client3) を指定します。なお、このマシンは依然として doc.com. ドメインのクライアントであり、sales.doc.com. ドメインのクライアントではありません。

このスクリプトの残りの出力については、118 ページの「ルート複製サーバーを作成する方法」を参照してください。

サブドメインの NIS+ クライアントマシンの初期設定

マスターサーバーのテーブルをファイルまたは NIS マップから生成すれば、NIS+ クライアントマシンを初期設定できます。この節では、デフォルト設定の `nisclient` スクリプトを使用して、新しいドメイン内の NIS+ クライアントを初期設定する方法を説明します。NIS+ クライアントマシンは、NIS+ マスターサーバーとは別のマシンです。

注 – 130 ページの「新しいサブドメインクライアントマシンを初期設定する方法」 で使用される `-i` オプションは、NIS+ クライアントを DNS を必要とするホスト名の解決ができるようには構成しません。DNS を使用する場合には、ネームサービススイッチファイルの中でクライアント用に、DNS を明確に指定しなければなりません。

新しいドメイン内のクライアントを初期設定するには、ルートドメイン内のクライアントの初期設定と同じ手順を使用します。この例では、新しいドメインのクライアントを初期設定するための入力だけを示します。スクリプトの残りの出力については、111 ページの「新しいクライアントマシンを初期設定する方法」を参照してください。

`nisclient` を実行するための前提条件

`nisclient` スクリプトを使用してユーザーを初期設定するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのマスターサーバーのテーブルが生成されている。少なくとも、`hosts` テーブルには新しいクライアントマシンのエントリが必要
- ドメイン内のクライアントマシンが初期設定されている
- クライアントマシンに「ユーザー」としてログインしている (この例では、ユーザー名は `user1`)

必要な情報

次の情報が必要です。

- ドメイン名 (この例では、`sales.doc.com.`)
- デフォルトのネットワークパスワード (この例では、`nisplus`)

- クライアントとなるマシンの root のパスワード
- クライアントのホームドメイン内の NIS+ サーバーの IP アドレス (この例では、マスターサーバー `client2` のアドレス)

▼ 新しいサブドメインクライアントマシンを初期設定する方法

- 新しいクライアントマシン上で新しいクライアントを初期設定するには、スーパーユーザーとして次のコマンドを入力します。

```
subclient1# nisclient -i -d sales.doc.com. -h client2
Initializing client subclient1 for domain "sales.doc.com.".
Once initialization is done, you will need to reboot your machine.
Do you want to continue? (type 'Y' to continue, 'N' to exit this script)
```

`-i` オプションはクライアントを初期設定します。`-d` オプションは新しい NIS+ ドメイン名を指定します。ドメイン名が指定されない場合、現在のドメイン名がデフォルトになります。`-h` オプションは NIS+ サーバーのホスト名を指定します。

このスクリプトの残りの出力については、111 ページの「新しいクライアントマシンを初期設定する方法」を参照してください。

サブドメインの NIS+ クライアントユーザーの初期設定

新しいドメイン内のユーザーを初期設定するには、ルートドメイン内のユーザーを初期設定する場合と同じ手順 (`nisclient`) を使用します。全ユーザーが NIS+ クライアントにならなければなりません。この例では、新しいドメインのユーザーを初期設定するための入力だけを示します。スクリプトの残りの出力については、113 ページの「NIS+ ユーザーを初期設定する方法」を参照してください。

`nisclient` を実行するための前提条件

`nisclient` スクリプトを使用してユーザーを初期設定するには、次の条件が必要です。

- ドメインがすでに構成され、そのマスターサーバーが実行されている
- ドメインのマスターサーバーのテーブルが生成されている。少なくとも、`hosts` テーブルには新しいクライアントマシンのエントリが必要

- ドメイン内のクライアントマシンが初期設定されている
- クライアントマシンに「ユーザー」としてログインしている (この例では、ユーザー名は user2)

必要な情報

次の情報が必要です。

- ユーザーのログイン名 (この例では、user2)
- デフォルトのネットワークパスワード (この例では、nisplus)
- NIS+ クライアントになるユーザーのログインパスワード

▼ NIS+ サブドメインユーザーを初期設定する方法

- **NIS+** クライアントとなるには、ユーザーとしてログインして、次のコマンドを実行します。

```
user2prompt% nisclient -u
At the prompt below, type the network password (also known as the
Secure-RPC password) that you obtained either from your administrator
or from running the nispopulate script.
Please enter the Secure-RPC password for user2:
```

このスクリプトの残りの出力については、113 ページの「NIS+ ユーザーを初期設定する方法」を参照してください。

NIS+ 名前空間サンプルで使用したコマンドのまとめ

名前空間サンプルの作成で使用したコマンドを表 4–4 にまとめます。各コマンドの前にあるプロンプトは、そのコマンドをどのマシンで入力するかを示します。

表 **4–4** NIS+ 名前空間サンプルのコマンド行のまとめ

目的	コマンド
/usr/lib/nis を含むように環境パスを設定 (C シェルまたは Bourne シェル)	**# setenv PATH $PATH:/usr/lib/nis** または **# PATH=$PATH:/usr/lib/nis; export PATH**
オプションで Diffie-Hellman キー長を設定する	master1# **nisauthconf dh640-0 des**
doc.com. ドメインのルートマスターサーバーを作成	master1# **nisserver -r -d doc.com.**
ルートマスターサーバーの NIS+ テーブルを、ファイルまたは NIS マップから生成	master1# **nispopulate -F -p /nis+files -d doc.com.** または master1# **nispopulate -Y -d doc.com. -h salesmaster -a \ 130.48.58.111 -y sales.doc.com.**
管理グループ (2) にメンバーを追加する	master1# **nisgrpadm -a admin.doc.com. topadmin.doc.com. \ secondadmin.doc.com.**
NIS+ データベースのチェックポイントを実行	master1# **nisping -C org_dir. doc.com.**
オプションで Diffie-Hellman キー長を設定する	client1# **nisauthconf dh640-0 des**
doc.com. ドメイン内の NIS+ クライアントマシンを初期設定	client1# **nisclient -i -d doc.com. -h master1**
ユーザーを NIS+ クライアントとして初期設定する	client1user1prompt% **nisclient -u**
NIS+ クライアントを、NIS 互換性あり、またはなし、あるいは NIS と DNS ありで NIS+ サーバーに変更	client1# **rpc.nisd** または client1# **rpc.nisd -Y** または client1# **rpc.nisd -Y -B**
ルート複製サーバーを作成	master1# **nisserver -R -d doc.com. -h client1**
サーバーを sales.doc.com. ドメインの非ルートのマスターサーバーに変更	master1# **nisserver -M -d sales.doc.com. -h client2**

表 **4–4** NIS+ 名前空間サンプルのコマンド行のまとめ　*(続き)*

目的	コマンド

新しいマスターサーバーの NIS+ テーブルを、ファイルまたは NIS マップから生成	`client2# nispopulate -F -p /nis+files -d sales.doc.com.`
	または
	`client2# nispopulate -Y -d sales.doc.com. -h \ businessmachine -a 130.48.58.242 -y business.doc.com.`

マスターサーバーの複製を作成	`client2# nisserver -R -d sales.doc.com. -h client3`
sales.doc.com.ドメイン内の NIS+ クライアントを初期設定	`subclient1# nisclient -i -d sales.doc.com. -h client2`
ユーザーを NIS+ クライアントとして初期設定する	`subclient1user2prompt% nisclient -u`

第 5 章

ルートドメインの設定

この章では、NIS+ コマンド群によるルートドメインと DES 認証の設定の手順を説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris™ 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

ルートドメインの設定方法の概要

ここでは、ルートマスターサーバーをセキュリティレベル 2 で稼動させるルートドメインの構成方法について説明します。

注 – ルートドメインの構成は、この章で説明する NIS+ コマンドより、NIS+ スクリプトを使用する方が簡単です。この章で説明する方法は、NIS+ に精通した管理者や、設定スクリプトでは提供されない標準以外の機能や構成を必要とする管理者だけが使用してください。

ルートドメインの設定には、次の 3 つの主な作業があります。

- ルートマスターサーバーの準備
- ルートドメインの作成
- ルートドメインの資格の作成

ただし、ルートドメインの設定はこれらの 3 つの作業を単にこの順で行うといった簡単なものではありません。3 つの作業はそれぞれ関連し合っています。たとえば、ルートディレクトリを作成する前にセキュリティパラメータの一部を指定しなければなりません。他のパラメータはその後で指定します。ルートドメインの構成を容易にするために、この節ではこれらの作業を個別の手順に分けて、一番効率の良い順番に並べています。

標準構成と NIS 互換構成の手順の相違

この章で説明する手順は、標準の NIS+ ルートドメインと NIS 互換のルートドメインの両方に適用できます。ただし、いくつかの重要な相違があります。NIS 互換ドメイン用の NIS+ デーモンは `-Y` オプションを使用して起動する必要があります。これによりルートマスターサーバーは、NIS クライアントからの要求に応えることができます。起動方法については「ルートドメインを構成する方法」の 手順 11 で説明します。標準の NIS+ ドメインについては 手順 12 で説明します。

また、NIS 互換ドメインでは、`passwd` テーブルは、未認証クラスに対する読み取り権が必要です。これにより NIS クライアントはテーブルのパスワード列にある情報にアクセスすることができます。これは 手順 14 で `nissetup` コマンドに `-Y` オプションを使用して行います。標準の NIS+ ドメインは同じコマンドを使用しますが、`-Y` オプションは付けません。

ルートドメインを確立する

各手順では詳細を説明し、また、関連する情報についても説明します。詳細な説明が必要ない場合は、150 ページの「ルートドメイン構成の要覧」に必要なコマンドをまとめたリストがありますので参照してください。

手順の要約

設定作業の手順を次にまとめます。

1. ルートマスターサーバーにスーパーユーザーとしてログインします。
2. ルートマスターサーバーのドメイン名をチェックします。
3. ルートマスターサーバーのスイッチ構成ファイルをチェックします。
4. オプションで Diffie-Hellman キー長を設定します。
5. NIS+ のファイルを削除しプロセスを終了します。

6. ルートドメインの管理グループを指定します。
7. ルートディレクトリを作成し、ルートマスターサーバーを初期設定します。
8. NIS+ デーモンを `-Y` で起動します (NIS の互換場合)。NIS+ デーモンを起動します (NIS+ の場合)。
9. NIS+ デーモンが実行されていることを確認します。
10. ルートドメインのサブディレクトリとテーブルを作成します。
11. ルートマスターサーバーの DES 資格を作成します。
12. ルートドメインの管理グループを作成します。
13. ルートドメインの管理 (admin) グループにルートマスターを追加します。
14. ルートドメインの公開鍵を更新します。
15. NIS+ キャッシュマネージャを起動します。
16. NIS+ デーモンをセキュリティレベル 2 で再起動します。
17. 自分の LOCAL 資格をルートドメインに追加します。
18. 自分の DES 資格をルートドメインに追加します。
19. 他のシステム管理者の資格を追加します。
20. ルートドメインの管理グループに自分と他のシステム管理者を追加します。

ルートドメインを確立する — 作業マップ

表 5–1 ルートドメインを確立する

作業	説明	指示の参照先
ルートドメインを確立する	`domainname` コマンドを使用して、ルートドメインを設定する。オプションで Diffie-Hellman キー長を長くする。`ncsd` デーモンの停止と起動を行う。`keyserv` デーモンの終了と再起動を行う。不要な NIS+ 情報を消去する	138 ページの「ルートドメインを構成する方法」

セキュリティについて

NIS+ はルートドメインに対しデフォルトのセキュリティを提供します。デフォルトのセキュリティレベルは、レベル 2 です。実際にユーザーが存在するネットワークは、常にセキュリティレベル 2 で運用してください。セキュリティレベル 0 およびレベル 1 は、構成およびテストの目的だけに使用します。本番環境のネットワークは、レベル 0 または 1 で稼動しないでください。

注 – NIS+ のセキュリティシステムは複雑です。NIS+ セキュリティを使い慣れていない場合は、第 11 章を参照してから NIS+ 環境を構成することをお勧めします。

前提条件

作業を開始する前に、以下のことを確認します。

- ルートマスターサーバー上の `/etc/passwd` ファイルには、この構成作業でルートドメインに資格を追加するシステム管理者である自分と、他のすべてのシステム管理者のエントリが含まれなければなりません。
- サーバーが NIS 互換モードで動作し、Solaris 1.x のクライアントで DNS 転送を使用する場合は、`/etc/resolv.conf` ファイルを正しく構成しておく必要があります。
- サーバーには、必ず他のユーザー ID と異なる、固有のマシン名を付けてください。
- サーバーに付けるマシン名には、ドットを使用しないでください。たとえば、マシン名に `sales.alpha` を使用できません。`sales-alpha` というマシン名は、使用できます。

必要な情報

ルートドメインを作成するには、次の内容について調べておく必要があります。

- ルートマスターサーバーとなるマシンのスーパーユーザーパスワード
- ルートドメイン名
- ルートドメインの管理グループ名
- 自分のユーザー ID とパスワード
- ルートドメインに資格を追加する予定の他のシステム管理者のユーザー ID

▼ ルートドメインを構成する方法

1. ルートマスターサーバーとするマシン上にスーパーユーザーとしてログインします。

   次の例では、`rootmaster` をルートマスターサーバーとして、`doc.com.` をルートドメインとして、それぞれ使用します。

2. ルートマスターサーバーのドメイン名をチェックします。

   `domainname` コマンドを使用して、ルートマスターサーバーが正しいドメイン名を使用していることを確認します。`domainname` コマンドは、マシンの現在のドメイン名を返します。

注意 – ドメインとホストで同じ名前を使用しないでください。たとえば、sales というドメインを使用している場合、sales という名前の付いたマシンは使用しないでください。同様に、home という名前のマシンを使用する場合は、home という名前のドメインを作成しないでください。これは、サブドメインの場合にもあてはまります。たとえば、マシンに west という名前を付けている場合、sales.west.myco.com というサブディレクトリを作成しないでください。

ドメイン名が正しくない場合は、変更してください。

```
rootmaster# domainname
strange.domain
rootmaster# domainname doc.com
rootmaster# domainname
rootmaster# doc.com
rootmaster# rm -f /etc/defaultdomain
rootmaster# domainname> /etc/defaultdomain
```

domainname コマンドの最後にドットを付けないでください。domainname コマンドは、NIS+ コマンドではありません。そのため、domainname コマンドは、ドメイン名の最後にドットを付ける NIS+ の表記規則には従っていません。

上記の例では、ルートマスターサーバーのドメイン名を strange.domain から doc.com に変更しています。ドメイン名を変更したり設定したりする場合は、doc ではなく doc.com のように、少なくとも 2 つのラベルを付けてください。末尾の要素はインターネットの組織コード (.com など) または地域識別子 (.jp、.uk など) にしてください。

3. ルートマスターサーバーのスイッチ構成ファイルをチェックします。

   ルートマスターサーバーが、NIS 互換モードで動作する場合であっても、NIS+ 用の nsswitch.conf ファイルを使用していることを確認してください。この手順で、ルートマスター用の情報の主情報源が確実に NIS+ テーブルとなります。

   ```
   rootmaster# more /etc/nsswitch.conf
   ```

   上のコマンドを実行すると、現在の nsswitch.conf ファイルの内容が表示されます。このファイルによって参照される主ネームサービスは nisplus です。ルートマスターサーバーの構成ファイルが主ネームサービスとして nisplus を使用していない場合は、45 ページの「構成ファイルの変更」を参照して、nisplus を使用する構成ファイルに置き換えてください。

4. オプションで **Diffie-Hellman** キー長を設定します。

   DES 認証を使用する場合には、Diffie-Hellman キー長をデフォルトの 192 ビットより長くすることができます。たとえば、640 ビットと 192 ビットの両方を許可する場合には、以下を入力します。

   ```
   rootmaster# nisauthconf dh640-0 des
   ```

5. nsswitch.conf ファイルに対する変更が終了したら、nscd デーモンを終了して再起動します。

nscd が nsswitch.conf ファイルの内容をキャッシュに格納しているため、ファイルの内容を変更したあとは、nscd を終了して再起動する必要があります。

詳細な手順については、第 1 章を参照してください。

6. 次のように入力し、keyserv を強制終了して再起動します。

```
rootmaster# cp /etc/nsswitch.nisplus /etc/nsswitch.conf
rootmaster# sh /etc/init.d/nscd stop
rootmaster# sh /etc/init.d/nscd start
rootmaster# ps -e | grep keyserv
 root 145 1 67 16:34:44 ? keyserv
 .
 .
rootmaster# kill -9 145
rootmaster# rm -f /etc/.rootkey
rootmaster# keyserv
```

7. **NIS+** のファイルを削除しプロセスを終了します。

現在作業しているマシンが、以前は NIS+ のサーバーまたはクライアントとして使用したものである場合、/var/nis 内にファイルがあればすべて削除し、キャッシュマネージャがまだ実行中であれば、そのプロセスを終了します。この例では、/var/nis 内にはコールドスタートファイルとディレクトリキャッシュファイルがまだ存在します。

```
rootmaster# ls /var/nis
NIS_COLD_START NIS_SHARED_CACHE
rootmaster# rm -rf /var/nis/*
rootmaster# ps -ef | grep nis_cachemgr
 root 295 260 10 15:26:58 pts/0 0:00 grep nis_cachemgr
 root 286 1 57 15:21:55 ? 0:01 /usr/sbin/nis_cachemgr
rootmaster# kill -9 286
```

この手順によって、/var/nis 内に残されたファイル、またはキャッシュマネージャによって格納されたディレクトリオブジェクトは完全に消去されるため、これらがこの設定作業で生成される新しい情報と重複することはありません。/var/nis 内に管理スクリプトを格納していた場合、ルートドメインの設定が終わるまでは、これらを一時的に他のどこかに格納しておくことができます。

8. サーバーのデーモンを終了します。

現在作業しているマシンが、すでに NIS+ サーバーとして使用されていた場合は、rpc.nisd または rpc.nispasswdd が実行されているかどうか確認してください。実行されている場合は、どちらのデーモンも終了してください。

9. ルートドメインの管理グループを指定します。

手順 16 までは、実際には管理グループを作成しませんが、ここで管理グループの名前を指定する必要があります。手順 13 で管理グループの名前を指定することによって、ルートドメインの org_dir ディレクトリオブジェクト、groups_dir ディレクトリオブジェクト、およびその全テーブルオブジェクトが 手順 14 で作成されたときに、適切なデフォルトグループが割り当てられます。

管理グループを指定するには、環境変数 NIS_GROUP の値に、ルートドメインの管理グループの名前を設定します。次に、csh ユーザー用と、sh/ksh ユーザー用の 2 つ

の例を示します。どちらも、NIS_GROUP を admin.doc.com. に設定します。

C シェルの場合

```
rootmaster# setenv NIS_GROUP admin.doc.com.
```

Bourne シェルまたは Korn シェルの場合

```
rootmaster# NIS_GROUP=admin.doc.com.
rootmaster# export NIS_GROUP
```

10. ルートディレクトリを作成し、ルートマスターサーバーを初期設定します。

この手順では、名前空間内の最初のオブジェクトであるルートディレクトリを作成し、マシンをルートマスターサーバーに変換します。次に示すように、nisinit -r コマンドを使用します。この場合に限り、ドメインのディレクトリオブジェクトの作成と、そのマスターサーバーの初期設定が 1 回の操作で行われます。実際には nisinit -r が nismkdir を実行し、自動的にルートディレクトリを作成します。いずれにしても、ルートマスターの場合を除くと、これらの 2 つのプロセスは別々の作業として行われます。

```
rootmaster# nisinit -r
This machine is in the doc.com. NIS+ domain
Setting up root server ...
All done.
```

/var/nis/data という名前の UNIX ディレクトリが生成されます。

この下には root.object. という名前のファイルがあります。

```
rootmaster# ls -l /var/nis/data
-rw-rw-rw- 1 root other 384 date root.object
```

これはルートディレクトリオブジェクトではありません。実際には、NIS+ が内部目的のために名前空間の root を記述するために使用するファイルです。NIS+ のルートディレクトリオブジェクトは、手順 11 または 手順 12 で作成します。

以降の手順では、この手順で作成されるディレクトリの下に、他のファイルが追加されます。UNIX ディレクトリを直接調べることによって、これらのファイルの存在を検証できますが、NIS+ にはもっと便利なコマンドがあります。以下の手順では、必要に応じてこれらのコマンドを実行します。

注意 – nisinit など NIS+ の構成手順によって作成された /var/nis、/var/nis/data といったディレクトリ、およびその下のファイルは、名前を変更しないでください。Solaris 2.4 以前では、/var/nis ディレクトリに *hostname*.dict、*hostname*.log という 2 つのファイルとサブディレクトリ /var/nis/*hostname* が存在していました。またサブディレクトリ /var/nis/*hostname* もありました。Solaris 2.5 では、これらの 2 つのファイル名は trans.log、data.dict とし、サブディレクトリ名は /var/nis/data となります。Solaris 2.5 ではこれらのファイルの内容も変更されており、Solaris 2.4 以前との互換性はなくなっています。したがって、これらのファイルやディレクトリを Solaris 2.4 での名前にしてしまうと、Solaris 2.4、2.5 双方の rpc.nisd で機能しなくなるので名前の変更をしないようにしてください。ディレクトリ名もファイル名も変更しないでください。

---

11. **NIS+** デーモンを `-Y` で起動します **(NIS** 互換の場合のみ**)**。

    この手順は、NIS 互換モードでルートドメインを設定している場合にだけ実行します。標準の NIS+ ドメインを設定する場合は、この代わりに 手順 12 を実行します。この手順には、NIS クライアントの DNS 転送機能をサポートするための操作手順が含まれます。

    「a」では、NIS+ デーモンを NIS 互換モードで起動します。「b」では、NIS+ デーモンはこのサーバーが再起動されたとき、確実に NIS 互換モードで再起動します。手順 b の「b」の後は、手順 14 に進んでください。

    a. `rpc.nisd` に `-Y`、`-B`、`-S 0` オプションを使用して実行します。

       ```
       rootmaster# rpc.nisd -Y -B -S 0 options
       ```

       `-Y` オプションを指定すると、NIS+ 要求だけでなく NIS 要求にも応答します。`-B` オプションを指定すると DNS 転送をサポートします。`-S 0` フラグは、サーバーのセキュリティレベルに 0 を設定します。この時点では、ブートストラップ動作のためにこの設定が必要です。`cred` テーブルはまだ存在しないため、NIS+ 主体は資格をもつことができません。これより高いセキュリティレベルを使用した場合、ユーザーがサーバーから締め出されてしまいます。

    b. `/etc/init.d/rpc` ファイルを編集します。

       `/etc/init.d/rpc` ファイル内で文字列 `EMULYP="Y"` を検索します。その行のコメント指定を解除し、DNS 転送機能を使用できるようにするために、`-B` フラグを追加します。

       DNS 転送を使用する `rpc` ファイルの場合

       ```
       EMULYP="-Y -B"
       ```

       DNS 転送を使用しない `rpc` ファイルの場合

       ```
       EMULYP="-Y"
       ```

       DNS 転送機能を使用する必要がない場合、この行をコメント解除しますが、`-B` フラグは追加しません。

12. **NIS+** デーモンを起動します **(**標準 **NIS+** の場合のみ**)**。

    `rpc.nisd` コマンドを使用しますが、必ず `-S 0` フラグを追加してください。

    ```
    rootmaster# rpc.nisd -S 0
    ```

    `-S 0` フラグは、サーバーのセキュリティレベルに 0 を設定します。この時点では、ブートストラップ動作のためにこの設定が必要です。`cred` テーブルはまだ存在しないため、NIS+ 主体は資格をもつことができません。これより高いセキュリティレベルを使用した場合、ユーザーがサーバーから締め出されてしまいます。

13. ルートオブジェクトが正しく作成されているかどうか確認します。

    手順 11 または 手順 12 を実行すると、名前空間には次のものが作成されます。

    - ルートディレクトリオブジェクト (`root.dir`)
    - NIS+ デーモン (`rootmaster`) を実行するルートマスターサーバー (`rpc.nisd`)
    - マスターサーバー用のコールドスタートファイル (`NIS_COLD_START`)

- トランザクションログファイル (trans.log)
- テーブル辞書ファイル (data.dict)

ルートディレクトリオブジェクトは、手順 10 で作成したディレクトリに格納されます。ls コマンドを使用して、ディレクトリの内容を確認してください。

```
rootmaster# ls -l /var/nis/data
-rw-rw-rw- 1 root other 384 date root.object
-rw-rw-rw- 1 root other 124 date root.dir
```

この時点で、ルートディレクトリは空です。つまり、サブディレクトリはありません。このことを確認するには、nisls コマンドを使用します。

```
rootmaster# nisls -l doc.com.
doc.com.:
```

ただし、いくつかの「オブジェクト」属性は存在し、niscat -o を使用すればこれを調べることができます。

```
rootmaster# niscat -o doc.com.
 Object Name : doc
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : Com.
Access Rights : r---rmcdrmcdr---
```

なお、ルートディレクトリオブジェクトは、「所有者」と「グループ」の両方には完全な (読み取り、変更、作成、削除) 権利を与え、「その他」と「未認証」には読み取り権だけを与えます。ディレクトリオブジェクトがこれらの権利を提供しない場合、nischmod コマンドを使用して変更できます。

NIS+ デーモンが動作していることを確認するには、ps コマンドを使用します。

```
rootmaster# ps -ef | grep rpc.nisd
root 1081 1 61 16:43:33 ? 0:01 rpc.nisd -S 0
root 1087 1004 11 16:44:09 pts/1 0:00 grep rpc.nisd
```

ルートマスターサーバーのインターネットアドレスが (最終的には公開鍵も) 収められているルートドメインの NIS_COLD_START ファイルは、/var/nis 内に置かれます。この内容を調べるための NIS+ コマンドはありませんが、この内容はサーバーのディレクトリキャッシュ (NIS_SHARED_DIRCACHE) に保存されます。これらの内容は、/usr/lib/nis/nisshowcache コマンドで調べることができます。

また、トランザクションログファイル (trans.log) と辞書ファイル (data.dict) も作成されます。マスターサーバーのトランザクションログは、前回の更新以降マスターサーバーとそのすべての複製サーバーによって実行されたすべてのトランザクションを格納しています。この内容を調べるには、nislog コマンドを使用します。辞書ファイルは、NIS+ が内部目的に使用するものであり、システム管理者には関係ありません。

14. ルートドメインのサブディレクトリとテーブルを作成します。

この手順では、ルートディレクトリオブジェクトの下に、org_dir ディレクトリと groups_dir ディレクトリ、および NIS+ テーブルを追加します。nissetup ユーティリティを使用してください。NIS 互換ドメインの場合、必ず -Y フラグを付けてください。両方の場合の例を次に示します。

標準 NIS+ のみの場合

```
rootmaster# /usr/lib/nis/nissetup
```

NIS 互換のみの場合

```
rootmaster# /usr/lib/nis/nissetup -Y
```

このユーティリティによって追加されたオブジェクトを表示すると、次のようになります。

```
rootmaster# /usr/lib/nis/nissetup
org_dir.doc.com. created
groups_dir.doc.com. created
auto_master.org_dir.doc.com. created
auto_home.org_dir.doc.com. created
bootparams.org_dir.doc.com. created
cred.org_dir.doc.com. created
ethers.org_dir.doc.com. created
group.org_dir.doc.com. created
hosts.org_dir.doc.com. created
mail_aliases.org_dir.doc.com. created
sendmailvars.org_dir.doc.com. created
netmasks.org_dir.doc.com. created
netgroup.org_dir.doc.com. created
networks.org_dir.doc.com. created
passwd.org_dir.doc.com. created
protocols.org_dir.doc.com. created
rpc.org_dir.doc.com. created
services.org_dir.doc.com. created
timezone.org_dir.doc.com. created
```

`-Y` オプションは、標準の NIS+ ドメインと同じテーブルとサブディレクトリを作成します。しかし、NIS クライアントからの未認証要求が `passwd` テーブルに含まれる暗号化されたパスワードにアクセスできるようにするため、`passwd` テーブルへの読み取り権を未認証クラスに割り当てます。

`nisls` でルートディレクトリの内容を調べたとき (手順 12)、これが空であったことを思い出してください。現在は 2 つのサブディレクトリがあります。

```
rootmaster# nisls doc.com.
doc.com.:
org_dir
groups_dir
```

サブディレクトリとテーブルのオブジェクト属性を調べるには、`niscat -o` コマンドを使用してください。また、フラグなしで `niscat` オプションを使用して、このテーブル内の情報を調べることもできますが、この時点では空です。

15. ルートマスターサーバーの **DES** 資格を作成します。

ルートマスターサーバーは、自分の要求が認証されるために、DES 資格を必要とします。これらの資格を作成するには、次に示すように `nisaddcred` コマンドを使用します。プロンプトに対して、サーバーの root パスワードを入力します。

```
rootmaster# nisaddcred des
DES principal name: unix.rootmaster@doc.com
Adding key pair for unix.rootmaster@doc.com
 (rootmaster.doc.com.).
Enter login password:
```

```
Wrote secret key into /etc/.rootkey
```

サーバーの root パスワードと異なるパスワードを入力した場合、警告メッセージとパスワードを再入力するプロンプトが表示されます。

```
Enter login password:
nisaddcred: WARNING: password differs from login password.
Retype password:
```

それでも同じパスワードを再び入力すると、NIS+ は資格を作成します。この新しいパスワードは /etc/.rootkey に格納され、キーサーバーが起動時に使用します。キーサーバーに新しいパスワードをすぐに登録するには、keylogin -r を実行します。第 12 章を参照してください。

最終的に自分のログインパスワードを使用する場合は、Control-C を押して、この手順をやり直します。Control-C を押さずにメッセージに従って自分のログインパスワードを入力すると、別の目的のエラーメッセージが表示され、混乱を招くことがあります。

```
nisaddcred: WARNING: password differs from login password.
Retype password:
nisaddcred: password incorrect.
nisaddcred: unable to create credential.
```

この手順の結果として、ルートサーバーの非公開鍵と公開鍵がルートドメインの cred テーブル (cred.org_dir.doc.com.) に格納され、その非公開鍵は /etc/.rootkey. に格納されます。cred テーブル内の資格を確認するには、niscat コマンドを使用します (第 8 章を参照)。デフォルトのドメイン名は doc.com. であるため、cred テーブルの完全指定名を入力する必要はありません。接尾辞の org_dir で十分です。ルートマスターの資格を探すためには、その Secure RPC ネット名を探してください。

16. ルートドメインの管理グループを作成します。

この手順では、手順 9 で指定された管理グループを作成します。nisgrpadm コマンドに -c オプションを付けて実行してください。次の例では admin.doc.com. グループを作成します。

```
rootmaster# nisgrpadm -c admin.doc.com.
 Group admin.doc.com. created.
```

この手順ではグループを作成するだけであり、そのメンバー名の指定は行いません。メンバー名の指定については、手順 17 で行います。グループのオブジェクト属性を見るには、niscat -o を使用します。しかし、グループ名では必ず groups_dir を使用します。たとえば、次のようになります。

```
doc.com.
Object Name : admin
Directory : groups_dir.doc.com
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : groups_dir.doc.com.
Access Rights : ----rmcdr---r---
Time to Live : 1:0:0
Object Type : GROUP
Group Flags :
Group Members :
```

17. ルートドメインの管理 **(admin)** グループにルートマスターを追加します。

この時点では、このルートマスターサーバーは DES 資格をもつ唯一の NIS+ 主体であるため、管理グループに追加すべき唯一のメンバーです。今度は、`nisgrpadm` コマンドに `-a` オプションを付けて実行します。第 1 引数はグループ名、第 2 引数はルートマスターサーバー名です。この例では、`rootmaster.doc.com.` を `doc.com` ドメインに追加します。

```
rootmaster# nisgrpadm -a admin.doc.com.
 rootmaster.doc.com.
Added rootmaster.doc.com. to group admin.doc.com.
```

ルートマスターがこの管理グループに属していることを確認するには、`nisgrpadm` コマンドを、`-l` オプションを指定して実行します。第 17 章を参照してください。

注 – `nisgrpadm` などのグループ関連コマンドでは、名前に `groups_dir` サブディレトリを含める必要はありません。`niscat` などのコマンドの場合、NIS+ オブジェクト一般に対して動作するように設計されているため、このディレクトリを含む必要があります。グループ関連コマンドは、`groups_dir` サブディレクトリを対象としています。

```
rootmaster# nisgrpadm -l admin.doc.com.
 Group entry for admin.doc.com. group:
 Explicit members:
 rootmaster.doc.com.
 No implicit members
 No recursive members
 No explicit nonmembers
 No implicit nonmembers
 No recursive nonmembers
```

18. ルートドメインの公開鍵を更新します。

ディレクトリオブジェクトは、すでに DES 資格をもつ NIS+ 主体によって作成されるのが普通です。しかしこの場合、`cred` テーブルを作成するまでは、自分の資格を格納する親ドメインがないため、ルートマスターサーバーは DES 資格を獲得できません。その結果、`root`、`org_dir`、および `groups_dir` という 3 つのディレクトリオブジェクトには、ルートマスターサーバーの公開鍵のコピーがありません。これについては、`niscat -o` コマンドにどれかのディレクトリオブジェクトを指定して実行することによって確認できます。「Public Key:」フィールドを探してください。手順については、第 18 章 を参照してください。

ルートマスターサーバーの公開鍵を、ルートドメインの `cred` テーブルからこれら 3 つのディレクトリオブジェクトに伝達するには、次に示すように、各ディレクトリオブジェクトに対して `/usr/lib/nis/nisupdkeys` ユーティリティを使用します。

```
rootmaster# /usr/lib/nis/nisupdkeys doc.com.
rootmaster# /usr/lib/nis/nisupdkeys org_dir.doc.com.
rootmaster# /usr/lib/nis/nisupdkeys groups_dir.doc.com.
```

コマンドを実行するたびに、次のような確認メッセージが表示されます。

```
Fetch Public key for server rootmaster.doc.com.
 netname = 'unix.rootmaster@doc.com.'
Updating rootmaster.doc.com.'s public key.
 Public key:
```

これらのディレクトリを (niscat -o を使用して) 表示すると、「Public key:」フィールドに 1 つまたは複数のエントリが見つかります。

```
Public key: Diffie-Hellman (192 bits)
```

19. NIS+ キャッシュマネージャを起動します。

キャッシュマネージャは、NIS+ クライアント (この場合、ルートマスターサーバー) に関する位置情報のローカルキャッシュを管理します。初期の情報をクライアントのコールドスタートファイル (手順 11 または 手順 12 で作成) から入手し、これを /var/nis 内の NIS_SHARED_DIRCACHE という名前のファイルに保存します。

キャッシュマネージャを起動するには、次に示すように、nis_cachemgr コマンドを入力します。

```
rootmaster# nis_cachemgr
```

キャッシュマネージャがいったん起動されたら、このプロセスを明示的に終了させた場合を除いて、再起動する必要はありません。クライアントが再起動されたときには、/var/nis 内の NIS_COLD_START ファイルがキャッシュマネージャを自動的に起動するため、キャッシュマネージャを再起動する必要はありません。

20. NIS+ デーモンをセキュリティレベル 2 で再起動します。

これで、ルートマスターサーバーには DES 資格があり、ルートディレクトリオブジェクトにはルートマスターの公開鍵のコピーがあるため、ルートマスターをセキュリティレベル 2 で再起動できます。まず既存のデーモンのプロセスを終了し、次にこれをセキュリティレベル 2 で再起動します。

標準 NIS+ ドメインの場合

```
rootmaster# ps -e | grep rpc.nisd
1081 ? 0:03 rpc.nisd -s 0
rootmaster# kill 1081
rootmaster# rpc.nisd
```

NIS 互換のルートドメインの場合、必ず -Y フラグ (および -B フラグ) を使用してください。

NIS 互換の NIS+ ドメインの場合

```
rootmaster# ps -e | grep rpc.nisd
1081 ? 0:03 rpc.nisd -Y -B -s 0
rootmaster# kill 1081
rootmaster# rpc.nisd -Y -B
```

セキュリティレベル 2 はデフォルトであるため、-S 2 フラグは不要です。

注 – 実際にユーザーが存在するネットワークは、常にセキュリティレベル 2 で運用してください。セキュリティレベル 0 およびレベル 1 は、設定およびテストの目的だけに使用します。本番環境ネットワークは、レベル 0 または 1 で稼動しないでください。

---

21. 自分の **LOCAL** 資格をルートドメインに追加します。

ユーザーはルートドメインの `cred` テーブルに対するアクセス権がないため、この動作はスーパーユーザーとして実行しなければなりません。さらに、「前提条件」で説明したように、ルートマスターの `/etc/passwd` ファイルには自分用のエントリが必要です。`nisaddcred` コマンドに `-p` と `-P` フラグを付けて使用します。その構文と例を次に示します。

```
nisaddcred -p uid -P principal-name local
```

principal-name はシステム管理者のログイン名とドメイン名で構成されます。この例では、UID が 11177 で NIS+ 主体名が `topadmin.doc.com.` のシステム管理者用の LOCAL 資格を追加します。

```
rootmaster# nisaddcred -p 11177 -P topadmin.doc.com. local
```

`nisaddcred` コマンドの詳細については、第 12 章を参照してください。

22. 自分の **DES** 資格をルートドメインに追加します。

ここでも `nisaddcred` コマンドを使用しますが、次のような構文となります。

```
nisaddcred -p secure-RPC-netname- P principal-name des
```

secure-RPC-netname は、接頭辞 `unix` に自分のユーザー ID、`@` 記号、およびドメイン名を付けて構成しますが、最後にドットは付けません。*principal-name* は LOCAL 資格の場合と同じで、ログイン名にドメイン名を付け、さらに最後にドットを付けます。

```
rootmaster# nisaddcred -p unix.11177@doc.com -P topadmin.doc.com. des
Adding key pair for unix.11177@doc.com (topadmin.doc.com.).
Enter login password:
```

ログインパスワードの入力後に「`password that differs from the login password`」という警告が表示され、入力したパスワードが正しいログインパスワードである場合は、このエラーメッセージを無視してください。NIS+ がパスワードの格納されている、保護された `/etc/shadow` ファイルを期待どおりに読み込めないため、このメッセージが表示されます。ユーザーパスワード情報を `/etc/passwd` ファイルに格納していない場合、このメッセージは表示されません。

23. 他のシステム管理者の資格を追加します。

そのルートドメインで作業する他のシステム管理者の LOCAL 資格と DES 資格を追加します。これには 以下の方法があります。

- 他の管理者に一時的な資格を作成するには、NIS+ モードで実行されている Solstice AdminSuite (使用できる場合) を使用すると簡単です。

- 2 つ目は、他の管理者にその管理者自身の資格を追加するよう要求する方法です。この方法は、スーパーユーザーとして実行する必要があります。ユーザー ID が 33355 で主体名が miyoko.doc.com. のシステム管理者の資格を追加する例を次に示します。

```
rootmaster# nisaddcred -p 33355 -P miyoko.doc.com. local
rootmaster# nisaddcred -p unix.33355@doc.com -P miyoko.doc.com. des
Adding key pair for unix.33355@doc.com (miyoko.doc.com.).
 Enter login password:
```

- 3 つ目は、ダミーのパスワードを使用して、他の管理者に一時的な資格を作成する方法です。他の管理者、この例では miyoko は、NIS+ passwd テーブルにエントリがなければなりません。エントリがない場合は、まず nistbladm を使用して、必ずエントリを作成してください。次の例で、その手順を示しています。

```
rootmaster# nistbladm -D owner=miyoko.doc.com. name=miyoko uid=33355 gcos=miyoko
home=/home/miyoko shell=/bin/tcsh passwd.org_dir
rootmaster# nisaddent -a -f /etc/passwd.xfr passwd
rootmaster# nisaddent -a -f /etc/shadow.xfr shadow
rootmaster# nisaddcred -p 33355 -P miyoko.doc.com. local
rootmaster# nisaddcred -p unix.33355@doc.com -P miyoko.doc.com. des
Adding key pair for unix.33355@doc.com (miyoko.doc.com.).
Enter miyoko's login password:
nisaddcred: WARNING: password differs from login passwd.
Retype password:
rootmaster# nischown miyoko.doc.com. '[name=miyoko],passwd.org_dir'
```

この場合、最初の nisaddent で passwd テーブルにエントリが生成されます (パスワード列は除く)。次の nisaddent により、shadow 列が生成されます。各システム管理者は、chkey コマンドを使用することによって、自分のネットワークパスワードを後で変更できます。この手順については、第 12 章を参照してください。

24. ルートドメインの管理グループに自分と他のシステム管理者を追加します。

この手順を実行するには、他のシステム管理者がダミーパスワードを変更するまで待つ必要はありません。-a オプションを付けて nisgrpadm コマンドを実行します。最初の引数はグループ名であり、残りの引数はシステム管理者の名前です。この例では、topadmin と miyoko の 2 人のシステム管理者を admin.doc.com. グループに追加します。

```
rootmaster# nisgrpadm -a admin.doc.com. topadmin.doc.com. miyoko.doc.com.
Added topadmin.doc.com. to group admin.doc.com.
Added miyoko.doc.com. to group admin.doc.com.
```

25. **NIS+** テーブルを格納するための、十分なスワップ空間を割り当てます。

スワップ空間は、rpc.nisd の最大サイズの 2 倍にする必要があります。rpc.nisd が使用するメモリ量を調べるには、次のコマンドを実行してください。

```
rootmaster# /usr/lib/nis/nisstat
```

rpc.nisd は、特定の条件のもとでは、自らのコピーを作成してフォークします。メモリーが不足すると、rpc.nisd は正しく動作しません。

また、NIS+ テーブルに必要なメモリーとスワップ空間のサイズも計算できます。たとえば、NIS+ テーブル内に、180,000 人のユーザーと 180,000 台のホストがある場合、これらの 2 つのテーブルが占有するメモリーは、約 190M バイトです。180,000 人のユーザーと180,000 台のホストに資格を追加する場合、`cred` テーブルには、540,000 のエントリ (ユーザーごとにローカルの資格と DES の資格、合わせて 2 つの資格、ホストごとに 1 つの資格) が入ります。そのため、`cred` テーブルが占有するメモリーは、約 285M バイトになります。この例では、`rpc.nisd` に必要なメモリー容量は、少なくとも、190M バイト + 285M バイト = 475M バイトになります。この結果、少なくとも 1G バイトのスワップ空間が必要になります。また、`rpc.nisd` 全体をすべてメモリー内に保持するには、少なくとも 500M バイトが必要です。

ルートドメイン構成の要覧

ルートドメインの構成に必要な手順を表 5–2 にまとめます。このまとめは、簡単な場合を想定しています。このまとめを参考として使用する前に、作業全体の説明を十分に理解しておいてください。また、ここでは、各コマンドに対するサーバーの応答を示していません。

表 **5–2** ルートドメインを設定する手順のまとめ

作業	コマンド
ルートマスターサーバーにスーパーユーザーとしてログインする	`rootmaster% su` `Password:` root パスワードを入力する
ドメイン名をチェックする	`# domainname`
スイッチファイルをチェックする	`# more /etc/nsswitch.conf`
NIS+ データを削除し、プロセスを終了する	`# rm -rf /var/nis*`
管理 (admin) グループを指定する	`# NIS_GROUP=admin.doc.com.; export NIS_GROUP`
ルートマスターサーバーを初期設定する	`# nisinit -r`
NIS 互換の場合のみ :	`# rpc.nisd -Y -B -S 0`
デーモンを `-Y -B, S 0` で起動する	`# vi /etc/inet.d/rpc`
`EMULYP=-Y -B` に変更する	
標準 NIS+ の場合のみ : デーモンを `-S 0` で起動する	`# rpc.nisd -S 0`
`org_dir` と、`groups_dir` テーブルを作成する	`# /usr/lib/nis/nissetup [-Y]`

表 **5–2** ルートドメインを設定する手順のまとめ　(続き)

作業	コマンド
ルートマスターサーバー用の DES 資格を作成する	#nisaddcred des Enter login password: ログインパスワードを入力する
管理グループを作成する	# nisgrpadm -c admin.doc.com.
ルートディレクトリに完全なグループ権を割り当てる	# nischmod g+rmcd doc.com.
ルートマスターを管理グループに追加する	# nisgrpadm -a admin.doc.com. rootmaster.doc.com.
ルートディレクトリの鍵を更新する。org_dir の鍵を更新する。groups_dir の鍵を更新する	# /usr/lib/nis/nisupdkeys doc.com. # /usr/lib/nis/nisupdkeys org_dir.doc.com. # /usr/lib/nis/nisupdkeys groups_dir.doc.com.
NIS+ キャッシュマネージャを起動する	# nis_cachemgr
既存のデーモンを終了する	# ps -ef | grep rpc.nisd # kill -9 *process-id*
NIS+ デーモンを再起動する (NIS 互換には -Y を、DNS には -B を使用する)	# rpc.nisd [-Y] [-B]
自分の LOCAL 資格を追加する	# nisaddcred -p 11177 -P topadmin.doc.com. local
自分の DES 資格を追加する	# nisaddcred -p unix.11177@doc.com -P topadmin.doc.com. desEnter login password:
他のシステム管理者の資格を追加する	# nisaddcred ... nisgrpadm -a admin.doc.com *members*
他のシステム管理者を管理グループに追加する	

第 6 章

NIS+ クライアントの構成

この章では、NIS+ コマンド群を使用した 3 通りの初期設定方法で NIS+ クライアントを設定する手順を説明します。この章の内容は、NIS+ モード、NIS+ 互換モードを問わず、ルートドメイン、サブドメインに共通するものです。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ クライアントの設定の概要

この章では、標準の NIS+ ドメインと NIS 互換ドメイン内のクライアントを構成する方法を説明します。手順の説明ではそれぞれの内容を詳細に説明し、関連する情報も示します。詳しい手順の説明が必要でない場合は、表 6–6 の「必要なコマンドの一覧」を参照してください。

注 – NIS+ クライアントを設定する作業は、この章で説明する NIS+ のコマンドセットを使用する方法よりも、パート I で説明した NIS+ 設定スクリプトを使用する方が簡単です。この章で説明する方法は、NIS+ に精通した管理者や、設定スクリプトでは提供されない標準以外の機能や構成を必要とする管理者だけが使用してください。Solstice AdminSuite がある場合には、これを使用すると NIS+ クライアントマシンの追加や設定の作業が簡単にできます。

クライアントを設定する作業のうち、手順 10 では、ブロードキャスト、ホスト名、またはコールドスタートファイルのうちのどれかを使用する方法を選択してください。それぞれ実装方法が異なるため、各作業について別々に説明します。クライアントを初期設定したあとは、手順 11 に戻ってクライアントの設定を続けてください。

この章の最後の作業では、マシンのドメイン名を変更する方法を取り上げています。

クライアントの構成

ここでは、ルートドメイン内であるかどうかとは関係なく、一般的な NIS+ クライアントの構成方法を説明します。ここでの説明は、通常の NIS+ クライアント、および後で NIS+ サーバーとなるクライアントに当てはまります。また、標準の NIS+ ドメイン内、および NIS 互換ドメイン内のクライアントにも当てはまります。

注意 – ドメインとホストで同じ名前を使用しないでください。たとえば、`sales` というドメインを使用している場合、`sales` という名前の付いたマシンは使用しないでください。同様に、`home` というホスト名がある場合には、ドメイン名に `home` を使用できません。これは、サブドメインの場合にもあてはまります。たとえば、マシンに `west` という名前を付けている場合、`sales.west.myco.com` というサブドメインを作成しないでください。

NIS+ クライアントの設定には、次の作業が必要です。

- クライアントの資格の作成
- マシンの準備
- マシンを NIS+ クライアントとして初期設定

ただし、ルートドメインの設定と同様、クライアントの設定も、これら 3 つの作業を順番に実行するような単純なものではありません。構成手続を実行しやすくするため、これらの作業を個々の手順に分割し、次に示すように、これらの手順を最も効率的な順に並べてあります。

1. ドメインのマスターサーバーにログインします。
2. 新しいクライアントマシン用の DES 資格を作成します。
3. マスターサーバーで使用されている Diffie-Hellman キー長を確認します。
4. クライアントにスーパーユーザーとしてログインします。
5. クライアントに新しいドメイン名を設定します。
6. `nscd` の停止と再起動を行います。
7. クライアントの `nsswitch.conf` ファイルの設定を確認します。
8. クライアントの Diffie-Hellman キーを設定します。
9. NIS+ のファイルを削除し、プロセスを終了します。
10. クライアントを初期設定します。
11. `keyserv` デーモンのプロセスを終了して再起動します。

12. keylogin を実行します。
13. クライアントを再起動します。

セキュリティについて

クライアントの設定には、セキュリティ上の主な必要要件が 2 つあります。つまり、システム管理者とクライアントの両方が、適切な資格とアクセス権を持つことです。そうでない場合、クライアントがセキュリティレベル 2 で実行しているドメインの資格を入手する唯一の方法は、クライアントのホームドメイン内での有効な DES 資格と cred テーブルに対する変更権とを持つシステム管理者が資格を作成することです。システム管理者は DES 資格を、クライアントのホームドメイン内、または自分のホームドメイン内に所持できます。

システム管理者がクライアントの資格を作成すると、そのクライアントは構成プロセスを終了できます。しかしクライアントは、ホームドメインのディレクトリオブジェクトに対する読み取り権を必要とします。第 5 章または第 8 章の手順に従ってクライアントのホームドメインを構成した場合、ディレクトリオブジェクトの作成に使用した NIS+ コマンド (nisinit と nismkdir) によって、読み取り権がその他のクラスに提供されています。

ディレクトリオブジェクトのアクセス権をチェックするには、niscat -o コマンドを使用します。このコマンドは、アクセス権などのディレクトリ属性を表示します。次にその例を示します。

```
rootmaster# niscat -o doc.com.
ObjectName : Doc
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : Com.
Access Rights : r---rmcdr---r---
```

ディレクトリオブジェクトのアクセス権は、オブジェクトに対する変更権を持っている場合は、nischmod コマンドを使用して変更できます。詳細については、第 15 章を参照してください。

前提条件

クライアントの資格を設定するシステム管理者は、次の条件をすべて満たしている必要があります。

- 有効な DES 資格を持っていること
- クライアントのホームドメインにある cred テーブルを修正する権利を持っていること

クライアントは次の条件をすべて満たしている必要があります。

- ホームドメインのディレクトリオブジェクトに対する読み取り権を持っていること

- クライアントのホームドメインがあらかじめ構成されており、NIS+ を実行していること
- マスターサーバーの /etc/hosts ファイル、/etc/inet/ipnodes ファイルまたはドメインの hosts テーブル、ipnodes テーブルの中にエントリが存在すること
- マシン名が、どのユーザーの ID とも重複しておらず、固有であること
- マシン名にドットが含まれていないこと。たとえば、sales.alpha というマシン名は使用できません。sales-alpha というマシン名なら使用できます。

必要な情報

- クライアントのホームドメイン名
- クライアントとなるマシンのスーパーユーザーのパスワード
- クライアントのホームドメイン内にある NIS+ サーバーの IP アドレス

クライアントの設定 — 作業マップ

表 **6–1** クライアントの設定

作業	説明	指示の参照先
クライアントの設定	クライアントの資格を作成する。クライアントマシンを準備して、NIS+ クライアントとして初期設定する	156 ページの「NIS+ クライアントを設定する方法 」

▼ NIS+ クライアントを設定する方法

1. ドメインのマスターサーバーにログインします。

   スーパーユーザーとして、または自分自身のユーザー名でログインします。どちらでログインするかは、どちらの NIS+ 主体がドメインの cred テーブルに資格を追加するための適切なアクセス権を所有しているのかに依存します。

2. 新しいクライアントマシン用の **DES** 資格を作成します。

   -p と -P の引数を付けた nisaddcred コマンドを実行します。

   ```
   nisaddcred -p secure-RPC-netname principal-name des [domain]
   ```

   secure-RPC-netname は、接頭辞 unix に、クライアントのホスト名、@ 記号、およびクライアントのドメイン名を付けて構成しますが、最後にドットは付けません。*principal-name* は、クライアントのホスト名とドメイン名によって構成され、最後にドットを付けます。このクライアントの所属するドメインが、コマンドを入力したサーバーとは異なる場合、2 番目の引数の後にクライアントのドメイン名を追加します。

   この例では、doc.com. ドメイン内の client1 という名前のクライアントマシンに対する DES 資格を追加します。

```
rootmaster% nisaddcred -p unix.client1@doc.com -P client1.doc.com. des
Adding key pair for unix.client1@doc.com (client1.doc.com.).
Enter client1.doc.com.'s root login passwd:
Retype password:
```

nisaddcred コマンドの詳細については、第 12 章を参照してください。

3. マスターサーバーで使用される **Diffie-Hellman** キー長を確認します。

   たとえば:

   ```
   rootmaster% nisauthconf dh640-0 des
   ```

4. クライアントにスーパーユーザーとしてログインします。

   これでクライアントマシンに資格ができたため、ユーザーはマスターサーバーからログアウトし、クライアント自体から作業を開始できます。この作業はローカルでもリモートでも可能です。

5. クライアントに新しいドメイン名を設定します。

   クライアントのドメイン名を設定する (変更する) 方法については、159 ページの「マシンのドメイン名を変更する」を参照し、次の 手順 6 に戻ります。

6. クライアントの nsswitch.conf ファイルをチェックします。

   クライアントが NIS+ バージョンの nsswitch.conf ファイルを使用していることを確認します。これによって、クライアント情報の 1 次ソースが NIS+ テーブルということが確認できます。NIS+ スイッチファイルの詳細については、例 1–1 を参照してください。

7. nsswitch.conf ファイルに何らかの変更を加えた場合 **(**または、新規にファイルにコピーした場合**)** 、必ず次の表のように入力して nscd を停止してから再起動する必要があります。

   ```
   client1# cp /etc/nsswitch.nisplus /etc/nsswitch.conf
   client1# sh /etc/init.d/nscd stop
   client1# sh /etc/init.d/nscd start
   ```

   キーサーバーをこの時点で終了および再起動する必要はありません。手順 11 で 行います。

8. 手順 **3** の情報を使用して、**Diffie-Hellman** キー長を設定します。

   たとえば、次のようにします。

   ```
   client# nisauthconf dh640-0 des
   ```

9. **NIS+** のファイルを削除しプロセスを終了します。

   現在作業しているマシンが、NIS+ のサーバーまたはクライアントとして以前に使用されていた場合、/var/nis 内にファイルがあればすべて削除し、キャッシュマネージャがまだ実行中であれば、そのプロセスを終了します。この例では、/var/nis 内にはコールドスタートファイルとディレクトリキャッシュファイルがまだ存在します。

   ```
   client1# ls /var/nis
   NIS_COLD_START NIS_SHARED_CACHE
   ```

```
client1# rm -rf /var/nis/*
client1# ps -ef | grep nis_cachemgr
 root 295 260 10 15:26:58 pts/0 0:00 grep nis_cachemgr
 root 286 1 57 15:21:55 ? 0:01 /usr/sbin/nis_cachemgr
client1# kill -9 286
```

/var/nis 内に残されたファイル、またはキャッシュマネージャによって保存されたディレクトリオブジェクトは、この手順によって完全に消去されます。したがって、この構成プロセスで生成された新しい情報と重複することはありません。/var/nis 内に管理スクリプトを格納していた場合、ルートドメインの設定が終わるまでは、これらを一時的に他のどこかに格納しておくことができます。

10. クライアントを初期設定します。

クライアントを初期設定するには、次の 3 つの方法があります。3 つの方法のうち、いずれかを選択して実行します。クライアントの初期設定が終了したら、手順 11 に進みます。

11. keyserv デーモンを終了してから再起動します。

この手順では、非公開鍵をキーサーバー上に格納します。

a. keyserv デーモンを終了します。

この手順によって、キーサーバーが保持していたクライアントに関するスイッチ情報も更新されます。

b. /etc/.rootkey ファイルを削除します。

c. キーサーバーを再起動します。

次の例では、手順 11 の内容を示しています。

```
client1# ps -e | grep keyserv
 root 145 1 67 16:34:44 ? keyserv
client1# kill 145
client1# rm -f /etc/.rootkey
client1# keyserv
```

d. keylogin -r を実行します。

この手順では、クライアントの非公開鍵をキーサーバーに格納します。また、クライアント上のスーパーユーザーが NIS+ を使用するために keylogin を行わなくてもすむように、/etc/.rootkey にコピーを保存します。-r オプションを付けて keylogin を実行します。パスワードの入力を求められたら、クライアントのスーパーユーザーパスワードを入力します。このパスワードは、クライアントの DES 資格を作成するために与えられたパスワードと同じでなければなりません。

```
client1# keylogin -r
Password:
Wrote secret key into /etc/.rootkey
```

e. クライアントを再起動します。

DNS 転送の設定

▼ NIS+ クライアントの DNS 転送機能を有効にするには

1. スーパーユーザーとしてログインします。
2. `/etc/resolve.conf` ファイルの `hosts` 行を構成します **(**たとえば、`hosts:nisplus dns files` とします**)**。

   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、この NIS 実装では DNS へ要求を転送するために、`ypstart` によって `ypserv` デーモンが `-d` オプション付きで「自動的に」起動されます。 DNS 転送を停止するには、`/usr/lib/netsvc/yp/ypstart` スクリプトを編集して、`ypserv` コマンドの `-d` オプションを削除します。そのあと、マシンをリブートしてください。

マシンのドメイン名を変更する

この作業では、マシンのドメイン名を変更します。マシンのドメイン名は通常インストール時に設定されるため、`domainname` コマンドを引数なしで実行して、マシンのドメイン名をチェックしてからこの作業を実行してください。

セキュリティについて

この作業は、ドメイン名を変更するマシン上のスーパーユーザーとして実行しなければなりません。

必要な情報

- マシンのスーパーユーザーのパスワード
- 新しいドメイン名

マシンのドメインの変更 - 作業マップ

表 **6–2** クライアントの構成

作業	説明	指示の参照先
マシンのドメインの変更	nisrestore を使用して、クライアントマシンのドメインを変更する	160 ページの「クライアントのドメイン名を変更する方法」

▼ クライアントのドメイン名を変更する方法

1. マシンにログインし、スーパーユーザーになります。

   この例では、マシンに client1 を、新しいドメイン名に doc.com. を使用します。

   ```
   client1% su
   Password:
   ```

2. マシンのドメイン名を変更します。

   domainname コマンドを使用して新しい名前を入力します。名前の最後にドットを入力しないでください。たとえば、マシンのドメインを doc.com に変更する場合、次のように入力します。

   ```
   client1# domainname doc.com
   ```

   マシンが NIS クライアントの場合は、NIS サービスを受けることはできません。

3. 結果を確認します。

   今度は、引数を付けずに domainname コマンドを実行し、サーバーの現在のドメインを表示させます。

   ```
   client1# domainname
   doc.com
   ```

4. 新しいドメイン名を保存します。

   domainname コマンドの出力を /etc/defaultdomain ファイルに書き込みます。

   ```
   client1# domainname> /etc/defaultdomain
   ```

5. 適当な時に、マシンを再起動します。

   /etc/defaultdomain ファイルに新しいドメイン名を入力した後でも、一部のプロセスは依然として古いドメイン名で動作している可能性があります。すべてのプロセスに新しいドメイン名を確実に使用させるため、マシンを再起動します。

   この作業は、他のいくつかの作業の流れの中で行うものです。リブートは、マシンでのすべての作業が完了したことを確認してから行なってください。確認を怠ると、何度もリブートが必要になる可能性があります。

   mountd などのデーモンを個別に再起動すれば、NFS の問題が解決されることがあります。ただし、構成の変更をデーモン間で同期化するために、マシンをリブートすることを強くお勧めします。マシンをリブートすることによって、構成の変更の非同期が原因で発生するアプリケーションの失敗を最小限に抑えることができます。

NIS+ クライアントを初期設定する

NIS+ クライアントを初期設定する方法には、以下の 3 つの種類があります。

- ブロードキャストを使用する方法 (161 ページの「ブロードキャストにより初期設定する」を参照)
- ホスト名を使用する方法 (162 ページの「ホスト名により NIS+ クライアントを初期設定する」を参照)
- コールドスタートファイルを使用する方法 (164 ページの「コールドスタートファイルを使用してクライアントを初期設定する」を参照)

ブロードキャストにより初期設定する

この方法では、クライアントの存在するサブネット上に IP ブロードキャストを送信して NIS+ クライアントを「初期化」します。

初期化の方法としてはこれが最も簡単ですが、最も安全性の低い方法でもあります。ブロードキャストに応答した NIS+ サーバーはクライアントが自分自身のコールドスタートファイルに格納する必要がある情報 (サーバーの公開鍵など) をすべて送信します。おそらくブローキャストに応答するのは NIS+ サーバーだけですが、クライアントからは、ブロードキャストに応答したワークステーションが確かに信用できるサーバーなのかどうか確認できません。そのため、この方法は小規模で、セキュリティが確保されたサイトでだけ使用することをお勧めします。

セキュリティについて

この作業は、クライアント上のスーパーユーザーとして実行しなければなりません。

前提条件

クライアントと同じサブネット上に、少なくとも 1 台の NIS+ サーバーが存在しなければなりません。クライアントは、マスターサーバーで使用するのと同じ Diffie-Hellman キー長を使用する必要があります。`nisauthconf(1M)` を参照してください。

必要な情報

クライアントのスーパーユーザーのパスワードが必要です。

NIS+ クライアントを初期設定する — 作業マップ

表 **6–3** NIS+ クライアントを初期設定する

作業	説明	指示の参照先
NIS+ クライアントを初期設定する	`nisclient` コマンドを使用して NIS+ クライアントを初期設定する	156 ページの「NIS+ クライアントを設定する方法 」

ブロードキャストによりクライアントを初期設定する方法

- クライアントを初期設定する

  この手順では、クライアントを初期設定し、その `/var/nis` ディレクトリ内に `NIS_COLD_START` ファイルを作成します。`nisinit` コマンドに `-c` と `-B` のオプションを付けて実行します。

```
client1# nisinit -c -B
This machine is in the doc.com. NIS+ domain.
Setting up NIS+ client ...
All done.
```

  同じサブネット上の NIS+ サーバーがブロードキャストに応答し、その位置情報をクライアントのコールドスタートファイルに追加します。

ホスト名により NIS+ クライアントを初期設定する

クライアントをホスト名によって初期化する場合、信頼できるサーバーの IP アドレスを明確に指摘します。そしてこのサーバー名、位置情報、公開鍵がクライアントのコールドスタートファイルに格納されます。

この方法は、クライアントがサーバーの IP アドレスを指定するので、自分で自分を識別してくるサーバーに応答するブロードキャストよりも安全です。しかし、クライアントと信頼できるサーバーの間にルーターが介在している場合、正しい IP アドレスへのメッセージを横取りし、不正なサーバーに送ることもあり得ます。

セキュリティについて

この作業は、クライアント上のスーパーユーザーとして実行しなければなりません。

前提条件

- NIS+ サービスはクライアントのドメインで実行されていなければなりません。

- クライアントは、/etc/hosts ファイルまたは /etc/inet/ipnodes ファイル内に信頼できるサーバーのエントリを持っていなければなりません。
- クライアントは、マスターサーバーで使用するのと同じ Diffie-Hellman キー長を使用する必要があります。nisauthconf(1M) を参照してください。

必要な情報

信頼できるサーバー名と IP アドレスが必要です。

NIS+ クライアントを初期設定する — 作業マップ

表 **6–4** NIS+ クライアントを初期設定する

作業	説明	指示の参照先
ホスト名によりクライアントを初期設定する	nisinit コマンドを使って、NIS+ クライアントをホスト名により初期設定する	163 ページの「ホスト名によりクライアントを初期設定する方法」

▼ ホスト名によりクライアントを初期設定する方法

1. クライアントの /etc/hosts ファイルまたは /etc/inet/ipnodes ファイルを確認します。

   クライアントが、信頼できるサーバーのエントリを持っていることを確認します。

2. クライアントを初期設定します。

   この手順では、クライアントを初期設定し、その /var/nis ディレクトリ内に NIS_COLD_START ファイルを作成します。nisinit コマンドに -c と -H のオプションを付けて実行します。次の例では、信頼できるサーバーとして rootmaster を使用します。

   ```
   Client1# nisinit -c -H rootmaster
   This machine is in the doc.com. NIS+ domain.
   Setting up NIS+ client ...
   All done.
   ```

   nisinit ユーティリティは、クライアントの /etc/hosts ファイルまたは /etc/inet/ipnodes ファイル内でサーバーのアドレスを探します。したがって、サーバーにドメイン名を付加しないでください。ドメイン名を付加した場合、このユーティリティはサーバーのアドレスを見つけることができません。

コールドスタートファイルを使用してクライアントを初期設定する

ここでは、NIS+ クライアントを初期設定するために、別の NIS+ クライアント (できれば同じドメインから) のコールドスタートファイルを使用します。NIS+ クライアントを設定する方法としてはこれが最も安全です。これにより、クライアントは、信頼できるサーバーから確実に NIS+ 情報を得ることができます。これはホスト名やブロードキャストによる初期化では保証されません。

セキュリティについて

この作業は、クライアント上のスーパーユーザーとして実行しなければなりません。

前提条件

コールドスタートファイルに指定されたサーバーは、すでに構成されており、NIS+ を実行していなければなりません。

クライアントは、マスターサーバーで使用するのと同じ Diffie-Hellman キー長を使用する必要があります。`nisauthconf(1M)` を参照してください。

必要な情報

コピーするコールドスタートファイルの名前と位置が必要です。

NIS+ クライアントを初期設定する — 作業マップ

表 **6–5** NIS+ クライアントを初期設定する

作業	説明	指示の参照先
コールドスタートファイル経由でクライアントを初期設定する	`nisinit` コマンドを使って、NIS+ クライアントをコールドスタートファイル経由で初期設定する	164 ページの「コールドスタートファイルを使用して NIS+ クライアントを初期設定する方法」

▼ コールドスタートファイルを使用して NIS+ クライアントを初期設定する方法

1. 他のクライアントのコールドスタートファイルをコピーします。

   他のクライアントのコールドスタートファイルを、新しいクライアントのディレクトリにコピーします。これを行うには、クライアント上のスーパーユーザーとしてではなく、自分のユーザー名でログインしている間に行う方が簡単です。クライアントを

初期設定する前に、必ずスーパーユーザーになってください。

ただし、NIS_COLD_START ファイルを /var/nis にコピーしないでください。初期設定中にこのファイルは上書きされます。次の例では、client1 のコールドスタートファイルを、初期設定されていない client2 の /tmp ディレクトリにコピーします。

```
client2# exit
client2% rcp client1:/var/nis/NIS_COLD_START /tmp
client2% su
```

2. コールドスタートファイルからクライアントを初期設定します。

次に示すように、nisinit コマンドに -c と -C のオプションを付けて実行します。

```
client2# nisinit -c  -C /tmp/NIS_COLD_START
This machine is in the doc.com. NIS+ domain.
Setting up NIS+ client ...
All done.
```

NIS+ クライアント構成の要覧

クライアントの構成に必要な手順を表 6–6 にまとめます。クライアントは doc.com ドメインにある client1 とします 。これは最も簡単なケースを想定しているため、このまとめを参考用として使用するには、その前に自分の実際の作業の詳細を理解することが必要です。簡略化のため、ここでは各コマンドに対するサーバーの応答を示していません。

表 **6–6** クライアントを設定する方法のまとめ

作業	コマンド
ドメインのマスターサーバーにログインする	rootmaster%
クライアントの DES 資格を作成する	rootmaster% nisaddcred -p unix.client1.doc.com -P client1.doc.com. des
Diffie-Hellman キー長を確認する	rootmaster% nisauthconf
クライアントにスーパーユーザーとしてログインする	client1% su Password: root パスワードを入力する
クライアントのドメイン名を設定する	client1# domainname doc.com client1# domainname> /etc/defaultdomain

表 **6–6** クライアントを設定する方法のまとめ　(続き)

作業	コマンド
クライアントのスイッチ構成ファイルが正しく設定されていることをチェックする	`client1# more /etc/nsswitch.conf`
Diffie-Hellman キー長を設定する	`client1# nisauthconf dh640-0 des`
`/var/nis` の下のファイルを削除する	`client1# rm -rf /var/nis/*`
クライアントを初期設定する	`client1# nisinit -c -H rootmaster`

キーサーバーのプロセスを終了して、再起動する	`client1# ps -ef | grep keyserv`
	`client1# kill -9` *process-id*
	`client1# keyserv`

クライアント上でキーログインを実行する	`client1# keylogin -r password:`
クライアントを再起動する	`client1# init 6`

第 7 章

NIS+ サーバーの構成

この章では、NIS+ コマンドセットを使って NIS+ サーバーを設定する手順と、既存の NIS+ ドメインに複製サーバーを追加する手順を説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ サーバーを初期設定する

NIS+ サーバーの初期設定は、この章で説明する NIS+ コマンドよりも、NIS+ スクリプトを使用した方が簡単に行うことができます。この章で説明する方法は、NIS+ に精通した管理者や、設定スクリプトでは提供されない標準以外の機能や構成を必要とする管理者だけが使用してください。

標準構成と NIS 互換構成の手順の相違

NIS 互換の NIS+ サーバーと標準の NIS+ サーバーの設定における違いは、ルートマスターサーバーの場合と同じです (136 ページの「標準構成と NIS 互換構成の手順の相違」を参照)。NIS 互換サーバー用の NIS+ デーモンは `-Y` オプション (DNS 転送を使用する場合は、`-B` オプションを追加する) を使用して起動しなければなりません。これによって、サーバーは NIS クライアントからの要求に応答できます。これについては、「NIS+ サーバーを構成する方法」の手順 2 (標準の NIS+ サーバーの場合は、手順 3) で説明します。

注 – -Y または -B のいずれかのオプションを使用して rpc.nisd を起動した場合、必ず rpc.nisd_resolv という副デーモンが生成され、名前の解決を行います。この副デーモンは、rpc.nisd 主デーモンを終了させた場合は、必ず別個に終了させなければなりません。

設定作業の手順を次にまとめます。

1. 新しくサーバーにするワークステーションにスーパーユーザーとしてログインします。
2. NIS+ デーモンを -Y で起動します (NIS 互換の場合のみ)。
3. NIS+ デーモンを起動します (標準の NIS+ の場合のみ)。

セキュリティについて

注 – NIS+ のセキュリティシステムは複雑です。NIS+ セキュリティを使い慣れていない場合は、第 11 章を参照してから NIS+ 環境を構成することをお勧めします。

この手順は、サーバー上のスーパーユーザーとして実行しなければなりません。起動したサーバーのセキュリティレベルによって、そのクライアントが備えるべき資格が決まります。たとえば、サーバーがセキュリティレベル 2 で構成された場合、サーバーがサポートするドメイン内のクライアントは、DES 資格を必要とします。このマニュアルの指示に従ってクライアントを構成した場合、そのクライアントは適切なドメインに DES 資格を持ち、セキュリティレベル 2 でサーバーを起動できます。

注 – セキュリティレベル 0 は、管理者による構成とテストの目的だけに使用します。セキュリティレベル 1 はサポートされていません。一般のユーザーが通常の業務を行う環境では、レベル 0 またはレベル 1 を使用せず、常にセキュリティレベル 2 を使用してください。

前提条件

- ルートドメインがあらかじめ構成されている (第 5 章を参照)
- サーバーにするには、NIS+ クライアントとして初期設定しておく (第 6 章を参照)
- サーバーを構成するには、そのマシンにスーパーユーザーとしてログインする必要がある

- サーバーが NIS 互換モードで稼動し、DNS 転送をサポートするためには、正しく構成された /etc/resolv.conf ファイルが必要である (『*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「DNS の管理 (参照情報)」を参照)

必要な情報

サーバーに変換するクライアントのスーパーユーザーパスワードが必要です。

▼ NIS+ サーバーを構成する方法

1 つのマスターサーバーまたは複製サーバーから複数のドメインにサービスを提供することは可能ですが、あまりお勧めしません。

1. 新しい複製サーバーにスーパーユーザーとしてログインします。

   以下の手順では、154 ページの「クライアントの構成」に従って、マシンを NIS+ クライアントとして設定した後、マシンを再起動したことを前提としています。マシンを再起動すると、次の手順の推奨前提条件であるキャッシュマネージャが起動します。マシンを再起動しなかった場合、nis_cachemgr を使用して、ここでキャッシュマネージャを再起動します。

2. **NIS+** デーモンを -Y で起動します **(NIS** 互換の場合のみ**)**。

   この手順は、サーバーを NIS 互換モードで設定する場合にだけ実行します。標準の NIS+ サーバーを設定する場合は、この代わりに 手順 3 を実行します。この手順には、NIS クライアントの DNS 転送機能をサポートするための操作説明も含まれています。

   この手順は、2 つに分かれています。最初の手順では、NIS+ デーモンを NIS 互換モードで起動します。2 つめの手順では、サーバーが再起動されたときに、NIS+ デーモンが NIS 互換モードで再起動するように設定します。

   a. rpc.nisd に -Y と -B のフラグを付けて実行します。

      ```
      compatserver# rpc.nisd -Y -B
      ```

      -Y オプションを指定すると、NIS+ 要求だけでなく NIS 要求にも応答します。-B オプションは DNS 転送をサポートします。

   b. /etc/init.d/rpc ファイルを編集します。

      /etc/init.d/rpc ファイル内で文字列 EMULYP="-Y" を検索して、この文字列を含む行のコメント指定を解除します。

      DNS 転送機能を使用するには、EMULYP="-Y" に -B フラグを追加します。DNS 転送機能が必要ない場合は、コメント指定の解除だけを実行し、-B フラグは追加しないでください。

      この手順によって、/var/nis/data という名前のディレクトリが作成されます。また、trans.log というトランザクションログファイルが作成され、/var/nis というディレクトリに格納されます。

```
compatserver# ls -F /var/nis
NIS_COLD_START data/ trans.log data.dict
```

trans.log ファイルは、トランザクションログです。トランザクションログの内容を確認するには、nislog コマンドを使用します。使用方法については、350 ページの「nislog コマンド」を参照してください。

注意 – /var/nis ディレクトリと /var/nis/data ディレクトリは、移動または名前の変更をしないでください。また、/var/nis/trans.log ファイルと /var/nis/data.dict ファイルについても、移動または名前の変更をしないでください。Solaris 2.4 以前からアップグレードする場合、それまで使っていた /hostname サブディレクトリは自動的に /var/nis/data に変換され、関連するファイルも必要に応じて変換されます。この自動変換がなされた後で、新しい名前に変更することは絶対にしないでください。

これでこのサーバーは、第 8 章の説明に従って、ドメインのマスターまたは複製に指定できます。NIS+ サーバーの設定は、この手順で完了です。作業の要約については 178 ページの「サーバー構成の要覧」を参照してください。

3. **NIS+ デーモンを起動します (標準の NIS+ の場合のみ)。**

   rpc.nisd コマンドを実行します。

   ```
   server# rpc.nisd
   ```

   NIS+ デーモンが本当に実行されていることを確認するには、次のように ps コマンドを実行します。

   ```
   server# ps -ef | grep rpc.nisd
   root 1081 1 16:43:33 ? 0:01 rpc.nisd
   root 1087 1004 11 16:44:09 pts/1 0:00 grep rpc.nisd
   ```

   この手順によって、/var/nis/data という名前のディレクトリが作成されます。また、trans.log というトランザクションログファイルが作成され、/var/nis ディレクトリに格納されます。

   ```
   compatserver# ls -F /var/nis
   NIS_COLD_START data/ trans.log data.dict
   ```

   compatserver.log ファイルは、トランザクションログです。トランザクションログの内容を確認するには、nislog コマンドを使用します。使用方法については、第 18 章を参照してください。

注意 – /var/nis ディレクトリと /var/nis/data ディレクトリは、移動または名前の変更をしないでください。また、/var/nis/trans.log ファイルと /var/nis/data.dict ファイルについても、移動または名前の変更をしないでください。Solaris 2.4 以前からアップグレードする場合、それまで使っていた /hostname サブディレクトリは自動的に /var/nis/data に変換され、関連するファイルも必要に応じて変換されます。この自動変換がなされた後で、新しい名前に変更することは絶対にしないでください。

これでこのサーバーは、第 8 章の説明に従って、ドメインのマスターまたは複製に指定できます。NIS+ サーバーの設定は、この手順で完了です。作業の要約については 178 ページの「サーバー構成の要覧」を参照してください。

既存のドメインに複製サーバーを追加する

NIS+ サービスを常に利用できる状態にしておきたいのであれば、ルート複製サーバーを少なくとも 1 つは作成しておくことをお勧めします。複製サーバーを作成すると複数のサーバーが存在することになり、要求の処理を分散させることができるため、ネットワーク要求の処理も高速化されます。

パフォーマンス上の理由から、1 つのドメインに多くの複製サーバーを置くことはお勧めできません。ネットワークが複数のサブネットで構成されている場合、あるいは広域ネットワーク (WAN) でリモートサイトに接続されている場合にだけ、複製サーバーを置くようにしてください。

- 「サブネット」複数のサブネットで構成されているドメインの場合、各サブネットに複製サーバーを少なくとも 1 つは作成することをお勧めします。そうしておけば、ネットワーク間の通信が一時的に途絶していても、接続が回復するまでの間、サブネットレベルの機能は維持されるからです。
- 「リモートサイト」。WAN によりリモートサイトに接続されているドメインの場合、WAN 接続の両側に複製サーバーを少なくとも 1 つは作成することをお勧めします。組織論的な見地からしても、同一の NIS+ ドメインに物理的に離れた 2 つのサイトがあるのは意味のあることです。たとえば、ドメイン内のマスターサーバーとその複製サーバーがすべて一方のサイトに置かれている場合、そのサイトともう一方のサイトとの間の NIS+ ネットワークトラフィックが増大するのは目に見えています。もう一方のサイトにも複製サーバーを置いておけば、ネットワークトラフィックが減るはずです。

複製サーバーの分散および最適な複製サーバー数については、616 ページの「サーバーの必要条件を決める」 を参照してください。既存のドメインに複製サーバーを追加するには、その複製サーバーを構成してから該当する名前空間の NIS+ データセットをロードします。

新しい複製サーバーを構成して NIS+ データセットをロードする方法には、次の 2 通りがあります。

- 「スクリプト」。`nisserver` スクリプトを実行するには、116 ページの「ルート複製サーバーの作成」の説明に従ってください。この方法では、NIS+ データセットが新しい複製サーバーにロードされて自動的に再同期がとられます。格段に簡単なので、こちらの方法をお勧めしますが、「NIS+ コマンドセット」と「バックアップと復元」を利用する方法に比べると、時間が長くかかることがあります。
- 「NIS+ コマンドセット」。NIS+ コマンドを使うには、172 ページの「NIS+ コマンドを使って複製サーバーを構成する」の説明に従ってください。`nisserver` スクリプトを実行する方法に比べると、NIS+ に対する深い知識が必要です。この NIS+ コマンドを使う方法には、きめの細かい設定と監視が可能であるという利点があります。そして、もう 1 つ、ドメインディレクトリを手作業で作成して複製サーバーを生成し、`nisbackup` と `nisrestore` を使って NIS+ データをロードできるという利点もあります。`nisbackup` と `nisrestore` を使うと、`nisserver` スクリプトを使うより、短時間でデータセットをロードできます。

  新たに構成した複製サーバーに NIS+ データセットをロードする方法には、次の 2 通りがあります。

  - 「`nisping`」`nisserver` スクリプトと NIS+ コマンドセットのどちらを使った場合でも、新しい複製サーバーの構成が完了すると、`nisping` を使用することで、該当する名前空間のデータセットがマスターサーバーにより、ネットワーク経由で自動的に新しい複製サーバーにロードされます。このとき、大きな名前空間では処理に長時間かかり、その間、名前管理情報の要求が遅れることがあります。詳細は、176 ページの「`nisping` を使ってデータを複製サーバーにロードする」を参照してください。
  - 「バックアップと復元」`nisping` によるデータ転送に割り込みをかけ、NIS+ のバックアップ機能と復元機能を使って、名前空間データを新たに構成した複製サーバーにロードできます (174 ページの「`nisrestore` を使ってデータを複製サーバーにロードする」を参照)。複製サーバーから複製サーバーにデータセットがロードされることになり、マスターサーバーから複製サーバーにネットワーク経由でデータセットをロードする場合に比べて格段に早く終わるので、こちらの方法をお勧めします。

NIS+ コマンドを使って複製サーバーを構成する

この節では、NIS+ コマンドを使って複製サーバーを既存のドメインに追加する方法について説明します。

セキュリティについて

この作業を実行する NIS+ 主体には、ドメインのディレクトリオブジェクトに対する変更権が必要です。

前提条件

- ドメインをあらかじめ構成し、マスターサーバーを稼動させておく
- 新しい複製サーバーが NIS+ サーバーとして構成されている (167 ページの「NIS+ サーバーを初期設定する」を参照)

必要な情報

- サーバー名
- ドメイン名

NIS+ コマンドを使って複製サーバーを構成する—作業マップ

表 **7–1** NIS+ コマンドを使って複製サーバーを構成する

作業	説明	指示の参照先
NIS+ サーバーを設定する	NIS+ コマンドを使って複製サーバーを設定する	173 ページの「NIS+ コマンドを使って複製サーバーを構成する方法」

▼ NIS+ コマンドを使って複製サーバーを構成する方法

この例では、マスターサーバー名を `master1`、新しい複製サーバー名を `replica2` とします。

1. ドメインのマスターサーバーにログインします。
2. `rpc.nisd` が稼働中であることを確認します。
3. ドメインに複製サーバーを追加します。

   `nismkdir` コマンドに `-s` オプションを付けて実行します。次の例では、`doc.com.` ドメインに `replica2` という名前の複製サーバーマシンを追加します。

```
master1# nismkdir -s replica2 doc.com.
master1# nismkdir -s replica2 org_dir.doc.com.
master1# nismkdir -s replica2 groups_dir.doc.com.
```

すでに存在するディレクトリオブジェクトに nismkdir コマンドを実行すると、ディレクトリは再作成されずに、与えられたフラグに基づいてディレクトリが変更されます。この場合、-s フラグはドメインに追加する複製サーバーを割り当てます。複製サーバーが追加されたことを確認するには、niscat -o コマンドを実行して、ディレクトリオブジェクトの定義を調べます。

注意 – nismkdir コマンドは必ずマスターサーバー上で実行してください。複製サーバー上で nismkdir コマンドを実行すると、マスターサーバーと複製サーバーとの間で通信上の問題が生じます。

これで新しい複製サーバーの構成は完了です。次は、構成した複製サーバーに NIS+ データセットをロードします。NIS+ データセットのロードには、2 通りの方法があります。

- 「nisping」。何もしなければ、マスターサーバーによって nisping コマンドが実行され、該当する名前空間データが新たに構成された複製サーバーにロードされます。名前空間が大きい場合は、データのロードに時間がかかることがあります。データのロード中は、ネーミング情報の要求は遅延することがあります。詳細は、176 ページの「nisping を使ってデータを複製サーバーにロードする」を参照してください。
- 「バックアップと復元」。nisping によるデータ転送に割り込みをかけ、NIS+ のバックアップ機能と復元機能を使って、名前空間データを新たに構成した複製サーバーにロードできます (174 ページの「nisrestore を使ってデータを複製サーバーにロードする」を参照)。他の方法に比べて格段に早く効率的なので、こちらの方法をお勧めします。

nisrestore を使ってデータを複製サーバーにロードする

この節では、NIS+ のバックアップ機能と復元機能を使って名前空間データを新しい複製サーバーにロードする方法について説明します。この方法を使ってデータを複製サーバーにロードすることをお勧めします。

セキュリティについて

この作業を実行する NIS+ 主体には、ドメインのディレクトリオブジェクトに対する変更権が必要です。

前提条件

- ドメインをあらかじめ構成し、マスターサーバーを稼動させておく

- 新しい複製サーバーが NIS+ サーバーとして構成されている (167 ページの「NIS+ サーバーを初期設定する」を参照)
- 新しい複製サーバーが複製サーバーとして構成されている (172 ページの「NIS+ コマンドを使って複製サーバーを構成する」を参照)

nisrestore を使ってデータを複製サーバーにロードする — 作業マップ

表 **7–2** nisrestore を使ってデータを複製サーバーにロードする

作業	説明	指示の参照先
nisrestore を使ってデータを複製サーバーにロードする	nisrestore を使って名前空間データをロードする	175 ページの「名前空間データをロードする方法—nisrestore による方法」

▼ 名前空間データをロードする方法—nisrestore による方法

この例では、マスターサーバー名を master1、新しい複製サーバー名を replica2 とします。

1. 複製サーバー上の rpc.nisd を終了させます。

   マスターサーバーから複製サーバーへの名前空間データの自動ロード (nisping による) が中断されます。

2. マスターサーバー上で **NIS+** バックアップ機能を実行します。

   この手順の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』を参照してください。以下の例では、nisbackup コマンドを使って master1 を /var/master1_bakup ディレクトリにバックアップします。

   ```
   master1# nisbackup -a /var/master1_bakup
   ```

   nisrestore を使って新しい複製サーバーを構成する最も簡単な方法は、マスターサーバーのデータを NFS にマウントされた (複製サーバーからアクセス可能な) ディレクトリにバックアップするというものです。この例では、マスターサーバーと新しい複製サーバーの両方に、/var/master1_bakup ディレクトリへのアクセス権が与えられているものと想定します。

   このほかに、tar コマンドを使って /var/master1_bakup ディレクトリからテープカートリッジなどの可搬記憶メディアにデータをコピーし、次に、その可搬記憶メディアから新しい複製サーバーにマウントされているディレクトリにデータをコピーしてから、そのディレクトリを nisrestore コマンドの情報源として使うという方法 (手順 3 を参照) もあります。

3. nisrestore コマンドを使って、**NIS+** データセットを新しい複製サーバーにロードします。

   この手順の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』を参照してください。以下の例では、nisrestore コマンドを使って NIS+ データを /var/master1_bakup ディレクトリから client2 にダウンロードします。

   ```
   replica2# nisrestore -a /var/master1_bakup
   ```

   作成している複製サーバーがルートドメインで使うものである場合、あるいは nisrestore が必要なデータを検証または見つけることができないという旨のエラーメッセージが出た場合は、次に示すように -f オプション付きで実行してみてください。

   ```
   replica2# nisrestore -f -a /var/master1_bakup
   ```

4. 新しい複製サーバー上で rpc.nisd を再実行します。

   169 ページの「NIS+ サーバーを構成する方法」 を参照してください。

nisping を使ってデータを複製サーバーにロードする

この節では、nisping コマンドを使って名前空間データを新しい複製サーバーにロードする方法について説明します。通常、このプロセスは自動的に実行されるため、nisping コマンドを実行する必要はまずありません。

nisping コマンドを使う方法の問題点は、マスターサーバーから複製サーバーへデータの再同期をとるために、NIS+ プロトコルを使ったネットワーク上のデータのやりとりが必要だということです。名前空間が大きい場合は、この処理に何時間もかかり、その間、名前管理情報の要求に対する応答が遅れることがあります。

セキュリティについて

この作業を実行する NIS+ 主体には、ドメインのディレクトリオブジェクトに対する変更権が必要です。

前提条件

- ドメインをあらかじめ構成し、マスターサーバーを稼動させておく
- 新しい複製サーバーが NIS+ サーバーとして構成されている (167 ページの「NIS+ サーバーを初期設定する」を参照)
- 新しい複製サーバーが複製サーバーとして構成されている (172 ページの「NIS+ コマンドを使って複製サーバーを構成する」を参照)

nisping を使ってデータを複製サーバーにロードする — 作業マップ

表 **7–3** nisping を使ってデータを複製サーバーにロードする

作業	説明	指示の参照先
nisping を使ってデータを複製サーバーにロードする	nisping を使ってデータを複製サーバーにロードする	177 ページの「名前空間データをロードする方法—nisping による方法」

▼ 名前空間データをロードする方法—nisping による方法

通常、名前空間データのロードは、マスターサーバーによって自動的に開始されます。マスターサーバーによるロードが行われなかった場合は、次の説明に従って nisping コマンドを実行してください。

- ディレクトリに対して nisping を実行します。

  この手順では、新しい複製サーバーにメッセージ「ping」を送信して、マスターサーバーに対して更新を要求するように通知します。複製サーバーがルートドメインに所属していない場合、必ずドメイン名を指定してください。次の例では、ドメイン名は完全を期すためにだけ記述してあります。この作業で使用する例は、ルートドメインに複製サーバーを追加しているため、次の例にあるドメイン名 doc.com. は必要ありません。

```
master1# nisping doc.com.
master1# nisping org_dir.doc.com.
master1# nisping groups_dir.doc.com.
```

  次のような画面が表示されます。

```
master1# nisping doc.com.
Pinging replicas serving directory doc.com. :
Master server is master1.doc.com.
 No last update time
Replica server is replica1.doc.com.
 Last update seen was Wed Nov 18 11:24:32 1992
 Pinging ... replica2.doc.com.
```

  大きな名前空間の場合、この処理に何時間もかかる場合があります。nisping の詳細については、第 18 章 を参照してください。

サーバー構成の要覧

表 7–4、表 7–5 では、この章で説明した作業のまとめを示しています。この 2 つの表は、最も簡単な場合を想定しているため、参考用として使用するには、実際の自分の作業の詳細を理解している必要があります。また、ここでは、各コマンドに対するサーバーの応答を示していません。

表 **7–4** 複製サーバー `replica2` を `doc.com.` に追加する

作業	コマンド
ドメインマスターサーバーにスーパーユーザーとしてログインする	`master1% su`

新しい複製サーバーを指定する	`# nismkdir -s replica2 doc.com.`
	`# nismkdir -s replica2 org_dir.doc.com.`
	`# nismkdir -s replica2 groups_dir.doc.com.`

複製サーバーに対して `nisping` を実行する	`# /usr/lib/nis/nisping doc.com.`
	`# /usr/lib/nis/nisping org_dir.doc.com.`
	`# /usr/lib/nis/nisping groups_dir.doc.com.`

注 – 上記の例で説明したように、新しい複製サーバーにデーターをロードする場合は、`nisping` を使用するより NIS+ のバックアップと復元機能を使用した方が簡単です。詳細については、174 ページの「`nisrestore` を使ってデータを複製サーバーにロードする」 を参照してください。

表 **7–5** 非ルートマスターサーバーを起動するには

作業	コマンド
root としてサーバーにログインする	`server% su`
NIS 互換モードの場合のみ: `-Y -B` フラグを使用してデーモンを起動する	`server# rpc.nisd -Y - B`
NIS 互換モードの場合のみ: `EMULYP= -Y -B` に変更する	`server# vi /etc/inet.d/rpc`
NIS+ モードの場合のみ: デーモンを起動する	`server# rpc.nisd`

第 8 章

非ルートドメインの構成

この章では、NIS+ コマンドセットを使ってサブドメイン (非ルートドメイン) を構成する方法 (マスターサーバーと複製サーバーを設定する方法を含む) を、手順を追って説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については
、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

非ルートドメインを設定する

注 – NIS+ クライアントを設定する作業は、この章で説明する NIS+ のコマンドセットを使用する方法よりも、パート I で説明した NIS+ 設定スクリプトを使用する方が簡単です。この章で説明する方法は、NIS+ に精通した管理者や、設定スクリプトでは提供されない標準以外の機能や構成を必要とする管理者だけが使用してください。

最初に非ルートドメインのサーバーを構成してから、非ルートドメインを構成してください。

非ルートドメインを設定するには、次の作業を行います。

- ドメインのセキュリティの設定
- ドメインのディレクトリの作成
- ドメインのテーブルの作成

- ドメインのサーバーの指定

ルートドメインの構成と同様に、これらの作業は連続して実行できません。構成プロセスを簡単にするため、これらを個々の手順に分割して、最も効率的な順序に並べています。

標準構成と NIS 互換構成の手順の相違

サブドメインにおける NIS 互換の NIS+ サーバーと標準の NIS+ サーバーとの違いは、ルートドメインの場合と同じです (136 ページの「標準構成と NIS 互換構成の手順の相違」参照)。

NIS 互換ドメインの各サーバーの NIS+ デーモンは、第 7 章の説明に従って、`-Y` オプションを使用して起動する必要があります。また、NIS 互換ドメインでは、ドメインのテーブルによって未認証クラスに読み取り権を提供する必要があります。これにより、NIS クライアントはテーブルに格納されている情報にアクセスできます。手順 4 で説明するとおり、`nissetup` コマンドに `-Y` オプションを追加すると、テーブル内の情報にアクセスできます。標準の NIS+ ドメインでも同じコマンドを使用しますが、`-Y` オプションは使用しません。これについても手順 4 で説明します。

設定作業の手順を次にまとめます。

1. ドメインのマスターサーバーにログインします。
2. ドメインの管理グループを指定します。
3. ドメインのディレクトリを作成し、そのサーバーを指定する
4. ドメインのサブディレクトリとテーブルを作成します。
5. ドメインの管理グループを作成します。
6. ディレクトリオブジェクトに完全なグループアクセス権を割り当てます。
7. ドメインの管理グループにサーバーを追加します。
8. 他のシステム管理者の資格を追加します。
9. ドメインの管理グループに管理者を追加します。

セキュリティ上の留意点

注 – NIS+ のセキュリティシステムは複雑です。NIS+ セキュリティを使い慣れていない場合は、第 17 章を参照してから NIS+ 環境を構成することをお勧めします。

多くのサイトでは、親ドメインのセキュリティを確保するために、その下にドメインを作成できるのは、親ドメインのマスターサーバー、または親ドメインの管理グループに所属するシステム管理者に限定しています。これは、管理方針であり NIS+ の必要条件ではありませんが、この章の操作説明ではこの作業を行う管理者がこの方針に従っているものと仮定します。もちろん、親ドメインの管理グループには、親ディレクトリオブジェクトに対する作成権が必要です。これを確認するには、`niscat -o` コマンドを実行します。

```
rootmaster# niscat -o doc.com.
Object Name : Doc
Owner : rootmaster
Group : admin.doc.com.
Domain : Com.
Access Rights : r---rmcdrmcdr---
:
```

安全性よりも便宜性を重視する場合、新しいドメインのマスターサーバーをその親ドメインの管理グループのメンバーとし、そのサーバーからすべて手順を実行できます。この場合、第 17 章で説明する `nisgrpadm` コマンドを使用します。

前提条件

- 親ドメインを構成し、実行していなければなりません。
- このドメインのマスターとして指定されるサーバーは、すでに初期設定され、NIS + を実行していなければなりません。
- 複製サーバーを指定する場合、マスターサーバーは、その `/etc/hosts` ファイル、`/etc/inet/ipnodes` ファイルまたはその NIS+ `hosts` テーブルを通じて、複製サーバーの IP アドレスを入手できなければなりません。

必要な情報

- 新しいドメインの名前 (手順 3)
- 新しいドメインのマスターサーバーと複製サーバー名
- 新しいドメインの管理グループ名 (手順 2)
- 新しいドメインの管理グループに所属する管理者のユーザー ID (UID) (手順 8)

非ルートドメインを設定する — 作業マップ

表 8–1 非ルートドメインを設定する

作業	説明	指示の参照先
非ルートドメインを設定する	NIS+ コマンドを使って非ルートドメインを設定する	181 ページの「非ルートドメインを設定する方法」

▼ 非ルートドメインを設定する方法

1. ドメインのマスターサーバーにログインします。

   新しいドメインのマスターにする予定のサーバーにログインします。この作業の手順では `smaster` という名前のサーバーを使用します。このサーバーは `doc.com.` ドメインに所属し、`sales.doc.com.` サブドメインのマスターサーバーになります。こ

の作業を実行する管理者は、admin.doc.com. グループのメンバーである nisboss.doc.com. です。このグループには、 doc.com. ディレクトリオブジェクトに対するフルアクセス権が割り当てられています。

2. ドメインの管理グループを指定します。

実際に管理グループを作成するのは、手順 5 の時点でですが、ここで管理グループを指定する必要があります。これによって、次の手順で使用される nismkdir コマンドは、このグループに対する適切なアクセス権をもつディレクトリオブジェクトを作成できます。またこれは、手順 4 で使用する nissetup ユーティリティに対しても同じことを行います。

環境変数 NIS_GROUP に、ドメインの管理グループ名を設定します。ここでは、C シェルユーザーの場合と Bourne シェルまたは Korn シェルユーザーの場合の 2 つの例を示します。いずれも NIS_GROUP に admin.sales.doc.com. を設定します。

「C シェルの場合」

```
smaster# setenv NIS_GROUP admin.sales.doc.com.
```

「Bourne シェルまたは Korn シェルの場合」

```
smaster# NIS_GROUP=admin.sales.doc.com.
smaster# export NIS_GROUP
```

3. ドメインのディレクトリを作成し、そのサーバーを指定する

nismkdir コマンドは、新しいドメインのディレクトリ作成と、そのサポートサーバーの指定を 1 つの手順で行います。この構文を次に示します。

```
nismkdir -m master -s replica domain
```

-m フラグはマスターサーバーを指定し、-s フラグは複製サーバーを指定します。この例を次に示します。

```
smaster# nismkdir -m smaster -s salesreplica sales.doc.com.
```

注意 – nismkdir は必ず (複製サーバーではなく) マスターサーバー上で実行してください。複製サーバーで実行すると、マスターサーバーと複製サーバーの間で通信上の問題が発生します。

ディレクトリオブジェクトは /var/nis にロードされます。内容を表示するには、niscat -o コマンドを実行します 。cat または more は使用しないでください。

```
smaster# niscat -o sales.doc.com.
Object Name : Sales
Owner : nisboss.doc.com.
Group : admin.sales.doc.com.
Domain : doc.com.
Access Rights : ----rmcdr---r---
.
```

ルートディレクトリとは異なり、このディレクトリオブジェクトには適切なグループが割り当てられています。したがって、nischgrp を実行する必要はありません。

4. ドメインのサブディレクトリとテーブルを作成します。

   この手順では、org_dir ディレクトリと groups_dir ディレクトリ、および NIS+ テーブルを新しいディレクトリオブジェクトの下に追加します。nissetup ユーティリティを使用しますが、新しいドメイン名の追加を忘れないでください。NIS 互換ドメインの場合、-Y フラグを指定します。

   「NIS 互換の場合」

```
smaster# /usr/lib/nis/nissetup -Y sales.doc.com.
```

   「標準 NIS+ の場合」

```
smaster# /usr/lib/nis/nissetup sales.doc.com.
```

   このユーティリティによって追加されたオブジェクトを表示すると、次のようになります。

```
smaster# /usr/lib/nis/nissetup
org_dir.sales.doc.com. created
groups_dir.sales.doc.com. created
auto_master.org_dir.sales.doc.com. created
auto_home.org.dir.sales.doc.com. created
bootparams.org_dir.sales.doc.com. created
cred.org_dir.sales.doc.com. created
ethers.org_dir.sales.doc.com. created
group.org_dir.sales.doc.com. created
hosts.org_dir.sales.doc.com. created
mail_aliases.org_dir.sales.doc.com. created
sendmailvars.org_dir.sales.doc.com. created
netmasks.org_dir.sales.doc.com. created
netgroup.org_dir.sales.doc.com. created
networks.org_dir.sales.doc.com. created
passwd.org_dir.sales.doc.com. created
protocols.org_dir.sales.doc.com. created
rpc.org_dir.sales.doc.com. created
services.org_dir.sales.doc.com. created
timezone.org_dir.sales.doc.com. created
```

   -Y オプションによって、標準の NIS+ ドメインの場合と同じテーブルとサブディレクトリが作成されますが、NIS クライアントからの要求が NIS+ テーブル内の情報にアクセスできるよう、nobody クラスに読み取り権が割り当てられます。

   /var/nis/salesmaster に相当する自分のマスターを調べることによって、org_dir ディレクトリと groups_dir ディレクトリが存在することを確認できます。これらのディレクトリは、ルートオブジェクトおよびその他の NIS+ ファイルと共に登録されています。テーブルは org_dir ディレクトリに存在します。任意のテーブルの内容を調べるには、第 9 章で説明する niscat コマンドを実行します 。ただしこの時点ではテーブルは空です。

5. ドメインの管理グループを作成します。

   この手順では、手順 2 で指定した管理グループを作成します。nisgrpadm コマンドに -c オプションを付けて実行してください。次の例では admin.sales.doc.com. グループを作成します。

```
smaster# nisgrpadm -c admin.sales.doc.com.
 Group admin.sales.doc.com. created.
```

この手順ではグループを作成するだけであり、そのメンバー名の指定は行いません。指定は 手順 9 で行います。

6. ディレクトリオブジェクトに完全なグループアクセス権を割り当てます。

   デフォルトでは、ディレクトリオブジェクトはそのグループに読み取り権を与えるだけであり、これではその他のカテゴリと同様、グループも使うことができません。クライアントとサブドメインの構成を簡単にするため、ディレクトリオブジェクトがそのグループに与えるアクセス権を、読み取り権のみから読み取り権、変更権、作成権、削除権に変更します。次に示すように、nischmod コマンドを実行します。

   ```
   smaster# nischmod g+rmcd sales.doc.com.
   ```

   nischmod コマンドの使用方法の詳細については、第 15 章を参照してください。

7. ドメインの管理グループにサーバーを追加します。

   この時点で、このドメインのグループにはメンバーがありません。-a オプションを付けて nisgrpadm コマンドを実行し、マスターサーバーと複製サーバーを追加します。最初の引数はグループ名であり、残りの引数は新しいメンバーの名前です。この例では、smaster.doc.com. とsalesreplica.doc.com. を admin.sales.doc.com. グループに追加します。

   ```
   smaster# nisgrpadm -a admin.sales.doc.com. smaster.doc.com.
   salesreplica.doc.com.
   Added smaster.doc.com. to group admin.sales.doc.com.
   Added salesreplica.doc.com. to group admin.sales.doc.com.
   ```

   このサーバーが管理グループに属していることを確認するには、nisgrpadm コマンドを -l オプションで実行します。手順については、第 17 章を参照してください。

   ```
   smaster# nisgrpadm -l admin.sales.doc.com.
    Group entry for admin.sales.doc.com. group:
    Explicit members:
    smaster.doc.com.
    salesreplica.doc.com.
    No implicit members
    No recursive members
    No explicit nonmembers
    No implicit nonmembers
    No recursive nonmembers
   ```

8. 他の管理者の資格を追加します。

   そのドメインで仕事をする他の管理者の資格を追加します。

   すでにもう 1 つのドメインで DES 資格をもつ管理者の場合、単に LOCAL 資格を追加します。このとき、-p フラグと -P フラグ付きの nisaddcred コマンドを実行します。たとえば、次のようになります。

   ```
   smaster# nisaddcred -p 33355 -P nisboss.doc.com. local
   ```

   まだ資格をもたない管理者の場合、2 つの方法があります。

   - 管理者に対して、自分の資格を追加するよう要求するのが 1 つの方法です。この方法は、スーパーユーザーとして実行する必要があります。ユーザー ID が 22244 で、主体名が juan.sales.doc.com. の管理者が、sales.doc.com. ドメインに自分の資格を追加する例を次に示します。

```
smaster# nisaddcred -p 22244 -P juan.sales.doc.com. local
smaster# nisaddcred -p unix.22244@sales.doc.com -P juan.sales.doc.com. des
Adding key pair for unix.22244@sales.doc.com.
Enter login password:
```

- もう 1 つの方法では、ダミーパスワードを使用して、他の管理者用の一時的な資格を作成します 。各管理者には NIS+ passwd テーブル内にエントリが必要です。

```
smaster# nisaddcred -p 22244 -P juan.sales.doc.com. local
smaster# nisaddcred -p unix.22244@sales.doc.com -P juan.sales.doc.com. des
Adding key pair for unix.22244@sales.doc.com.
Enter juan's login password:
nisaddcred: WARNING: password differs from login passwd.
Retype password:
```

各管理者は、後で chkey コマンドを実行して、自分のネットワークパスワードを変更できます。パスワードの変更方法については、第 12 章と第 13 章を参照してください。

注 – 上記の 手順 8 の 2 つの例で、小文字の -p フラグに続くドメイン名の終わりにドットを付けないでください。また、大文字の -P フラグに続くドメイン名の終わりにはドットを必ず付けてください。

---

9. ドメインの管理グループに管理者を追加します。

この手順を実行するには、他のシステム管理者がダミーパスワードを変更するまで待つ必要はありません。-a オプションを付けて nisgrpadm コマンドを実行します。最初の引数はグループ名、残りの引数は管理者名です。この例では、管理者 juan を admin.sales.doc.com. グループに追加します。

```
smaster# nisgrpadm -a admin.sales.doc.com. juan.sales.doc.com.
Added juan.sales.doc.com. to group admin.sales.doc.com.
```

10. **NIS+** テーブルを格納するための、十分なスワップ空間を割り当てます。

スワップ空間は、rpc.nisd の最大サイズの 2 倍にする必要があります。rpc.nisd が使用するメモリ量を調べるには、次のコマンドを実行してください。

```
rootmaster# /usr/lib/nis/nisstat
```

rpc.nisd は、特定の条件のもとでは、自らのコピーを作成してフォークします。メモリーが不足すると、rpc.nisd は正しく動作しません。

また、NIS+ テーブルに必要なメモリーとスワップ空間のサイズも計算できます。たとえば、NIS+ テーブル内に、180,000 人のユーザーと 180,000 台のホストがある場合、これらの 2 つのテーブルが占有するメモリーは、約 190M バイトです。180,000 人のユーザーと180,000 台のホストに資格を追加する場合、cred テーブルには、540 ,000 のエントリ (ユーザーごとにローカルの資格と DES の資格、合わせて 2 つの資格、ホストごとに 1 つの資格) が入ります。そのため、cred テーブルが占有するメモリーは、約 285M バイトになります。この例では、rpc.nisd に必要なメモリー容量は、少なくとも、190M バイト + 285M バイト = 475M バイトになります。この結果、少なくとも 1G バイトのスワップ空間が必要になります。また、rpc.nisd 全体をすべてメモリー内に保持するには、少なくとも 500M バイトが必要です。

サブドメイン構成の要覧

表 8–2 は、サブドメインの構成に必要な手順のまとめです。これは最も簡単なケースを想定しているため、このまとめを参考用として使用するには、その前に自分の実際の作業の詳細を理解することが必要です。また、ここでは、各コマンドに対するサーバーの応答を示していません。

表 **8–2** まとめ - サブドメインを設定する方法

作業	コマンド
ドメインマスターサーバーにスーパーユーザーとしてログインする	`smaster% su`
ドメインの管理グループを指定する	`# NIS_GROUP=admin.sales.doc.com.` `# export NIS_GROUP`
ドメインのディレクトリを作成し、そのサーバーを指定する	`# nismkdir -m smaster -s salesreplica sales.doc.com.`
org_dir 、groups_dirおよびテーブルを作成する。NIS 互換の場合、-Y を使用する	`# /usr/lib/nis/nissetup sales.doc.com.`
管理グループを作成する	`# nisgrpadm -c admin.sales.doc.com.`
ドメインのディレクトリに対して、完全なグループ権を割り当てる	`# nischmod g+rmcd sales.doc.com.`
管理グループにサーバーを追加する	`# nisgrpadm -a admin.sales.doc.com. smaster.doc.com. sreplica.doc.com.`
他のシステム管理者の資格を追加する	`# nisaddcred -p 22244 -P juan.sales.doc.com. local` `# nisaddcred -p unix.22244@sales.doc.com. juan.sales.doc.com. DES`
ドメインの管理グループに管理者を追加する	`# nisgrpadm -a admin.sales.doc.com. juan.sales.doc.com.`

第 9 章

NIS+ テーブルの設定

この章では、NIS+ コマンドセットを使用して、/etc ディレクトリ内のファイルや NIS マップからマスターサーバー上に NIS+ テーブルを作成する方法について説明します。また、この章では NIS+ テーブルから NIS マップへ情報を戻す方法と、passwd テーブルのパスワード列へのアクセスを制限する方法を説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、http://www.sun.com/directory/nisplus/transition.html を参照してください。

テーブルの設定

注 – NIS+ テーブルを生成する作業は、この章で説明する NIS+ コマンドセットを使用する方法よりも、パート I で説明した NIS+ 設定スクリプトを使用した場合の方が簡単です。この章で説明する方法は、NIS+ に精通した管理者や、設定スクリプトでは提供されない標準以外の機能や構成を必要とする管理者だけが使用してください。さらに、Solstice AdminSuite ツールをお持ちの場合は、NIS+ テーブルに関連する作業が簡単になります。

NIS+ テーブルを生成するには次の 4 種類の方法があります。

- ファイルから作成する (189 ページの「NIS+ テーブルをファイルから生成する」参照)
- NIS マップから作成する (195 ページの「 NIS+ テーブルを NIS マップから生成する」参照)

- `nispopulate` スクリプトを使用して作成する (103 ページの「NIS+ テーブルの生成 (populate)」および 124 ページの「新しいサブドメインのテーブルの生成」)
- Solstice AdminSuite ツールを使用する

マップまたはファイルからテーブルを生成する場合、第 5 章および第 8 章で説明する手順に従って、ルートまたはサブドメインの設定作業の中で、あらかじめテーブルを作成しておく必要があります。ドメインテーブルは、テーブルの作成後いつでも生成できますが、ドメインの設定が完了したらすぐに生成しておくことをお勧めします。テーブルを生成すると、クライアントに関する必要な情報がすでにドメインテーブル内に格納されることになるため、クライアントの追加が簡単にできます。

テーブルの生成の方法 (必要な場合)

ファイルまたは NIS マップからテーブルを生成する場合、次の 3 つの方法を使用できます。

- 「置換」この方法を使用すると、NIS+ は、初めにテーブル内の既存のエントリをすべて削除してから、情報源からのエントリを追加します。サイズの大きいテーブルの場合は、マスターサーバーの `/var/nis/trans.log` ファイルにエントリの大きなセット (既存のエントリを削除するためのセットと新しいエントリを追加するためのセット) が追加されます。`/var/nis` 内のディスク空間を大幅に使用するため、複製サーバーへの転送時間はさらに長くなります。
- 「追加」 この方法は、NIS+ テーブルに情報源からのエントリを追加します。既存のエントリが影響されることはありません。
- 「マージ」この方法は、置換オプションと結果は同じですが、内部的な処理が異なります。既存のエントリを削除後、置換するのではなく、更新します。このオプションでは、NIS+ は次の 3 種類のエントリを異なる方法で処理します。
  - 情報源内だけに存在するエントリは、テーブルに追加する
  - 情報源とテーブルの両方に存在するエントリは、テーブルで更新する
  - NIS+ テーブルだけに存在するエントリは、テーブルから削除する

  大きなテーブルを、内容があまり変わらないファイルやマップで更新する場合、マージ用のオプションを使用すると、サーバーの処理を大幅に減らすことができます。情報源内で重複していないエントリだけを削除する (置換オプションでは、単純にすべてのエントリを削除する) ため、重複するエントリに対する削除と追加の処理を 1 回ずつ少なくできます。このため、この方法をお勧めします。

NIS+ テーブルをファイルから生成する

これは /etc/hosts などの ASCII ファイルの内容を NIS+ テーブルに転送する作業です。

手順を次に示します。

1. データを転送する各ファイルの内容を確認します。
2. 各ファイルのコピーを作成します。作成したコピーを実際の転送に使用します。このマニュアルでは、hosts.xfr のように、転送されるファイルのコピーの最後に .xfr を付けます。
3. NIS+ クライアントにログインします。テーブルを更新するための資格とアクセス権が必要です。詳細は、190 ページの「ファイルのセキュリティ上の留意点」を参照してください。
4. このシェルのコマンド検索パスに /usr/lib/nis を追加します。
5. nisaddent を使用して、次の必要なファイルを 1 つずつ転送します。

```
aliases
bootparams
ethers
group
hosts
netgroup
netmasks
networks
passwd
protocols
rpc
services
shadow
ipnodes
```

注 – 新しい /etc/inet/ipnodes ファイルには、IPv4 および IPv6 のアドレスが含まれています。nisaddent を実行して、 /etc/inet/ipnodes ファイルを ipnodes.org_dir テーブルに変換してください。

---

6. publickey ファイルを転送します。
7. オートマウンタ情報を転送します。
8. 複製サーバーに対して nisping を実行します。
9. テーブルに対しチェックポイントを実行します。

ファイルのセキュリティ上の留意点

NIS+ テーブルは、NIS+ クライアントまたは NIS+ ルートマスターサーバーから生成できます。NIS+ テーブルを生成するには、スーパーユーザー (root) としてログインする必要はありませんが、一定の資格とアクセス権は必要です。テーブル内のエントリをテキストファイルのエントリによって置換またはマージする場合、そのテーブルへの作成権と削除権が必要です。新しいエントリを追加する場合、作成権だけが必要です。

注 – NIS+ のセキュリティシステムは複雑です。NIS+ セキュリティを使い慣れていない場合は、第 11 章を参照してから NIS+ 環境を構成することをお勧めします。

この処理が完了した後、テーブルエントリは、動作を実行した NIS+ 主体と、環境変数 `NIS_GROUP` によって指定されたグループによって所有されます。

前提条件

- ドメインをあらかじめ設定していて、そのマスターサーバーを実行している
- ドメインのサーバーには、新しいテーブル情報を収容できるだけの十分なスワップ領域が必要です。
- ファイル内の情報は、ロード先のテーブルに合った書式で書かれていなければなりません。対応する NIS+ テーブルに転送するテキストファイルで要求される書式については、104 ページの「`nispopulate` を実行するための前提条件」を参照してください。ローカルの `/etc` 内のファイルは、正しい書式で書かれているのが普通ですが、いくつかのコメントを削除しなければならないこともあります。
- マシン名とユーザー名は重複していない。ユーザーとマシンは、すべて固有の名前を付ける必要がある。また、ユーザーと同じ名前の付いたマシンは使用できない
- マシン名にはドット (ピリオド) および下線は使用できません。たとえば、マシン名に `sales.alpha` を使用できません。ドットや下線の代わりにハイフンを使うことはできます。たとえば、`sales-alpha` というマシン名は有効です。

必要な情報

転送されるテキストファイルの名前と位置が必要です。

NIS+ テーブルをファイルから生成する — 作業マップ

表 **9–1** NIS+ テーブルをファイルから生成する

作業	説明	指示の参照先
NIS+ テーブルをファイルから生成する	NIS+ テーブルをファイルから生成する	191 ページの「NIS+ テーブルをファイルから生成する方法」

▼ NIS+ テーブルをファイルから生成する方法

1. データを転送する各ファイルをチェックします。

   不正なエントリがないことを確認します。正しいデータが、所定の場所に正しい書式で記録されていることを確認します。エントリのうち、古いもの、無効なもの、破損しているものは削除します。また、不完全なエントリや一部のみのエントリも削除します。不完全なエントリは、設定を完了した後に追加する方が、不完全なエントリや破損したエントリを転送するよりも簡単です。

2. 転送する各ファイルの作業用コピーを作成します。

   このセクションで説明する実際のファイル転送の手順では、作成した作業用コピーを使用します。各作業用コピーには、同じファイル名拡張子を付けます (たとえば、`.xfr`)。

   ```
   rootmaster% cp /etc/hosts /etc/hosts.xfr
   ```

   万が一に備えて、使用する予定のすべてのファイルを `/etc` 以外のディレクトリにコピーしておくのもよいでしょう。`nisaddent` コマンドと `nispopulate` コマンドでは、ファイルソースディレクトリを指定できます。

3. **NIS+** クライアントにログインします。

   この作業はどの NIS+ クライアントからでも実行できます。ただし、そのクライアントは、情報の転送先となるテーブルと同じドメインに所属していなければなりません。ここで示す例では、ルートマスターサーバーを使用します。これらの例では、システム管理者はスーパーユーザーとしてログインしているため、この操作を実際に実行している (適切な資格とアクセス権を必要とする) NIS+ 主体は、ルートマスターサーバーです。

   しかし、NIS+ テーブルを更新するだけであれば、あえてルートマスターサーバーにスーパーユーザー (root) としてログインする必要はありません。スーパーユーザーであれ、一般ユーザーであれ、一定の資格、権限、許可さえあれば、どのマシンからでも NIS+ テーブルを更新できます。

4. このシェルのコマンド検索パスに `/usr/lib/nis` を追加します。

   テーブルごとに `/usr/lib/nis/nisaddent` コマンドを使用するため、検索パスに `/usr/lib/nis` を追加しておくと、毎回これを入力する必要がありません。ここでは、C シェルユーザーの場合の例と Bourne シェルまたは Korn シェルのユーザーの場合の例を示します。

C シェルの場合

```
rootmaster# setenv PATH $PATH:/usr/lib/nis
```

Bourne シェルまたは Korn シェルの場合

```
rootmaster# PATH=$PATH:/usr/lib/nis
rootmaster# export PATH
```

5. nisaddent を使用して、次のファイルを **1** 度に **1** つずつ転送します。

```
aliases
bootparams
ethers
group
hosts
ipnodes
netgroup
netmasks
networks
protocols
rpc
services
```

publickey、automounter、passwd および shadow の各ファイルは、実行する手順がそれぞれ少し異なります。publickey ファイルの場合は、手順 6 に進んでください。automounter ファイルの場合は、手順 7 に進んでください。 passwd および shadow ファイルの場合は、手順 8 に進んでください。

デフォルトでは、nisaddent はファイル情報をテーブル情報に追加します。置換またはマージを行うには、-r または -m オプションを使用します。

置換する場合、次のように入力します。

```
rootmaster# nisaddent -r -f filename table [domain]
```

追加する場合、次のように入力します。

```
rootmaster# nisaddent -a -f filename table  [domain]
```

マージする場合、次のように入力します。

```
rootmaster# nisaddent -m -f filename table  [domain]
```

初めてテーブルを生成する場合は、デフォルトの -a オプションが最適です。NIS+ テーブルを NIS マップまたは /etc 内のファイルと同期させるための最適なオプションは -m (マージ) オプションです。

- *filename* はファイル名です。通常は、nisaddent によって作成されたことを示す .xfr をファイル名の最後につけます。
- *table* は NIS+ テーブル名です。*domain* 引数は省略可能で、テーブルを別のドメインに生成するときに使用します。ルートドメインのマスターサーバーから入力されたいくつかの例を次に示します。これらのファイルは、/etc 内のファイルを編集したものです。

```
rootmaster# nisaddent -m -f /etc/hosts.xfr hosts
rootmaster# nisaddent -m -f /etc/groups.xfr groups
```

この作業をルート以外のサーバーから実行する場合、ルート以外のサーバーは、サポートするドメインの上のドメインに所属することに注意してください。つまり、ルート以外のサーバーはもう一つのドメインのクライアントです。たとえば、sales.doc.com. マスターサーバーは doc.com. ドメインに所属します。そのマスターサーバーからテーブルを sales.doc.com. ドメインに生成するには、nisaddent 文に sales.doc.com. ドメイン名を追加しなければなりません。

```
salesmaster# nisaddent -f /etc/hosts.xfr hosts Sales.doc.com.
```

この作業を sales.doc.com. ドメインのクライアントとして実行した場合、このコマンドにドメイン名を指定する必要はありません。

エントリが NIS+ テーブルに転送されたことを確認するには、niscat コマンドを使用します。詳細については、第 19 章を参照してください。

```
rootmaster# niscat groups.org_dir
root::0:root
other::1::
bin::2:root,bin,daemon
.
```

障害追跡のこつ： niscat を実行しても表示された内容が反映されてない場合、変更がマスターサーバーから複製サーバーに送られていないためかもしれません。そのような場合は、5 〜 10 分後にもう一度 niscat の実行を試みるか、niscat に -M オプションを付けて実行してください。-M オプションを付けると、マスターサーバーのデータが取得されます。

6. publickey ファイルを転送する

   ドメインの cred テーブルには、すでにいくつかの資格が格納されているため、この cred テーブルに転送する publickey テキストファイルの内容によって、これらの資格が上書きされないよう確認する必要があります。これを避けるには、publickey テキストファイルからこれらの資格を取り除きます。rootmaster で次のような行がある場合は、すべて削除してください。

   ```
   unix.rootmaster@doc.com public-key:private-key [alg-type]
   ```

   こうすれば、publickey ファイルの内容を cred テーブルに転送できます。nisaddent に -a (追加) オプションを付けて実行します。

```
rootmaster# nisaddent -a -f /etc/publickey.xfr -t cred.org_dir publickey [domain]
```

   ただし、この操作は DES 資格を cred テーブルに転送するだけです。主体は、cred テーブルに対する自分の LOCAL 資格を作成する必要があります。

7. オートマウンタ情報を転送します。

   nissetup ユーティリティは auto_master テーブルと auto_home テーブルを作成しますが、これらは標準の NIS+ テーブルとはみなされません。したがって、これらのテーブルに情報を転送するには、少し異なる構文が必要となります。特に、-t フラグを使用し、そのテーブルが key-value 形式であることを指定しなければなりません。

```
rootmaster# nisaddent -f auto.master.xfr -t auto_master.org_dir key-value
rootmaster# nisaddent -f auto.home.xfr -t auto_home.org_dir key-value
```

8. **NIS+** passwd テーブルを作成します。

NIS+ の passwd テーブルは、/etc/passwd および /etc/shadow の両方のディレクトリのファイルから引き出されるデータで構成されます。このため、nisaddent を 2 回実行して passwd テーブルを作成しなければなりません。passwd テーブルを対象として /etc/passwd ファイルからデータを引き出す場合と、shadow テーブルを対象として /etc/shadow ファイルからデータを引き出す場合の 2 回実行します。shadow ファイルに対して nisaddent を実行する場合、shadow テーブルが存在しなくても、ターゲットテーブルに shadow を指定すること、そしてデータが実際に passwd テーブルの shadow 列に配置されることに注意してください。

```
rootmaster# nisaddent -m -f /etc/passwd.xfr passwd
rootmaster# nisaddent -m -f /etc/shadow.xfr shadow
```

9. nisping を実行して更新内容を複製サーバーに送ります。

nisping を実行すると、複製サーバーに変更が反映されます。

```
master1# nisping domain
master1# nisping org_dir.domaincom.
master1# nisping groups_dir.domain
```

10. テーブルに対しチェックポイントを実行します。

ここまでの手順で、マスターサーバーと複製サーバーの NIS+ データセットがメモリ内で更新されました。今度はそれをディスク上のテーブルファイルに書き込みます。この作業を「チェックポイント」といいます。チェックポイントは必ずしもここで実行しなければならないわけではありません。定期的に実行しさえすればいいのですが、重要な変更を加えた場合は、できるだけ早い機会にディスクに書き込むことをお勧めします。

この手順により、ドメインをサポートしている全サーバーが、それらの .log ファイルからディスク上のテーブルのコピーに新しい情報を転送します。ルートドメインを設定したばかりの場合、そのルートドメインにはまだ複製サーバーがないため、この手順はルートマスターサーバーだけが対象となります。チェックポイントを実行するには nisping コマンドに -C (大文字) オプションを付けて実行します。

```
rootmaster# nisping -C org_dir
Checkpointing replicas serving directory org_dir.doc.com. :
Master server is rootmaster.doc.com.
 Last update occurred at July 14, 1997
Master server is rootmaster.doc.com.
checkpoint succeeded.
```

スワップ空間が不足している場合、サーバーはチェックポイントを正常に実行できませんが、そのことを通知してくれません。スワップ空間が十分あることを確認する方法として、niscat コマンドを使ってテーブルの内容をリスト表示する方法があります。スワップ空間が不足している場合、次のエラーメッセージが表示されます。

```
can't list table: Server busy, Try Again.
```

このメッセージ内容からはわかりませんが、これはスワップ空間の不足を示しています。スワップ空間を増やし、このドメインに再びチェックポイントを実行します。

NIS+ テーブルを NIS マップから生成する

ここでは、NIS マップの内容を NIS+ テーブルに転送します。手順を次に示します。

1. データを転送する各 NIS マップの内容を確認します。
2. NIS+ クライアントにログインします。
3. このシェルのコマンド検索パスに `/usr/lib/nis` を追加します。
4. `nisaddent` を使用して、次のマップを 1 度に 1 つずつ転送します。

```
aliases
bootparams
ethers
group
hosts
netgroup
netmasks
networks
passwd
protocols
rpc
services
```

5. `publickey` マップを転送します。
6. オートマウンタ情報を転送します。
7. `nisping` を実行して変更内容を複製サーバーに反映させます。
8. テーブルに対しチェックポイントを実行します。

マップのセキュリティ上の留意点

この作業を行うシステム管理者 (またはクライアント上のスーパーユーザー) に適切な資格とアクセス権がある限り、この作業は、どの NIS+ クライアントからでも実行できます。テーブル内のエントリを NIS マップのエントリで置換またはマージする場合、そのテーブルへの作成権と削除権が必要です。新しいエントリを追加する場合、作成権だけが必要です。

この作業が完了した後、テーブルエントリは、作業を実行した NIS+ 主体 (システム管理者である自分、またはスーパーユーザーとしてログインした場合はそのクライアント) と、環境変数 `NIS_GROUP` によって指定されたグループによって所有されます。

前提条件

- ドメインをあらかじめ設定していて、そのマスターサーバーを実行している

- NIS+ テーブルにロードしようとしている NIS マップ用の dbm ファイル (.pag および .dir ファイル) は、/var/yp のサブディレクトリ内にある
- マシン名とユーザー名は重複していない。ユーザーとマシンは、すべて固有の名前を付ける必要がある。また、ユーザーと同じ名前の付いたマシンは使用できない
- マシン名にはドット (ピリオド) を使用できない。たとえば、マシン名に sales.alpha を使用できない。sales-alpha というマシン名は、使用できる

必要な情報

NIS マップの名前と位置が必要です。

NIS+ テーブルを NIS マップから生成する — 作業マップ

表 **9–2** NIS+ テーブルを NIS マップから生成する

作業	説明	指示の参照先
NIS+ テーブルを NIS マップから生成する	NIS+ テーブルを NIS マップから生成する	196 ページの「NIS+ テーブルをマップから生成する方法」

▼ NIS+ テーブルをマップから生成する方法

1. データを転送する各 **NIS** マップをチェックします。

   不正なエントリがないことを確認します。正しいデータが、所定の場所に正しい書式で記録されていることを確認します。エントリのうち、古いもの、無効なもの、破損しているものは削除します。また、不完全なエントリや一部のみのエントリも削除します。不完全なエントリは、設定を完了した後に追加する方が、不完全なエントリや破損したエントリを転送するよりも簡単です。

2. **NIS+** クライアントにログインします。

   この作業はどの NIS+ クライアントからでも実行できます。ただし、そのクライアントは、情報の転送先となるテーブルと同じドメインに所属していなければなりません。ここで示す例では、ルートマスターサーバーを使用します。これらの例では、システム管理者はスーパーユーザーとしてログインしているため、この操作を実際に実行している (適切な資格とアクセス権を必要とする) NIS+ 主体は、ルートマスターサーバーです。

3. このシェルのコマンド検索パスに /usr/lib/nis を追加します。

   テーブルごとに /usr/lib/nis/nisaddent コマンドを使用するため、コマンド検索パスに /usr/lib/nis を追加しておくと、毎回これを入力する必要がありません。ここでは、C シェルユーザーの場合の例と Bourne シェルまたは Korn シェルのユーザーの場合の例を示します。

C シェルの場合

```
rootmaster# setenv PATH $PATH:/usr/lib/nis
```

Bourne シェルまたは Korn シェルの場合

```
rootmaster# PATH=$PATH:/usr/lib/nis
rootmaster# export PATH
```

4. `nisaddent` を使用して、次のマップを **1** 度に **1** つずつ転送します。

```
aliases
bootparams
ethers
group
hosts
netgroup
netmasks
networks
passwd
protocols
rpc
services
```

`publickey` とオートマウンタマップでは、少し手順が異なります。`publickey` ファイルの場合は、「NIS+ テーブルをファイルから生成する方法」の 手順 6 に、オートマウンタファイルの場合は、手順 7 に進んでください。

デフォルトでは、`nisaddent` はファイル情報をテーブル情報に追加します。置換またはマージを行うには、`-r` または `-m` のオプションを使用します。

置換する場合、次のように入力します。

```
rootmaster# nisaddent -r -y nisdomain table
```

追加する場合、次のように入力します。

```
rootmaster# nisaddent -a -y nisdomain table
```

マージする場合、次のように入力します。

```
rootmaster# nisaddent -m -y nisdomain table
```

初めてテーブルを生成するときに最適なオプションは、デフォルトの `-a` オプションです。NIS+ テーブルを NIS マップまたは `/etc` 内のファイルと同期させるための最適なオプションは `-m` (マージ) オプションです。

`-y` (小文字) オプションは、テキストファイルではなく、NIS ドメインを示します。*nisdomain* 引数は、ユーザーが NIS+ テーブルに転送しようとしているマップを持つ NIS ドメインの名前です。実際のマップを指定する必要はありません。`nisaddent` ユーティリティは、*table* 引数に対応する NIS マップを自動的に選択します。次に例をいくつか示します。

```
rootmaster# nisaddent -m -y olddoc hosts
rootmaster# nisaddent -m -y olddoc passwd
rootmaster# nisaddent -m -y olddoc groups
```

最初の例では、`olddoc` (NIS) ドメイン内の `hosts.byname` マップと `hosts.byaddr` マップの内容を、ルートドメイン (NIS+) 内の NIS+ `hosts` テーブルに転送します。2 番目の例では、パスワード関連情報を格納している NIS マップ

を、NIS+Passwd テーブルに転送します。3 番目の例では、グループ関連情報で同じことを実行します。nisaddent コマンドの詳細については、第 19 章を参照してください。

5. publickey マップを転送します。

   ドメインの cred テーブルには、すでにいくつかの資格が格納されているため、cred テーブルに転送する publickey マップの内容によって、これらの資格が上書きされないように確認する必要があります。

   a. 初めに、publickey マップをファイルにダンプします。続いて、テキストエディタでそのファイルをオープンします。

      ```
      rootmaster# makedbm -u /var/yp/olddoc/publickey.byname /etc/publickey.xfr
      rootmaster# vi /tmp/publickey.tmp
      ```

   b. publickey マップから、ログインしているマシンの資格を削除します。

      rootmaster に対しては、次のような行は、すべて削除してください。

      ```
      unix.rootmaster@doc.com public-key:private-key [alg-type]
      ```

   c. これにより、マップではなく「ファイル」の内容を cred テーブルに転送できます。nisaddent に -a (追加) オプションを付けて実行します。

```
rootmaster# nisaddent -a -f /etc/publickey.xfr -t cred.org_dir Publickey
```

      ただし、この操作は DES 資格を cred テーブルに転送するだけです。cred テーブルに対する自分の LOCAL 資格は自分で作成する必要があります。

6. オートマウンタ情報を転送します。

   nissetup ユーティリティは auto_master テーブルと auto_home テーブルを作成しますが、これらは標準の NIS+ テーブルとはみなされません。したがって、これらのテーブルに情報を転送するには、少し異なる構文が必要となります。

```
rootmaster# nisaddent -y olddoc -Y auto.master -t auto_master.org_dir key-value
rootmaster# nisaddent -y olddoc -Y auto.home -t auto_home.org_dir key-value
```

   NIS ドメイン名 (この例では olddoc) と同様、-m と -y のオプションが必要です。しかし、NIS マップ名 (auto.master など) の前には -Y (大文字) を付けなければなりません。次に、オートマウンタの「テキストファイル」を転送するときに必要なように、標準の NIS+ テーブルであることを示す -t オプションを使用しなければなりません。この引数は、NIS+ ディレクトリオブジェクト (auto_master.org_dir) とテーブルの種類 (key-value) です。NIS+ テーブル名の後ろには必ず接尾辞 org_dir を追加してください。

7. nisping を実行して更新内容を複製サーバーに送ります。

   nisping を実行すると、複製サーバーに変更が反映されます。

   ```
   master1# nisping domain
   master1# nisping org_dir.domaincom.
   master1# nisping groups_dir.domain
   ```

8. テーブルに対しチェックポイントを実行します。

この手順により、ドメインをサポートしている全サーバーが、それらの .log ファイルからディスク上のテーブルのコピーに新しい情報を転送します。ルートドメインを設定したばかりの場合、そのルートドメインにはまだ複製サーバーがないため、この手順はルートマスターサーバーだけが対象となります。nisping コマンドに -C (大文字) オプションを付けて実行します。

```
rootmaster# nisping -C org_dir
Checkpointing replicas serving directory org_dir.doc.com. :
Master server is rootmaster.doc.com.
 Last update occurred at July 14, 1994
Master server is rootmaster.doc.com.
checkpoint succeeded.
```

スワップ空間が不足している場合、サーバーはチェックポイントを正常に実行できませんが、そのことを通知しません。スワップ空間が十分あることを確認する方法として、niscat コマンドを使ってテーブルの内容をリスト表示する方法があります。スワップ空間が不足している場合、次のエラーメッセージが表示されます。

```
can’t list table: Server busy, Try Again.
```

このメッセージ内容からはわかりませんが、これはスワップ空間の不足を示しています。スワップ空間を増やし、このドメインに再びチェックポイントを実行します。

NIS+ から NIS に情報を転送する

ここでは、NIS+ テーブルの内容を、Solaris 1.x の NIS マスターサーバー上の NIS マップに転送します。手順を次に示します。

1. NIS+ サーバーにログインします。
2. NIS+ テーブルを出力ファイルに転送します。
3. 出力ファイルの内容を NIS マップに転送します。

NIS から NIS+ に情報を転送する際のセキュリティ上の留意点

この作業を実行するには、内容を転送する各テーブルへの読み取り権が必要です。

前提条件

マップを NIS サーバー上にあらかじめ作成しておかなければなりません。

NIS+ から NIS に情報を転送する — タスクマップ

表 **9–3** NIS+ から NIS に情報を転送する

作業	説明	指示の参照先
NIS+ から NIS に情報を転送する	NIS+ テーブルから Solaris 1.x の NIS のマスターサーバー上の NIS マップに情報を転送する	200 ページの「NIS+ から NIS へ情報を転送する方法」

▼ NIS+ から NIS へ情報を転送する方法

1. **NIS+** サーバーにログインします。
   この例では、`dualserver` という名前のサーバーを使用します。
2. **NIS+** テーブルを出力ファイルに転送します。
   次に示すように、各テーブルごとに 1 回、`-d` オプションを付けた `nisaddent` コマンドを使用します。

   ```
   dualserver% /usr/lib/nis/nisaddent -d -t table tabletype> filename
   ```

   `-d` オプションは、*table* の内容を `/etc` 内の標準のファイル形式に変換して *filename* に出力します。
3. 出力ファイルの内容を **NIS** マップに転送します。
   NIS+ の出力ファイルは、NIS マップ用の入力ファイルとして使用できる ASCII ファイルです。これらを NIS マスターの `/etc` ディレクトリにコピーし、通常の方法で `make` を実行します。

   ```
   dualserver# cd /var/yp
   dualserver# make
   ```

所有者および管理者に対する Passwd 列へのアクセス制限

ここでは、`passwd` テーブルのパスワードに関係する列に対する読み取り権を所有者とテーブルの管理者だけに制限し、しかも `passwd` テーブル内の残りの列に対する認証主体 (アプリケーションを含む) の読み取り権に影響を与えない方法を説明します。

この作業では、次のアクセス権を設定します。

```
                        Nobody  Owner   Group  World
Table Level Rights:     ----    rmcd    rmcd   ----
Passwd Column Rights:   ----    rm--    rmcd   ----
Shadow Column Rights:   ----    rm--    rmcd   ----
```

Passwd 列のセキュリティ上の留意点

- ドメインを NIS 互換モードで動作させないでください。
- ドメインのすべての NIS+ クライアントには DES 資格が必要です。
- ドメインのすべてのクライアントで、Solaris 2.3 以降のリリースが実行されている必要があります。
- ユーザーのネットワークパスワード (DES 資格の暗号化に使用) は、ログインパスワードと同じディレクトリでなければなりません。

前提条件

- passwd テーブルをあらかじめ設定しておかなければなりません。ただし、情報が入っている必要はありません。
- この作業を実行する NIS+ 主体には、passwd テーブルへの変更権が必要です。

必要な情報

必要なのは passwd テーブルの名前だけです。

所有者および管理者に対する Passwd 列へのアクセス制限 — 作業マップ

表 **9–4** 所有者および管理者に対する Passwd 列へのアクセス制限

作業	説明	指示の参照先
所有者および管理者に対する Passwd 列へのアクセス制限	NIS+ のコマンドを使って passwd.org_dir を変更し、所有者および管理者に対する passwd 列へのアクセスを制限する	201 ページの「Passwd 列へのアクセスを制限する方法」

▼ Passwd 列へのアクセスを制限する方法

1. ドメインのマスターサーバーにログインします。
   この例ではルートマスターサーバー rootmaster を使用します。
2. 現在のテーブルと列のアクセス権を確認します。
   niscat -o コマンドを実行します。

```
rootmaster# niscat -o passwd.org_dir
```

この作業では、次のような既存のアクセス権になっていることを前提としています。

```
Access Rights    : ----rmcdrmcdr---
Columns          :
                       [0]  Name              : name
                            Access Rights : r-----------r---
                       [1]  Name              : passwd
                            Access Rights : -----m----------
                       [2]  Name              : uid
                            Access Rights : r-----------r---
                       [3]  Name              : gid
                            Access Rights : r-----------r---
                       [4]  Name              : gcos
                            Access Rights : r----m------r---
                       [5]  Name              : home
                            Access Rights : r-----------r---
                       [6]  Name              : shell
                            Access Rights : r-----------r---
                       [7]  Name              : shadow
                            Access Rights : r-----------r---
```

自分のアクセス権が異なる場合、別の構文を使用する必要があります。手順については、第 15 章 を参照してください。

3. テーブルのアクセス権を変更します。

   nischmod コマンドを実行して、テーブルのオブジェクトレベルのアクセス権を ----rmcdrmcd---- に変更します。

```
rootmaster# nischmod og=rmcd,nw= passwd.org_dir
```

4. 列のアクセス権を変更します。

   nistbladm コマンドを -u オプションを付けて使用して、passwd 列 および shadow 列のアクセス権を次のように変更します。

```
passwd ---- rm-- ---- ----
shadow ---- r--- ---- ----
rootmaster# nistbladm -u passwd=o+r, shadow=o+r passwd.org_dir
```

5. 新しいアクセス権を確認します。

   手順 2 の場合と同様、niscat -o コマンドを実行します。アクセス権は、手順 2 の出力と同じでなければなりません。

テーブルの生成のまとめ

NIS+ テーブルの生成に必要な手順のまとめを次に示しています。これは、最も簡単な場合を想定しています。このため、まとめを参考として使用する前に、作業の詳細を十分に理解しておいてください。また、このまとめでは各コマンドに対するサーバーの応答は示していません。

表 **9–5** まとめ : NIS+ テーブルへのファイルの転送

作業	コマンド
NIS+ クライアントにログインする	rootmaster%
転送するファイルの作業用コピーを作成する	% cp /etc/hosts /etc/hosts.xfr
検索パスに /usr/lib/nis を追加する	% PATH=$PATH:/usr/lib/nis; export PATH
各ファイルを 1 つずつ転送する	% nisaddent -m -f /etc/hosts.xfr hosts
publickey ファイルから古いサーバー資格を削除する	% vi /etc/publickey.xfer
資格テーブルに publickey ファイルを転送する	% nisaddent -a -f /etc/publickey.xfr cred
オートマウンタファイルを転送する	% nisaddent -f auto.master.xfr -t auto_master.org_dir key-value % nisaddent -f auto.home.xfr -t auto_home.org_dir key-value
テーブルディレクトリにチェックポイントを実行する	% nisping -C org_dir

表 **9–6** まとめ : NIS+ テーブルへのマップの転送

作業	コマンド
NIS+ クライアントにログインする	rootmaster%
検索パスに /usr/lib/nis を追加する	% PATH=$PATH:/usr/lib/nis; export PATH
各マップを 1 つずつ転送する	% nisaddent -m -y olddoc hosts
publickey マップをファイルにダンプする	% makedbm -u /var/yp/olddoc/publickey.byname> /etc/publickey.xfr
新しい資格を削除する	% vi /etc/publickey.xfr
publickey ファイルを転送する	% nisaddent -a -f /etc/publickey.xfr -t cred.ortg_dir publickey
オートマウンタマップを転送する	% nisaddent -y olddoc -Y auto.master -t auto_master.org_dir key-value % nisaddent -y olddoc -Y auto.home -t auto_home.org_dir key-value

表 **9–6** まとめ : NIS+ テーブルへのマップの転送　(続き)

作業	コマンド
テーブルディレクトリにチェックポイントを実行する	`% nisping -C org_dir`

表 **9–7** まとめ : NIS+ テーブルを NIS マップに転送する

作業	コマンド
NIS+ サーバーにログインする	`dualserver%`
NIS+ テーブルをファイルに転送する	`% /usr/lib/nis/nisaddent -d [-t` *table*`]` *tabletype* `>` *filename*
ファイルを NIS マップに転送する	`% makedbm` *flags output-file NIS-dbm-file*

表 **9–8** まとめ : `Passwd` 列へのアクセス権を変更する

作業	コマンド
ドメインのマスターサーバーにログインします。	`rootmaster#`
テーブルの既存の権利をチェックする	`# niscat -o passwd.org_dir`
テーブルに新しい権利を割り当てる	`# nischmod og=rmcd,nw= passwd.org_dir`
列に新しい権利を割り当てる	`# nistbladm -u passwd=o+r, shadow=n+r passwd.org_dir`
新しい権利を確認する	`# niscat -o passwd.org_dir`

パート III　NIS+ の管理

このパートでは、Solaris オペレーティング環境上での NIS+ ネームサービスの管理および問題解決について説明します。

第 10 章

NIS+ のテーブルと情報

この章では、これらのテーブルの構造と設定方法の概要について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ テーブル構造

NIS+ は、さまざまなネットワーク情報をテーブルに格納します。NIS+ テーブルには、単純なテキストファイルやマップには見られない機能がいくつかあります。これらのテーブルは列エントリ構造をもち、検索パスを使用することや、リンクすることができ、しかも複数の設定方法があります。NIS+ にはあらかじめ構成された 16 個のシステムテーブルがあり、独自のテーブルを作成することもできます。表 10–1 は、事前構成した NIS+ テーブルを示します。

表 10–1 NIS+ テーブル

テーブル	テーブル内の情報
`hosts`	ドメイン内にあるすべてのマシンのネットワークアドレスとホスト名
`bootparams`	ドメイン内の全ディスクレスクライアントのルート、スワップ、およびダンプパーティションの位置
`passwd`	ドメイン内の全ユーザーに関するパスワード情報

表 **10–1** NIS+ テーブル　*(続き)*

テーブル	テーブル内の情報
cred	ドメインに属する主体の資格
group	ドメイン内の全 UNIX グループのグループ名、グループパスワード、グループ ID、およびメンバー
netgroup	ドメイン内のマシンやユーザーが所属できるネットグループ
mail_aliases	ドメイン内のユーザーのメール別名に関する情報
timezone	ドメイン内の全マシンの時間帯
networks	ドメイン内のネットワークとこれらの標準の名前
netmasks	ドメイン内のネットワークとこれらに関連付けられているネットマスク
ethers	ドメイン内の全マシンの Ethernet アドレス
services	ドメイン内で使用される IP サービス名とそれらのポート番号
protocols	ドメイン内で使用される IP プロトコルのリスト
RPC	ドメインで使用できる RPC サービスの RPC プログラム番号
auto_home	ドメイン内の全ユーザーのホームディレクトリの位置
auto_master	自動マウンタのマップ情報

cred テーブルには NIS+ のセキュリティに関係する情報だけが入っています。これについては第 12 章で説明します。

これらのテーブルには、ユーザー名からインターネットサービスまでの幅広い情報が格納されています。この情報の多くは、設定時または構成の手順の中で生成されます。たとえば、password テーブル内のエントリは、ユーザーアカウントが設定されたときに作成されます。hosts テーブル内のエントリは、マシンがネットワークに追加されたときに作成されます。また、networks テーブル内のエントリは、新しいネットワークが設定されたときに作成されます。

この情報はこのような広範囲にわたる操作によって生成されるため、このマニュアルでは詳細を説明していません。ただし、使いやすさを考慮して、テーブルの各列に含まれる情報は、第 23 章に要約してあります。詳細な説明は、グループを NIS+ グループやネットグループと区別する場合など、理解しにくい内容についてのみ行います。テーブルの情報の詳細は、Solaris の「システム管理マニュアルセット」と「ネットワーク管理マニュアルセット」を参照してください。

注 – ドメインにさらに多くのオートマウントマップを作成できますが、必ず NIS+ テーブルとして格納し、`auto_master` テーブルの中に入れてください。オートマウントマップを作成して、ユーザー用に作成された `auto_master` に追加するときは、列名にはキーと値が必要です。オートマウントの詳細は、オートマウントまたは NFS ファイルシステムの関連マニュアルを参照してください。

注 – ネームサービスとして、NIS+ テーブルはオブジェクト自体ではなく、オブジェクトの参照情報を格納するように設計されています。そのため、NIS+ はエントリの大きなテーブルをサポートできません。テーブルに非常に大きなエントリがある場合、`rpc.nisd` は機能しません。

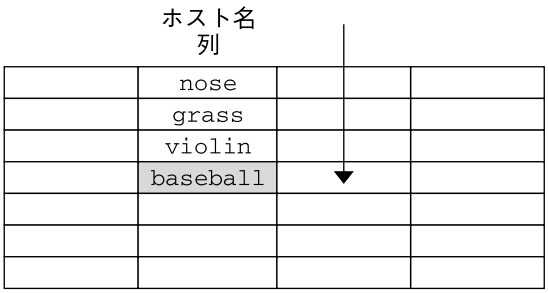
列とエントリ

NIS+ テーブルにはさまざまな種類の情報が格納されていますが、これらの基本構造はすべて同じで、行と列から構成されます。この列は、「エントリ」または「エントリオブジェクト」と呼ばれます。

ホスト名
列

	nose		
	grass		
	violin		
	baseball		

クライアントは、キー、または検索可能な任意の列によりアクセスできます。たとえば、`baseball` という名前のマシンのネットワークアドレスを見つけるには、ホスト名の列を探して `baseball` を見つけます。



次に、baseball のエントリ行を移動して、そのネットワークアドレスを見つけます。

クライアントは、任意のレベルでテーブル情報にアクセスできる (つまり、テーブルに全体としてアクセスできる) ため、NIS+ は 3 つのレベルすべてにセキュリティ機構を提供しています。たとえば、管理者は、オブジェクトレベルで全員にテーブルの読み取り権を割り当て、列レベルで所有者に変更権を割り当て、エントリレベルでグループに変更権を割り当てることができます。テーブルのセキュリティの詳細は、第 15 章で説明します。

検索パス

テーブルには、その「ローカル」ドメインについての情報だけが収められています。たとえば、doc.com. ドメイン内のテーブルには、doc.com. ドメインのユーザー、クライアント、およびサービスについての情報だけが収められています。たとえば、sales.doc.com. ドメイン内のテーブルには、sales.doc.com. ドメインのユーザー、クライアント、およびサービスについての情報だけが収められています。

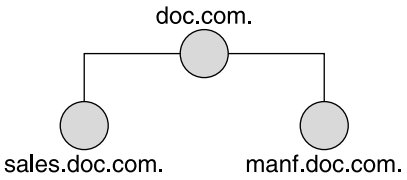
あるドメイン内のクライアントが、別のドメインに格納されている情報を見つけようとした場合、完全指定名を使用しなければなりません。84 ページの「環境変数 NIS_PATH」に説明されているように、環境変数 NIS_PATH が正しく設定されていれば、NIS+ サービスによって自動的に処理されます。

情報を探すときにサーバーがたどる「検索パス」をどの NIS+ テーブルでも指定できます。この検索パスは、NIS+ テーブルをコロンで区切って順番に並べたものです。

table:*table*:*table*...

検索パス内のテーブル名は、完全指定する必要はありません。テーブル名は、コマンド行で入力された名前と同じように展開されます。サーバーが自分のローカルテーブル内に情報を検出できなかった場合、サーバーはクライアントにテーブルの検索パスを返します。クライアントはそのパスを使用して、その情報が見つかるかまたは最後まで、検索パスに名前があるすべてのテーブルを順番に探していきます。

検索パスのメリットを次に示します。次のようなドメイン階層があるとします。



下位にある 3 つのドメインの hosts テーブルの内容は次のようになります。

表 **10–2** hosts テーブルの例

sales.doc.com.		**manf.doc.com.**	
127.0.0.1	localhost	127.0.0.1	localhost
111.22.3.22	luna	123.45.6.1	sirius
111.22.3.24	phoebus	123.45.6.112	rigel
111.22.3.25	europa	123.45.6.90	antares
111.22.3.27	ganymede	123.45.6.101	polaris
111.22.3.28	mailhost	123.45.6.79	mailhost

ここで、sales.doc.com. ドメイン内の luna にログインしたユーザーが、別のクライアントにリモートログインする場合を考えてみます。このユーザーが完全指定名を指定しない場合、そのユーザーがリモートログインできるのは localhost、phoebus、europa、ganymede、および mailhost の 5 台のマシンだけです。

sales.doc.com. ドメインの hosts テーブルの検索パスに、manf.doc.com. ドメインの hosts テーブルが指定されていると仮定します。

hosts.org_dir.manf.doc.com.

これで sales.doc.com. ドメイン内のユーザーは、たとえば、rlogin sirius と入力することができ、NIS+ サーバーはこれを検出します。NIS+ サーバーはまずローカルドメインで sirius を探し、見つからなければ manf.doc.com. ドメイン内を探

します。クライアントは manf.doc.com. ドメインをどのように認識するのでしょうか。第 2 章で説明したように、この情報はディレクトリキャッシュに格納されています。これがディレクトリキャッシュにない場合、クライアントは、第 2 章で説明したプロセスに従ってこの情報を入手します。

ただし、検索パスを指定する方法には、若干の欠点があります。ユーザーがたとえば、rlogin luba (luna でなく) などと間違った名前を入力した場合、サーバーは 1 つのテーブルではなく、3 つのテーブルを探さなければエラーメッセージを返せません。名前空間の全域に検索パスを設定した場合、1 つの操作は 2、3 のドメインではなく、10 個のドメイン内のテーブルを検索してから終了します。もう 1 つの欠点は、多数のクライアントが NIS+ テーブルにアクセスする必要があるとき、複数のサーバーのセットと通信するために、パフォーマンスの低下を招くという点です。

また、mailhost は別名として使用されることが多いため、特定のメールホストについての情報を探したいときは、その完全指定名 (mailhost.sales.doc.com. など) を使用しなければなりません。そうしないと、NIS+ は検索したすべてのドメインで見つけたすべてのメールホストを返します。そうしないと、 NIS+ は、検索したすべてのドメインで見つかったメールホストをすべて返します。

テーブルの検索パスを指定するには、356 ページの「nistbladm コマンド」で説明するように、nistbladm コマンドに -p オプションを付けて実行します。

テーブルの設定方法

NIS+ テーブルの設定には 3 つまたは 4 つの作業が伴います。

1. org_dir ディレクトリの作成
2. システムテーブルの作成
3. システムテーブル以外のテーブルの作成 (必要に応じて)
4. 情報を入力してテーブルを生成 (populate) する

62 ページの「NIS+ のファイルとディレクトリ」で説明したように、NIS+ のシステムテーブルは org_dir ディレクトリに格納されます。したがって、テーブルを作成するには、その前にテーブルを入れる org_dir ディレクトリを作成しなければなりません。これには 3 つの方法があります。

- nisserver スクリプトを使用。nisserver スクリプトは、ディレクトリとシステムテーブのフルセットを作成します。nisserver スクリプトを実行することを推奨します。
- nismkdir コマンドを使用。nismkdir コマンドは単にディレクトリを作成するものです。
- /usr/lib/nis/nissetup ユーティリティを使用。nissetup ユーティリティは org_dir および groups_dir ディレクトリとシステムテーブルのフルセットを作成します。

nisserver スクリプトと、nissetup および nismkdir ユーティリティについては、336 ページの「nismkdir コマンド」 を参照してください。また、nismkdir コマンドの詳細については、336 ページの「nismkdir コマンド」 を参照してください。

nissetup ユーティリティのメリットは、サーバーが NIS 互換モードで動作しているドメインのテーブルに対して、適切なアクセス権を割り当てる機能です。-Y フラグを付けて実行すると、このユーティリティは作成するオブジェクトの「未認証」カテゴリに読み取りアクセス権を割り当てます。その結果、未認証の NIS クライアントは、そのドメインの NIS+ テーブルから情報を得ることができます。

NIS+ の 16 個のシステムテーブルと、これらに格納される情報の種類については、第 10 章で説明します。これらを作成するには、前述の 3 つの方法を使用できます。nistbladm コマンドは NIS+ テーブルの作成と変更を行います。nistbladm コマンドで名前空間内のすべてのテーブルを作成できますが、入力量が多くなり、列の正確な名前やアクセス権を知っている必要があります。nisserver スクリプトを使う方がはるかに簡単です。

システムテーブル以外のテーブル、つまり NIS+ によってまだ構成されていないテーブルを作成するには、nistbladm コマンドを使用します。オートマウントマップを追加する場合は、キーの列と値の列が必要です。

NIS+ テーブルを生成するには、NIS マップから行う方法、ASCII ファイル (/etc 内のファイルなど) から行う方法、および手作業の 3 つの方法があります。

NIS サービスからアップグレードしている場合、すでにネットワーク情報の大部分が NIS マップに格納されています。この情報を NIS+ テーブルに手作業で再入力する必要はありません。nispopulate スクリプトか nisaddent ユーティリティで自動的に転送できます。

他のネットワーク情報サービスを使用していない場合で、しかもネットワークデータを /etc 内のファイルで管理しているときも、この情報を再入力する必要はありません。nispopulate スクリプトか nisaddent ユーティリティを使用すれば自動的に転送できます。

初めてネットワークを設定する場合、十分なネットワーク情報がどこにも格納されていないことがあります。この場合、まず情報を入手して、次に NIS+ テーブルに手作業で入力する必要があります。これを行うには、nistbladm コマンドを使用できます。また、特定のテーブルの全情報を「入力ファイル」(これは本質的に /etc 内のファイルと同じです) に入力し、次に、nispopulate スクリプトか nisaddent ユーティリティを使用してファイルの内容を転送することによっても同じことが行えます。

テーブルの更新方法

ドメインが設定されると、サーバーはそのドメインの NIS+ テーブルの最初のバージョンを受け取ります。これらのバージョンはディスクに格納されますが、サーバーは起動されると、これらをメモリーにロードします。サーバーがテーブルに対する更新を受信すると、サーバーはそのメモリーにあるテーブルをすぐに更新します。サーバーが情報の要求を受信すると、その応答のためにメモリーにあるコピーを使用します。

もちろん、サーバーはその更新をディスクにも格納する必要があります。ディスクにあるテーブルを更新するには時間がかかるため、NIS+ の全サーバーはテーブル用の「ログ」ファイルを持っています。このログファイルは、テーブルに対して行われた変更内容を、ディスク上で更新できるまで、一時的に格納するように設計されています。ログファイルは、テーブル名を接頭辞として使用し、これに `.log` を追加します。たとえば:

```
hosts.org_dir.log
bootparams.org_dir.log
password.org_dir.log
ipnodes.org_dir.log
```

ログファイルが大きくなりすぎて大量のディスク領域を占有しないように、ディスクにあるテーブルのコピーは毎日更新しなければなりません。このプロセスは「チェックポイントを実行する」と呼ばれます。これには、`nisping -C` コマンドを使用します。

第 11 章

NIS+ のセキュリティの概要

この章では、NIS+ のセキュリティシステムと、システムが NIS+ 名前空間全体に与える影響について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については
、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

Solaris のセキュリティ - 概要

Solaris のセキュリティは、ユーザーが Solaris 環境に入るために通過しなければならないゲートと、内部でそのユーザーが何をできるのかを判定するアクセス権マトリックスの 2 つによって確保されています。このセキュリティシステムの概略については、図 11–1 を参照してください。

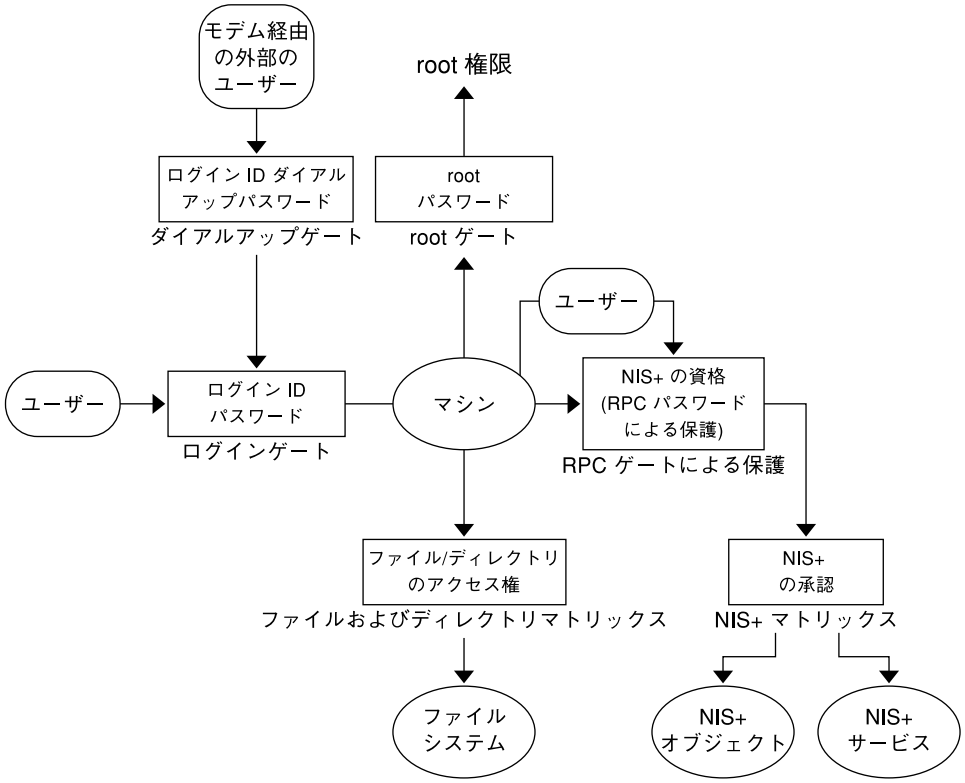


図 **11–1** Solaris のセキュリティゲートとフィルタ

上の図に示すように、Solaris システムは 4 つのゲートと 2 つのアクセス権マトリックスから構成されます。

- 「ダイアルアップゲート」モデムや電話回線により Solaris 環境にアクセスするには、正しいログイン ID とダイアルアップパスワードが必要です。
- 「ログインゲート」Solaris 環境に入るには、正しいログイン ID とユーザーパスワードが必要です。
- 「ファイルおよびディレクトリマトリックス」Solaris 環境に入ってから、ファイルやディレクトリに関して、読み取り、実行、変更、作成、削除等の各種操作を行う権限はアクセス権マトリックスによってコントロールされます。
- 「root ゲート」root 権限にアクセスするには、正しいスーパユーザー (root) パスワードが必要です。
- 「RPC ゲートによる保護」セキュリティレベル 2 (デフォルト値) の NIS+ 環境においては、NIS+ サービスを使って NIS+ オブジェクト (サーバー、ディレクトリ、テーブル、テーブルエントリなど) にアクセスしようとする場合、ユーザー ID は Secure RPC プロセスを使った NIS+ によって確認されます。

ダイヤルアップ、ログイン、ルートゲート、およびファイルとディレクトリのアクセス権マトリクスの詳細は、Solaris の各マニュアル及び資料を参照してください。

Secure RPC ゲートをパスするには、Secure RPC パスワードが必要です。「Secure RPC パスワード」は、「ネットワークパスワード」と呼ばれる場合もあります。Secure RPC パスワードとログインパスワードは通常同一なので、ゲートをパスした場合はパスワードを再入力する必要がありません (2 つのパスワードが異なった場合についての説明は、239 ページの「Secure RPC パスワードとログインパスワードの問題」 を参照)。

ユーザー要求が Secure RPC ゲートを自動的にパスするのに、1 組の「資格」セットが使われます。ユーザーの資格を作成、提示、判定するプロセスを、そのプロセスがユーザーが誰であり、正しい RPC パスワードを所持しているかを確認することから、「認証 (authentication)」と呼びます。この認証プロセスは、ユーザーが NIS+ サービスを要求するごとに自動的に実行されます。

NIS 互換モード (YP 互換モードでもある) の NIS+ 環境では、ユーザーが正しい資格を所持しているかにかかわらず (認証プロセスによって、ID が確認され、Secure RPC パスワードがチェックされたかにかかわらず)、誰でもすべての NIS+ オブジェクトを読み取ることができ、またそれらエントリを変更できるので、Secure RPC ゲートによるガードは極めて弱くなります。誰もがすべての NIS+ オブジェクトを読み取りかつそれらエントリを変更できるので、互換モードの NIS+ ネットワークは通常モードのよりも安全性が低くなります。

NIS+ の認証と資格の作成および管理については、第 12 章を参照してください。

- 「NIS+ オブジェクトマトリックス」一度正しく認証されると、ユーザーが NIS+ オブジェクトを読み取り、変更、作成、削除する権限はアクセス権マトリックスに管理されます。このプロセスをNIS+ の「承認 (authorization)」と呼びます。

  NIS+ のアクセス権および承認の詳細は、第 15 章を参照してください。

NIS+ のセキュリティ - 概要

NIS+ のセキュリティは NIS+ の名前空間全体にかかわっています。セキュリティと名前空間を別に設定することはできません。このため、セキュリティの設定方法は名前空間の各構成要素を設定する手順と密接に関係することになります。一度 NIS+ のセキュリティ環境を設定すると、その後はユーザーの追加および削除、アクセス権の変更、グループメンバーの再割当やネットワーク展開を管理するために必要なその他すべての管理ルーチンタスクを実行できます。

NIS+ のセキュリティ機能は名前空間内の情報と、名前空間そのものの構造を不正アクセスから保護するものです。このセキュリティ機能がなければ、どんな NIS+ クライアントであっても名前空間に保存された情報を獲得することも変更することもできないだけでなく、ダメージを与える可能性があります。

NIS+ セキュリティには次の 2 つの機能があります。

- 「認証 (authentication)」認証は NIS+ 主体を識別するのに使われます。主体 (ユーザーまたはマシン) が NIS+ オブジェクトにアクセスしようとするときはいつも、ユーザー ID と Secure RPC パスワードが確認されチェックされます。
- 「承認 (authorization)」承認はアクセス権を識別するのに使われます。NIS+ 主体が NIS+ オブジェクトにアクセスしようとするときはいつも、所有者、グループ、その他、未認証の 4 つのクラスの 1 つに位置付けられています。NIS+ セキュリティシステムは NIS+ 管理機能により、各クラスに NIS+ オブジェクトに対する読み取り、変更、作成、削除の各権限を指定します。たとえば、あるクラスは `passwd` テーブルの特定列を変更できるが、その列を読み取ることはできないように、あるいは別のクラスはいくつかのエントリを読み取ることはできるが、それ以外はできないように設定できます。

NIS+ のセキュリティ機能は 2 段階のプロセスを用意しています。

1. 「認証 (authentication)」。NIS+ は資格 (credential) を使ってユーザーを確認します。
2. 「承認 (authorization)」。承認プロセスがユーザー ID を確認すると、NIS+ はクラスを判定します。NIS+ オブジェクトまたはサービスに対して何が行えるのかは、ユーザーがどのクラスに属しているかに依存します。これは UNIX の標準的なファイルおよびディレクトリのアクセス権システムと同様の考え方に基づいています (クラスの詳細は、223 ページの「承認クラス」 を参照)。

このプロセスは、マシン A の root 権限を持つユーザーが `su` コマンドを使って、第 2 の NIS+ アクセス権で NIS+ オブジェクトにアクセスすることを防ぐものです。

ただし、NIS+ は、他人のユーザーログインパスワードを知っている者が、そのユーザーになりすましてユーザー ID と NIS+ アクセス権を使用することを妨ぐことはできません。また root 権限を持つユーザーが「同一」マシンからログインしている別のユーザーになりすますのを防ぐこともできません。

図 11–2 に、このプロセスの詳細を示します。

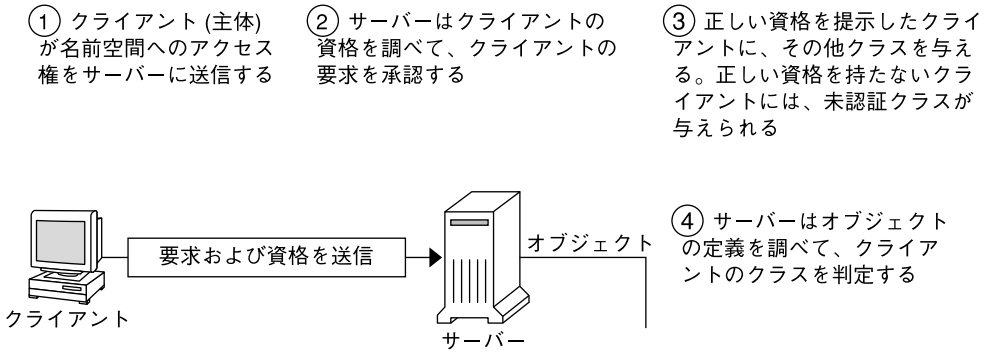


図 **11–2** NIS+ セキュリティプロセスの概要

NIS+ の主体について

NIS+ の主体とは、NIS+ サービス要求を依頼するエンティティ (クライアント) です。NIS 主体には、クライアントマシンに一般ユーザーとしてログインしたユーザー、スーパーユーザーとしてログインしたユーザー、NIS+ クライアントマシンにおいてスーパーユーザー特権で動作しているプロセスを含みます。つまり NIS+ の主体はクライアントユーザーの場合もクライアントマシンの場合もあります。

NIS+ の主体は、NIS+ サーバーから NIS+ サービスを提供する主体を指すこともあります。すべての NIS+ サーバーは NIS+ クライアントにもなるので、サーバーが NIS+ の主体となる場合もあります。

NIS+ セキュリティレベル

NIS+ サーバーは、2 つのセキュリティレベルのどちらかで動作します。セキュリティレベルにより、要求を送信して認証を取得しなければならない資格主体の種類が決まります。NIS+ は、最高セキュリティレベル (セキュリティレベル 2) で動作するように設計されています。レベル 0 は、テスト、設定、およびデバッグのときにだけ使用します。セキュリティレベルについては、表 11–1 を参照してください。

表 **11–1** NIS+ セキュリティレベル

セキュリティレベル	説明
0	セキュリティレベル 0 は、NIS+ 名前空間の初期テストと初期設定を行うレベル。セキュリティレベル 0 で動作している NIS+ サーバーは、ドメイン内の全 NIS+ オブジェクトに対する完全なアクセス権をどの NIS+ 主体にも与える。レベル 0 はシステム管理者だけが設定管理のためにだけ使用する。レギュラーユーザーはネットワークにおける通常のオペレーションに使用できない
1	セキュリティレベル 1 は AUTH_SYS セキュリティを利用する。このレベルは NIS+ のサポートを受けられないため、利用すべきではない
2	セキュリティレベル 2 はデフォルトで、NIS+ が現在提供している最高レベルのセキュリティ。これは DES 資格を持たない要求には未認証クラスが与えられ、未認証クラスに割り当てられたアクセス権が与えられる。無効な DES 資格を使用する要求だけを認証する。資格を使用する要求は再試行される。有効な DES 資格獲得に繰り返し失敗すると、無効資格を持った要求として承認エラーになる (主体が原因となって資格が無効になる場合、その原因として要求をした主体に対してそのマシンで `keylogin` が実行されていなかったり、クロックの同期がとれていなかったり、鍵が不一致であるなどの原因が考えられる)。

セキュリティレベルとパスワードコマンド

Solaris リリース 2.0 から 2.4 の環境では、`nispasswd` を使用してパスワードを変更します。ただし、資格がなければ、`nispasswd` は機能しません (より上位のレベルでの資格があった場合を除き、セキュリティレベル 0 より下では機能しない)。Solaris 2.5 以降の場合は、セキュリティレベルや資格ステータスにかかわらず、`passwd` コマンドを使ってパスワードを変更できます。

NIS+ の認証と資格 - 紹介

NIS+ 資格の目的は、NIS+ サービスや NIS+ オブジェクトへのアクセスを要求する各主体の ID を「認証」(確認) することにあります。つまり、NIS+ 資格および承認プロセスは Secure RPC システムを実装することです。

資格および認証システムは他人になりすますのを防ぐシステムです。あるマシンの root 権限を持つユーザーが `su` コマンドを使って、ログインしていないもしくは別のマシンにログインしている第 2 のユーザーになりすまし、NIS+ アクセス権を使用して NIS+ オブジェクトにアクセスすることを妨げます。

サーバーが主体を認証すると、主体は 4 つの認証クラスのどれかに置かれ、次にサーバーは承認されている動作を知るためにアクセスしようとする NIS+ オブジェクトをチェックします。承認の詳細は、223 ページの「NIS+ の承認とアクセス - 紹介」を参照してください。

ユーザーおよびマシンの資格

主体には、「ユーザー」と「マシン」の 2 つの基本的な種類があります。

- 「ユーザー *(user)* 資格」。一般ユーザーとして NIS+ クライアントにログインした場合、NIS+ サービス要求には当人の「ユーザー」資格を含みます。
- 「マシン *(machine)* 資格」。スーパーユーザーとして NIS+ クライアントにログインした場合、サービス要求は「クライアントマシン」資格を使います。

DES と LOCAL 資格

NIS+ 主体には DES と LOCAL の 2 つの資格が用意されています。

DES 資格 (credential)

注 – 認証を行う仕組みとしては DES 資格が唯一のものです。将来は他の仕組みも利用できるようになるかもしれません。DES 資格と NIS+ 資格は同等のものではありません。

このマニュアルでは、DES 資格という用語は、鍵の長さにかかわりなく Diffie - Hellman 鍵をベースにした認証を表すものとして一搬的に使われています。鍵の長さはあらかじめ決められたセットから指定することができます。Diffie - Hellman 鍵の長さの設定や表示を行うには `nisauthconf(1M)` を使用します。

DES (データ暗号化規格) 資格は資格の 1 種であり、保護認証を提供するものです。このマニュアルで NIS+ 主体を認証するために資格をチェックすると記述している場合は、NIS+ がチェックしている DES 資格のことを意味します。

主体が NIS+ サービスを要求したり、NIS+ オブジェクトにアクセスするごとに、ソフトウェアは格納してある資格情報を使って主体の資格を作成します。DES 資格は NIS + 管理機能によって各主体に作られた情報から作成されます。第 12 章を参照してください。

- 主体の DES 資格の正当性を NIS+ が確認すると、その主体は「認証」されます。
- 主体は所有者、グループ、その他のいずれかの承認クラスを得るために認証されなければなりません。つまり、これらクラスの 1 つを得るために、有効な DES 資格を持つ必要があるのです。有効な DES 資格を持たない主体は自動的に未認証クラスに割り当てられます。

- 主体がクライアントユーザーであるかクライアントマシンであるかにかかわらず、DES 資格情報は常に主体のホームドメインの `cred` テーブルに格納されます。

LOCAL 資格

LOCAL 資格は、ユーザー ID 番号とホームドメイン名を含む NIS+ 主体名とのマップです。ユーザーがログインすると、システムは DES 資格が格納されているホームドメインを特定する LOCAL 資格をチェックします。システムはその情報を使ってユーザーの DES 資格情報を獲得します。

ユーザーがリモートドメインにログインした場合、その要求はユーザーのホームドメインを示す LOCAL 資格を使います。NIS+ は次にユーザーの DES 資格情報を知るためにユーザーのホームドメインを問い合わせます。こうして、たとえユーザーの DES 資格情報がリモートドメインに格納されていなくても、リモートドメインで認証されます。

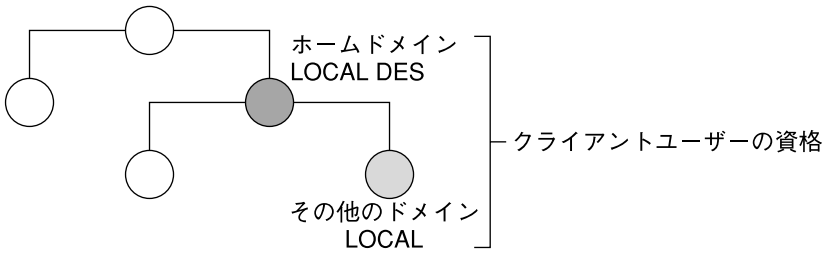


図 **11–3** 資格とドメイン

LOCAL 資格情報はどのドメインにも格納できます。実際、リモートドメインにログインし認証されるためには、クライアントユーザーはリモートドメインの `cred` テーブルに LOCAL 資格を持っている必要があります。ユーザーが、アクセスしようとするリモートドメインに LOCAL 資格を持っていなかった場合、NIS+ はユーザーの DES 資格を獲得するためにユーザーのホームドメインに入ることができなくなります。そのような場合、ユーザーは認証されず未認証クラスを与えられることになります。

ユーザータイプと資格種類

ユーザーは両方の資格を持つことができますが、マシンは DES 資格しか持てません。

すべてのマシンの root UID は常に 0 であるため、root は他のマシンに対して NIS+ アクセスできません。もしマシン A の root (UID=0) がマシン B にアクセスしようとすると、すでに存在しているマシン B の root (UID=0) と衝突します。このため、LOCAL 資格は、クライアント「マシン」については意味を持たないため、クライアント「ユーザー」にだけ認められています。

表 11–2 資格のタイプ

資格のタイプ	クライアントユーザー	クライアントマシン
DES	有効	有効
LOCAL	有効	無効

NIS+ の承認とアクセス - 紹介

NIS+ の承認の基本的目的は、各 NIS+ 主体が各 NIS+ オブジェクトとサービスに対して持っているアクセス権を特定することにあります。

一度主体の NIS+ 要求が認証されると、NIS+ は承認クラスを与えます。NIS+ オブジェクトに対して行うことのできる動作を指定するアクセス権は各クラスに与えられます。クラスごとに別のアクセス権が与えられる場合、各承認クラスはそれぞれアクセス権を持つということです。

- 「承認クラス」。承認クラスには次の 4 つがあります。所有者、グループ、その他、未認証 (詳細は、次の223 ページの「承認クラス」を参照)。
- 「アクセス権」。アクセス権には次の 4 つがあります。作成 (CREATE)、削除 (DESTROY)、変更 (MODIFY)、読み取り (READ) (詳細は、227 ページの「NIS+ アクセス権」を参照)。

承認クラス

NIS+ オブジェクトは NIS+ 主体に直接アクセス権を与えずに、主体の 4 つの「クラス」にアクセス権を与えます。

- 「所有者」。オブジェクトの所有者になった主体が所有者クラスの権限を与えられます。
- 「グループ」。各 NIS+ オブジェクトグループは関連した 1 つのグループを持ちます。オブジェクトグループのメンバーは NIS+ 管理機能が指定します。オブジェクトグループクラスに属している主体がグループクラスの権限を与えられます。この場合の「グループ」とは NIS+ グループを表します。UNIX のグループやネットグループのことではありません。
- 「その他」。その他クラスは、サーバーが認証できるすべての NIS+ 主体を包含するクラスです。所有者とグループクラスを除く、すべての認証されたものを意味します。
- 「未認証」。認証されていないものすべてを意味します。

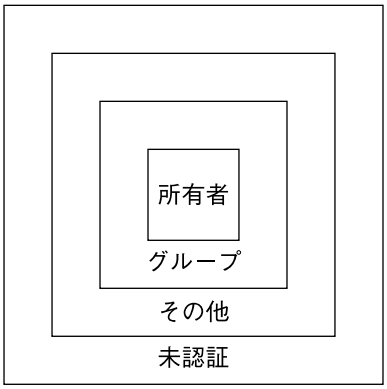


図 **11–4** 承認クラス

システムは、NIS+ 要求を受信すると、要求を出している主体が属しているクラスを判定します。主体は、判定されたクラスに属しているすべてのアクセス権を使用できます。

オブジェクトに対するアクセス権は、各クラスで任意の組み合わせが可能です。ただし、通常は上位クラスが下位クラスのすべての権限に追加権限を持つように割り当てられます。

たとえば、未認証およびその他クラスには読み取りアクセスを許可し、グループクラスには読み取りおよび変更アクセスを許可し、所有者クラスには読み取り、変更、作成、および削除アクセスを許可することができます。

4 つのクラスについては、以下に詳しく説明します。

所有者クラス

所有者は「 単一」の NIS+ 主体です。

NIS+ オブジェクトへのアクセスを要求する主体は、所有者アクセス権を得る前に認証され (有効な DES 資格を提示し) なければなりません。

デフォルトではオブジェクトの所有者は、オブジェクトを作成した主体になります。ただし、オブジェクト所有者は 2 つの方法で別の主体に所有権を譲ることができます。

- オブジェクトが作成されたときに、主体が別の所有者を指定する方法 (281 ページの「コマンドによるアクセス権の指定」を参照)
- オブジェクトの作成後に、主体がオブジェクトの所有権を変更する方法 (291 ページの「オブジェクトとエントリの所有権の変更」を参照)

一度主体が所有権を放棄すると、その主体はオブジェクトに対する所有者のアクセス権すべてを放棄することになり、グループ、その他、未認証のいずれかの所有権を持つことになります。

グループクラス

オブジェクトグループは 1 つの NIS+ グループです。この場合の「グループ」とは、UNIX のグループやネットグループのことではありません。

NIS+ オブジェクトにアクセスを要求する主体は、認証され (有効な DES 資格を提示し) かつグループアクセス権を与えられる前にそのグループに属していなければなりません。

1 つの NIS+ グループは NIS+ 主体の集合であり、名前空間へのアクセス権を与えるのに都合の良いようにまとめられたものです。NIS+ グループに与えられるアクセス権はそのグループのメンバーである主体すべてに適用されます。オブジェクトの所有者が、そのオブジェクトのグループに属している必要はありません。

オブジェクトが作成された時、そのオブジェクトはデフォルトグループに割り当てられます。オブジェクトが作成された時でもその後でも、そのオブジェクトをデフォルト以外のグループに割り当てることができます。オブジェクトグループはいつでも変更できます。

注 – NIS+ グループに関する情報は NIS+ `group` テーブルに格納されないことに注意してください。NIS+ `group` テーブルには UNIX グループに関する情報が格納されます。NIS+ グループに関する情報は、適切な `groups_dir` ディレクトリのオブジェクトに格納されます。

NIS+ グループの情報は、各 NIS+ ドメインの `groups_dir` サブディレクトリにある、NIS+ グループ「オブジェクト」に格納されます。

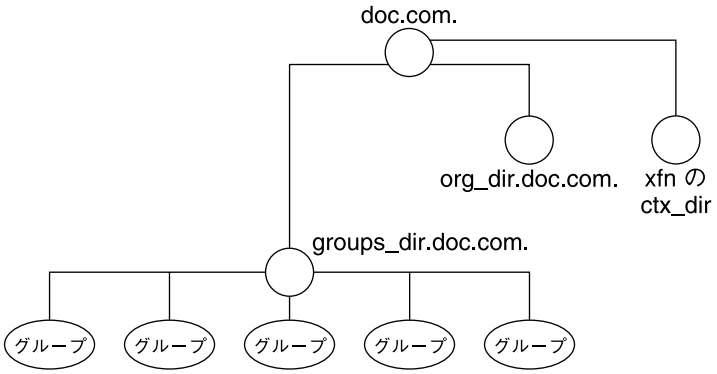


図 **11–5** NIS+のディレクトリ構造

NIS+ グループの管理については、第 17 章を参照してください。

その他クラス

その他クラスは、NIS+ によって認証されたすべての NIS+ 主体のクラスです。すなわち、所有者およびグループクラスのすべてと有効な DES 資格を提示したものを加えたものです。

その他に与えられたアクセス権は、認証されたすべての主体に適用されます。

未認証クラス

未認証クラスは、正しく認証されなかったものすべてで構成されます。すなわち、有効な DES 資格を提示しなかったものです。

承認クラスと NIS+ オブジェクトの階層

NIS+ オブジェクトと承認クラスには各レベルに適用される階層があります。標準的な NIS+ ディレクトリ階層のデフォルトは次のようになります。

- 「ディレクトリレベル」。各 NIS+ ドメインには `groups_dir` と `org_dir` という 2 つの NIS+ ディレクトリオブジェクトがあります。各 `groups_dir` ディレクトリオブジェクトはさまざまなグループで構成されます。各 `org_dir` ディレクトリオブジェクトにはさまざまなテーブルが含まれます。
- 「グループレベルまたはテーブルレベル」。グループは個々のエントリやグループで構成されます。テーブルには列と個々のエントリが含まれます。
- 「列レベル」。テーブルは 1 つ以上の列で構成されます。

- 「エントリ (行) レベル」。グループもしくはテーブルは 1 つ以上のエントリを持ちます。

各レベルには 4 つの承認クラスが適用されます。ディレクトリオブジェクトはそれ自身の所有者とグループを持ちます。ディレクトリ内の個々のテーブルは、ディレクトリの所有者およびグループと異なる所有者およびグループを持ちます。テーブル内では、1 エントリ (行) が個々の所有者もしくはグループを持ちます。この所有者もしくはグループは、テーブル全体の所有者やグループとは異なり、またオブジェクト全体の所有者やグループとも異なります。テーブル内の個々の列は、テーブル全体と同じ所有者やグループを持ちます。

NIS+ アクセス権

NIS+ オブジェクトのオブジェクト定義には、オブジェクトのアクセス権を指定します (オブジェクト定義は、`niscat -o` コマンドで確認できます)。

NIS+ オブジェクトは、UNIX ファイルが UNIX ユーザーに対するアクセス権を指定するのと同じ方法で、NIS+ 主体に対するアクセス権を指定します。アクセス権は、NIS+ 主体が NIS+ オブジェクトについて実行することが許されている動作の種類を表します。

NIS+ の動作はオブジェクトの種類によって異なりますが、読み取り、変更、作成、および削除の 4 つのクラスに分類されます。

- 「読み取り」- オブジェクトの読み取り権を持った主体はオブジェクトの内容を読み取ることができます。
- 「変更」- オブジェクトに対する変更権を持った主体がオブジェクトの内容を変更できます。
- 「削除」- オブジェクトの削除権を持った主体はオブジェクトの内容を削除することができます。
- 「作成」- 上位レベルのオブジェクトに対する作成権を持った主体が、そのレベルの下位に新規のオブジェクトを作成できます。すなわち、NIS+ ディレクトリオブジェクトの作成権を持っていれば、そのディレクトリ内に新しいテーブルを作成できます。NIS+ テーブルの作成権を持っていれば、そのテーブル内に新しい列とエントリを作成できます。

NIS+ クライアントから NIS+ サーバーへのすべての通信は、実際には特定の NIS+ オブジェクトに対してこれらの動作のうちの 1 つを実行してほしいという要求です。たとえば、NIS+ 主体が他のマシンの IP アドレスを要求した場合、実際には、この種の情報を格納している `hosts` テーブルオブジェクトへの読み取り権を要求しています。主体が NIS+ 名前空間にディレクトリを追加するようサーバーに要求した場合、実際には、そのディレクトリの親オブジェクトに対して「変更」権を要求しています。

これらの権限は論理的には、ディレクトリ、テーブル、列とエントリのように下位に展開するものであることに注意してください。たとえば、新規にテーブルを作成するには、そのテーブルを格納する NIS+ ディレクトリオブジェクトに対する作成権が必

要です。そのテーブルを作成した場合、そのデフォルト所有者になります。所有者として、自分自身にそのテーブルに対する作成権を与え、テーブルに新規のエントリを作成できます。テーブル内に新規のエントリを作成した場合、それらのエントリの所有者になります。所有者として、他のユーザーにテーブルレベルの作成権を与えることもできます。たとえば、テーブルのグループクラスにテーブルレベルの作成権を与えることができます。その場合、テーブルのグループのすべてのメンバーがテーブル内に新規のエントリを作成できます。新規のテーブルエントリを作成したグループの個々のメンバーは、そのエントリのデフォルト所有者になります。

NIS+ 管理者

NIS+管理者は、NIS+オブジェクトに対して「管理者権限」を持つユーザです。管理者権限は、オブジェクトに対する作成および削除権限で構成されます。また、オブジェクトによっては変更権限が含まれる場合もあります (NIS+ アクセス権については、227 ページの「NIS+ アクセス権」 を参照してください)。

NIS+ オブジェクトを作成するものは誰でもそのオブジェクトに対する初期アクセス権を持つと設定します。もし作成者が管理権限をオブジェクトの所有者に限った場合 (初期状態では作成者) は、所有者だけがそのオブジェクトに対する管理権限を持ちます。一方、作成者が管理権限をオブジェクトのグループに与えた場合は、グループの全員がそのオブジェクトに対して管理権限を持つことになります。

あるオブジェクトに対して管理権限を持つものは誰でもそのオブジェクトの NIS+ 管理者とみなされます。

すなわち、NIS+ ソフトウェアは NIS+ 管理者を 1 人にしようとするものではないということです。

理論的には、管理権限をその他クラスに与えることも、未認証クラスに与えることもでき、ソフトウェア上では可能です。しかし、管理権限をグループクラス以上に広げて与えてしまうと、NIS+ のセキュリティを実質的に無効にしてしまいます。もし管理権限をその他クラスや未認証クラスに与えた場合、NIS+ のセキュリティは保証されません。

NIS+ のパスワード、資格、およびコマンド

管理者のパスワード、資格、および鍵には次のコマンドを使用します。各コマンドの説明についてはマニュアルを参照してください。

- `chkey` - 主体の Secure RPC 鍵の組み合わせを変更します。必要のない場合は `chkey` を使わずに、`passwd` を使ってください。詳細については、255 ページの「NIS+ 主体の鍵の変更」 を参照してください。
- `keylogin` - `keyserv` を使って主体の非公開鍵を復号化し格納します。
- `keylogout` - 格納されている非公開鍵を `keyserv` から削除します。
- `keyserv` - サーバーに非公開暗号鍵を格納します。詳細については、254 ページの「キーログイン」を参照してください。
- `newkey` - 公開鍵データベースに新しい鍵ペアを作成します。
- `nisaddcred` - NIS+ 主体に資格 (credential) を作成します。詳細については、243 ページの「資格情報の作成」および、251 ページの「NIS+ 資格情報の管理」を参照してください。
- `nisauthconf` - Diffie - Hellman 鍵の長さの表示あるいは設定を行います。
- `nisupdkeys` - ディレクトリオブジェクト内の公開鍵を更新します。詳細については、259 ページの「公開鍵の更新」を参照してください。
- `passwd` - 主体のパスワードを変更し管理します。詳細については、第 16 章を参照してください。

第 12 章

NIS+ 資格の管理

この章では、NIS+ 資格とその管理方法について説明します。

注 – NIS+ セキュリティタスクには、Solstice AdminSuite ツールがあればより簡単に実行できるものがあります。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ 資格について

NIS+ 資格は、NIS+ のユーザーを識別するために使用します。この章では、NIS+ セキュリティシステム全体と、特にシステムで資格が果たす役割について、十分理解しているユーザーを想定しています。

これらタスクを実行するのに使われるコマンドの構文、オプションなどの詳細については、NIS+ のマニュアルを参照してください。

注 – この章の DES 資格の説明は、192 ビット Diffie - Hellmaun DES 資格に適用することができます。これらは似てはいますが、その他の鍵の長さを使用した認証は細部では異なっています。コマンド行インタフェースを使用して鍵を操作するときは、その違いはユーザーにもシステム管理者にも意識されることはありません。指定した鍵の長さの表示や設定は `nisauthconf(1M)` を使って行います。

資格の機能

注 – この章では、NIS+ のコマンド一式を使用してクライアント情報を作成する方法について説明します。

資格および認証システムは他人になりすますのを防ぐシステムです。あるマシンの root 権限で `su` コマンドを使って、ログインしていないもしくは別のマシンにログインしている第 2 のユーザーになりすまし、第 2 のユーザーの NIS+ アクセス権を使用して NIS+ オブジェクトにアクセスすることを防ぎます。

注意 – NIS+ には、他人のユーザーログインパスワードを知っている者が、そのユーザーになりすましてユーザー ID と NIS+ アクセス権を使用することを妨ぐことはできません。また root 権限を持つユーザーが「同一」マシンにログインしている別のユーザーになりすますのを防ぐこともできません。

資格と資格情報

DES 資格をどのように作成しそれがどのように機能するのかを理解するには、実際の資格と、資格を作成しチェックするのに使われる情報とを区別する必要があります。

- 「資格情報」。DES 資格を作成するのに使われるデータであり、サーバーが資格をチェックするのに使われる
- 「DES 資格」。主体を認証するために、主体からサーバーへ渡される一連の数字。主体が NIS+ 要求を行うたびに作成されチェックされる。DES 資格については、236 ページの「DES 資格の詳細」を参照してください。

認証コンポーネント

資格および認証プロセスを機能させるには、次の要素が必要です。

- 「主体の DES 資格情報」。この情報は各主体に NIS+ 管理者が最初に作成します。この情報は主体のホームドメインの `cred` テーブルに格納されます。主体の DES 資格情報は次のもので構成されます。
  - 「主体名」。ユーザーの完全ログイン ID もしくはマシンの完全ホスト名
  - 「主体の *Secure RPC* ネット名」。各主体は固有の Secure RPC ネット名を持っています (Secure RPC ネット名の詳細については、236 ページの「DES 資格の Secure RPC ネット名」を参照)。
  - 「主体の公開鍵」
  - 「主体の非公開暗号鍵」
- 「主体の LOCAL 資格」
- 「サーバーの公開鍵」。各ディレクトリオブジェクトには、当該ドメインのすべてのサーバーの公開鍵の複製が格納されています。`cred` テーブルには各サーバーの DES 資格も格納されていることに注意してください。
- 「主体の公開鍵のキーサーバーの複製」。キーサーバーは、その時点でログインしている主体 (ユーザーまたはマシン) の公開鍵の複製を持っています。

主体の認証方法

承認プロセスには 3 つのフェーズがあります。

- 「準備フェーズ」。NIS+ 管理機能がユーザーのログインに先立って行う設定動作。たとえば、ユーザーのための資格情報を作成
- 「ログインフェーズ」。ユーザーがログインした時にシステムが行う動作
- 「要求フェーズ」。NIS+ 主体が、NIS+ サービスもしくは NIS+ オブジェクトに要求を発する時にソフトウェアが行う動作

これら 3 つのフェーズの詳細については、次の項目を参照してください。

資格準備フェーズ

ユーザーの資格情報は、`nisclient` スクリプトを使用すれば最も簡単に作成できます。この節では、NIS+ のコマンド一式を使用してクライアント情報を作成する方法について説明します。

NIS+ 主体がログインする前に、NIS+ 管理者は当該主体 (ユーザーまたはマシン) のための資格情報を作成する必要があります。管理者は次のようにする必要があります。

- 各主体の公開鍵と非公開暗号鍵を作成します。これらの鍵は主体のホームドメインの `cred` テーブルに格納されます。これは `nisaddcred` コマンドを使って行われます (247 ページの「NIS+ 主体の資格情報を作成する方法」を参照)。
- サーバーの公開鍵を作成します (259 ページの「公開鍵の更新」 を参照)。

ログインフェーズ - 詳細

主体がシステムにログインする時、次のステップが自動的に実行されます。

1. 主体のために `keylogin` プログラムが起動します。`keylogin` プログラムは `cred` テーブルから主体の非公開暗号鍵を取得し、主体のログインパスワードを使って復号化します。

注 – 主体のログインパスワードが Secure RPC パスワードと異なる場合、`keylogin` コマンドはパスワードを復号化できず、「`cannot decrypt`」などのエラーとなるか、メッセージなしでコマンドが失敗します。この問題については、239 ページの「Secure RPC パスワードとログインパスワードの問題」を参照してください。

---

2. 復号化された主体の非公開暗号鍵は、要求フェーズでの利用に備えてキーサーバーに渡され格納されます。

注 – 復号化された非公開暗号鍵はユーザーが `keylogout` コマンドを実行するまでキーサーバーに格納されます。ユーザーが `keylogout` を実行せずにログアウトした (またはログアウトせずに帰宅してしまった) 場合には、復号化された非公開鍵がサーバーに格納されたままになっています。もしユーザーマシンの root 権限を持った誰かがユーザーログイン ID に `su` コマンドを使った場合、その人間はユーザーの復号化された非公開鍵を使用でき、ユーザーのアクセス承認を使って NIS+ オブジェクトにアクセスできることになります。このような事態を避けるため、ユーザーが業務を終了する時は確実に `keylogout` コマンドを使ってログアウトするように注意する必要があります。同様にもしユーザーがログオフする場合は、業務に戻る時はログバックするだけで良いのです。もしユーザーが確実にログオフしなかった場合、業務に戻る時に確実に `keylogin` を実行する必要があります。

要求フェーズ - 詳細説明

NIS+ 主体の要求が NIS+ オブジェクトにアクセスするつど、その主体を認証するために NIS+ ソフトウェアは複数段階のプロセスを実行します。

1. NIS+ はオブジェクトドメインの `cred` テーブルをチェックします。
   - 主体が LOCAL 資格情報を持っている場合、NIS+ では LOCAL 資格に含まれるドメイン情報を使用して主体のホームドメイン の `cred` 資格を検索し、必要な情報を取得します。
   - もし主体が資格情報を持っていなかった場合、残りのプロセスは実行されず、主体の承認アクセスクラスには未認証クラスが与えられます。
2. ユーザーのホームドメインの `cred` テーブルから、NIS+ がユーザーの DES 資格を取得します。ユーザーパスワードを使って非公開暗号鍵が復号化され、キーサーバーに格納されます。

3. NIS+ が NIS+ ディレクトリオブジェクトからサーバーの公開鍵を獲得します。
4. キーサーバーが、主体の復号化された非公開鍵とオブジェクトのサーバー (オブジェクトが格納されているサーバー) の公開鍵を使って、「共通鍵」を作成します。
5. 共通鍵を使用して、暗号化された「DES 鍵」が作成されます。これは、Secure RPC が乱数を発生させ、これを共通鍵を使って暗号化することで行われます。このため、DES 鍵は「乱数鍵」もしくは「乱数 DES 鍵」として参照される場合があります。
6. NIS+ が主体のサーバーの時刻情報に基づいてタイムスタンプ (DES 鍵を使って暗号化される) を作成します。
7. NIS+ が 15 秒のタイムウィンドウ (DES 鍵を使って暗号化される) を作成します。この「ウィンドウ」はタイムスタンプとサーバーの内部クロックとの間で許容される最長時間になります。
8. NIS+ が主体の DES 資格を作成します。これは次のもので構成されます。
   - 主体の cred テーブルに基づく Secure RPC ネット名 (`unix.` *identifier@domain*)
   - キーサーバーから得た暗号化された主体の DES 鍵
   - 暗号化されたタイムスタンプ
   - 暗号化されたタイムウィンドウ
9. NIS+ が NIS+ オブジェクトの格納されているサーバーに次の情報を渡します。
   - アクセス要求
   - 主体の DES 資格
   - ウィンドウ判定子 (暗号化済)。ウィンドウ判定子は暗号化されたウィンドウに 1 を加えたもの
10. オブジェクトのサーバーがこの情報を受信します。
11. オブジェクトのサーバーが、資格の中の Secure RPC ネット名の一部を使って、主体のホームドメインにある `cred` テーブル内にある主体の公開鍵をチェックします。
12. サーバーが主体の公開鍵とサーバーの非公開鍵を使ってもう一度共通鍵を作成します。この共通鍵は主体の非公開鍵とサーバーの公開鍵を使って作成されている共通鍵と一致する必要があります。
13. 共通鍵を使って、主体の資格の一部として受信している DES 鍵を暗号化します。
14. サーバーが、新しく復号化した DES 鍵を使って主体のタイムスタンプを復号化し、ウィンドウ判定子を使ってそれをチェックします。
15. サーバーが、復号化しチェックしたタイムスタンプと自分の内部時計とを比較し、次のプロセスを実行します。
    a. サーバーとの時間差がウィンドウ許容値を越えていた場合、要求は拒否され、プロセスが中止されて、エラーメッセージが表示されます。たとえば、タイムスタンプが 9:00 am であり、ウィンドウが 1 分であったとします。もしその要求をサーバーが受信して復号化したのが 9:01 am であれば、その要求は拒否されます。

b. タイムスタンプがウィンドウ許容値内である場合、サーバーは主体から前に受信した要求のタイムスタンプよりも大きいかチェックします。これによって、NIS+ 要求を正しい順序で処理しているか確認できます。

- 順序誤りの要求はエラーメッセージとともに拒否されます。たとえば、タイムスタンプが 9:00 am でありその主体から最後に受信した要求のタイムスタンプが 9:02 am だった場合、その要求は拒否されます。
- 最後に受信した要求と同じタイムスタンプであった場合もエラーメッセージが表示されて拒否されます。これによって、要求を 2 度処理することはなくなります。たとえば、タイムスタンプが 9:00 am であり最後にその主体から受信した要求のタイムスタンプも 9:00 am だった場合、この要求は拒否されます。

16. タイムスタンプがウィンドウ許容値内であり、その主体からの最後の要求よりも大きかった場合に、サーバーはその要求を受け入れます。
17. サーバーは次に、要求をコンパイルし、タイムスタンプをその主体から受信した最後のものとして格納し、要求に基づいて動作します。
18. 要求に対する応答としてサーバーから受信した情報が信頼できるサーバーからのものであるかを主体が確認できるようにするため、サーバーは主体の DES 鍵を使ってタイムスタンプを暗号化し、データとともに主体に送り返します。
19. 主体側では、主体の DES 鍵を使って送り返されたタイムスタンプを復号化します。

- 復号化が成功した場合は、サーバーからの情報を要求者へ戻します。
- 何らかの理由で復号化に失敗した場合は、エラーメッセージが表示されます。

DES 資格の詳細

DES 資格は次の内容で構成されます。

- 主体の「Secure RPC ネット名」。236 ページの「DES 資格の Secure RPC ネット名」を参照
- 「確認 (verification)」フィールド。237 ページの「DES 資格確認フィールド」を参照

DES 資格の Secure RPC ネット名

- 「Secure RPC ネット名」。資格のこの部分は NIS+ の主体を特定するのに使用します。Secure RPC ネット名はすべて次の 3 つの要素で構成されます。
  - 「接頭辞」。接頭辞は常に `unix` になります。

- 「識別子」。主体がクライアントユーザーの場合、ID フィールドはユーザーの UID です。主体がクライアントマシンの場合、ID フィールドはマシンのホスト名です。
- 「ドメイン名」。主体の DES 資格を含んでいるドメイン名がドメイン名になります (主体のホームドメイン)。

注 – NIS+ 主体名は常にドット (.) で終わりますが、Secure RPC ネット名はドット (.) で終わらないことに注意してください。

表 **12–1** Secure RPC ネット名のフォーマット

主体	接頭辞	識別子	ドメイン	例
ユーザー	`unix`	ユーザー ID	ユーザーのパスワードエントリと DES 資格そのもののあるドメイン	`unix.24601@sales.doc.com`
マシン	`unix`	ホスト名	ドメイン名は、マシンで `domainname` コマンドを実行すると返される	`unix.machine7@sales.doc.com`

DES 資格確認フィールド

確認フィールドは資格を確かめるために使用するフィールドです。`cred` テーブルに格納された資格情報に基づいて作成されます。

確認フィールドの構成は次のとおりです。

- 主体の非公開鍵と NIS+ サーバーの公開鍵に基づく、暗号化された主体の DES 鍵 (234 ページの「要求フェーズ - 詳細説明」参照)
- 暗号化されたタイムスタンプ
- タイムウィンドウ

DES 資格の作成方法

DES 資格を作成するには、`keylogin` コマンドを実行する必要がありますが、これは主体が資格を作成する「前」に実行します。`keylogin` コマンド (単に「`keylogin`」とする場合が多い) は、NIS+ 主体がログインする時に自動的に実行されます。図 12–2 を参照してください。

注 – 主体のログインパスワードが Secure RPC パスワードと異なる場合は、keylogin コマンドは成功しないことに注意してください。詳しくは、239 ページの「Secure RPC パスワードとログインパスワードの問題」を参照してください。

keylogin コマンドの目的は、主体が自身の非公開鍵にアクセスできるようにすることにあります。keylogin コマンドを実行すると、cred テーブルから主体の非公開鍵を取り込み (fetch)、主体の Secure RPC パスワード (非公開鍵は主体の「Secure RPC パスワード」を使って最初に暗号化される) を使ってそれを復号化し、将来の NIS+ 要求に備えてキーサーバーに格納します。

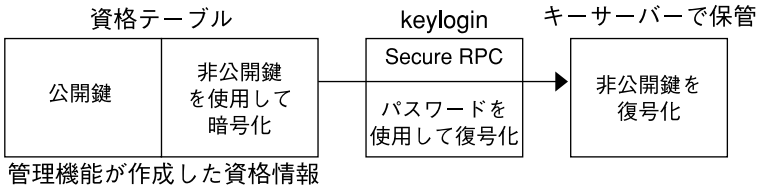


図 12–1 keylogin コマンドで主体の非公開鍵を作成

主体が DES 資格を作成するには、主体が要求を送る相手サーバーの公開鍵が必要です。この情報は主体のディレクトリオブジェクトに格納されています。主体がこの情報を獲得すれば資格の確認フィールドを作成できます。

まず、主体が種々の資格情報を暗号化するためにランダム DES 鍵を作成します。主体は自分の非公開鍵 (キーサーバーに格納されている) とサーバーの公開鍵を使って、ランダム DES 鍵を作成し暗号化するための共通鍵を作成します。次に DES 鍵で暗号化したタイムスタンプを作成し、それを確認フィールドで他の資格関連情報の中に入れます。

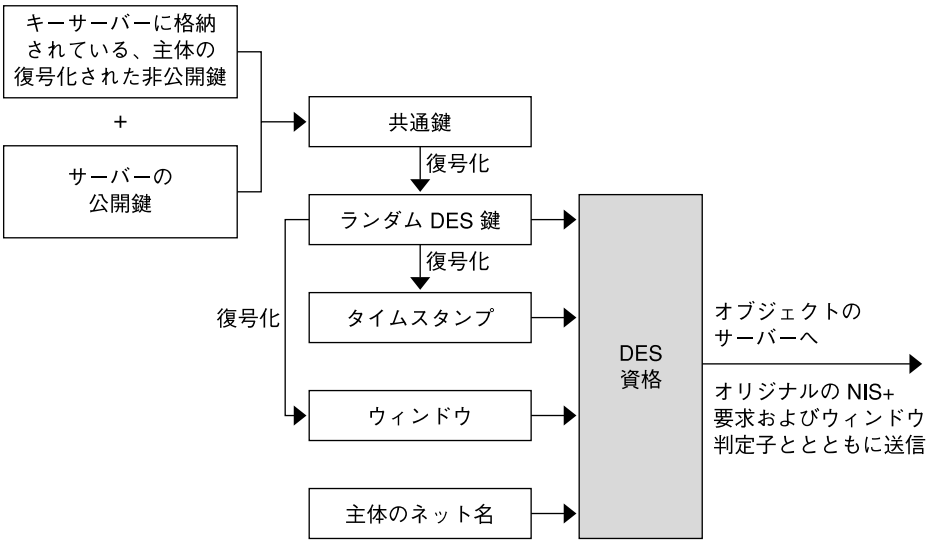


図 **12–2** DES 資格の作成

Secure RPC パスワードとログインパスワードの問題

主体のログインパスワードと Secure RPC パスワードが一致しないと、ログイン時に `keylogin` はパスワードを復号化できません。`keylogin` はデフォルトで主体のログインパスワードを使用することになっていて、また非公開鍵は主体の Secure RPC パスワードを使用して暗号化されているからです。

この場合でも主体がシステムにログインすることは可能ですが、キーサーバーがそのユーザーのための復号化された非公開鍵を持っていませんから、未認証クラスが与えられることになります。ほとんどの NIS+ 環境において、未認証クラスにはほとんどの NIS+ オブジェクトに対する作成権、削除権、変更権を与えないように設定されますから、結果としてユーザーが NIS+ オブジェクトにアクセスしても「アクセス権拒否」などのエラーになります。

注 – 「ネットワークパスワード」を「Secure RPC パスワード」の同意語として使用する場合があります。ネットワークパスワードの入力を求められたら Secure RPC パスワードを入力します。

別の承認クラスを得るには、この場合 `keylogin` プログラムを実行し、パスワード入力のプロンプトに主体の Secure RPC パスワードを入力する必要があります (254 ページの「キーログイン」の項を参照)。

しかし keylogin を実行しても、現在のログインセッション以外には無効で、一時的な解決にしかなりません。キーサーバーはこの方法によって、復号化された形でユーザーの非公開鍵を持つようになるのですが、ユーザーの cred テーブル中の非公開鍵が Secure RPC パスワード (ユーザーのログインパスワードとは異なっている) を使用して暗号化されているという点に変わりはないからです。次にログインした時には同じ問題が発生します。この問題を永久的に解決するには、ユーザーの Secure RPC パスワードではなくユーザーログイン ID を使って、cred テーブルにある非公開鍵を再度暗号化する必要があります。このためにはユーザーが chkey -p コマンドを実行します (255 ページの「NIS+ 主体の鍵の変更」の項を参照)。

Secure RPC パスワードがログインパスワードと異なる問題を永久的に解決するには、ユーザー (またはユーザーに対応している管理者) が次の手順を実行してください。

1. ログインパスワードを使用してログインします。
2. keylogin プログラムを実行して、キーサーバーに保存される非公開鍵を一時的に復号し、一時的な NIS+ アクセス権を得ます。
3. chkey -p を実行し、cred テーブル内の暗号化された非公開鍵を、ユーザーログインパスワードに基づいたものに永久的に変更します。
4. ログインセッションの終了の準備ができたら、システムをログオフする前に keylogout を実行します。
5. logout コマンドを使って、システムからログオフします。

キャッシュに保存された公開鍵の問題

正しい資格を作成し、正しいアクセス権を与えられたのに、主体の要求が拒否される場合があります。この原因のほとんどは、サーバーの公開鍵が古いバージョンだった時に作られた、古いオブジェクトが残っていることにあります。この問題は通常次の方法で解決します。

- アクセスしようとしているドメインで、nisupdkeys を実行します(nisupdkeys コマンドの使用法については 259 ページの「nisupdkeys コマンド」を、この種の問題の解決方法については 454 ページの「資格情報が古い」を参照)。
- マシン上で nis_cachemgr を終了します。次に /var/nis/NIS_SHARED_DIRCACHE を削除してから、nis_cachemgr を再起動します。

資格に関連する情報が格納されている場所

この節では、NIS+ 名前空間のどこに資格関連情報が格納されるのかを説明します。

公開鍵など、資格に関する情報は、名前空間内の様々な場所に保存されています。NIS+ は、情報を格納しているオブジェクトの生存期間に応じてこの情報を定期的に更新しますが、場合によっては、更新の間に同期が失われることがあります。その結果、システムが正常に動作しなくなることがあります。資格関連の情報を格納するすべてのオブジェクト、テーブル、ファイル、および再設定方法を、表 12–2 に示します。

表 **12–2** 資格に関連する情報が格納されている場所

項目	保存対象	リセットおよび更新の方法
cred テーブル	NIS+ 主体の秘密非公開鍵と公開鍵。これらの鍵のマスターコピーとなる	nisaddcred を使用して新しい資格を作成する。これによって既存の資格が更新される。chkey を使用しても同様のことが行える
ディレクトリオブジェクト	個々のサーバーの公開鍵のコピー	ディレクトリオブジェクトに対して /usr/lib/nis/nisupdkeys コマンドを実行する
キーサーバー	その時点でログインされている NIS+ 主体の非公開鍵	主体ユーザーに対して keylogin を実行する。あるいは、主体マシンに対して keylogin -r を実行する
NIS+ デーモン	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	rpc.nisd デーモンとキャッシュマネージャを終了し、 /var/nis から NIS_SHARED_DIRCACHE を削除する。その後両方を再起動する
ディレクトリキャッシュ	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	NIS+ キャッシュマネージャを終了した後、nis_cachemgr -i コマンドを使用して再起動する。-i オプションを指定すると、コールドスタートファイルからディレクトリキャッシュがリセットされた後、キャッシュマネージャが再起動される
コールドスタートファイル	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	ルートマスターで NIS+ デーモンを終了した後、再起動する。デーモンは、既存の NIS_COLD_START ファイルに新しい情報を再ロードする。クライアントでは、まず /var/nis から NIS_COLD_START と NIS_SHARED_DIRCACHE ファイル削除し、次にキャッシュマネージャのプロセスを終了する。その後、nisinit -c でクライアントを再び初期設定する。クライアントの信頼できるサーバーは、クライアントマシンの NIS_COLD_START ファイルに新しい情報を再ロードする

表 12–2 資格に関連する情報が格納されている場所　(続き)

項目	保存対象	リセットおよび更新の方法
passwd テーブル	ユーザーのパスワード	passwd -r nisplus コマンドを使用する。これによって、NIS+ passwd テーブル、cred テーブルの中でパスワードが更新される
passwd ファイル	ユーザーのパスワードまたはマシンのスーパーユーザーのパスワード	passwd -r nisplus コマンドを使用する。スーパーユーザーとしてログインしていても、自分のユーザー名でログインしていてもよい
passwd マップ (NIS)	ユーザーのパスワード	passwd -r nisplus コマンドを使用する

cred テーブルの詳細

主体の資格情報は「cred テーブル」に格納されています。cred テーブルは NIS+ が標準で持つ 16 のテーブルの 1 つです。cred テーブルはドメインにつき 1 つ存在し、ドメインに属するクライアントマシンおよびマシンにログインできるユーザー (つまり、ドメイン中の主体) の資格に関する情報を持っています。cred テーブルはドメイン中の org_dir サブディレクトリにあります。

注意 – cred テーブルをリンクしないでください。各 org_dir ディレクトリはそれ自身の cred テーブルを持つ必要があります。他の org_dir の cred テーブルとのリンクも使用しません。

ドメイン内のすべてのマシンにログインできるユーザーすべての LOCAL 資格情報が cred テーブルに格納されています。cred テーブルにはまた、ホームドメインとしてとしてドメインを持っているユーザーの DES 資格情報も格納されています。

cred テーブルの内容は niscat コマンドを使って見ることができます。方法については、第 19 章を参照してください。

表 12–3 に示すように、cred テーブルには 5 つの列があります。

表 **12–3** cred テーブルの資格情報

	NIS+ 主体名	認証タイプ	認証名	公開データ	非公開データ
列名	cname	auth_type	auth_name	public_data	private_data
ユーザー	完全主体名	LOCAL	UID	グループ ID リスト	
対象マシン	完全主体名	DES	Secure RPC ネット名	公開鍵	暗号化された非公開鍵

2 列目の認証の種類で他の 4 列の値の種類を判定します。

- 「LOCAL」。認証タイプが LOCAL の場合、残りの列には主体ユーザー名、UID、GID が入ります (最後の列は空白)。
- 「DES」。認証タイプが DES の場合、残りの列には主体名、Secure RPC ネット名、公開鍵、暗号化非公開鍵が入ります。これらの鍵は他の情報と組み合わされ、DES 資格の暗号化および復号化に使用されます。

資格情報の作成

資格情報を作成したり管理したりするには次の方法があります。

- Solstice AdminSuite ツールを使用します。このツールを使用すると容易に資格を管理できます。個々の資格を管理する場合はこのツールの使用を推奨します。
- nisclient スクリプトを使用します。1 つの主体の資格を容易に作成し変更できます。便利な方法なので、個々の資格を管理する方法として推奨します。本書では nisclient スクリプトの使用方法を手順ごとに示しながら、資格情報を作成していきます。
- nispopulate スクリプトを使用します。これはすでに NIS+ マップまたは /etc ファイルに資格情報を格納してある 1 つ以上の主体の資格を作成したり、変更したりする便利な方法です。NIS+ 主体グループの資格を管理する方法として推奨します。本書では nispopulate スクリプトの使用方法を手順ごとに示しながら、資格情報を作成していきます。103 ページの「NIS+ テーブルの生成 (populate)」を参照してください。
- nisaddcred コマンドを使用します。次の節では、nisaddcred コマンドを使って資格と資格情報を作成する方法を説明します。

nisaddcred コマンド

nisaddcred コマンドは資格情報を作成するコマンドです。

注 – nispopulate スクリプトと nisclient スクリプトを使って資格情報を作成することもできます。これらスクリプトは nisaddcred コマンドよりも容易に使えて効果的なものです。ネットワークが特別な機能を必要とするのでなければ、これらのスクリプトを使用してください。

---

nisaddcred コマンドは、LOCAL 資格と DES 資格情報の作成、更新、および削除を行います。資格情報を作成するには、適切なドメインの cred テーブルに対する作成権が必要です。資格を更新するには、cred テーブル、または少なくとも cred テーブルの特定エントリに対する変更権が必要です。資格を削除するには、cred テーブル、または cred テーブルのエントリに対する削除権が必要です。

- 別の NIS+ 主体資格を作成または更新する場合

  LOCAL 資格情報

  ```
  nisaddcred -p uid -P principal-name local
  ```

  DES 資格情報

  ```
  nisaddcred -p rpc-netname -P principal-name des
  ```

- 自分の資格を更新する場合

  LOCAL 資格情報

  ```
  nisaddcred -local
  ```

  DES 資格

  ```
  nisaddcred des
  ```

- 資格を削除する場合

  ```
  nisaddcred -r principal-name
  ```

関連コマンド

この章で説明する nisaddcred コマンドに加えて、資格に関して有用な情報を提供するコマンドが 2 つ用意されています。

表 **12–4** その他の資格関連コマンド

コマンド	説明	参照する項目
niscat -o	ディレクトリの属性を一覧表示する。ディレクトリのサーバーの公開鍵フィールドをチェックすることで、ディレクトリオブジェクトに公開鍵が格納されているかわかる	334 ページの「ディレクトリのオブジェクト属性を表示する」

表 12–4 その他の資格関連コマンド　(続き)

コマンド	説明	参照する項目
nismatch-	cred テーブル上で実行すると主体の資格情報を表示する	373 ページの「nismatch と nisgrep コマンド」

nisaddcred コマンドを使って資格情報を作成する方法

nisaddcred コマンドを使用して、LOCAL および DES 資格情報を作成します。

LOCAL 資格情報

LOCAL 資格情報を作成するために nisaddcred コマンドを使用すると、nisaddcred コマンドは主体のログインレコードから主体ユーザーの UID (および GID) を抽出し、ドメインの cred テーブルに置きます。

DES 資格情報

DES 資格情報を作成するのに使用した場合、nisaddcred は 2 つのプロセスを実行します。

1. 主体の Secure RPC ネット名を作成します。Secure RPC ネット名は、パスワードレコードから取得した主体のユーザー ID 番号とドメイン名 (たとえば、unix.1050@doc.com) を結合して作成されます。
2. 主体の非公開鍵と公開鍵を生成します。

nisaddcred が非公開鍵を暗号化するには、Secure RPC パスワードが必要です。nisaddcred コマンドを引数 -des で呼び出すと、主体の Secure RPC パスワードの入力を要求されます。通常このパスワードは主体のログインパスワードと同じです (異なる場合は、239 ページの「Secure RPC パスワードとログインパスワードの問題」に示す手順に従ってさらに操作が必要となります)。

nisaddcred コマンドは 1 対の乱数 (Difie-Hellman 方式を使った 192 ビットの認証鍵) を作成します。この鍵は Difie-Hellman の鍵ペア (key-pair) または省略して単に「鍵ペア」と呼びます。

この鍵ペアの一方が「非公開鍵」であり、もう一方が「公開鍵」になります。公開鍵は cred テーブルの公開データフィールドに置かれます。非公開鍵は主体の Secure RPC パスワードで暗号化された後に非公開データに置かれます。

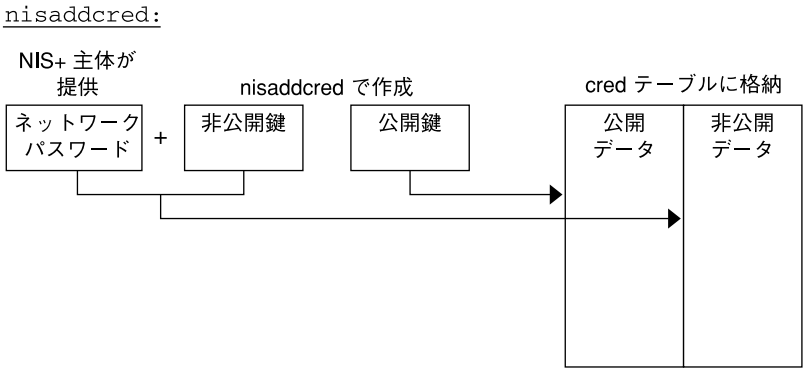


図 **12–3** nisaddcred コマンドを使って主体の鍵を作成する方法

デフォルトでは、cred テーブルをすべての NIS+ 主体 (未認証主体であっても) が読み取ることができるので、セキュリティ上の予防措置として、主体の非公開鍵を暗号化しています。

Secure RPC ネット名と NIS+ 主体名

資格情報を作成する際、何度も主体の「RPC ネット名 (rpc-netname)」と「主体名 (principal-name)」を入力しなければなりません。どちらも独自の構文が必要です。

- 「Secure RPC ネット名」Secure RPC ネット名は Secure RPC プロトコルで判定されますから、NIS+ のネーミング規則には従いません。
  - ユーザーの場合： unix. *uid@domain*
  - マシンの場合： unix *.hostname@domain*

  Secure RPC ネット名がユーザーを識別する場合、ユーザーの UID が必要です。Secure RPC ネット名がマシンを識別する場合、マシンのホスト名が必要です。nisaddcred コマンドに指定するときは、Secure RPC ネット名の前に -p (小文字) フラグを入力してください。

  Secure RPC ネット名は常に unix (すべて小文字) で始まりドメイン名で終わります。Secure RPC プロトコルに従うため、ドメイン名にはドットを付けません。
- 「主体名」NIS+ 主体名は通常の NIS+ 命名規約に従いますが、常に完全指定名でなければなりません。構文は、*principal.domain* になります。

識別する対象がクライアントユーザーであっても、クライアントマシンであっても、主体名で始まり、その後にドットと完全なドメイン名が続き、最後はドットで終わります。資格の作成に使用する場合、常に先頭は -P (大文字) フラグで始まります。資格の削除に使用する場合、-P フラグは使用しません。

管理者のために資格情報を作成する方法

名前空間を最初に設定する場合、ドメインをサポートする管理者の資格情報を最初に作成します。管理者の資格情報を一度作成すると、他の管理者、クライアントマシン、およびクライアントユーザーの資格情報を作成できます。

自分の資格情報を作成しようとすると、堂々めぐりに陥ります。つまり、ドメインの cred テーブルに対する作成権がなければ自分の資格情報を作成できず、もし NIS+ 環境が適切に設定されていれば、資格を持つまではそのような権限を持つこともできません。この循環から抜け出さなくてはなりません。これには、次の 2 つの方法があります。

- ドメインのマスターサーバーにスーパーユーザーとしてログインしている間に、資格情報を作成する
- ダミーパスワードを使用して、他の管理者に自分の資格を作成してもらってから chkey コマンドを使用してパスワードを変更する

どちらの場合も、別の NIS+ 主体に資格情報を作成してもらうことになります。自分の資格情報を作成するには、247 ページの「NIS+ 主体の資格情報を作成する方法」の項を参照してください。

NIS+ 主体の資格情報を作成する方法

NIS+ の資格情報はドメインの設定後いつでも作成できます。すなわち、cred テーブルがあれば作成できます。

NIS+ 主体の資格情報を作成する

- 主体のホームドメインの cred テーブルに対する作成権が必要
- サーバーは、主体の存在を認識している必要がある。この場合「サーバーが認識している」とは以下のことを意味する
  - 主体がユーザーの場合、ドメインの NIS+ passwd テーブルかサーバーの /etc/passwd ファイルのどちらかにエントリが必要
  - 主体がマシン場合、ドメインの NIS+ host テーブルかサーバーの /etc/hosts ファイルのどちらかにエントリが必要

これらの条件を満たせば、-p と -P の両方のオプション付きで nisaddcred コマンドを実行できます。

LOCAL 資格情報

```
nisaddcred -p uid -P principal-name local
```

DES 資格情報

```
nisaddcred -p rpc.netname -P principal-name des
```

次の原則に注意してください。

- 主体ユーザーの場合、LOCAL 資格と DES 資格の両方の情報を作成できます。
- 主体マシンの場合、DES 資格情報しか作成できません。
- 主体のホームドメイン (ユーザーまたはマシン) でだけ DES 資格情報を作成できます。
- ユーザーの LOCAL 資格情報はユーザーのホームドメインでも他のドメインでも作成できます。

ユーザー主体 - 例

UID が 11177 の morena という名前の NIS+ ユーザーの LOCAL および DES 資格情報を作成する例を次に示します。彼女の所属するドメインは doc.com. で、 この例では、このドメインの主体マシンから彼女の資格情報を入力します。

```
client# nisaddcred -p 11177 -P morena.doc.com. local
client# nisaddcred -p unix.11177@sales.doc.com \
   -P morena.doc.com. des
Adding key pair for unix.11177@sales.doc.com
   (morena.doc.com.).
Enter login password:
```

Enter login password: プロンプトに対する正しい応答は morena のログインパスワードです。彼女のログインパスワードを知らない場合は、ダミーパスワードを使用します。ダミーパスワードは、次の例のように、後で chkey コマンドを使って変更できます。

ダミーパスワードと chkey の使い方 - 例

ユーザーのログインパスワードを知らない場合、次に説明するようにダミーパスワードを使うことができます。

表 12–5 は、ダミーパスワードを使って資格情報を作成した管理者が、chkey コマンドを使ってパスワードを変更する方法を示します。この例では、UID が 119 の eiji という名前の管理者の資格情報を作成します。eiji はルートドメインに属しているので、rootmaster という名前のルートマスターサーバーから彼の資格情報を入力します。

表 **12–5** 管理者の資格情報を作成する作業 : コマンドのまとめ

作業	コマンド
eiji の LOCAL 資格情報を作成する	rootmaster# nisaddcred -p 119 -P eiji.doc.com. local

表 **12–5** 管理者の資格情報を作成する作業 : コマンドのまとめ　(続き)

作業	コマンド
eiji の DES 資格情報を作成する	rootmaster# nisaddcred -p unix.119@doc.com -P eiji.doc.com. des Adding key pair for unix.119@doc.com (eiji.doc.com.).
eiji のダミーパスワードを入力する	Enter eiji's login password: nisaddcred: WARNING: password differs from login passwd
ダミーパスワードを再度入力する	Retype password:
使用したダミーパスワードを eiji に伝える	
eiji がルートマスターにログイン	rootmaster% login:**eiji**
eiji が本当のログインパスワードを入力する	Password:
eiji はエラーメッセージを受けたが、ログインは許可される	Password does not decrypt secret key for unix.119@doc.com.
eiji がログインを実行	rootmaster% **keylogin**
eiji がダミーパスワードを入力	Password:*dummy-password*
eiji が chkey を実行	rootmaster% chkey -p Updating nisplus publickey database Generating new key for'unix.119@doc.com'.
eiji が本当のログインパスワードを入力	Enter login password:
eiji が本当のログインパスワードを再入力	Retype password: Done.

まず、通常の方法で eiji の資格情報を作成しますが、ダミーのログインパスワードを使用します。NIS+ は警告を発して、再入力を要求します。再入力を行うと、この操作は完了します。ドメインの cred テーブルには、ダミーパスワードに基づく eiji の資格情報が入っています。しかし、ドメインの passwd テーブル (または /etc/passwd ファイル) には、まだ彼のパスワードエントリが残っているので、彼はシステムにログオンできます。

次に、eiji は本当のログインパスワードを入力して、ドメインのマスターサーバーにログインします (ログイン手順では、passwd テーブルまたは /etc/passwd ファイルのパスワードをチェックするため)。そこから eiji は、まずダミーパスワードを使用して keylogin を実行し (cred テーブルをチェックするため)、chkey -p コマンドを使用して cred エントリを本当のパスワードに変更します。

別のドメインでの作成 - 例

これらの 2 つの例では、主体ユーザーのホームドメインのマスターサーバーにログインしている間に、主体ユーザーの資格情報を作成しました。しかし、適切なアクセス権がある場合、別のドメインに資格を作成できます。次の構文でドメイン名を付加するだけです。

LOCAL 資格情報

`nisaddcred -p` *uid* `-P` *principal-name* `local` *domain-name*

DES 資格情報

`nisaddcred -p` *rpc-netname* `-P` *principal-name* `des` *domain-name*

次の例では、まず chou という名前のシステム管理者の LOCAL および DES 資格情報を、そのシステム管理者のホームドメイン (これは、たまたまルートドメイン) に作成し、次にその LOCAL 資格を doc.com ドメインに追加します。chou の UID は 11155 です。このコマンドは、ルートマスターサーバーから入力します。簡単にするため、chou の正しいログインパスワードを入力しているものとします。

```
rootmaster# nisaddcred -p 11155 -P chou.doc.com. local
rootmaster# nisaddcred -p unix.11155@doc.com -P chou.doc.com. des
Adding key pair for unix.11155@doc.com (chou.doc.com.).
Enter login password:
rootmaster#  nisaddcred -p 11155 -P chou.doc.com. local doc.com.
```

LOCAL 資格情報は、UID を NIS+ 主体名にマップします。クライアントユーザーである NIS+ 主体は、さまざまなドメインにさまざまな UID を持てますが、NIS+ 主体名は 1 つしか持てません。したがって、chou などの NIS+ 主体が自分のホームドメイン以外のドメインからログインする場合、そのドメインにパスワードエントリを持つだけではなく、そのドメインの cred テーブルに LOCAL 資格も持っていなければなりません。

マシン の場合—例

主体「マシン」の資格情報を作成する例を次に示します。このホスト名は starshine1 で、ルートドメインに所属しています。したがって、この資格はルートマスターサーバーから作成されます。この例では、ルートマスターに root としてログインしている間に資格情報を作成します。しかし、有効な資格情報と適切なアクセス権をすでに持っている場合、自分のユーザー名でログインしているときに、資格を作成できます。

```
rootmaster# nisaddcred -p unix.starshine1@doc.com -P \
starshine1.doc.com. des
Adding key pair for unix.starshine1@doc.com
(starshine1.doc.com.).
Enter starshine1.doc.com.'s root login password:
Retype password:
```

パスワードプロンプトに対しては、主体マシンのスーパーユーザーパスワードを入力してください。もちろん、ダミーパスワードを使用することもできます。このダミーパスワードは、その主体マシンにスーパーユーザーとしてログインすれば、後で変更できます。

NIS+ 資格情報の管理

以下のいくつかの節では、nisaddcred コマンドで資格情報を管理する方法について説明します。nisaddcred で資格情報を管理するには、cred テーブルに対する作成権、変更権、読み取り権、削除権が必要です。

自分の資格情報を更新する方法

自分の資格情報を更新することは、資格を作成することよりはるかに簡単です。自分のユーザー名でログインしているときに、次のような nisaddcred コマンドを入力するだけです。

```
# nisaddcred des
# nisaddcred local
```

別のユーザーの資格情報を更新する場合は、そのユーザーの資格情報を作成するのと同じ操作を行います。

資格情報を削除する方法

nisaddcred コマンドは、主体の資格情報を削除しますが、コマンドを実行しているローカルドメインだけから削除できます。

システム全体から完全に主体を削除するには、主体のホームドメインおよび LOCAL 資格情報のあるすべてのドメインから主体の資格情報を明示的に削除しなければなりません。

資格情報を削除するには、ローカルドメインの cred テーブルへの変更権が必要です。-r オプションを使い、完全 NIS+ 主体名で主体を指定します。

```
# nisaddcred -r principal-name
```

次の 2 つの例では、システム管理者 morena.doc.com. の LOCAL と DES の資格情報を削除します。最初の例では、システム管理者のホームドメイン (doc.com.) から両方の資格情報を削除し、2 番目の例では、sales.doc.com. ドメインから LOCAL 資格情報を削除します。該当するドメインのマスターサーバーから、それぞれがどう入力されるかに注意してください。

```
rootmaster# nisaddcred -r morena.doc.com.
salesmaster# nisaddcred -r morena.doc.com.
```

資格が本当に削除されたことを確かめるには、次に示すように、cred テーブルに対して nismatch を実行します。nismatch の詳細については、第 19 章を参照してください。

```
rootmaster# nismatch morena.doc.com. cred.org_dir
salesmaster# nismatch morena.doc.com. cred.org_dir
```

第 13 章

NIS+ 鍵の管理

この章では、NIS+ の鍵と、鍵を管理する方法について説明します。

注 – NIS+ セキュリティタスクには、Solstice AdminSuite ツールがあればより簡単に実行できるものがあります。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ 鍵

NIS+ 鍵は NIS+ の関連情報を暗号化するために使用します。

この章では、NIS+ セキュリティシステム全体と、特にシステムで鍵が果たす役割について、十分理解しているユーザーを想定しています (詳しくは、第 11 章を参照してください)。

NIS+ 鍵に関するコマンドとその構文やオプションについての詳しい説明は、nis+(1) のマニュアルページを参照してください。また、`nisaddcred` コマンドも鍵に関連のある働きをします。詳細については、 第 12 章を参照してください。

キーログイン

主体がログインする時、パスワードの入力を求めるプロンプトが現れます。このパスワードはユーザーがセキュリティゲートをパスし、ネットワークにアクセスするために必要なものです。ログインプロセスは、ユーザーのホームドメインの `cred` テーブルにあるユーザーの非公開鍵を復号化しキーサーバーへ渡します。キーサーバーはその復号化された非公開鍵を使って、ユーザーが NIS+ オブジェクトにアクセスするつど、ユーザーの認証を行います。

通常は、この場合だけ主体にパスワードが要求されます。しかし、`cred` テーブルにある主体の非公開鍵がユーザーのログインパスワードと異なるパスワードを使用して暗号化されていると、ログインプロセスはそれを `login` の際にログインパスワードを使って復号化できないので、キーサーバーに復号化した非公開鍵を提供できません。(これは `cred` テーブル内にあるユーザーの非公開鍵が、ログインパスワードと異なる Secure-RPC パスワードを使用して暗号化された場合に多く起こります。)

注 – 「ネットワークパスワード」と「Secure-RPC パスワード」を同意語として使用する場合があります。

この問題を一時的に解決するには、ログインの後には必ず主体が `keylogin` コマンドを使用して、キーログインを実行する必要があります。`-r` フラグを使って、スーパーユーザー主体にログインし、ホストの `/etc/.rootkey` にスーパーユーザーの鍵を格納します。

主体ユーザーの場合

```
keylogin
```

主体マシンの場合 (一度だけ)

```
keylogin -r
```

ただし、オリジナルのパスワードを使用して確実にキーログインを実行しても、そのログインセッションにおいて一時的に問題が解決するだけです。`cred` テーブルの非公開鍵はユーザーのログインパスワードと異なるパスワードを用いて暗号化されたままであり、次にユーザーがログインした時に同じ問題が起こります。この問題を永久的に解決するには、`chkey` コマンドを使用して、非公開鍵の暗号化に使ったパスワードをユーザーのログインパスワードに変更します (255 ページの「NIS+ 主体の鍵の変更」を参照)。

NIS+ 主体の鍵の変更

chkey コマンドは cred テーブルに格納されている NIS+ 主体の公開および非公開鍵を変更します。このコマンドは passwd テーブル内もしくは /etc/passwd ファイル内のどちらのエントリにも影響を与えません。

chkey コマンドについて

- 新規に鍵を作成し、パスワードを用いて非公開鍵を暗号化します。-p オプションをつけて実行した場合、chkey コマンドは既存の非公開鍵を新規のパスワードで再暗号化します。
- 新規に Diffie-Hellman の鍵ペアを作成し、提供されたパスワードを使用して非公開鍵を暗号化します (各主体に複数の Diffie-Hellman の鍵のペアが存在することは可能) 。しかしほとんどの場合、ユーザーは新規の鍵ペアを求めず、既存の非公開鍵を新規のパスワードを使って再暗号化しようとします。これを行うには、-p オプション付きで chkey を実行します。

詳細については、マニュアルを参照してください。

注 – NIS+ 環境では、どんな管理ツールまたは passwd (あるいは nispasswd) コマンドを用いてログインパスワードを変更しても、cred テーブル内の非公開鍵は新規のパスワードを使用して自動的に再暗号化されます。従って、ログインパスワードの変更後に chkey コマンドを実行する必要はありません。

---

chkey コマンドはキーサーバー、cred テーブル、および passwd テーブルとの関連で動作します。chkey を実行するには次のようにします。

- ホームドメインの passwd テーブルにエントリが必要です。この条件を満たさなければ、エラーメッセージが返されます。
- キーサーバーに復号化された非公開鍵のあることを確認するために、keylogin を実行する必要があります。
- cred テーブルの変更権を持っていなければなりません。この権限がない場合は「permission denied」などのエラーメッセージが返されます。
- cred テーブル内の非公開鍵の暗号化に使われた、オリジナルのパスワードを知らなくてはなりません (ほとんどの場合、これは Secure RPC パスワード)。

ログインパスワードを使って非公開鍵を再暗号化するために chkey コマンドを使用するには、最初にオリジナルのパスワードを用いて keylogin を実行し、次に表 13–1 に示す手順で chkey -p を実行します。表 13–1 は、主体ユーザーとして keylogin と chkey を実行する方法を示しています。

表 **13–1** 非公開鍵の再暗号化：コマンドの説明

作業	コマンド
ログインする	`Sirius% login` *Login-name*
ログインパスワードを入力する	`Password:`
ログインパスワードと Secure RPC パスワードが異なっている場合、`keylogin` を実行する	`Sirius% keylogin`
非公開鍵の暗号化に使用したオリジナルパスワードを入力する	`Password:` *Secure RPC password*
`chkey` を実行する	Sirius% `chkey -p` Updating nisplus publickey database Updating new key for 'unix.1199@Doc.com'.
ログインパスワードを入力する	`Enter login password:` *login-password*
ログインパスワードを再入力する	`Retype password:`

鍵の変更

次の節では、NIS+ 主体の鍵の変更方法を説明します。

注 – サーバーの鍵を変更するときは常に、ドメイン内の全クライアントの鍵情報も更新する必要があります。その方法は、262 ページの「クライアントの鍵情報の更新」で説明します。

ルートからのルート鍵の変更

ルートマスターから (root として) ルートマスターサーバーの鍵を変更するには、表 13–2 の手順を実行します。

表 **13–2** ルートマスター鍵の変更：コマンドの説明

作業	コマンド
新規 DES 資格を作成	`rootmaster# nisaddcred des`
`rpc.nisd` のプロセス ID を発見	`rootmaster# ps -e | grep rpc.nisd`
NIS+ デーモンを終了	`rootmaster# kill` *pid*

表 13–2 ルートマスター鍵の変更：コマンドの説明　(続き)

作業	コマンド
セキュリティなしで NIS+ デーモンを再起動	`rootmaster# rpc.nisd -S0`
`keylogout` を実行 (以前の `keylogin` はタイムアウト)	`rootmaster# keylogout -f`
マスターがディレクトリに保管していた鍵を更新	`rootmaster# nisupdkeys` *dirs*
`rpc.nisd` のプロセス ID を発見	`rootmaster# ps -e | grep rpc.nisd`
NIS+ デーモンを終了	`rootmaster# kill` *pid*
デフォルトセキュリティで NIS+ デーモンを再起動	`rootmaster# rpc.nisd`
`keylogin` を実行	`rootmaster# keylogin`

引数の意味はそれぞれ以下のとおりです。

- *pid* は `ps -e | grep rpc.nisd` コマンドで通知されるプロセス ID 番号
- *dirs* は更新するディレクトリオブジェクト (`rootmaster` によって保管されたディレクトリオブジェクト)

表 13–2 に示すプロセスの最初の手順では、`nisaddcred` がルートマスターの `cred` テーブルを更新し、`/etc/.rootkey` を更新し、ルートマスターのキーログインを実行します。この時点では、マスターに保管されたディレクトリオブジェクトが更新されておらず、その資格情報とルートマスターとは同期がとれていません。表 13–2 に示すその後の手順は、すべてのオブジェクトを更新するのに必要です。

注 – サーバーの鍵を変更するときは常に、ドメイン内の全クライアントの鍵情報も更新する必要があります。方法については、262 ページの「クライアントの鍵情報の更新」で説明します。

別のマシンからルート鍵を変更する手順

別のマシンからルートマスターの鍵を変更する場合、それに必要な NIS+ 資格とそれを行う承認を得ていなくてはなりません。

表 13–3 別のマシンによるルートマスターの鍵の変更 - コマンドの説明

作業	コマンド
新規の DES 資格を作成	`othermachine% nisaddcred -p` *principal* `-P` *nisprincipal* `des`
ディレクトリオブジェクトを更新	`othermachine% nisupdkeys` *dirs*

表 13–3 別のマシンによるルートマスターの鍵の変更 - コマンドの説明　(続き)

作業	コマンド
/etc.rootkey を更新	othermachine% keylogin -r
クライアントとして他のマシンを再度初期化	othermachine% nisinit -cH

引数の意味はそれぞれ以下のとおりです。

- *principal* はルートマシンの Secure RPC ネット名。たとえば、unix.rootmaster@doc.com (ドットなし)
- *nis-principal* はルートマシンの NIS+ 主体名。たとえば、rootmaster.doc.com. (ドットで終わる)
- *dirs* は更新するディレクトリオブジェクト (rootmaster で保管されたディレクトリオブジェクト)

nisupdkeys を実行する場合、関連したディレクトリオブジェクトのすべてを同時に更新するように注意します。すなわち、すべてを 1 つのコマンドで行います。分割して更新すると、認証エラーになります。

注 – サーバーの鍵を変更するときは常に、ドメイン内の全クライアントの鍵情報も更新する必要があります。方法については、262 ページの「クライアントの鍵情報の更新」で説明します。

複製からルート複製の鍵を変更する手順

複製からルート複製の鍵を変更する手順を次に示します。

```
replica# nisaddcred des
replica# nisupdkeys dirs
```

引数の意味はそれぞれ以下のとおりです。

- *dirs* は更新するディレクトリオブジェクト (replica で保管されたディレクトリオブジェクト)

nisupdkeys を実行する場合、関連したディレクトリオブジェクトのすべてを同時に更新するように注意します。すなわち、すべてを 1 つのコマンドで行います。分割して更新すると、認証エラーになります。

注 – サーバーの鍵を変更するときは常に、ドメイン内の全クライアントの鍵情報も更新する必要があります。方法については、262 ページの「クライアントの鍵情報の更新」で説明します。

ルート以外のサーバーの鍵の変更手順

サーバーからルート以外のサーバー (マスターまたは複製) の鍵を変更する手順を以下に示します。

```
subreplica# nisaddcred des
subreplica# nisupdkeys parentdir dirs
```

引数の意味はそれぞれ以下のとおりです。

- *parentdir* はルート以外のサーバーの親ディレクトリ (subreplica の NIS+ サーバーを持つディレクトリ)
- *dirs* は更新しようとするディレクトリオブジェクト (subreplica で保管されたディレクトリオブジェクト)

nisupdkeys を実行する場合、関連したディレクトリオブジェクトのすべてを同時に更新するように注意します。すなわち、すべてを 1 つのコマンドで行います。分割して更新すると、認証エラーになります。

注 – サーバーの鍵を変更するときは常に、ドメイン内の全クライアントの鍵情報も更新する必要があります。方法については、262 ページの「クライアントの鍵情報の更新」で説明します。

公開鍵の更新

NIS+ サーバーの公開鍵は名前空間のあちこちに格納されています。サーバーに新規の資格情報を作成する場合、新規の鍵ペアが作成され cred テーブルに格納されます。しかし、名前空間ディレクトリオブジェクトには、まだサーバーの古い公開鍵のコピーが残っています。nisupdkeys コマンドを使用して、これらのディレクトリオブジェクトのコピーを更新します。

nisupdkeys コマンド

古い鍵ペアの保全が危うくなったり、非公開鍵の暗号化に使ったパスワードを忘れてしまったりして新規の鍵ペアを作成する場合、nisupdkeys を使用してディレクトリオブジェクト内の古い公開鍵を更新できます。

nisupdkeys コマンドで次の操作を行うことができます。

- 1 台の特定サーバーの鍵を更新する
- NIS+ ディレクトリオブジェクトをサポートしているサーバーすべての鍵を更新する

- サーバーの公開鍵をディレクトリオブジェクトから削除する
- サーバーの IP アドレスが変更された場合にそれを更新する

ただし、nisupdkeys は主体マシン上の NIS_COLD_START ファイルを更新できません。サーバーの鍵のコピーを更新するには、NIS+ クライアントが nisclient コマンドを実行しなければなりません。あるいは、NIS+ キャッシュマネージャを実行中でかつコールドスタートファイル内で 1 つ以上のサーバーを利用できる場合、主体はディレクトリオブジェクト上で生存期間がタイムアウトするまで待つことができます。タイムアウトが発生すると、キャッシュマネージャはコールドスタートファイルを自動的に更新します。生存期間のデフォルトは 12 時間です。

nisupdkeys を使うには、NIS+ ディレクトリオブジェクトへの変更権が必要です。

公開鍵の引数の更新と例

nisupdkeys コマンドは /usr/lib/nis にあります。nisupdkeys コマンドは次の引数を使います。nisupdkeys コマンドの詳細と引数すべてのリストは、nisupdkeys(1M) のマニュアルページを参照してください。

表 **13–4** nisupdkeys の引数

引数	用途
(引数なし)	カレントドメインのサーバーの鍵をすべて更新する
directoryname	ディレクトリ名で指定したディレクトリオブジェクトの鍵を更新する
-H *servername*	カレントドメインのディレクトリオブジェクト内のサーバー名で指定したサーバーの鍵を更新する。他のドメインにあるサーバーの鍵を更新する場合は、完全ホスト名を使用する
-s - H *servername*	サーバー名で指定したサーバーで保管されたディレクトリオブジェクトすべての鍵を更新する
-C	鍵をクリアする

表 13–5 で公開鍵の更新手順の例を示します。

表 **13–5** 公開鍵の更新: コマンド例

作業	コマンド
カレントドメイン (doc.com) のすべてのサーバーのすべての鍵を更新	rootmaster# /usr/lib/nis/nisupdkeys Fetch Public key for server rootmaster.doc.com. netname='unix.rootmaster@doc.com' Updating rootmaster.doc.com.'s public key. Public key: *public-key*
sales.doc.com ドメインのディレクトリオブジェクトをサポートしているすべてのサーバーの鍵を更新	salesmaster# nisupdkeys sales.doc.com (画面上には何も表示されません)
すべてのディレクトリ内のサーバー名が master7 であるサーバーの鍵を更新	rootmaster# nisupdkeys -H master7
sales.doc.com ディレクトリオブジェクトで保管された鍵をクリア	rootmaster# nisupdkeys -C sales.doc.com
カレントドメインのディレクトリオブジェクトのサーバー名が master7 であるサーバーの鍵をクリア	rootmaster# nisupdkeys -C -H master7

IP アドレスの更新

サーバーの IP アドレスを変更するかアドレスを追加する場合、NIS+ アドレス情報を更新するために nisupdkeys を実行する必要があります。

1 つ以上のサーバーの IP アドレスを更新するには、-a オプション付きで nisupdkeys を使用します。

指定されたドメインのサーバーの IP アドレスを更新。

```
rootmaster# nisupdkeys -a domain
```

特定サーバーの IP アドレスを更新。

```
rootmaster# nisupdkeys -a -H server
```

クライアントの鍵情報の更新

サーバーの鍵を変更するときは、常に全クライアントの鍵情報も更新する必要があります。NIS+ のサーバーは NIS+ のクライアントでもあります。そのため、あるサーバーの鍵を更新した場合は、それが NIS+ のサーバーであるか通常のクライアントであるかにかかわらず、ドメイン内にあるほかのすべてのマシンの鍵情報を更新する必要があります。

クライアントの鍵情報を更新するには、以下の 3 通りの方法があります。

- クライアント鍵情報は、クライアントの `nisclient` スクリプトを実行すれば、最も簡単に作成できます。
- 別の方法で個々のクライアントの鍵情報を更新するには、クライアント側で `nisinit` コマンドを実行します。344 ページの「クライアントを初期設定する」を参照してください。
- ドメイン内にある全てのマシンのクライアントの鍵情報を一括で更新するには、ドメインのディレクトリオブジェクトの生存期間の値を短くします。262 ページの「クライアントの鍵情報を一括で更新する」を参照してください。

クライアントの鍵情報を一括で更新する

サーバーの鍵を変更したあとで、ドメイン内にあるすべてのマシンのクライアントの鍵情報を一括更新できます。

1. `nischttl` コマンドを使用して、ドメインのディレクトリオブジェクトの生存期間 **(TTL)** を、すぐに満了するような小さな値にしてください。

   たとえば、`sales.doc.com.` ドメイン内のサーバーの鍵を変更した場合、ディレクトリの TTL 値を 1 分にするには、以下のように入力します。

   ```
   client% nischttl 60 sales.doc.com.
   ```

2. ディレクトリの **TTL** 値が満了すると、キャッシュマネージャはエントリを終了し、クライアントのために新しく更新された情報を入手します。

3. ディレクトリオブジェクトの **TTL** 値が満了したあとで、ディレクトリオブジェクトの **TTL** 値をデフォルト値に戻します。

   たとえば、`sales.doc.com.` ドメインのディレクトリオブジェクトの TTL 値を 12 時間に戻すには、以下のように入力します。

   ```
   client% nischttl 12h sales.doc.com.
   ```

   TTL 値の使用の詳細は、351 ページの「`nischttl` コマンド」を参照してください。

第 14 章

拡張セキュリティ資格の管理

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については
、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

Diffie-Hellman 拡張鍵

NIS+ は、RPCSEC_GSS `RPC(3N)` セキュリティフレイバーを使用して、192 ビット Diffie-Hellman [`RPC(3N)` セキュリティフレイバー AUTH_DES] を超える `RPC(3N)` 層におけるより厳格なセキュリティを提供します。システム上で使用可能なセキュリティメカニズムのリストについては、`nisauthconf(1M)` コマンドを参照してください。また、これらのセキュリティメカニズムは、より厳格な暗号と各 NIS+ トランザクションの完全性を提供します。すなわち、各 NIS+ トランザクションのデータが変更されていないことが検証されます。

システム管理者は、後述のガイドラインに従って、NIS+ サーバー環境を構築する前または後に `nisauthconf(1M)` を実行することによって、より厳格なセキュリティメカニズムの利点を活用することができます。

新しい公開鍵ベースのセキュリティメカニズムへの移行

Diffie-Hellman 640 ビット (dh640-0) のような公開鍵暗号ファミリーのより厳格なセキュリティメカニズムを使用するには、既存の `cred` テーブルに各主体の新しい資格を追加することが必要です。後述の手順は、現在 Diffie-Hellman 192 ビット (RPC セキュリティフレイバー AUTH_DES) セキュリティを実行しているシステムを、Diffie-Hellman 640 ビット (RPC セキュリティフレイバー RPCSEC_GSS) セキュリティを実行するように変換する場合のものです。この移行手順は、最もよくある場合を説明したものですが、公開鍵暗号ファミリーのどれか 1 つのセキュリティメカニズムタイプから、公開鍵暗号ファミリーの別のセキュリティメカニズムに変換する場合、原理は同じです。

注 – 以下の例では、$PATH に `/usr/lib/nis` があることを前提にしています。

NIS+ セキュリティメカニズムの構成

NIS+ セキュリティの構成は、`nisauthconf(1M)` コマンドを使用して行います。`nisauthconf` はセキュリティメカニズムのリストを設定した順に取ります。セキュリティメカニズムは、`secure_rpc(3N)` で指定された 1 つまたは複数の認証フレイバーを使用することができます。**`des`** が唯一の指定されたメカニズムである場合には、NIS+ は他の NIS+ クライアントおよびサーバーの認証に AUTH_DES だけを使用します。NIS+ は、`nisaddcred(1M)` の場合を除いて、des より後のメカニズムは無視します。

```
nisauthconf [-v] [mechanism, ...]
```

この場合の `mechanism` は、システム上で使用可能な RPC セキュリティメカニズムです。使用可能なメカニズムのリストについては、`nisauthconf(1M)` を参照してください。メカニズムを指定しないと、現在構成されているメカニズムのリストが出力されます。

新しいセキュリティメカニズム資格の作成

NIS+ ユーザーおよびホスト主体ごとに、新しいメカニズムの資格情報を作成する必要があります。これを行うには、システムが現在のメカニズムによる認証を継続している間に、NIS+ を実行しているマシンの 1 つで `nisauthconf(1M)` コマンドを実行して新しい資格を作成しなければなりません。資格作成の基本の詳細については、247 ページの「NIS+ 主体の資格情報を作成する方法」も参照してください。

新しいセキュリティメカニズム資格 - 例

des から dh640-0 へ変換します。`nisauthconf` がスーパーユーザーとして実行されており、`nisaddcred` が、ホームディレクトリの中で作成権を持っているどれかの主体として実行されていることが必要です。サーバー名は `server1`、ユーザーの主体名は `morena` です。ユーザー `morena` の UID は 11177 です。

```
client# nisauthconf des dh640-0
client% nisaddcred -P server1.doc.com. -p unix.server1@doc.com dh640-0
            (画面上には何も表示されない)
client% nisaddcred -P morena.doc.com. -p unix.11177@doc.com -ldummy-password    dh640-0
            (画面上には何も表示されない)
```

NIS+ ディレクトリオブジェクトへの新しい鍵の追加

すべてのサーバーの新しい資格が生成されたら、`nisupdkeys(1M)` を実行して、これらのサーバーが提供するすべてのディレクトリオブジェクトに新しい公開鍵を追加します。`nisupdkeys(1M)` コマンドを使用するには、その NIS+ ディレクトリオブジェクトに対する変更権を持っていなければなりません。詳細については、259 ページの「公開鍵の更新」を参照してください。

注意 – これらの NIS+ ディレクトリを提供するすべてのサーバー、およびこれらのディレクトリにアクセスするすべてのクライアントは、Solaris 7 以降を実行していなければなりません。

NIS+ ディレクトリオブジェクトへの新しい公開鍵の追加 - 例

この例では、新しい公開鍵を使用してサーバーが提供しているディレクトリは、doc.com、org_dir.doc.com.、および groups_dir.doc.com. です。更新は、マスターサーバー主体として行われます。新しいメカニズムを実行する前に、nisauthconf(1M) を使用して nisupdkeys を構成することが必要です。この例では、現在の認証メカニズムは des で、新しいメカニズムは dh640-0 です。

```
masterserver# nisauthconf dh640-0 des
masterserver# nisupdkeys doc.com.
            (画面上には何も表示されない)
masterserver# nisupdkeys org_dir.doc.com.
            (画面上には何も表示されない)
masterserver# nisupdkeys groups_dir.doc.com.
            (画面上には何も表示されない)
```

新しいセキュリティメカニズム資格を受け入れるようにする NIS+ サーバーの構成

各サーバー上で NIS+ 認証を構成して、新旧両方の資格を受け入れるようにします。そのためには、nisauthconf(1M) および keylogin(1) を実行し、keyserv(1M) を再起動することが必要です。keylogin(1) コマンドは、各メカニズムの鍵を /etc/.rootkey に格納します。keylogin の基本の詳細については、254 ページの「キーログイン」を参照してください。

新しいセキュリティメカニズム資格を受け入れるようにする NIS+ サーバーの構成 - 例

この例では、現在の認証メカニズムは des で、新しいメカニズムは dh640-0 です。ここでは順序が重要です。クライアントおよびサーバーの NIS+ 認証では des より後のエントリは無視されます。

```
server# nisauthconf dh640-0 des
server# keylogin -r
            (画面上には何も表示されない)
server# /etc/reboot
```

新しいセキュリティメカニズム資格を使用するようにするマシンの構成

新しい資格を受け入れられるようになったので、マシンを新しい資格によって認証するように変換することができます。そのためには、nisauthconf(1M) および keylogin(1) をスーパーユーザーとして実行し、再起動します。

新しいセキュリティメカニズム資格を受け入れるようにするマシンの構成 - 例

この例では、新しいメカニズムは dh640-0 ですが、システムは、dh640-0 資格が使用可能でないか、あるいは dh640-0 による認証に成功しなかった場合には、des 資格による認証も試みます。

```
workstation# nisauthconf dh640-0 des
workstation#  keylogin -r
            (画面上には何も表示されない)
workstation# /etc/reboot
```

次の例では、新しいメカニズムは dh640-0 で、このメカニズムによる認証だけが行われます。システムを新しいメカニズムだけで認証するように構成する前に、キャッシュに書き込まれているディレクトリオブジェクトが新しいメカニズムの鍵を含むように、リフレッシュされることが必要です。これは、nisshowcache(1M) によって検証することができます。キャッシュに書き込まれているディレクトリオブジェクトがタイムアウトしてリフレッシュされるのを待つ代わりとして、次の方法があります。nis_cachemgr(1M) を終了し、続いて nisinit(1M) を使用して新しい NIS_COLD_START を構築し、続いて niscachemgr(1M) を再起動します。

ディレクトリオブジェクトの手動によるリフレッシュ - 例

ディレクトリオブジェクトを手動でリフレッシュするには、次のようにします。

```
# pkill -u 0 nis_cachemgr
# nisinit -cH masterserver
# niscachemgr -i
```

注意 – マシン主体およびこのマシンのすべてのユーザーが cred テーブルの中に dh640-0 資格を持っていなければ、dh640-0 だけで認証を行うようにシステムを構成することはできません。

```
workstation# nisauthconf dh640-0
workstation#  keylogin -r
            (画面上には何も表示されない)
```

```
workstation# /etc/reboot
```

新しい資格を保護するパスワードの変更

ダミーパスワードを持った新しい資格を与えられた各ユーザーは、chkey(1) を実行してログインパスワードに変換することが必要です。詳細については、255 ページの「NIS+ 主体の鍵の変更」を参照してください。

新しい資格を保護するパスワードの変更 - 例

chkey(1) を実行してログインパスワードに変換します。

```
# chkey -p
              (画面上には何も表示されない)
```

新しいセキュリティメカニズム資格だけを受け入れるようにするサーバーの構成

低グレードのセキュリティメカニズムから高グレードのものに変換する場合には、新しい高グレードのセキュリティメカニズムタイプの資格だけを受け入れるように NIS+ サーバーを構成することによって、最大限のセキュリティが達成されます。新旧両方のメカニズムによって認証を行うようにサーバーを構成した後で、変換します。

新しいメカニズムだけを使用して認証を行うようにシステムを構成する前に、キャッシュに書き込まれているディレクトリオブジェクトをリフレッシュして、新しいメカニズムの鍵を含むようにし、nisshowcache(1M) を使用してこれを検証します。

新しいセキュリティメカニズム資格だけを受け入れるようにするサーバーの構成 - 例

各 NIS+ サーバー上で nisauthconf(1M) を実行し、再起動します。この例では、NIS+ サーバーは dh640-0 資格の認証だけを受け入れるように構成されます。

```
server#  nisauthconf dh640-0
server# /etc/reboot
```

この段階で、オプションで、ディレクトリオブジェクトを更新して、古い公開鍵を削除できます。これはマスターサーバーから行う必要があり、新しいセキュリティメカニズムだけを使用して認証を行うサーバーが管理するディレクトリごとに一度、nisupdkeys(1M) を実行しなければなりません。この例では、更新されるディレクトリは、doc.com、org_dir.doc.com.、および groups_dir.doc.com. です。

```
masterserver#    nisupdkeys doc.com.
            (画面上には何も表示されない)
masterserver#  nisupdkeys org_dir.doc.com.
            (画面上には何も表示されない)
masterserver#    nisupdkeys groups_dir.doc.com.
```

cred テーブルからの古い資格の削除

必要に応じて、cred テーブルから古いセキュリティメカニズムの資格を削除できます。そのためには、ローカルドメインの cred テーブルに対する変更権が必要です。詳細については、「資格情報の削除」の項を参照してください。

cred テーブルからの古い資格の削除 - 例

この例では、主体 morena.doc.com が自分の des 資格を cred テーブルから削除します。

```
master# nisaddcred -r morena.doc.com. dh192-0
```

第 15 章

NIS+ のアクセス権の管理

この章では、NIS+ アクセス権とその管理方法について説明します。

注 – NIS+ セキュリティタスクには、Solstice AdminSuite ツールがあればより簡単に実行できるものがあります。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ のアクセス権について

NIS+ アクセス権によって、NIS+ ユーザーが実行できる操作とアクセス可能な情報が決定されます。この章では、NIS+ セキュリティシステムとそのアクセス権について、十分に理解していることを前提としています。NIS+ セキュリティシステムについては、第 11 章を参照してください。

NIS+ アクセス関連のコマンドとその構文、オプションについては、`nis+(1)` のマニュアルページを参照してください。

承認およびアクセス権について

NIS+ 名前空間のセキュリティは、承認、アクセス権、および NIS+ の資格と認証の相互作用によって確保されています。これらの詳細については、223 ページの「NIS+ の承認とアクセス - 紹介」を参照してください。

承認クラス - 復習

223 ページの「承認クラス」に説明のあるように、NIS+ のアクセス権は各クラスに与えられるものです。4 つの NIS+ クラスが用意されています。

- 「所有者」。所有者クラスは 1 つの NIS+ 主体です。デフォルトではオブジェクトの所有者は、オブジェクトを作成した主体になります。しかし、オブジェクトの所有者は所有権を別の主体に譲ることで所有者を変更できます。
- 「グループ」。グループクラスは 1 つ以上の NIS+ 主体の集まりです。1 つの NIS+ オブジェクトは 1 つだけの NIS+ グループを持つことができます。
- 「その他」。その他クラスには、NIS+ に認証された NIS+ 主体のすべて (すべての所有者クラスとグループクラス、および有効な DES 資格を提示したものすべて) が含まれます。
- 「未認証」。未認証クラスは、適切な認証を受けられなかったものすべてで構成されます。すなわち、有効な DES 資格を提示できなかったすべてのものです。

アクセス権 - 復習

227 ページの「NIS+ アクセス権」で詳述したように、NIS+ には 4 種類のアクセス権があります。

- 「読み取り」。オブジェクトに対する読み取り権を持った主体がオブジェクトの内容を読み取れます。
- 「変更」。オブジェクトに対する変更権を持った主体がオブジェクトの内容を変更できます。
- 「削除」。オブジェクトに対する削除権を持った主体がオブジェクトを削除できます。
- 「作成」。上位レベルのオブジェクトに対する作成権を持った主体が、そのレベルの下位に新規のオブジェクトを作成できます。すなわち、NIS+ ディレクトリオブジェクトの作成権を持っていれば、そのディレクトリ内に新しいテーブルを作成できます。NIS+ テーブルに対する作成権があれば、そのテーブル内に新規の列とエントリを作成できます。

これらの権限は論理的には、ディレクトリ、テーブル、列とエントリのように下位に展開するものであることを銘記してください。たとえば、新規にテーブルを作成するには、そのテーブルを格納する NIS+ ディレクトリオブジェクトに対する作成権が必

要です。そのテーブルを作成した場合、そのデフォルト所有者になります。所有者として、自分自身にそのテーブルに対する作成権を与え、テーブルに新規のエントリを作成できます。テーブル内に新規のエントリを作成した場合、それらのエントリの所有者になります。テーブルの所有者として、テーブルレベルの作成権を他の人に与えることもできます。たとえば、自身のテーブルのグループクラスのテーブルレベルの作成権を与えることができます。その場合、テーブルのグループのすべてのメンバーがテーブル内に新規のエントリを作成できます。新規のテーブルエントリを作成した、グループの個々のメンバーはそのエントリのデフォルト所有者になります。

アクセス権の連鎖

承認クラスは連鎖の関係にあります。つまり、上位クラスは通常下位クラスにも属しており、自動的に下位クラスの権限を得ることになります。次のように機能します。

- 「所有者クラス」。オブジェクトの所有者はそのオブジェクトのグループに所属していても、いなくてもかまいません。所有者があるグループに属していると、そのグループに与えられている権限をすべて得ることになります。オブジェクトの所有者が自動的に、その他と未認証のクラスにも属することになり、自動的にこれら 2 つのクラスに与えられている権限を獲得することになります。
- 「グループクラス」。オブジェクトのグループメンバーは自動的にその他と未認証クラスに所属します。したがって、グループメンバーは自動的にその他クラスと未認証クラスのメンバーがそのオブジェクトに持っている権限を獲得します。
- 「その他クラス」。その他クラスは自動的に、未認証クラスがオブジェクトに対して持っている権限と同じ権限を持つことになります。
- 「未認証クラス」。未認証クラスは、オブジェクトが未認証クラスに与えている権限を持つだけです。

この基本原則は、下位クラスのアクセス権は自動的に上位クラスに与えられるということです。つまり、上位クラスは下位クラスよりも「多く」の権限を持つことができ、権限が「少なくなる」ことはないということです (この原則の例外は、もし所有者がグループのメンバーでなければ、所有者の持っていない権限をグループクラスに与えることが可能になる場合)。

アクセス権の割り当てと変更方法

NIS+ オブジェクトを作成した時、NIS+ はそのオブジェクトに所有者クラスとグループクラスのアクセス権のデフォルトセットを与えます。デフォルトでは、所有者はそのオブジェクトを作成した NIS+ 主体です。デフォルトのグループは、環境変数 `NIS_GROUP` の中で名前をつけられたグループです。

異なるデフォルト権限の指定

NIS+ は NIS+ オブジェクトが作成された時に自動的に付与されたデフォルト権限を変更する 2 つの異なった方法を提供しています。

- NIS_DEFAULTS 環境変数。NIS_DEFAULTS はセキュリティに関するデフォルト値を保管し、その 1 つはアクセス権です。このデフォルトアクセス権は、オブジェクトが作成された時にオブジェクトに自動的に付与されるものです (詳細については、284 ページの「NIS+ デフォルトの表示 - nisdefaults コマンド」を参照してください)。

  NIS_DEFAULTS 環境変数の値を変更すると、変更後に作成されたオブジェクトに新規の値が与えられます。しかし、以前に作成されたオブジェクトは影響を受けません。

- -D オプション。-D オプションは、いくつかの NIS+ コマンドで使用されます。NIS+ オブジェクトを作成するコマンドに -D オプションを使用すると、NIS_DEFAULTS 環境変数が指定したデフォルト権限を上書きします。詳細については、287 ページの「デフォルトを無効にする」を参照してください。

既存オブジェクトへのアクセス権を変更する

NIS+ オブジェクトを作成する場合、既存のデフォルトアクセス権 (NIS_DEFAULTS 環境変数か -D オプションの指定かのどちらかによる) に対処する必要があります。デフォルト権限を変更するコマンドは次のとおりです。

- nischmod コマンド
- テーブルの列の場合は nistbladm コマンド

テーブル、列、およびエントリのセキュリティ

NIS+ テーブルに対するアクセス権を指定する方法には、次の 3 通りあります。

- 「テーブル」全体を対象にアクセス権を指定する
- 「エントリ」(行) 単位でアクセス権を指定する
- 「列」単位でアクセス権を指定する

列とエントリ (行) が交差する部分をフィールドといいます。データ値はすべてこの交差領域、つまりフィールドに入力します。

列とエントリレベルのアクセス権を持っていると、テーブルレベルのアクセス権の制限があっても個々の行と列にアクセスできますが、テーブル全体へのアクセス権以上に列とエントリレベルの権限を制限することはできません。

- 「テーブル」。テーブルレベルは基本的なレベルです。テーブルレベルに付与されたアクセス権は、列ごとまたはエントリごとに特に変更された場合を除き、テーブル内のすべての部分に適用されます。テーブルレベルの権限は最も厳格であるべきです。

注 – 承認クラスは連結しているということに注意してください。上位クラスは、下位クラスに割り当てられた権利を取得していることになります。273 ページの「アクセス権の連鎖」を参照してください。

- 「列」。列レベルの権限を持っていると、列ごとに追加アクセス権を持つことになります。たとえば、その他クラスと未認証クラスにテーブルレベルの権限が何も付与されていなかったとします。この場合、この 2 つのクラスはテーブル内のデータに対して、読み取り権、変更権、作成権、削除権を持ちません。列レベルの権限を持てば、テーブルレベルの権限の制限を超えて、その他クラスのメンバーであっても特定の列のデータを読み取ることができます。

  一方、所有者クラスとグループクラスにテーブル全体の読み取り権が付与されている場合、列レベルの権限を使ってグループクラスの列の読み取り権を妨げることはできません。列のグループはテーブルのグループやエントリのグループと同じにはなりません。

- 「エントリ *(*行*)*」。エントリレベルの権限があれば、行ごとに追加アクセス権を持つことになります。たとえば、個々のユーザーに指定したエントリに限って変更する権限を与えることができるのです。

  エントリのグループはテーブルのグループとは同じである必要はなく、別のグループにできます。そうすることによって、特定のグループのメンバーに、他のグループに属するエントリに影響を与えないで、あるエントリのセットにアクセスする権限を付与できます。

テーブル、列、エントリの例

列またはエントリレベルのアクセス権は、追加アクセスを次の 2 つの方法で提供できます。1 つは権限を持つ主体を増やす方法で、もう 1 つは同じ主体に権限を追加する方法です。もちろん、これらを組み合わせることも可能です。以下にその例を示します。

テーブルオブジェクトが、そのテーブルの所有者に対して読み取り権を与えたとします。

表 15–1 テーブル、列、エントリの例 1

	未認証	所有者	グループ	その他
テーブルのアクセス権	`----`	`r---`	`----`	`----`

このことは、テーブルの所有者だけがテーブル全体の内容を読み取れることを意味します。所有者でない主体は、アクセスできません。テーブルのエントリ 2 にグループクラスに対して読み取り権を与えることができます。

表 **15–2** テーブル、列、エントリの例 2

	未認証	所有者	グループ	その他
テーブルのアクセス権	`----`	`r---`	`----`	`----`
エントリ 2 のアクセス権	`----`	`----`	`r---`	`----`

テーブルの内容をすべて読み取れるのは所有者だけですが、このテーブルのグループのメンバーであれば、この特定エントリの内容を読み取ることができます。次に、特定の列がその他クラスに読み取り権を与えたとします。

表 **15–3** テーブル、列、エントリの例 3

	未認証	所有者	グループ	その他
テーブルのアクセス権	`----`	`r---`	`----`	`----`
エントリ 2 のアクセス権	`----`	`----`	`r---`	`----`
列 1 のアクセス権	`----`	`----`	`----`	`r---`

その他クラスのメンバーは列 1 のすべてのエントリを読み取ることができます。グループクラスのメンバーは (その他クラスにも属しているので) 列 1 のすべてとエントリ 2 の全列を読み取ることができます。`*NP*` になっているセルは、「グループ」、「その他」いずれのクラスも読み取りができません (どちらにもアクセス権がない)。表 15–4 を参照してください。

表 **15–4** テーブル、列、エントリの例 4

	列 **1**	列 **2**	列 **2**
エントリ 1	読み取り	`*NP*`	`*NP*`
エントリ 2	読み取り	読み取り	読み取り
エントリ 3	読み取り	`*NP*`	`*NP*`
エントリ 4	読み取り	`*NP*`	`*NP*`
エントリ 5	読み取り	`*NP*`	`*NP*`

異なるレベルの権限

この節では 4 つの権限 (読み取り、作成、変更、削除) が 4 つのアクセスレベル (ディレクトリ、テーブル、列、エントリ) とどのようにかかわるのかを説明します。

種々の権限とレベルに関係した機能を次の表 15–5 にまとめます。

表 **15–5** アクセス権、アクセスレベル、およびその機能

	ディレクトリ	テーブル	列	エントリ
読み取り	ディレクトリ内容のリスト	テーブル内容の読み取り	列内容の読み取り	エントリ (行) 内容の読み取り
作成	新規ディレクトリまたはテーブルオブジェクトの作成	新規エントリ (行) の追加	列に新規のデータを入力	エントリ (行) に新規のデータを入力
変更	オブジェクトの移動とオブジェクト名の変更	テーブル内の任意のデータを変更	列内のデータを変更	エントリ (行) 内のデータを変更
削除	テーブル等のディレクトリオブジェクトの削除	エントリ (行) の削除	列内のデータの削除	エントリ (行) 内のデータの削除

読み取り権

- 「ディレクトリ」。ディレクトリの読み取り権があれば、ディレクトリの内容を表示できます。
- 「テーブル」。テーブルの読み取り権があれば、テーブル内のすべてのデータを読み取れます。
- 「列」。列の読み取り権があれば、その列のすべてのデータを読み取れます。
- 「エントリ」。エントリの読み取り権があれば、そのエントリのすべてのデータを読み取れます。

作成権

- 「ディレクトリ」。ディレクトリレベルの作成権があれば、テーブル等の新規オブジェクトをディレクトリ内に作成できます。
- 「テーブル」。テーブルレベルの作成権があれば、テーブル内に新規のエントリを作成できますが、列は作成できません。テーブルレベルの作成権だけでは、テーブルに新規のエントリを追加できますが、新規の列を追加することはできません。
- 「列」。列の作成権があれば、列内のフィールドに新規のデータを入力できます。新規の列を作成できません。
- 「エントリ」。エントリの作成権があれば、その行のフィールドに新規のデータを入力できます。(エントリレベルの作成権では新規の行を作成することはできません。)

変更権

- 「ディレクトリ」。ディレクトリレベルの変更権があれば、ディレクトリオブジェクトの移動と名前の変更ができます。

- 「テーブル」。テーブルレベルの変更権があれば、テーブル内のデータをすべて変更できます。新規の行を作成 (追加) できますが、新規の列は作成できません。空白フィールドにデータを入力することも可能です。
- 「列」。列の変更権があれば、その列の任意のフィールドのデータを変更できます。
- 「エントリ」。エントリの変更権があれば、その行の任意のフィールドのデータを変更できます。

削除権

- 「ディレクトリ」ディレクトリレベルの削除権があれば、テーブル等のディレクトリ内の既存オブジェクトを削除できます。
- 「テーブル」テーブルレベルの削除権があれば、テーブル内の既存のエントリ (行) を削除できますが、列は削除できません。削除できるのは既存のエントリだけで、既存の列は削除できません。
- 「列」列の削除権があれば、その列の任意のフィールドのデータを削除できます。
- 「エントリ」エントリの削除権があれば、その行の任意のフィールドのデータを削除できます。

アクセス権の格納場所

オブジェクトのアクセス権は、そのオブジェクトの定義の一部として指定され、格納されます。この情報は NIS+ テーブルには格納されません。

NIS+ オブジェクトのアクセス権の読み取り

アクセス権を読み取るには `niscat` コマンドを使用します。

`niscat -o` *objectname*

アクセス権を読み取るオブジェクト名を指定します。

このコマンドは、NIS+ オブジェクトに関する次の情報を返します。

- 「所有者」。所有権を持っている NIS+ 主体。通常はオブジェクトを作成した人ですが、元の所有者が所有権を渡した場合もある
- 「グループ」。オブジェクトの NIS+ 主体
- 「未認証クラスのアクセス権」。認証された (有効な DES 資格を提示した) か否かにかかわらず、全員が持つ権限
- 「所有者クラスのアクセス権」。オブジェクトの所有者に付与されたアクセス権
- 「グループクラスのアクセス権」。オブジェクトのグループに付与されたアクセス権

- 「その他クラスのアクセス権」。認証された NIS+ 主体全てに付与されたアクセス権

4 つの承認クラスのアクセス権は、次のように 16 文字の文字列で表示されます。

```
r---rmcdr---r---
```

各文字がアクセス権の種類を表します。

- r は読み取り権
- m は変更権
- d は削除権
- c は作成権
- - はアクセス権のないことを表す

先頭の 4 文字は未認証に、次の 4 文字は所有者に、その次の 4 文字はグループに、そして最後の 4 文字はその他に、それぞれ与えられたアクセス権を表します。

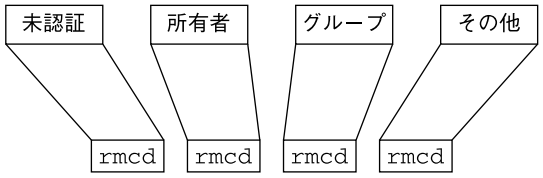


図 **15–1** アクセス権の表示

注 – UNIX ファイルシステムとは異なり、先頭のアクセス権は未認証用であり、所有者用ではありません。

デフォルトのアクセス権

オブジェクトを作成すると、NIS+ はそのオブジェクトにデフォルトの所有者、グループ、および アクセス権を割り当てます。デフォルトの所有者は、そのオブジェクトを作成する NIS+ 主体です。デフォルトのグループは、環境変数 NIS_GROUP の中で名前をつけられたグループです。デフォルトのアクセス権は次のようになります。

表 **15–6** デフォルトのアクセス権

未認証	所有者	グループ	その他
-	読み取り権	読み取り権	読み取り権
-	変更権	-	-

表 **15–6** デフォルトのアクセス権　(続き)

未認証	所有者	グループ	その他
-	作成	-	-
-	削除	-	-

環境変数 NIS_DEFAULTS のセットがある場合、NIS_DEFAULTS 内の値が新規のオブジェクトに適用されるデフォルト値を決定します。コマンド行でオブジェクトを作成した場合は、-D フラグを使ってデフォルト値以外を設定できます。

テーブルに対するアクセス権をサーバーが割り当てる方法

この節では、読み取り、変更、削除、作成の操作が行われる際、テーブルオブジェクト、エントリ、列に対するアクセス権をサーバーが各クラスにどのように割り当てるかについて説明します。

注 – セキュリティレベル 0 では、サーバーはアクセス権を実行しないため、すべてのクライアントがテーブルオブジェクトに対する完全なアクセス権を付与されます。セキュリティレベル 0 は管理者だけがテストの目的で使用します。通常の業務にはレベル 0 を使用しないでください。

サーバーがアクセスを許可するか否かを決定する 4 つの要素があります。

- 主体が要求する処理の種類
- 主体がアクセスしようとしているテーブル、エントリ、または列
- その特定のオブジェクトに対して、主体が所属する承認クラス
- テーブル、エントリ、または列がその主体の承認クラスに割り当てたアクセス権

認証後に主体は、主体が有効な DES 資格を所持しているかを確認することで要求を行い、NIS+ サーバーは処理の種類と要求のオブジェクトを決定します。

- 「ディレクトリ」。オブジェクトがディレクトリかグループの場合、サーバーは 4 つのクラスに付与されている権限を知るためにオブジェクトの定義をチェックし、どのクラスに主体が所属しているか判定し、主体のクラスとそのクラスに付与された権限に基づいて、その要求を受け入れるかまたは拒否します。
- 「テーブル」。オブジェクトがテーブルの場合、サーバーは 4 つのクラスに付与されているテーブルレベルの権限を知るためにテーブルの定義をチェックし、どのクラスに主体が所属しているか判定します。所属しているクラスが要求処理を行うテーブルレベルの権限を持っていない場合は、サーバーは次にどの行または列にかかわる処理かを判定し、要求処理に必要な該当する行または列レベルのアクセス権があるかを決定します。

コマンドによるアクセス権の指定

ここでの説明では、NIS+ 環境のセキュリティレベルが 2 (デフォルト) であるものと想定しています。

この節では、この章で説明するコマンドを使用するときにアクセス権や所有者、グループ所有者、オブジェクトを指定する方法を説明します。

アクセス権の構文

この節では、承認とアクセス権に関係する NIS+ コマンドに使われるアクセス権の構文について説明します。

クラス、演算子、および権限の構文

アクセス権は、環境変数で指定する場合もコマンドで指定する場合も、「クラス (class)」、「演算子 (operator)」、「権利 (right)」という 3 種類の引数で区別されます。

- 「クラス」。クラスは、「権利」が適用される NIS+ 主体のカテゴリを意味します。

表 **15–7** アクセス権の構文 - クラス

クラス	説明
`n`	未認証: すべての未認証要求
`o`	オブジェクトまたはテーブルエントリの所有者
`g`	オブジェクトまたはテーブルエントリのグループ所有者
`w`	その他: すべての認証済み主体
`a`	すべて: 所有者、グループ、およびその他の省略形。これはデフォルト

- 「演算子」。 演算子は、「権限」について行われる操作を示します。

表 **15–8** アクセス権の構文 - 演算子

演算子	説明
`+`	「権利」によって指定されたアクセス権を追加する
`-`	「権利」によって指定されたアクセス権を取り消す

表 **15–8** アクセス権の構文 - 演算子　(続き)

演算子	説明
=	この演算子権利によって指定されたアクセス権に変更する。つまり、既存の「権利」をすべて取り消し、新しいアクセス権に置き換える

- 「権限」。アクセス権そのものです。使用可能な値は次のとおりです。

表 **15–9** アクセス権の構文 - 権限

権利	説明
r	オブジェクト定義またはテーブルエントリを読み取る
m	オブジェクト定義またはテーブルエントリを変更する
c	テーブルエントリまたは列を作成する
d	テーブルエントリまたは列を削除する

コンマ (,) で区切ることで、複数のコマンドを 1 つのコマンド行にまとめることができます。

表 **15–10** クラス、演算子、権限の構文 - 例

操作	構文
読み取りアクセス権を「所有者」クラスに追加する	`o+r`
変更アクセス権を所有者、グループ、およびその他のクラスに追加する	`a=m`
読み取りと変更の権限をその他と未認証クラスに追加する	`wn+m`
グループ、その他、および未認証クラスから 4 つの権限をすべて削除する	`gwn-rmcd`
所有者クラスに作成と削除の権限を追加し、その他と未認証クラスに読み取り権と変更権を追加する	`o+cd,wn+rm`

所有者とグループの構文

- 「所有者」。NIS+ グループを指定するには、NIS+ グループ名にドメイン名を付けて使います。
- 「グループ」。NIS+ グループを指定するには、NIS+ グループ名にドメイン名を付けて使います。

  主体名が完全指定されていることに注意してください。(*principalname.domainname*)

  所有者

principalname

グループ

groupname.domainname

オブジェクトとテーブルエントリの構文

オブジェクトとテーブルエントリは異なる構文を使います。

- オブジェクトは単純なオブジェクト名を使います。
- テーブルエントリはインデックス付きの名前を使います。

オブジェクト

objectname

テーブルエントリ

[*columnname=value*] , *tablename*

注 – この場合、角括弧 ([ ]) は構文の一部であり、オプション記号ではありません。

インデックス付きの名前では、列と値のペアを複数指定できます。その場合、操作はすべての列と値のペアに一致するエントリにだけ適用されます。列と値のペアが増えると、検索条件が厳しくなります。

例を次に示します。

表 **15–11** オブジェクトとエントリ - 例

種類	例
オブジェクト	`hosts.org_dir.sales.doc.com.`
テーブルエントリ	`'[uid=33555],passwd.org_dir.Eng.doc.com.'`
2 つの値のテーブルエントリ	`'[name=sales,gid=2],group.org_dir.doc.com.'`

列はインデックス付きの名前の特殊な形式になっています。列の操作は `nistbladm` コマンドでだけ可能なため、詳細については 356 ページの「`nistbladm` コマンド」を参照してください。

NIS+ デフォルトの表示 - nisdefaults コマンド

nisdefaults コマンドは、名前空間内で現在有効な 7 つのデフォルトを表示します。これらのデフォルトは次のいずれかです。

- NIS+ ソフトウェアが提供するプリセット値
- 環境変数 NIS_DEFAULTS 値で指定 されるデフォルト (変数 NIS_DEFAULTS のセットがある場合)

オブジェクトを作成する時に -D オプション付きのコマンドを使って上書きしなければ、このマシン上でオブジェクトを作成すると自動的にデフォルト値を獲得することになります。

表 **15–12** 7 つの NIS+ デフォルト値と nisdefaults オプション

デフォルト	オプション	元データ	説明
ドメイン	-d	/etc/defaultdomain	コマンドを入力したマシンのホームドメインを表示する
グループ	-g	環境変数 NIS_GROUP	このシェルが作成する次のオブジェクトに付与されるグループを表示する
ホスト	-h	uname -n	マシンのホスト名を表示する
主体	-p	gethostbyname()	nisdefaults コマンドを入力した NIS+ 主体の完全指定ユーザー 名またはホスト名を表示する
アクセス権	-r	環境変数 NIS _DEFAULTS	このシェルが作成する次のオブジェクトまたはエントリに付与されるアクセス権を表示する フォーマット : ----rmcdr---r---
検索パス	-s	環境変数 NIS_PATH	検索パスの構文を表示する。これは NIS+ が情報を検索する時のドメインを示す。もし設定してあれば、環境変数 NIS_PATH の値を表示する
生存期間	-t	環境変数 NIS _DEFAULTS	このシェルが作成する次のオブジェクトに付与される生存期間を表示する。デフォルトは 12 時間
全部 (簡潔)	-a		7 つのデフォルトすべてを簡潔フォーマットで表示する
冗長	-v	指定した値を冗長モードで表示する	

これらのオプションを使用して、すべてのデフォルト値もしくはそのサブセットを表示できます。

- すべての値を冗長フォーマットで表示するには、引数なしで nisdefaults コマンドを実行します。

```
master% nisdefaults
Principal Name : topadmin.doc.com.
Domain Name : doc.com.
Host Name : rootmaster.doc.com.
Group Name : salesboss
Access Rights : ----rmcdr---r---
Time to live : 12:00:00:00:00
Search Path : doc.com.
```

- すべての値を簡潔フォーマットで表示するには、-a オプションを付けます。
- 値のサブセットを表示するには、適切なオプションを使用します。その値は簡潔モードで表示されます。権限と検索パスのデフォルトを簡潔モードで表示する例を次に示します。

```
rootmaster% nisdefaults -rs
----rmcdr---r---
doc.com.
```

- 値のサブセットを冗長モードで表示するには、-v フラグを使用します。

デフォルトセキュリティ値の設定

この節では、nisdefaults コマンド、環境変数 NIS_DEFAULTS、および -D オプションに関連したタスクを実行する方法を説明します。環境変数 NIS_DEFAULTS は次のデフォルト値を指定します。

- 所有者
- グループ
- アクセス権
- 生存期間

環境変数 NIS_DEFAULTS に設定した値はデフォルトとなり、そのシェルを使用して作成したすべての NIS+ オブジェクトに適用されます (-D オプション付きでコマンドを実行してデフォルト値に上書きした場合を除く)。

環境変数 NIS_DEFAULTS を指定することで、デフォルト値 (所有者、グループ、アクセス権、および生存期間) を指定できます。一度 NIS_DEFAULTS の値を設定するとそのシェルから作成したすべてのオブジェクトは、-D オプション付きでコマンドを実行して上書きした場合を除きそのデフォルトに設定されます。

NIS_DEFAULTS の値を表示する

echo コマンドを使って、環境変数の値をチェックできます。以下にその例を示します。

```
client% echo $NIS_DEFAULTS
owner=butler:group=gamblers:access=o+rmcd
```

nisdefaults コマンドを使用して、名前空間でアクティブな NIS+ デフォルトの一般的リストを表示することも可能です。284 ページの「NIS+ デフォルトの表示 - nisdefaults コマンド」を参照してください。

デフォルトを変更する

環境変数 NIS_DEFAULTS の値を変更することで、アクセス権、所有者、およびグループのデフォルトを変更できます。ユーザーのシェルに適切な環境コマンド (C シェルには setenv、Bourne シェルと Korn シェルには $NIS_DEFAULTS=, export) を次の引数を付けて使用します。

- access=アクセス権 (281 ページの「コマンドによるアクセス権の指定」で説明されたフォーマットを使って)
- owner=所有者名
- group=グループ名

複数の引数をまとめる場合は、コロン (:) で区切ります。

owner= 主体名 :group= グループ名

表 15–13 に例をいくつか示します。

表 **15–13** デフォルトの変更 - 例

作業	例
所有者のデフォルトアクセス権に読み取り権を設定	client% setenv NIS_DEFAULTS access=o+r
デフォルトの所有者をホームドメインが doc.com. である abe に設定	client% setenv NIS_DEFAULTS owner=abe.doc.com.
2 つのコード行をまとめる	client% setenv NIS_DEFAULTS access=o+r:owner =abe.doc.com.

デフォルトを変更したシェルから作成されるすべてのオブジェクトとエントリは、指定した新規の値になります。テーブルの列またはエントリに対してはデフォルトを指定できません。列とエントリはデフォルトをそのまま継承します。

NIS_DEFAULTS の値を再設定する

変数 NIS_DEFAULTS をオリジナルの値に再設定するには、ユーザーのシェルに適したフォーマットを使って、引数なしで変数の名前を入力します。

C シェルの場合

```
client# unsetenv NIS_DEFAULTS
```

Bourne シェルまたは Korn シェルの場合

```
client$ NIS_DEFAULTS=; export NIS_DEFAULTS
```

デフォルトを無効にする

次の NIS+ コマンドのどれかを使えば、NIS+ オブジェクトまたはテーブルエントリを作成する時に、いつでもデフォルトのアクセス権、所有者、およびグループを無効にできます。

- nismkdir - NIS+ ディレクトリオブジェクトを作成
- nisaddent - エントリを NIS+ テーブルに転送
- nistbladm - NIS+ テーブル内にエントリを作成

デフォルト値を無効にするには、281 ページの「コマンドによるアクセス権の指定」の説明のように、コマンドの構文に -D オプションを挿入します。

デフォルトを設定する時と同様、複数の引数を 1 行のコマンド行で指定できます。なお、列とエントリの所有者とグループは常に同じであるため、これらを無効にすることはできません。

nismkdir コマンドを使用して、sales.doc.com ディレクトリを作成し、デフォルトアクセス権を無効にして所有者にだけ読み取り権を付与するには、次のように入力します。

```
client% nismkdir -D access=o+r sales.doc.com
```

オブジェクトとエントリのアクセス権を変更する

nischmod コマンドは、NIS+ オブジェクトまたはテーブルエントリのアクセス権を変更します。テーブル列のアクセス権は操作しません。列の場合、nistbladm コマンドに -D オプションを付けて実行してください。nischmod コマンドを使用するには、そのオブジェクトかエントリに対する変更権が必要です。

nischmod コマンドを使用して権限を追加する

オブジェクトまたはエントリに権限を追加する例を次に示します。

オブジェクト

`nischmod` *class+right object-name*

テーブルエントリ

`nischmod class+`*right* `[column-name=`*value*`],`*table-name*

`sales.doc.com.` ディレクトリオブジェクトのグループに読み取り権と変更権を追加する場合は、次のように入力します。

```
client% nischmod g+rm sales.doc.com.
```

`hosts.org_dir.doc.com.` テーブル内の `name=abe` エントリのグループに読み取り権と変更権を追加する場合は、次のように入力します。

```
client% nischmod g+rm '[name=abe],hosts.org_dir.doc.com.'
```

nischmod を使用して権限を削除する

オブジェクトまたはエントリの権限を削除するには、次のように入力します。

オブジェクト

`nischmod` *class-right object-name*

エントリ

`nischmod` *class-right* `[`*column-name=value*`],`*table-name*

`sales.doc.com.` ディレクトリオブジェクトのグループから作成権と削除権を削除するには次のように入力します。

```
client% nischmod g-cd sales.doc.com.
```

たとえば、`hosts.org_dir.doc.com.` テーブル内の `name=abe` エントリのグループから削除権を削除する場合は、次のようにします。

```
client% nischmod g-d '[name=abe],hosts.org_dir.doc.com.'
```

列アクセス権を指定する

nistbladm コマンドは NIS+ テーブルに種々の操作を行うことができます。これについては 356 ページの「nistbladm コマンド」にほとんどの説明があります。このコマンドは -c と -u の 2 つのオプションを使用してセキュリティ関連のタスクを実行できます。

- -c オプション。nistbladm コマンドを使用して テーブルを作成する際に、-c オプションを使用すると最初の列アクセス権を指定できます。
- -u オプション。-u オプションを nistbladm コマンドとともに使用すると、列アクセス権を変更できます。

テーブル作成時の列権限設定

テーブルが作成された時、列にはテーブルオブジェクトと同じ権限が付与されます。このテーブルレベルの権限は環境変数 NIS_DEFAULTS で指定されるか、またはテーブルを作成したコマンドの一部で指定されます。nistbladm コマンドでテーブルを作成する際に -c オプションを使用すれば、最初の列アクセス権を指定できます。このオプションを使用するには、テーブルを作成しようとするディレクトリの作成権が必要です。テーブル作成時に列権限を指定するには、次のように入力します。

```
nistbladm -c type 'columnname=[flags] [,access]... tablename'
```

引数の意味は、それぞれ以下のとおりです。

- *type* は、テーブルの種類を示す文字列です。テーブルの *type* は何にでもできます。
- *columnname* は列名です。
- *flags* には、列の種類を指定します。使用できるフラグには次のものがあります。
  - S (検索可能)
  - I (大文字と小文字を区別しない)
  - C (暗号化)
  - B (2 進データ)
  - X (XDR 符号化データ)
- *access* は、281 ページの「コマンドによるアクセス権の指定」で説明した構文を使用したこの列のアクセス権です。
- ... は、それぞれの型と権限のセットを持った複数の列を指定できることを示しています。
- *tablename* は、完全指定名で表した作成しようとするテーブル名です。

テーブルを作成する際、その列にはテーブルオブジェクトと同じ権限が付与されます。列に独自の権限を付与するには、列の型とコンマの後の各列の等号記号にアクセス権を追加し、列をスペースで区切ります。

column=type, *rights column= type*, *rights column = type*, *rights*

以下の例では、div 特性で depts という名前のテーブルが doc.com ディレクトリに作成されます。列は Name、Site および Manager の 3 つです。第 2、第 3 列のグループには、変更権が追加されます。

```
rootmaster% nistbladm -c div Name=S Site=S,g+m Manager=S,g+m depts.doc.com
```

nistbladm と -c オプションの詳細については、第 19 章を参照してください。

既存のテーブル列に権限を追加する

既存の NIS+ テーブルの列にアクセス権を追加するには、nistbladm -u オプション付きの nistbladm コマンドを使用します。このオプションを使用する場合、テーブル列の変更権が必要です。入力例を次に示します。

```
nistbladm -u [column= access,...],tablename
```

引数の意味はそれぞれ以下のとおりです。

- *column* は列名です。
- *access* は、281 ページの「コマンドによるアクセス権の指定」で説明した構文を使用したこの列のアクセス権です。
- ... は、それぞれの type と権限のセットを持った複数の列を指定できることを示しています。
- *tablename* は、完全指定名で表した作成しようとするテーブル名です。

権限を更新する列ごとに *column=access* のペアを使用します。複数の列を更新するにはコンマ (,) で区切り全体を角括弧 ([]) で囲みます。

```
[column=access, column=access, column=access]
```

このオプションの完全な構文については、第 2 章を参照してください。

次の例は、hosts.org_dir.doc.com. テーブルの name および addr 列のグループに、作成権と変更権を追加するものです。

```
client% nistbladm -u '[name=g+rm,addr=g+rm],hosts.org_dir..doc.com.'
```

テーブル列から権限を削除する

NIS+ テーブル列のアクセス権を削除するには、上記 290 ページの「既存のテーブル列に権限を追加する」で説明したように -u オプションを使用します。

次の例は、hosts.org_dir.doc.com. テーブルの hostname 列のグループの読み取り権と変更権を削除するものです。

```
client% nistbladm -u 'name=g-rm,hosts.org_dir.doc.com.'
```

オブジェクトとエントリの所有権の変更

nischown コマンドは、1 つ以上のオブジェクトまたはエントリの所有者を変更します。このコマンドを使用するには、オブジェクトまたはエントリに対する変更権が必要です。テーブルの列の所有者はテーブルの所有者であるため、nischown コマンドでは列の所有者を変更できません。列の所有者を変更するには、テーブルの所有者を変更する必要があります。

nischown コマンドを使用してオブジェクトの所有者を変更する

オブジェクトの所有者を変更するには、次のように入力します。

nischown *new-owner object*

引数の意味はそれぞれ以下のとおりです。

- *new-owner* は、そのオブジェクトの新規の所有者のユーザー ID で完全指定します。
- *object* は、オブジェクトの完全指定名です

オブジェクト名と新規所有者名にドメイン名を必ず追加します。

次の例は、doc.com. ドメイン内の hosts テーブルの所有者を、ホームドメインが doc.com. でユーザー名 lincoln であるユーザーに変更するものです。

```
client% nischown lincoln.doc.com. hosts.org_dir.doc.com.
```

nischown コマンドを使用してテーブルエントリの所有者を変更する

テーブルエントリの所有者を変更する構文は、エントリを特定するのにインデックス付きエントリを使います。次に例を示します。この構文の詳細については、283 ページの「オブジェクトとテーブルエントリの構文」を参照してください。

nischown *new-owner* [*column=value*,...], *tablename*

引数の意味はそれぞれ以下のとおりです。

- *new-owner* は、そのオブジェクトの新規の所有者のユーザー ID で完全指定します。
- *column* は、所有者を変更するエントリ (行) を特定する値を持った列名です。
- *value* は、所有者を変更するエントリ (行) を特定するデータ値です。

- ... は、所有者を変更する複数のエントリを示します。
- *tablename* は、所有者を変更するエントリを含むテーブルの完全指定名です。

所有者名とテーブル名にドメイン名を必ず追加します。

次の例は、doc.com. ドメインのホストテーブル内のエントリの所有者を、ホームドメインが doc.com. であるユーザー takeda に変更するものです。そのエントリの name 列の値は virginia です。

```
client% nischown takeda.doc.com. '[name=virginia],hosts.org_dir.doc.com.'
```

オブジェクトまたはエントリグループの変更

nischgrp コマンドは、1 つ以上のオブジェクトまたはテーブルエントリのグループ所有者を変更します。このコマンドを使用するには、オブジェクトまたはエントリに対する変更権が必要です。テーブルの列に割り当てられたグループは、テーブルに割り当てられたグループと同じであるため、nischgrp コマンドは、列のグループを変更できません。列のグループ所有者を変更するには、テーブル所有者を変更する必要があります。

nischgrp コマンドを使用してオブジェクトのグループを変更する

オブジェクトのグループを変更するには、次の構文を使用します。

nischgrp *group object*

引数の意味はそれぞれ以下のとおりです。

- *group* は、オブジェクトの新規のグループの完全指定名です。
- *object* は、オブジェクトの完全指定名です。

オブジェクト名と新規のグループ名にはドメイン名を必ず追加します。

次の例は、doc.com. ドメイン内の hosts テーブルのグループを admins.doc.com. に変更するものです。

```
client% nischgrp admins.doc.com. hosts.org_dir.doc.com.
```

nischgrp コマンドを使用してテーブルエントリのグループを変更する

テーブルエントリのグループを変更する構文は、エントリを識別するためにインデックス付きのエントリを使用します (この構文については、283 ページの「オブジェクトとテーブルエントリの構文」に詳細説明があります)。以下に構文を示します。

nischgrp *new-group* [*column=value*,...], *tablename*

引数の意味はそれぞれ以下のとおりです。

- *new-group* は、オブジェクトの新規グループの完全指定名です。
- *column* は、変更される特定のエントリ (行) を識別する値を持った列名です。
- *value* は、変更されるグループの特定の エントリ (行) を識別するデータ値です。
- *tablename* は、変更されるグループのエントリを含むテーブルの完全指定名です。
- ... は、複数のエントリで変更するグループを指定できることを示します。

新規グループ名とテーブル名にはドメイン名を必ず追加します。

次の例は、`doc.com.` ドメインのホストテーブル内のエントリのグループを `sales.doc.com.` に変更するものです。そのエントリの `name` 列の値は `virginia` です。

```
client% nischgrp sales.doc.com. '[name=virginia],hosts.org_dir.doc.com.'
```

第 16 章

パスワードの管理

この章では、一般ユーザー (NIS+ 主体) の観点から見た `passwd` コマンドの使用方法と、NIS+ 管理者によるパスワードシステムの管理方法を説明します。

注 – NIS+ セキュリティタスクには、Solstice AdminSuite ツールが利用可能であればもっと簡単に実行できるものがあります。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

パスワードの使用

マシンへのログインの際には、ユーザー名 (「ログイン ID」とも呼ばれる)、およびパスワードを入力する必要があります。ログイン ID は公開の情報ですが、パスワードを知っているのは所有者だけです。

ログインの方法

システムへのログインは以下の手順で行います。

1. **Login:** プロンプトで、ログイン **ID** を入力します。

2. **Password:** プロンプトで、パスワードを入力します。
   (秘密を守るため、入力してもパスワードは画面に表示されません)
   ログインに成功すると、本日のメッセージ (ない場合もある)、続いてコマンド行プロンプト、ウィンドウシステム、通常のアプリケーションなどが表示されます。

Login incorrect メッセージ

「Login incorrect」というメッセージは以下のことを意味します。

- 「誤ったログイン ID あるいはパスワードを入力した」。最も一般的な理由です。スペルを確認してもう一度やり直します。誤入力の回数は、ほとんどのシステムで「5」に制限されているので注意してください。
  - 誤入力の回数が制限を超えると、「Too many failures - try later」というメッセージが表示され、一定の時間が経過するまで再試行ができなくなります。
  - 指定された時間内にログインが成功しないと、「Too many tries; try again later」というメッセージが表示され、一定の時間が経過するまで再試行できなくなります。
- 「管理者がパスワードをロックしている」。管理者によってパスワードがロックされた場合、ロックを解除してもらうまで使用することができません。ログイン ID、パスワードともに正しく入力しているにもかかわらず「Login incorrect」メッセージが表示される場合は、システム管理者に問い合わせてください。
- 「パスワード使用権がシステム管理者によって取り消された」。ログイン ID、パスワードともに正しく入力しているにもかかわらず「Login incorrect」メッセージが表示される場合は、システム管理者に問い合わせてください。

will expire メッセージ

「Your password will expire in *N* days」メッセージ (または「Your password will expire within 24 hours」というメッセージ) は、「パスワードが、*N* 日あるいは 24 時間以内に有効期限に達する」ということを意味します。

このメッセージが表示されたら、パスワードをすぐに変更する必要があります (297 ページの「パスワードの変更方法」を参照)。

Permission denied メッセージ

ログイン ID とパスワードを入力後、「Permission denied」というメッセージが表示されて login: プロンプトに戻る場合があります。これは、管理者がパスワードをロックしたか、管理者がユーザーのアカウントを終了したか、パスワード権限の有効期限が切れたために、ログインに失敗したことを意味します。このような場合、管理者がパスワードロックを解除するか、アカウントを復旧するまではログインができません。システム管理者に問い合わせてください。

パスワードの変更方法

セキュリティを確保するため、パスワードは定期的に変更してください (新しいパスワードを作成する場合の必要条件、および基準については、298 ページの「パスワードの選択」を参照)。

注 – 現在の `passwd` コマンドでは、以前 `nispasswd` で行なっていた操作がすべて行えます。NIS+ の名前空間に特有な操作を行うには、`passwd -r nisplus` を使用します。

パスワードの変更は以下の手順で行います。

1. システムプロンプトから `passwd` コマンドを実行します。
2. `Enter login password` **(多少異なる場合がある)** プロンプトで、従来のパスワードを入力します。

   キー入力の内容は画面には表示されません。

   - 「`Sorry: less than` *N* `days since the last change`」というメッセージが表示されたら、「現在使用中のパスワードは、作成されてからまだ十分な時間が経過していないため変更できない」ということを意味します。この場合はシステムプロンプトに戻ります。システム管理者に「パスワードは、作成後何日経過すれば変更できるのか」を問い合わせてください。
   - 「`You may not change this password`」というメッセージが表示された場合は、変更がネットワーク管理者によって禁止されているということを意味します。
3. `Enter new password` プロンプトで、新しいパスワードを入力します

   キー入力の内容は画面には表示されません。

   この時点で、新しいパスワードが必要条件を満たしているかどうかがシステムによって確認されます。

   - 満たしている場合は再入力を求められます。
   - 満たしていない場合は、その旨を知らせるメッセージが表示されます。このときは、必要条件を満たす別のパスワードを入力する必要があります。

   パスワードの必要条件については、299 ページの「パスワードの必要条件」を参照してください。
4. `Re-enter new password` プロンプトで、新しいパスワードを再入力します。

   キー入力の内容は画面には表示されません。

   1 回目と 2 回目で入力したパスワードが異なっていた場合は、手順 1 から繰り返すようにプロンプトが表示されます。

注 – root のパスワードを変更した場合は、その直後に必ず chkey -p を実行する必要があります (詳細については、256 ページの「ルートからのルート鍵の変更」、および、257 ページの「別のマシンからルート鍵を変更する手順」を参照してください)。root のパスワードを変更したあと chkey -p を実行しないと、root で正常にログインできなくなります。

このメッセージは、「パスワードが有効期限を過ぎている」ということを意味します。つまり、パスワードを作成してから時間が経ちすぎているので、すぐに作成し直す必要があるということです (新しいパスワードを作成する場合の必要条件については、298 ページの「パスワードの選択」を参照してください)。

この場合、新しいパスワードの作成は以下の手順で行います。

1. Enter login password (多少異なる場合がある) プロンプトで、従来のパスワードを入力します。
   キー入力の内容は画面には表示されません。
2. Enter new password プロンプトで、新しいパスワードを入力します
   キー入力の内容は画面には表示されません。
3. Re-enter new password プロンプトで、新しいパスワードを再入力します。
   キー入力の内容は画面には表示されません。

パスワード変更の失敗

システムの中には、パスワード変更の試行回数、所要時間に制限を設けているものもあります (他の人が試行錯誤によって勝手にパスワードを変更してしまうのを防ぐため)。

パスワード変更、ログインの試行回数、または所要時間が指定の範囲を超えた場合は、「Too many failures - try later」、または「Too many tries: try again later」といったメッセージが表示されます。この種のメッセージが表示されると、一定の時間 (システム管理者が指定) が経過するまでログイン、パスワードの変更は行えなくなります。

パスワードの選択

コンピュータのセキュリティの侵害の例としては、「他のユーザーのパスワードを推測によって盗む」というものが多くみられます。passwd コマンドには、パスワードを推測しにくいものにするための基準がいくつか設けられていますが、ユーザーに関していくつかの情報を得るだけでパスワードがわかってしまう人もいるのです。したがって、「自分にとって覚えやすく他人にとって推測しにくい」というのが良いパスワードです。逆に悪いパスワードとは、「自分にとって覚えにくく (メモしないと覚えられない)、自分を知る他人にとっては推測しやすい」というものです。

パスワードの必要条件

パスワードの必要条件は以下のとおりです。

- 「長さ」は 6 文字以上にします (デフォルト)。また意味を持つのは最初の 8 文字だけです (つまり、「8 文字より長いパスワードを作成することはできるが、システムは最初の 8 文字のみをチェックする」ということです)。パスワードの長さの最小値 (文字数) はシステム管理者によって変更される可能性があり、システムによっては 6 文字ではない場合があります。
- 「英文字」2 つ以上 (大文字、小文字どちらでも良い) と数字または記号 (`@`、`#`、`%` など) 1 つ以上で構成します。つまり、`dog#food`、`dog2food` といったパスワードは使用できますが、`dogfood` というパスワードは使用できません。
- 「ログイン ID と同じにならないようにします」。またログイン ID の文字を並べ替えたものも使用できません (この場合、大文字と小文字は同じものとして考える)。大文字と小文字は区別されません。たとえば、ログイン ID が `Claire2` の場合は、 `e2clair` はパスワードとして使用できません。
- 「古いパスワードと同じにならないようにします」。古いパスワードを、少なくとも 3 文字以上入れ替える必要があります (この場合、大文字と小文字は同じものとして考える)。古いパスワードが `Dog#fooD` なら、新しいパスワードにたとえば `dog#Meat` は使用できますが、`daT#Food` は使用できません。

悪いパスワードの例

悪いパスワードの例としては以下のようなものが考えられます。

- 自分の名前をもとにしたもの
- 家族やペットの名前
- 運転免許証の番号
- 電話番号
- 社会保険の番号
- 従業員番号
- 趣味に関係のある名前
- 季節に関係のある名前 (たとえば 12 月に `Santa` を使うなど)
- 普通の辞書に載っている単語

良いパスワードの例

良いパスワードの例としては以下のようなものが考えられます。

- フレーズに数字や記号を加えたもの (例：`beam#meup`)
- フレーズを構成する単語の頭文字に番号や記号を加えた意味のない言葉 (例：SomeWhere Over The RainBow の頭文字に 7 を加えた `swotrb7`)
- 単語の一部を数字や記号に置き換えたもの (例：`snoopy` ではなく、`sn00py` にする)

パスワードの管理

この節では、NIS+ 名前空間でパスワードを管理する方法について説明します。この節では、NIS+ セキュリティシステムと、そのログインパスワードについて十分に理解していることを想定しています (NIS+ セキュリティシステムについては、第 11 章を参照してください)。

注 – 現在の `passwd` コマンドでは、以前 `nispasswd` で行なっていた操作がすべて行えます。NIS+ の名前空間に特有な操作を行うには、`passwd -r nisplus` を使用します。

---

nsswitch.conf ファイルの必要条件

`passwd` コマンドの使用や、パスワードの使用期間に関する設定を正しく行うには、`nsswitch.conf` ファイル中の `passwd` エントリがすべてのマシンにおいて正しくなければなりません。`passwd` コマンドが「パスワード情報をどこに要求するか」および「パスワード情報をどこで更新するか」は、このエントリによって決定されます。

`passwd` エントリの設定として考えられるのは、以下の 5 種類だけです。

- `passwd: files`
- `passwd: files nis`
- `passwd: files nisplus`
- `passwd: compat`
- `passwd: compatpasswd_compat: nisplus`

注意 – 使用しているネットワークのマシン上に存在するすべての `nsswitch.conf` ファイルは、必ず上記の `passwd` 構成のうちの 1 つを使用していなければなりません。別の方法で、`passwd` エントリを構成した場合、ユーザーがログインできない可能性があります。

---

nispasswd コマンド

現在の `passwd` コマンドでは、以前 `nispasswd` で行なっていた操作がすべて行えます。コマンド行で実行するときは、`nispasswd` ではなく `passwd` を使用します。

旧バージョンとの互換性を確保するため、`nispasswd` も完全な形で残っている点に注意してください。

yppasswd コマンド

現在の passwd コマンドでは、以前 yppasswd で行なっていた操作がすべて行えます。コマンド行で実行するときは、yppasswd ではなく passwd を使用します。

旧バージョンとの互換性を確保するため、yppasswd も完全な形で残っている点に注意してください。

passwd コマンド

passwd コマンドでは、パスワードに関する様々な操作が行えます。現在の passwd コマンドは、nispasswd コマンドの代わりとして使用できます。従来 nispasswd で行なっていた操作にも、passwd コマンドを使用するようにしてください。passwd コマンドのフラグ、オプション、引数の詳細については、マニュアルページを参照してください。

一般ユーザーが passwd コマンドで行える操作には以下のものがあります。

- 自らのパスワードの変更
- 自らのパスワード情報の表示

管理者が passwd コマンドで行える操作には以下のものがあります。

- 他のユーザーにログイン時のパスワード変更を強制する
- 他のユーザーのパスワードをロック (使用不可能に) する
- 「パスワード作成後、どのくらいの期間変更禁止にするか」を指定する
- 「パスワードが間もなく無効になる」という警告を出すタイミングを指定する
- パスワードの有効期間を指定する

passwd コマンドと nsswitch.conf ファイル

passwd などのコマンドがパスワード情報をどこから得て、どこに保存するのかということは、nsswitch.conf ファイルで設定します。nsswitch.conf ファイルの passwd エントリの設定に使用する文字列はそれぞれ以下のことを意味します。

- nisplus。パスワード情報の獲得、変更、保存は、該当するドメインの passwd テーブルおよび cred テーブルで行う
- nis。パスワード情報の獲得、変更、保存は、passwd マップで行う
- files。パスワード情報の獲得、変更、保存は、/etc/passwd ファイルおよび /etc/shadow ファイルで行う

passwd -r オプション

passwd コマンドの -r nisplus、-r nis、-r files といった引数は、nsswitch.conf ファイルの設定よりも優先されます。これらの引数をつけて passwd コマンドを実行すると、「nsswitch.conf ファイルより優先される」とい

うことを知らせる警告メッセージが表示されます。警告メッセージ表示後も操作を続行すると、nsswitch.conf ファイルのシーケンスは無視され、-r によって指定された場所でパスワード情報の更新が行われます。

たとえば、nsswitch.conf ファイルにおいて passwd エントリが以下のように指定されている場合を考えてみましょう。

```
passwd: files nisplus
```

この場合、passwd コマンドを -r を使用しないで実行すると、パスワード情報のソース (情報の獲得、変更、保存が行われる場所) は /etc/passwd ファイルになります。しかし -r nisplus オプションを使用して passwd コマンドを実行すると、パスワード情報のソースは /etc/passwd ファイルから NIS+ の passwd テーブルに変更されます。

-r オプションは、「検索シーケンスが誤っていて nsswitch.conf ファイルが使用できない」という場合にのみ使用するようにします。たとえば、2 カ所に格納されているパスワード情報を更新する必要がある場合、1 つ目については nsswitch.conf ファイルで指定されているソースを使用できますが、2 つ目については別のソースを使用する必要があります。

変更しようとすると、次のメッセージが表示されます。

```
Your specified repository is not defined in the nsswitch file!
```

このメッセージは、「パスワード情報の更新は、-r オプションで指定された場所において行われるが、nsswitch.conf ファイルでその場所がソースとして指定されるまでは更新の影響がまったく現れない」ということを意味します。 たとえば、nsswitch.conf ファイルにおいて passwd: files nis という指定が行われているときに、-r nisplus オプションでパスワード使用期間の設定を行なったとします (つまり設定は NIS+ の passwd テーブルにおいて行われることになる)。

パスワード情報のソースが nsswitch.conf ファイルにおいて NIS+ passwd テーブル以外の場所に指定されているため、この設定による影響はまったく現れません。

passwd コマンドと NIS+ 環境

この章で「NIS+ 環境」とは、「nsswitch.confファイルでパスワード情報のソースが nisplus に設定されている」、「passwd コマンドが -r nisplus という引数をつけて実行されている」という状況を指します。

passwd コマンドと資格

passwd コマンドは NIS+ 環境 (前節参照) で実行された場合、ユーザーに資格があってもなくても機能するよう設計されています。ただし資格のないユーザーが passwd コマンドで行えるのは、自らのパスワードの変更だけです。他のパスワード操作は、資格のある (認証された)、必要なアクセス権を持った (承認された) ユーザーだけが行えます。

passwd コマンドとアクセス権

承認およびアクセス権については、ユーザーがすべて適切な資格を持っているという前提で説明をします。

通常の NIS+ 環境では、passwd テーブルの所有者はいつでも制約なしにパスワード情報の変更ができます (デフォルトの場合)。つまり passwd テーブルの所有者は、読み取り、作成、変更、削除に関して完全に承認されている (アクセス権を与えられている) ということになります。また所有者は以下のことも行えます。

- nischown コマンドを使用して、テーブルの所有権を別のユーザーに与える
- テーブルのグループ、その他、未認証といったクラスに、読み取り権、作成権、変更権、削除権などを与える (この種の権利をその他クラス、未認証クラスに与えると NIS+ のセキュリティが弱くなる)
- 任意のクラスに与えられたアクセス権を、nisdefaults、nischmod、nistbladm などのコマンドを使用して変更する

注 – 与えられているアクセス権には関わりなく、その他クラス、未認証クラスのユーザーはすべてパスワードの使用期間の制約に従います。つまり、自分のパスワードであろうと他のユーザーのパスワードであろうと、作成されてから一定の時間が経過するまでは変更ができないということです。また有効期間の過ぎたパスワードを変更しなければならないという点も、グループ、その他、未認証といったクラスのメンバーすべてに共通です。ただし、パスワード使用期間に関する上記のような制約は、passwd テーブルの所有者には適用されません。

NIS+ 環境で passwd コマンドを使用する場合、行おうとする操作に関する承認 (アクセス権) が必要です。

表 16–1 passwd コマンドに関するアクセス権

操作の種類	必要な権利	アクセス対象となるオブジェクト
情報を表示する	読み取り権	passwd テーブルのエントリ
情報を更新する	変更権	passwd テーブルのエントリ
情報を追加する	変更権	passwd テーブル

passwd コマンドと鍵

NIS+ 環境で passwd コマンドを使用して主体のパスワードを変更しようとすると、主体の非公開鍵が cred テーブル中で更新されます。

- cred テーブルの DES エントリに対してユーザーが変更権を持っていて、ログインパスワードと Secure RPC パスワードが同じであれば、passwd コマンドによって cred テーブルの非公開鍵が更新されます。

- cred テーブルの DES エントリに対する変更権がユーザーになく、ログインパスワードと Secure RPC パスワードが異なっている場合は、passwd コマンドによってパスワードだけが変更され、非公開鍵は変更されません。

  つまり cred テーブルの非公開鍵が、passwd テーブルに格納されているものとは異なったパスワードで作られることになります。この場合は、chkey コマンドを実行する、ログイン後に毎回 keylogin を実行するといった方法で、鍵を変更する必要があります。

passwd コマンドと他のドメイン

他のドメインの passwd テーブルに対して操作をするには、passwd コマンドを以下のように使用します。

```
passwd [options] -D domainname
```

nistbladm コマンド

nistbladm は、passwd テーブルをはじめとする NIS+ テーブルに関する情報を作成、変更、表示するのに使用するコマンドです。

注意 – nistbladm コマンドを使用してパスワード操作をするには、nistbladm を passwd テーブルのシャドウ列に適用する必要があります。nistbladm をシャドウ列に適用するのは複雑で微妙な作業になります。passwd コマンド、admintool、または Solstice AdminSuite ツールで容易に行える操作には、nistbladm コマンドを使用しない方が良いでしょう。

つまり以下のような操作には、nistbladm ではなく passwd コマンドや Solstice AdminSuite ツールを使用してください。

- パスワードの変更
- パスワードの使用期間の設定
- パスワード作成後から変更が可能になるまでの時間の設定
- 「パスワードが間もなく無効になる」という警告が表示されるタイミングの設定
- パスワードの使用期間に関する設定の解除

nistbladm コマンドには以下の機能があります。

- 新しい passwd テーブルエントリの作成
- 既存のエントリの削除
- passwd テーブル中の UID フィールド、GID フィールドの更新
- passwd テーブルの、アクセス権などセキュリティに関する属性の更新

- ユーザーアカウントに関して「どれだけの期間使用できるか」、「どれだけの期間使用されなければ無効になるか」を設定する (316 ページの「パスワード使用権の有効期限」、318 ページの「ログインの間隔の最大値の指定」を参照)

nistbladm とシャドウ列のフィールド

nistbladm コマンドでは、シャドウ列の様々なフィールドの値を指定することによってパスワードパラメータの設定を行います。シャドウ列のフィールドの値は、以下のような形式で設定します。

```
                          最小値     警告      期限
                            |         |         |
nistbladm -m shadow=n1 :n2 :n3 :n4 :n5 :n6 :n7 [name=login], passwd.org_dir
                       |     |       |      |
                  最終変更日 最大値    間隔    未使用
```

引数の意味はそれぞれ以下のとおりです。

- 「n1 最終変更日」は、パスワードが最後に変更されたときの日付です。1970 年 1 月 1 日からの日数で表されます。このフィールドの値は、パスワードが変更されると自動的に更新されます (日数に関する詳細な情報は、307 ページの「nistbladm と日数」に示されています)。このフィールドが空白であったり、指定されている値が 0 の場合は、過去に一度もパスワードの変更が行われていないことを意味します。

  このフィールドの値は、残りのフィールドの設定の基礎となることに注意してください。このフィールドの値が不当に変更されると、残りのフィールドの設定も不当なものになります。

- 「n2 最小値」は、パスワードが変更されてから、次の変更が可能になるまでの期間です。たとえば、パスワード作成時の最終変更日フィールドの値が 9201 (1970 年 1 月 1 日から 9201 日経過したことを示す) で、最小値フィールドの値が 8 だとすると、9209 日目まではパスワードの変更ができません (詳細については、313 ページの「パスワードの変更禁止期間の設定」を参照)。

  この値としては以下の 3 とおりが考えられます。

  - 「0」は、変更禁止期間がないことを意味します。
  - 「0 より大きい数字」は、この数字がパスワードの変更禁止期間 (単位：日) になります。
  - 「最大値 (n3) より大きい値」はこのフィールド値が最大フィールドの値より大きい場合、パスワードを変更できません。変更しようとすると、「`You may not change this password`」というメッセージが表示されます。

- 「n3 最大値」は、パスワードが作成されてから、使用できなくなるまでの日数を示します。この日数を過ぎると、ログインの際新しいパスワードの設定が強制されます。たとえば、パスワード作成時の最終変更日の値が 9201 の最大値が 30 である場合、9231 (9201+30) 日目が経過した後は、ログイン時に新しいパスワードの設定が強制されます (詳細については、313 ページの「パスワードの有効期間の設定」を参照)。

この値としては以下の 3 とおりが考えられます。

- 「0」は、次回ログイン時のパスワード変更を強制する。使用期間に関する設定は解除される
- 「0 より大きい値」は、パスワードの有効期間の日数
- 「-1」は、パスワードの使用期間に関する設定を解除します。つまり **passwd -x -1** *username* というコマンドを実行すると、以前に行われたパスワードの有効期間の設定が解除されることになります。このフィールドを空白にしておくと、-1 として扱われます。

- 「n4 警告」は、「パスワードが間もなく無効になる」という警告メッセージが表示されるタイミングを示します。パスワードが作成されてからの日数で指定します。たとえば、最終変更日時フィールドの値が 9201 、最大値フィールドの値が 30、警告フィールドの値が 5 であると仮定します。この場合、日付が 9226 日 (9201+30-5=9226) 以降になると、ログインするたびに「パスワードを変更してください」という警告が表示されます。パスワードの警告時期の詳細については、314 ページの「警告期間の設定」 を参照してください。

  この値としては以下の 2 とおりが考えられます。

  - 「0」は、警告メッセージが表示される期間がないことを示します。
  - 「0 より大きい値」は、この数字が、パスワードが作成されてから警告メッセージが表示されるまでの期間 (日数) になります。

- 「n5 間隔」は、ログインとログインの間隔 (日数) の最大値です。ここで指定された日数を超える期間ログインを行わないと、ログインができなくなります。たとえば、間隔に 6 を指定した場合、6 日間ログインを行わないと 7 日目にはログインができなくなります (詳細については、318 ページの「ログインの間隔の最大値の指定」を参照)。

  この値としては以下の 2 とおりが考えられます。

  - 「マイナス 1 (-1)」は、最大ログイン間隔機能を解除にします。任意の日数の間オフにすることができ、ログイン権限を失うことはありません。これはデフォルトです。
  - 「0 より大きい値」は、最大ログイン間隔 (日数) を指定します。

- 「n6 期限」は、パスワードの有効期限を示します。1970 年 1 月 1 日からの日数で表されます。この日を過ぎると、ログインができなくなります。たとえば、期限が 9739 (1995 年 9 月 1 日) に設定されている場合、1995 年 9 月 2 日 (GMT) になるとログインができなくなります。このときログインを試行すると 「`Login incorrect`」というメッセージが表示されます (詳細については、316 ページの「パスワード使用権の有効期限」を参照)。

  この値としては以下の 2 とおりが考えられます。

  - 「-1」は、有効期限の設定を解除します。パスワードが有効期限を過ぎている場合でも、-1 を設定すると再び使えるようになります。有効期限を設定したくないときは、`-1` を指定するようにしてください。
  - 「0 より大きい値」は、有効期限の日付 (「1970 年 1 月 1 日から数えて何日目か」) です。現在以前の日付を設定すると、パスワードはすぐに無効になります。

- 「n7 未使用」は、このフィールドは現在使用されていません。値を設定しても無視されます。
- 「*login*」はユーザのログイン ID

注意 – `nistbladm` を使用して `passwd` テーブルのシャドウ列を設定する場合、すべてのフィールドに適切な値を指定する必要があります。空白のまま残したり、0 を入力したりしても「変更なし」という意味にはなりません。

ユーザー amy が、パスワードの最終変更日が 1995 年 5 月 1 日 (1970 年 1 月 1 日から数えて 9246 日目)、パスワード作成後の変更禁止期間が 7 日、パスワードの有効期間が 30 日、「パスワードが間もなく無効になる」という警告が表示されるのがパスワード作成から 26 日目以降、ログインとログインの間隔の最大値が 15 日、アカウントの有効期限が (1970 年 1 月 1 日から) 9255 日目という設定にするには、以下のように入力します。

`nistbladm` と日数

パスワードの使用期間に関するパラメータは、日数で表されるものがほとんどです。日数を指定する際には以下の規則を守る必要があります。

- 1970 年 1 月 1 日を 0 とします。つまり 1970 年 1 月 2 日は 1 になります。
- NIS+ では、日付の計算、カウントに GMT (Greenwich Mean Time) が使用されます。つまり日付が変更されるのは、GMT の午前 0 時です。
- 日数の指定には整数を使用します。分数は使用できません。
- パスワードのロックなどの動作は、指定された日付に行われます。たとえば、パスワード使用権の有効期限を1995 年 1 月 2 日 (1970 年 1 月 1 日から 9125日目) に設定した場合、この日は「ユーザーがパスワードを使用できる最後の日」となります。パスワードが使用できなくなるのは、次の日からです。

最終変更日、期限のどちらのフィールドも、入力するのは 1970 年 1 月 1 日から数えた日数です。たとえば :

表 16–2 1970 年 1 月 1 日からの日数

日付	日数
1970 年 1 月 1 日	0
1970 年 1 月 2 日	1
1971 年 1 月 2 日	365
1997 年 1 月 1 日	9863

関連コマンド

passwd および nistbladm コマンドと類似の機能を持つコマンドは他にもあります。それぞれについて表 16–3 にまとめてあります。

表 **16–3** 関連コマンド

コマンド	説明
yppasswd	現在は passwd コマンドにリンク。yppasswd を起動すると passwd コマンドが起動される
nispasswd	現在は passwd コマンドにリンク。nispasswd を起動すると passwd コマンドが起動される
niscat	テーブルの内容を表示するのに使用

パスワード情報の表示

ドメイン中のユーザーのパスワード情報を表示するには、passwd コマンドを使用します。情報は、全ユーザーについて同時に表示することも、ユーザーごとに表示することもできます。

自分のパスワード情報

```
passwd -s
```

ドメインの全ユーザーの情報

```
passwd -s -a
```

個々のユーザー

```
passwd -s username
```

表示できるのは、エントリおよび列のうち読み取り権を持っているものだけです。エントリの表示形式は以下のとおりです。

- 使用期間に関する情報が省略されたもの： *username status*
- 使用期間に関する情報が付加されたもの： *username status mm/dd/yy min max warn expire inactive*

表 **16–4** NIS+ パスワード情報表示の形式

列	説明	参照箇所
username	ユーザーのログイン名	

表 **16–4** NIS+ パスワード情報表示の形式　(続き)

列	説明	参照箇所
status	パスワードの状態。PS は、そのアカウントにパスワードがあることを示す。LK は、そのパスワードがロックされていることを示す。NP は、そのアカウントにパスワードがないことを示す	310 ページの「パスワードのロック」
mm/dd/yy	パスワードが最後に変更された日付 (GMT で表す)	
min	パスワード作成後の変更禁止期間 (日数)	313 ページの「パスワードの変更禁止期間の設定」
max	パスワードの有効期間 (日数)	313 ページの「パスワードの有効期間の設定」
warn	「パスワードが間もなく無効になる」という警告が表示されるまでの日数	314 ページの「警告期間の設定」
expire	パスワードの有効期限 (日付)	316 ページの「パスワード使用権の有効期限」
inactive	ログインとログインの間隔の最大値 (日数)。ログインとログインの間隔がここで指定した日数を超えると、ログインができなくなる	318 ページの「ログインの間隔の最大値の指定」

別のドメインの `passwd` テーブルのエントリを表示するには、`-D` オプションを使用します。

ドメイン中の全ユーザー

```
passwd -s -a -D domainname
```

個々のユーザー

```
passwd -s -D domainname username
```

パスワードの変更

新しいパスワードを作成するときは、299 ページの「パスワードの必要条件」に示された必要条件を満たすようにします。

自分のパスワードの変更

自分のパスワードを変更するには、以下のように入力します。

```
station1% passwd
```

「古いパスワードの入力」、「新しいパスワードの入力」、「新しいパスワードの再入力 (確認のため)」という順にプロンプトが表示されるので、それに従って作業します。

他人のパスワードの変更

他人のパスワードを変更するには、以下のように入力します。

同じドメイン内の他のユーザー

passwd *username*

他のドメインのユーザー

passwd -D *domainname username*

NIS+ 環境 (302 ページの「passwd コマンドと NIS+ 環境」参照) で passwd コマンドを使用して他人のパスワードを変更する場合、passwd テーブルにおける該当ユーザーエントリへの変更権が必要になります。つまり、該当する passwd テーブルに対して変更権を持つグループのメンバーになる必要があります。このとき、該当ユーザーの古いパスワードや自分のパスワードを入力する必要はありません。確認のため新しいパスワードの入力を求めるプロンプトが 2 回表示されます。一致しない場合は、さらに 2 回入力する必要があります。

root のパスワードの変更

passwd コマンドで root のパスワードを変更した場合は、その直後に chkey -p を実行する必要があります。chkey -p を実行しないと、root で正しくログインできなくなります。

root のパスワードの変更手順は以下のとおりです。

1. **root** でログインします。
2. passwd コマンドで **root** のパスワードを変更します。
   nispasswd は使用しないでください。
3. chkey -p を実行します。
   必ず -p オプションを使用します。

パスワードのロック

NIS+ 環境 (302 ページの「passwd コマンドと NIS+ 環境」を参照) で passwd コマンドを使用してパスワードのロックができるのは、該当ユーザーの passwd テーブル中のエントリに対する変更権を持っている管理者 (グループのメンバー) です。パスワードがロックされると、そのアカウントは使用できなくなります。また、パスワードがロックされているユーザー名を使ってログインをしようとすると、「Login incorrect」というメッセージが表示されます。

該当ユーザーがすでにログインしている場合、パスワードをロックすることによる影響は現れないという点に注意してください。ただ、パスワードの入力が必要になる login、rlogin、ftp、telnet などの機能は使用できなくなります。

すでにログインしているユーザーのパスワードをロックしても、そのユーザーが passwd コマンドでパスワードを変更するとロックは解除されます。

以下のようなことが有効です。

- 休暇などで不在になっているユーザーのパスワードを一時的にロックする。他の人に不正に使用されることを防止できる
- セキュリティ上の問題が発生している可能性があれば、すぐに一部のユーザーのパスワードをロックする
- 解雇になったユーザーのパスワードをすぐにロックする。ユーザーのアカウントを削除するより速く容易に行える上、そのアカウントに格納されているデータをそのまま容易に保管できる
- UNIX プロセスにパスワードを設定している場合には、この種のパスワードもロックできる。パスワードがロックされてもプロセスの実行はできるが、プロセスを使用してログインすることは (たとえパスワードを知っていても) できなくなる。多くの場合、プロセスは NIS+ 主体の形では設定されない。しかしプロセスのパスワード情報は、/etc ディレクトリのファイルに保存される。このとき /etc に格納されたパスワードをロックするにはファイルモードで passwd コマンドを実行する必要がある

パスワードのロックは以下のように行います。

passwd -l *username*

パスワードロックの解除

パスワードのロックは、パスワードを変更すれば解除されます。この場合、「変更」とは必ずしもパスワードを新しいものに変えるという意味ではなく、パスワード変更の手順を踏むということです (元と同じパスワードに「変更」することも可能です)。ただし、新しいパスワードに変更することも可能です。

たとえば、jody というユーザーのパスワードのロックを解除するには、以下のように入力します。

```
station1% passwd jody
```

パスワードの使用期間に関する設定

設定により、パスワードの定期的な変更をユーザーに強制できます。

たとえば、以下のような設定が可能です。

- 次回のログインの際に強制的にパスワードを変更させる (詳細については、312 ページの「パスワードの強制的な変更」を参照)
- パスワードの有効期間 (日数) を設定する (詳細については、313 ページの「パスワードの有効期間の設定」を参照)

- パスワード作成後の変更禁止期間を設定する (詳細については、313 ページの「パスワードの変更禁止期間の設定」を参照)
- パスワードの有効期間が終わる前に「パスワードが間もなく無効になる」という警告メッセージが (ログイン時に) 表示されるタイミングを設定する (詳細については、314 ページの「警告期間の設定」を参照)
- ログインとログインの間隔の最大値 (日数) を指定する。指定された日数を超えてログインが行われないと、パスワードがロックされる (詳細については、318 ページの「ログインの間隔の最大値の指定」を参照)
- パスワードの有効期限 (日付) を設定する。この日を過ぎると、該当ユーザはシステムにログインできなくなる (詳細については、316 ページの「パスワード使用権の有効期限」を参照)

さまざまな最大日数や期間に到達していても、それまでにすでにログインしているユーザーは、上記の機能には影響されないことを覚えておいてください。そのまま普通に動作し続けます。

影響が現れるのは、次回のログイン時か、ログインを必要とする機能 (以下に例を示す) を使用した時です。

- `login`
- `rlogin`
- `telnet`
- `ftp`

パスワードの使用期間に関する設定は、ユーザーごとに行います。パスワードの使用期間に関する必要条件はユーザーによって違っている可能性があります。ユーザーは、一般のデフォルト設定を適用することもできます。パラメータについての詳細については、311 ページの「パスワードの使用期間に関する設定」を参照してください。

パスワードの強制的な変更

次回ログイン時のパスワード変更をユーザーに強制するには、以下の 2 種類の方法があります。

使用期間に関する設定を引き続き有効にする場合

`passwd -f` *username*

使用期間に関する設定を解除する場合

`passwd -x 0` *username*

パスワードの有効期間の設定

パスワードの有効期間を設定するには、passwd コマンドの引数 *max* を使用します。有効期間は日数で指定します。有効期間が終了した後は、新しいパスワードをユーザーが、作成しなければなりません。有効期間終了後に同じパスワードでログインしようとしても、「Your password has been expired for too long」というメッセージが表示され、新しいパスワードを作成しない限りログインを完了できません。

引数 *max* は以下の形式で使用します。

passwd -x *max username*

引数の意味はそれぞれ以下のとおりです。

- *username* は、ユーザーのログイン ID
- *max* は、以下のうちいずれかになります。
  - 「0 より大きい値」は、パスワードの有効期間の日数
  - 「0」は、次回ログイン時のパスワード変更を強制する。使用期間に関する設定は解除される
  - 「-1」は、パスワードの使用期間に関する設定を解除します。つまり **passwd -x -1** *username* と入力すると、使用期間に関して該当ユーザーに行われていた設定はすべて解除されます。

たとえば、schweik というユーザーに 45 日ごとのパスワード変更を強制するには、以下のように入力します。

```
station1% passwd -x 45 schweik
```

パスワードの変更禁止期間の設定

パスワード作成後の変更禁止期間を設定するには、引数 *min* を使用します。変更禁止期間が終了しないうちにパスワードを変更しようとすると、「Sorry less than N days since the last change」というメッセージが表示されます。

引数 *min* は以下の形式で使用します。

passwd -x *max* **-n** *min username*

引数の意味はそれぞれ以下のとおりです。

- *username* は、ユーザーのログイン ID
- *max* は、パスワードの有効期間 (前のセクションを参照)
- *min* は、パスワード作成後の変更禁止期間

たとえば、ユーザー eponine のパスワードの有効期間を 45 日間、変更禁止期間を 7 日間に設定する場合は、以下のように入力します。

```
station1% passwd -x 45 -n 7 eponine
```

min 引数には以下の規則があります。

- *min* 引数は必ずしも使用しなくて良い (つまり、パスワード変更禁止期間は必ずしも指定しなくて良い)
- *min* 引数は、必ず `-max` 引数と組み合わせて使用する。つまり、変更禁止期間を設定するには、有効期間も設定する必要がある
- *min* に *max* より大きい値を設定すると (例：`passwd -x 7 -n 8`)、ユーザーはパスワードを変更できなくなる。変更しようとすると、「`You may not change this password`」というメッセージが表示される。*min* に *max* より大きな値を設定すると、次の 2 つの効果が得られる
  - ユーザーによるパスワードの変更を禁止する。つまり、管理特権を有する一部の人物以外は、パスワードを変更できなくなる。たとえば、複数のユーザーに共通のグループパスワードを割り当て、そのパスワードに対して *min* 値より大きな *max* 値を設定すると、グループ内のユーザーは誰一人としてパスワードを変更できなくなる
  - パスワードは *max* 値で設定される期限内のみ有効となる。しかも、*min* 値が *max* 値より大きいため、ユーザーはパスワードを変更できない。つまり、*max* 値の示す有効期限が満了する前にパスワードの期限延長を図る手だては、ユーザーには一切与えられない。実際、有効期限が満了した後は、管理者の介在なしにユーザーはログインできなくなる

警告期間の設定

パスワードの有効期限以前の一定期間、ログイン時に「`Your password will expire in N days`」(*N* は日数) というメッセージが表示されるようにするには、引数 *warn* を使用します。

たとえば、パスワードの有効期限が 30 日間 (引数 `-max` で指定する) で、warn が 7 に設定されている場合、パスワード作成後 24 日目のログイン時に「`Your password will expire in 7 days`」という警告メッセージが表示されます。また翌日 (パスワード作成後 25 日目) には、警告メッセージは「`Your password will expire in 6 days`」となります。

警告メッセージは、電子メールで送信されるものでも、ユーザーのコンソールウィンドウに表示されるものでもないという点に注意してください。表示されるのは、ユーザーのログイン時のみです。つまり指定の期間ユーザーがログインしなければ警告メッセージは表示されません。

warn の値は、*max* の値との「関連によって機能する」ものであるという点に注意してください。つまり、「*max* によって設定される有効期限から逆算されるもの」ということです。*warn* の値が 14 であれば、「`Your password will expire in N days`」というメッセージの表示が開始されるのは、有効期限の 2 週間前です。

warn の値は *max* の値との関連によって機能します。*max* 値の指定がある場合のみ有効です。そのため、*max* 値の指定がない場合は、*warn* 値は意味がなくなります。

引数 *warn* は以下の形式で使用します。

passwd -x *max* **-w** *warn username*

引数の意味はそれぞれ以下のとおりです。

- *username* は、ユーザーのログイン ID
- *max* は、パスワードの有効期限 (日数) (313 ページの「パスワードの有効期間の設定」を参照)
- *warn* は、「有効期限の何日前から警告メッセージの表示を開始するか」を指定する

たとえば、nilovna というユーザーの、パスワードの有効期間を 45 日間とし、警告メッセージの表示を有効期限の 5 日前から開始する場合、以下のように入力します。

```
station1% passwd -x 45 -w 5 nilovna
```

warn 引数には以下の規則があります。

- *warn* 引数は必ずしも使用しなくて良い。*warn* の値を設定しなければ、警告メッセージは表示されない
- *warn* 引数は、必ず *max* 引数と組み合わせて使用する。つまり、警告メッセージの表示期間を指定するには、有効期間の指定が必要になる

注 – Solstice AdminSuite を使用して、*warm* 値をユーザーのパスワード用に設定することもできます。

パスワードの使用期間に関する設定の解除

ユーザーのパスワードの有効期間を解除するには、以下の 2 つの方法があります。

有効期間を取り消す。パスワードはそのまま使用できる

passwd -x -1 *username*

次回ログイン時にパスワードを強制的に変更させた後、有効期間を取り消す

passwd -x 0 *username*

上記の方法では、*max* の値を 0 か -1 に設定する (詳細については、313 ページの「パスワードの有効期間の設定」を参照)

たとえば、ユーザー mendez のパスワードを次回ログイン時に強制的に変更させ、その後パスワードの有効期間を解除するには、以下のように入力します。

```
station% passwd -x 0 mendez
```

注 – Solstice AdminSuite を使用して、このパラメータをユーザーのパスワード用に設定できます。

この値を設定するには、nistbladm コマンドを使用できます。たとえば、ユーザー otsu のパスワードの有効期間を取り消し、変更の必要がないようにするには、以下のように入力します。

```
station1% nistbladm -m 'shadow=0:0:-1:0:0:0:0' [name=otsu],passwd.org_dir
```

nistbladm コマンドの使用方法の詳細については、304 ページの「nistbladm コマンド」を参照してください。

パスワード使用権の有効期限

引数 *expire* には「ユーザーのパスワード使用権がいつ無効になるか」を日付で指定します。ユーザーのパスワード使用権が無効になると、そのユーザーは有効なパスワードを所有できなくなります。該当ユーザーのログインをこの日以降禁止し、システムから締め出します。

たとえば、ユーザー pete の有効期限を 1997 年 12 月 31 日に設定すると、どんなパスワードを使用しても 1998 年 1 月 1 日以降 pete という ID ではログインができなくなります。ログインをしようとしても、「Login incorrect」というメッセージが表示されます。

「有効期間」と「有効期限」の違い

パスワード使用権の有効期限は、パスワードの有効期間とは異なっています。

- 「パスワードの有効期間」。厳密に区別をする必要のない場合には、有効期間が終了した後も変更されないパスワードのことを「有効期限の過ぎたパスワード」と呼ぶことがあります。しかしこのパスワードは、もう一度だけログインに使用できます。ただしその際に変更を強制されます。
- 「パスワード使用権の有効期限」。ユーザーのパスワード「使用権」が無効になると、そのユーザーはすべてのパスワードでログインできなくなります。つまり、ネットワークにログインする権限の有効期限が切れたことになります。

有効期限の設定

パスワード使用権が無効になっても、その影響が現れるのは、該当ユーザーがログインをしようとするときだけです。該当ユーザーがすでにログインしていた場合には、パスワード使用権が無効になったことによる影響は現れません。ただし、rlogin、telnet などログインを必要とする機能は使用できなくなり、いったんログアウトするとログインできなくなります。以上のことから、パスワード使用権に有効期限を設ける場合、毎日のワークセッション終了時には必ずログアウトするようユーザー全員に指示してください。

注 – Solstice AdminSuite ツールを使用して有効期限の設定ができるときは、nistbladm を使用しないでください。Solstice AdminSuite ツールの方が使いやすく、設定を誤る可能性が低くなります。

nistbladm コマンドでパスワード使用権の有効期限を指定するには、以下のように入力します。

```
nistbladm -m 'shadow=n:n:n:n:n:n6:n' [name=login],passwd.org_dir
```

引数の意味はそれぞれ以下のとおりです。

- *login* は、ユーザーのログイン ID です。
- *n* は、シャドウ列の (n5 以外の) フィールドの値です。
- *n6* は、ユーザーのパスワード使用権の有効期限 (日付) を設定します。「1970 年 1 月 1 日から数えて何日目か」で表します (表 16–2 を参照)。このフィールドに指定する値には、以下の 2 種類があります。
  - 「-1」は、有効期限の設定を解除します。ユーザーのパスワードがすでに有効期限を過ぎていても、-1 を設定することによって再び使用できるようになります。有効期限を設定したくないときは、初めから -1 を指定しておきます。
  - 「0 より大きい値」は、有効期限の日付 (「1970 年 1 月 1 日から数えて何日目か」) です。現在以前の日付を指定すると、ユーザーのパスワードはすぐに無効になります。

たとえば、ユーザー pete の有効期限を 1995 年 12 月 31 日に設定する場合は、以下のように入力します。

```
station1% nistbladm -m 'shadow  =n:n:n:n:n:9493:n' \
[name=pete],passwd.org_dir
```

注意 – 空白のフィールドや、値の正しくないフィールドがあると機能しません。

パスワード使用権の有効期限の解除

有効期限を過ぎたパスワード使用権を再び使用できる状態にするには、nistbladm コマンドを使用して該当フィールド (n6) に -1 を指定します。例を以下に示します。

```
station1% nistbladm -m 'shadow=  n:n:n:n:n:-1:n' \
[name=huck],passwd.org_dir
```

また、nistbladm で「n6」フィールドを将来の日付に設定し直すという方法も使用できます。

ログインの間隔の最大値の指定

引数 *inactive* には、「ログインからログインまでの間隔を最大何日あけることができるか」を指定します。この引数に指定された日数を超えてログインが行われなかった場合、該当ユーザーはマシンにログインできなくなります。ログインしようとすると、「Login incorrect」というメッセージが表示されます。

この設定はネットワーク全体ではなく、個々のマシンに対して行われます。したがって、NIS+ 環境においてログイン間隔の最大値を設定するには、該当ユーザーのエントリをホームドメインの passwd テーブルに指定します。この設定はネットワーク上のすべてのマシンで該当ユーザーに対して適用されます。

たとえば、ユーザー sam の最大ログイン間隔を 10 日間に設定した場合を考えてみましょう。sam は 1 月 1 日にマシン A とマシン B にログインをした後、ログアウトしました。続いて 3 日後の 1 月 4 日、sam はマシン B にログインした後ログアウトしました。さらに 9 日後の 1 月13日になると、sam はマシン B にはログインできますがマシン A にはログインできません。マシン B においては最終ログインから 9 日間しか経過していませんが、マシン A においては 13 日間が経過しているからです。

最大ログイン間隔の設定は、まだ一度もログインをしていないマシンには適用されないという点に注意してください。一度もログインをしていないマシンには、引数 *inactive* の設定および他のマシンへのログイン実績がどのようになっていようとログインが可能です。

注意 – 「毎日の作業終了時には必ずログアウトをせよ」という指示がユーザーに対して行われていない場合、最大ログイン間隔の設定はしないでください。この設定はログインにだけ関係しており、他の形でシステムを使用する場合のことは考慮されていません。仮にユーザーが作業終了後もログインしたままの状態でシステムを放置しておくと、ログインが発生しないため指定のログイン間隔をすぐ超えてしまいます。指定のログイン間隔を超えた状態でリブートやログアウトをすると、再びログインできなくなります。

注 – Solstice AdminSuite ツールを使用して最大ログイン間隔の設定ができるときは、nistbladm を使用しないでください。Solstice AdminSuite ツールの方が使いやすく、設定を誤る可能性が低くなります。

最大ログイン間隔の設定を nistbladm コマンドで行う場合は、以下の形式を使用します。

nistbladm -m 'shadow= *n:n:n:n:n5:n:n'* [name=*login*], **passwd.org_dir**

引数の意味はそれぞれ以下のとおりです。

- *login* は、ユーザーのログイン ID です。
- *n* は、シャドウ列の (n5 以外の) フィールドの値です。

- *n5* は、最大ログイン間隔 (日数) です。指定される値には、以下の 2 種類があります。
  - 「マイナス 1 (-1)」は、最大ログイン間隔機能を解除にします。任意の日数の間オフにすることができ、ログイン権限を失うことはありません。これはデフォルトです。
  - 「0 より大きい値」は、最大ログイン間隔 (日数) を指定します。

たとえば、ユーザー sam の最大ログイン間隔を 7 日間に指定する場合は、以下のように入力します。

```
station1% nistbladm -m 'shadow= n:n:n:n:n:7:n:n' [name=sam],passwd.org_dir
```

最大ログイン間隔の設定を解除する (ログインできなくなったユーザーを再びログインできるようにする) には、nistbladm で *inactive* を -1 に指定します。

パスワードの使用規則の設定 (およびそのデフォルト)

パスワードの使用規則に関する設定およびそのデフォルトについて説明します。

/etc/defaults/passwd ファイル

パスワード情報の獲得先が nsswitch.conf ファイルで files に指定されているすべてのユーザーのパスワードについて、4 つのデフォルト設定を行うためのファイルです。/etc/defaults/passwd ファイルで行われたデフォルト設定は、/etc ディレクトリのファイルからパスワード情報を獲得しているユーザーにだけ適用されます。パスワード情報の獲得先が NIS マップあるいは NIS+ テーブルであるようなユーザーには適用されません。NIS+ サーバー上の /etc/defaults/passwd ファイルは、パスワード情報をローカルファイルから獲得しているローカルユーザーにだけ影響を与えます。NIS+ 環境または、nsswitch.conf ファイルでパスワード情報の獲得先が nis、nisplus に設定されているユーザーには影響を与えません。

/etc/defaults/passwd ファイルで行われる「パスワードに関する 4 つのデフォルト設定」とは、具体的には以下のものを指します。

- 有効期間 (週単位)
- 変更禁止期間 (週単位)
- 「パスワードが間もなく無効になる」という警告メッセージの表示される期間 (週単位)
- 最低文字数

/etc/defaults/passwd ファイルのデフォルト設定には、以下の規則があります。

- パスワード情報をローカルの /etc ファイルから得ているユーザーの場合、passwd コマンド、Solstice AdminSuite、admintool で個々に行われた設定の方が、/etc/defaults/passwd のデフォルト設定よりも優先します。つまり

/etc/defaults/passwd ファイルのデフォルト設定は、passwd テーブル中のエントリで (/etc/defaults/passwd の設定に対応した) 個別の設定が行われていないユーザーにだけ適用されるということです。

- /etc/defaults/passwd ファイルのすべてのデフォルト設定は、パスワード長を除いて、週単位として表現されます。ただし、パスワードの有効期間は、日数として表現されます。
- MAXWEEKS、MINWEEKS、 および WARNWEEKS のデフォルト値は、 の最終変更日を起点として計算されます。ただし、各警告値は、有効期限を起点として逆算されます。

/etc/defaults/passwd ファイルには、デフォルトでは以下のエントリがすでに含まれています。

```
MAXWEEKS=
MINWEEKS=
PASSLENGTH=
```

設定に必要な作業は、= の後に適切な数字を入力することだけです。= の後に数字が入っていないエントリは無効です。たとえば、MAXWEEKS に 4 を指定するには、以下のように入力します。

```
MAXWEEKS=4
MINWEEKS=
PASSLENGTH=
```

有効期限 (週単位)

/etc/defaults/passwd ファイルの MAXWEEKS により、ユーザーのパスワードが有効な有効期限を週単位で設定できます。ユーザーのパスワードの有効期限を週単位で指定するためのエントリです。以下に示すとおり、"=" の後に適切な数字を入力するだけで指定ができます。

```
MAXWEEKS=N
```

N は週の数です。たとえば、MAXWEEKS=9 というようにします。

変更禁止期間 (週単位)

/etc/defaults/passwd ファイルの MINWEEKS デフォルト値には、ユーザーのパスワードの変更禁止期間を週単位で設定できます。パスワードの変更禁止期間を週単位で指定するためのエントリです。以下に示すとおり、"=" の後に適切な数字を入力するだけで指定ができます。

```
MINWEEKS=N
```

N は週の数です。たとえば、MINWEEKS=2 というようにします。

警告期間 (週単位)

/etc/defaults/passwd ファイルに WARNINWEEKS デフォルト値を追加し、パスワードの有効期間が切れて無効になる前に警告を表示する期間を、週単位で設定できます。たとえば、MAXWEEKS デフォルト値に 9 を設定したときに、パスワードが無効になる前の 2 週間に警告する場合は、MAXWEEKS に 7 を設定します。

MAXWEEKS のデフォルトを設定しない限り、WARNWEEKS のデフォルトを設定することはありません。

WARNWEEKS は、 MAXWEEKS に指定された有効期限ではなく、パスワードの最終変更日を起点と考えて設定することに注意してください。したがって WARNWEEKS に、MAXWEEKS 以上の値を設定することはできません。

WARNWEEKS のデフォルトは、MAXWEEKS のデフォルトも一緒に設定されていない限り無効です。

WARNWEEKS は、以下に示すとおり "=" の後に適切な数字を入力するだけで指定ができます。

WARNWEEKS=*N*

N は週の数です。たとえば、WARNWEEKS=1 というようにします。

最低文字数

passwd コマンドには、デフォルトで「パスワードの長さは 6 文字以上にする」という規則があります。しかし、/etc/defaults/passwd ファイルの PASSLENGTH を使用することによってこの規則を変更することが可能です。

パスワードの最低文字数を 6 以外の値にするには、以下に示すとおり PASSLENGTH= の後に適切な値を入力します。

PASSLENGTH=*N*

N は文字数です。たとえば、PASSLENGTH=7 というようにします。

パスワード変更時の制限

パスワード変更の際の入力試行回数と所要時間には、制限を設定できます。設定は rpc.nispasswdd デーモン起動時の引数で行います。

入力試行回数を制限したり、タイムウィンドウを設定したりすることにより、「権利のない人が、パスワードを試行錯誤で知ることのできるものに変えてしまう」という危険をある程度 (完璧ではない) 回避できます。

試行回数の制限

パスワード変更時の誤入力試行回数を制限するには、`rpc.nispasswdd` に `-a` *number* という引数 (*number* は試行回数) を指定します。`rpc.nispasswdd` を起動するには、NIS+ マスターサーバー上にスーパーユーザー特権が必要です。

誤入力試行回数を 4 に制限する (デフォルトは 3) 場合、以下のように入力します。

```
station1# rpc.nispasswdd -a 4
```

この場合、4 回目のパスワード入力が失敗すると、「`Too many failures - try later`」というメッセージが表示され、一定の時間が経過するまで該当ユーザーのパスワード変更はできなくなります。

所要時間の制限

パスワード変更時の所要時間 (ログインの所要時間) を制限するには、`rpc.nispasswdd` に `-c` *minutes* という引数 (*minutes* には時間を分単位で指定する) を指定します。`rpc.nispasswdd` を起動するには、NIS+ マスターサーバー上にスーパーユーザー特権が必要です。

たとえば、ユーザーが 2 分以内にログインしなければならないように設定するには、以下のように入力します。

```
station1# rpc.nispasswdd -c 2
```

この場合、2 分以内にパスワードの変更に成功しないとエラーメッセージが表示され、一定の時間が経過するまで該当ユーザーのパスワード変更はできなくなります。

第 17 章

NIS+ グループの管理

この章では、NIS+ グループとその管理方法について説明します。

注 – NIS+ セキュリティグループのタスクには、Solstice AdminSuite ツールを利用するともっと簡単に実行できるものもあります。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

Solaris グループ

Solaris/NIS+ 環境には、UNIX グループ、ネットグループ、NIS+ グループの 3 種類のグループがあります。

- 「UNIX グループ」。UNIX グループとは、特別な UNIX アクセス権を与えられたユーザーの集まりのことです。NIS+ 名前空間では、UNIX グループに関する情報は `org_dir` ディレクトリオブジェクト (`group.org_dir`) 内のグループテーブルに格納されます。UNIX グループのメンバーの追加、変更、または削除方法については、第 19 章を参照してください。
- 「ネットグループ」。ネットグループとは、他のマシン上でリモート操作を実行する権限を与えられているマシンとユーザーの集まりのことです。NIS+ 名前空間では、ネットグループに関する情報は `org_dir` ディレクトリオブジェクト (`netgroup.org_dir`) 内のグループテーブルに格納されます。ネット グループの

メンバーの追加、変更、または削除方法については、第 19 章を参照してください。

- 「NIS+ グループ」。NIS+ グループとは、NIS+ オブジェクトに対する特別なアクセス権 (ネームスペースの管理を許されることが多い) を与えられている NIS+ ユーザーの集まりのことをいいます。NIS+ グループに関する情報は `groups_dir` ディレクトリオブジェクト内のテーブルに格納されます。

NIS+ グループ

NIS+ グループは、NIS+ オブジェクトに対するアクセス権を NIS+ の主体に割り当てるために考えられた概念です。(これらのアクセス権については、第 11 章を参照してください。NIS+ グループに関する情報は、NIS+ `groups_dir` ディレクトリオブジェクト内のテーブルに格納されます。各グループは個別のテーブルを持ち、グループ名とテーブル名はそれぞれ対応します。たとえば、`admin` グループに関する情報は `admin.groups_dir` テーブルに格納されます。

NIS+ グループを作成したら、どれか 1 つには `admin` という名前を付けることをお勧めします。そして、この `admin` グループには、NIS+ アクセス権を持たせるユーザーを割り当ててください。必ずしも `admin` という名前にしなければならないわけでもないのですが、NIS+ マニュアルでは、NIS+ 管理権限を持たせるユーザーのグループを `admin` と呼んでいます。異なるユーザーと異なる権限を組み合わせて複数の NIS+ グループを作成することもできます。

注 – NIS+ グループのメンバー (ユーザー) の管理には、`nisgrpadm` コマンドを使います。グループテーブルの管理には、`nisls` コマンドと `nischgrp` コマンドを使います。グループテーブルに対して `nistbladm` コマンドを使うことはできないので注意してください。

NIS+ グループ関連のコマンドとその構文、オプションの詳細は、`nis+(1)` のマニュアルページを参照してください。

関連コマンド

`nisgrpadm` コマンドを使ってほとんどのグループ管理作業を実行できますが、グループ管理に関連するコマンドには次のものがあります。

表 17–1 グループに関連するコマンド

コマンド	説明	参照する項目
nissetup	ドメインのグループが格納されるディレクトリである groups_dir を作成する	
nisls	groups_dir ディレクトリの内容、つまり、ドメイン内の全グループを表示する。各グループは groups_dir に個別のテーブルを持ち、グループ名とテーブル名はそれぞれ対応する	334 ページの「nisls コマンドでディレクトリを表示する」
nischgrp	グループを任意の NIS+ オブジェクトに割り当てる	292 ページの「オブジェクトまたはエントリグループの変更」
niscat	NIS+ グループのオブジェクト属性とメンバーを表示する	327 ページの「NIS+ グループについて niscat を使用する」
nisdefaults	新しい NIS+ オブジェクトに割り当てられるグループを表示する	284 ページの「NIS+ デフォルトの表示 - nisdefaults コマンド」

以上のコマンドの詳細 (構文、オプションなど) は、nis+(1) のマニュアルページを参照してください。

注 – NIS+ グループテーブルに対して nistbladm コマンドを使うことはできません。

NIS+ グループメンバーのタイプ

NIS+ グループには明示的 (explicit)、暗黙的 (implicit)、および再帰的 (recursive) という 3 タイプのメンバーがあります。またメンバー以外の主体にも、同様の 3 つのタイプがあります。メンバーのタイプは、メンバーを追加、削除する際に使用されます (328 ページの「nisgrpadm コマンド」を参照)。

メンバーのタイプ

- 「明示的なもの」。個々の NIS+ 主体です。これらは、すべてのグループ管理コマンドで、主体名によって識別されます。主体名は、デフォルトドメインから入力される場合、完全指定名を使用する必要はありません。
- 「暗黙的なもの」。NIS+ ドメインに所属するすべての NIS+ 主体です。これらは、* 記号とドットで始まるドメイン名によって識別されます。選択した処理は、グループ内のすべてのメンバーに適用されます。

- 「再帰的なもの」。他の NIS+ グループのメンバーの、すべての NIS+ 主体です。これらは、@ 記号で始まる NIS+ グループ名によって識別されます。選択した処理は、グループ内のすべてのメンバーに適用されます。

NIS+ グループは、明示的、暗黙的、および再帰的の 3 つすべてのカテゴリでのメンバー以外も受け付けます。メンバー以外の主体とは、「メンバーになることができるが、指定によってグループから排除されているもの」を指します。

メンバー以外の主体のタイプ

メンバー以外の主体はマイナス記号で始まり、メンバー主体と区別されます。

- 「明示的なもの」。名前の先頭に「-」だけをつけます。
- 「暗黙的なもの」。「-」の後に「*」、「.」をつけます。
- 「再帰的なもの」。「-」の後に「@」をつけます。

グループの構文

同じ主体について「グループに含まれる」という指定と「グループに含まれない」という指定があった場合、指定の順序に関係なく「含まれない」という指定が優先されます。たとえば、同じ主体について「グループに含まれる暗黙的なドメインのメンバーである」という指定と「グループに含まれない再帰的なグループのメンバーである」という指定の両方がある場合、後者の方が優先されます。

`nisgrpadm` コマンドを使用する場合、主体のタイプは表 17–2 のように指定します。

表 17–2 主体のタイプの指定方法

主体のタイプ	構文
明示的なメンバー	*username.domain*
暗黙的なメンバー	`*.` *domain*
再帰的なメンバー	*@groupname.domain*
メンバー以外 (明示的なもの)	*-username.domain*
メンバー以外 (暗黙的なもの)	`-*.` *domain*
メンバー以外 (再帰的なもの)	*@groupname.domain*

NIS+ グループについて niscat を使用する

niscat -o コマンドを使用して、NIS+ グループのオブジェクト属性を表示できます。

グループのオブジェクト属性を表示する方法

グループのオブジェクト属性を表示するには、そのグループが格納されている groups_dir ディレクトリへの読み取り権が必要です。ここでは、niscat -o とグループの完全指定名を使用します。この完全指定名には、次に示すようにその groups_dir サブディレクトリを含んでいなければなりません。

niscat -o *group-name*.groups_dir.*domain-name*

たとえば:

```
rootmaster# niscat -o sales.groups_dir.doc.com.
Object Name : sales
Owner : rootmaster.doc.com.
Group : sales.doc.com.
Domain : groups_dir.doc.com.
Access Rights : ----rmcdr---r---
Time to Live : 1:0:0
Object Type : GROUP
Group Flags :
Group Members : rootmaster.doc.com.
  topadmin.doc.com.
  @.admin.doc.com.
  *.sales.doc.com.
```

注 – nisgrpadm -1 コマンドを使うと、メンバーリストはさらに整理されて表示されます。

グループ属性のいくつかは、環境変数 NIS_DEFAULTS から継承されます。ただし、このグループの作成時に環境変数が無効になっている場合を除きます。Group Flags フィールドは、現在使用されていません。グループメンバーのリストでは、* 記号はメンバーのドメインを、@ 記号はメンバーのグループをそれぞれ示します。

nisgrpadm コマンド

nisgrpadm コマンドは、NIS+ グループを作成したり、削除したり、さまざまな管理作業を実行します。nisgrpadm コマンドを使用するには、操作に必要なアクセス権を取得しなければなりません。

表 **17–3** nisgrpadm に必要なアクセス権

操作の種類	必要なアクセス権	アクセス対象となるオブジェクト
グループの作成	作成	groups_dir ディレクトリ
グループの削除	削除	groups_dir ディレクトリ
メンバーの表示	読み取り	グループオブジェクト
メンバーの追加	変更	グループオブジェクト
メンバーの削除	変更	グループオブジェクト

nisgrpadm には、グループ作業用とグループメンバー作業用に 2 つの形式があります。

グループの作成または削除、あるいはメンバーの表示

```
nisgrpadm -c group-name.domain-name
nisgrpadm -d group-name
nisgrpadm -l group-name
```

メンバーの追加または削除、あるいはメンバーがグループに所属するかどうかの判定 (*member...* には、表 17–2 に示した、6 種類のメンバーのどのような組み合わせでも指定できます)。

```
nisgrpadm -a group-name member...
nisgrpadm -r group-name member...
nisgrpadm -t group-name member...
```

作成 (-c) 以外のすべての処理には、部分指定名 *group-name* を使用できます。ただし、-c オプションの場合でも、nisgrpadm では *group-name* 引数に groups_dir を使用する必要はありません。実際これは受け付けられません。

NIS+ グループを作成する

NIS+ グループを作成するには、グループのドメインの groups_dir ディレクトリに対する作成権が必要です。-c オプションと完全指定グループ名を使用します。

nisgrpadm -c *group-name.domainname*

グループを作成すると、指定した名前の NIS+ グループテーブルが groups_dir に作成されます。nisls コマンドを使うと、対応するテーブルが groups_dir に作成されているかどうかを確認できます。また、niscat コマンドを使うと、そのテーブル中のグループメンバーを一覧表示できます。

新しく作成したグループにはメンバーがありません。どのメンバーをどのグループに追加するかについては、330 ページの「NIS+ グループにメンバーを追加する」を参照してください。

次の例では、admin という名前の 3 つのグループを作成します。最初のグループは、doc.com. ドメインに作成します。2 番目のグループは、sales.doc.com. ドメインに作成します。3 番目のグループは、manf.doc.com. に作成します。これらはすべて、それぞれのドメインのマスターサーバーから作成されます。

```
rootmaster# nisgrpadm -c admin.doc.com.
Group admin.doc.com. created.
salesmaster# nisgrpadm -c admin.sales.doc.com.
Group admin.sales.doc.com. created.
manfmaster# nisgrpadm -c admin.manf.doc.com.
Group admin.manf.doc.com. created.
```

作成されるグループは、変数 NIS_DEFAULTS で指定されたオブジェクト属性をすべて継承します。つまり、その所有者、所有グループ、アクセス権、生存期間、および検索パスです。これらのデフォルトを表示するには、nisdefaults コマンドを使用します (第 15 章を参照)。オプションなしで使用すると、次のように出力されます。

```
rootmaster# nisdefaults
Principal Name : rootmaster.doc.com.
Domain Name : doc.com.
Host Name : rootmaster.doc.com.
Group Name :
Access Rights : ----rmcdr---r---
Time to live : 12:0:0
Search Path : doc.com.
```

所有者は Principal Name: フィールドに表示されます。所有者グループは、環境変数 NIS_GROUP を設定した場合にだけ表示されます。たとえば、C シェルを想定して NIS_GROUP を fns_admins.doc.com に設定する場合は、次のように入力します。

```
rootmaster# setenv NIS_GROUP fns_admins.doc.com
```

もちろん、グループの作成時に -D オプションを使用することによって、これらのデフォルトはどれでも変更できます。

```
salesmaster# nisgrpadm -D group=special.sales.doc.com.-c
admin.sales.doc.com. Group admin.sales.doc.com. created.
```

NIS+ グループを削除する

NIS+ グループを削除するには、グループのドメイン内の groups_dir ディレクトリに対する削除権が必要です。-d オプションを使用します。

```
nisgrpadm -d group-name
```

デフォルトドメインが正しく設定されている場合、完全指定グループ名を使用する必要はありません。ただし、別のドメイン内のグループを誤って削除しないように、まず (nisdefaults を使用して) チェックしなければなりません。次の例では test.sales.doc.com. グループを削除します。

```
salesmaster% nisgrpadm -d test.sales.doc.com.
Group test.sales.doc.com. destroyed.
```

NIS+ グループにメンバーを追加する

NIS+ グループにメンバーを追加するには、グループオブジェクトに対する変更権が必要です。-a オプションを使用します。

```
nisgrpadm -a group-name members. . .
```

325 ページの「NIS+ グループメンバーのタイプ」の記述どおり、主体 (明示的なメンバー)、ドメイン (暗黙的なメンバー)、およびグループ (再帰的なメンバー) を追加できます。デフォルトドメインに所属するメンバー名またはグループ名は、完全指定する必要がありません。この例では、デフォルトドメイン sales.doc.com. の NIS+ 主体 panza と valjean、および manf.doc.com. ドメインの主体 makeba を top-team.sales.doc.com. グループに追加します。

```
client% nisgrpadm -a Ateam panza valjean makeba.manf.doc.com.
Added panza.sales.doc.com to group Ateam.sales.doc.com
Added valjean.sales.doc.com to group Ateam.sales.doc.com
Added makeba.manf.doc.com to group Ateam.sales.doc.com
```

この動作を確認するには、nisgrpadm -l オプションを使用します。「明示的なメンバー」のカテゴリにあるメンバーを探してください。

次の例では、doc.com. ドメイン内のすべての NIS+ 主体を staff.doc.com. グループに追加します。これは doc.com. ドメイン内のクライアントから入力します。ドメイン名の前にある * 記号とドットに注意してください。

```
client% nisgrpadm -a Staff *.doc.com.
Added *.doc.com. to group Staff.manf.doc.com.
```

次の例では、NIS+ グループ admin.doc.com. を admin.manf.doc.com. グループに追加します。これは manf.doc.com. ドメインのクライアントから入力します。グループ名の前にある @ 記号に注意してください。

```
client% nisgrpadm -a admin @admin.doc.com.
Added @admin.doc.com. to group admin.manf.doc.com.
```

NIS+ グループのメンバーを表示する

NIS+ グループのメンバーを表示するには、グループオブジェクトに対する読み取り権が必要です。-l オプションを使用します。

nisgrpadm -l *group-name*

次の例では、admin.manf.doc.com. グループのメンバーを表示します。これは manf.doc.com. グループ内のクライアントから入力します。

```
client% nisgrpadm -l admin
Group entry for admin.manf.doc.com. group:
  No explicit members
  No implicit members:
  Recursive members:
  @admin.doc.com.
  No explicit nonmembers
  No implicit nonmembers
  No recursive nonmembers
```

NIS+ グループからメンバーを削除する

NIS+ グループからメンバーを削除するには、グループオブジェクトに対する変更権が必要です。-l オプションを使用します。

nisgrpadm -r *group-name members*. . .

次の例では、Ateam.sales.doc.com グループから NIS+ 主体 allende と hugo.manf.doc.com. を削除します。これは sales.doc.com.domain ドメイン内のクライアントから入力します。

```
client% nisgrpadm -r Ateam allende hugo.manf.doc.com.
Removed allende.sales.doc.com. from group Ateam.sales.doc.com.
Removed hugo.manf.doc.com. from group Ateam.sales.doc.com.
```

次の例では、admin.manf.doc.com. グループから admin.doc.com. グループを削除します。これは manf.doc.com. ドメイン内のクライアントから入力します。

```
client% nisgrpadm -r admin @admin.doc.com.
Removed @admin.doc.com. from group admin.manf.doc.com.
```

NIS+ グループのメンバーかどうかを調べる

ある NIS+ 主体が特定の NIS+ グループのメンバーであるどうかを調べるには、そのグループオブジェクトに対する読み取り権が必要です。-t オプションを使用します。

nisgrpadm -t *group-name members*. . .

次の例では、NIS+ 主体 topadmin が admin.doc.com. グループに所属するかどうかを調べます。これは doc.com. ドメイン内のクライアントから入力します。

```
client% nisgrpadm -t admin topadmin
topadmin.doc.com. is a member of group admin.doc.com.
```

次の例では、sales.doc.com. ドメインの NIS+ 主体 jo が admin.sales.doc.com. グループに所属するかどうかを調べます。 これは doc.com. ドメイン内のクライアントから入力します。

```
client% nisgrpadm -t admin.sales.doc.com. jo.sales.doc.com.
jo.sales.doc.com. is a member of group admin.sales.doc.com.
```

第 18 章

NIS+ ディレクトリの管理

この章では、NIS+ ディレクトリ オブジェクトとその管理方法について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ ディレクトリ

NIS+ ディレクトリオブジェクトには、NIS+ ドメインに関連する情報が格納されます。NIS+ ドメインには、それぞれ NIS+ ディレクトリ構造が関連付けられます。NIS+ ディレクトリの詳細については、第 2 章を参照してください。

NIS+ ディレクトリ関連のコマンドとその構文、オプションについては、`nis+(1)` のマニュアルページを参照してください。

ディレクトリに対して `niscat` コマンドを使用する

`niscat -o` コマンドを使えば、NIS+ ディレクトリのオブジェクト属性を表示できます。このコマンドを使うには、ディレクトリオブジェクトの読み取り権が必要です。

ディレクトリのオブジェクト属性を表示する

ディレクトリのオブジェクト属性を表示するには、niscat -o とディレクトリ名を使います。

```
niscat -o directory-name
```

たとえば:

```
rootmaster# niscat -o doc.com.
Object Name : doc
Owner : rootmaster.doc.com.
Group :
Domain : Com.
Access Rights : r---rmcdr---r---
Time to Live : 24:0:0
Object Type : DIRECTORY
.
.
```

nisls コマンドでディレクトリを表示する

nisls コマンドは、NIS+ ディレクトリの内容を表示します。このコマンドを使うには、ディレクトリオブジェクトに対する読み取り権が必要です。

簡潔形式での表示

```
nisls [-dgLmMR] directory-name
```

詳細形式での表示

```
nisls -l [-gm] [-dLMR] directory-name
```

表 18–1 nisls コマンドのオプション

オプション	種類
-d	ディレクトリオブジェクト。ディレクトリの内容を表示するのではなく、別のオブジェクトのように扱う
-L	リンク。ディレクトリ名が実際にはリンクである場合、コマンドはリンクをたどり、リンクされたディレクトリの情報を表示する
-M	マスター。マスターサーバーだけから情報を取得する。これによって最新の情報が得られるが、マスターサーバーが使用中の場合は長時間を要することがある

表 **18–1** nisls コマンドのオプション　(続き)

オプション	種類
-R	再帰的 (recursive)。ディレクトリを再帰的に表示する。つまり、ディレクトリ中に他のディレクトリがある場合、それらの内容も表示する
-l	長形式。情報を長形式で表示する。長形式では、オブジェクトのタイプ、作成時間、所有者、およびアクセス権を表示する
-g	グループ。情報を長形式で表示するとき、ディレクトリ所有者ではなく、そのグループ所有者を表示する
-m	変更時間。情報を長形式で表示するとき、ディレクトリ作成時間ではなく、その変更時間を表示する

ディレクトリの内容を表示する (簡潔形式)

ディレクトリ内容をデフォルトの短形式で表示するには、次に示すオプションの内の 1 つまたは複数と、ディレクトリ名を使います。ディレクトリ名を指定しない場合、NIS+ はデフォルトのディレクトリを使います。

```
nisls [-dLMR] directory-name
```

または

```
nisls [-dLMR]
```

たとえば、次の nisls の場合は、ルートドメイン doc.com. のルートマスターサーバーから入力します。

```
rootmaster% nisls doc.com.:
org_dir
groups_dir
```

次に、ルートマスターサーバーから入力した例をもう 1 つ示します。

```
rootmaster% nisls -R sales.doc.com.
sales.doc.com.:
org_dir
groups_dir
groups_dir.sales.doc.com.:
admin
org_dir.sales.doc.com.:
auto_master
auto_home
bootparams
cred
.
```

ディレクトリの内容を表示する (詳細形式)

ディレクトリの内容を詳細形式で表示するには -l オプション、および次に示すオプションのうちの 1 つまたは複数を使います。-g と -m のオプションは、表示される属性を変更します。ディレクトリ名を指定しない場合、NIS+ はデフォルトのディレクトリを使います。

```
nisls -l [-gm] [-dLMR] directory-name
```

または

```
nisls -l [-gm] [-dLMR]
```

次に示すのは、ルートドメイン doc.com. のマスターサーバーから入力した例です。

```
rootmaster% nisls -l
doc.com.
D r---rmcdr---r--- rootmaster.doc.com. date org_dir
D r---rmcdr---r--- rootmaster.doc.com. date groups_dir
```

nismkdir コマンド

注 – この節では、nismkdir コマンドを使用して既存のドメインに非ルートサーバーを追加する方法を説明します。nisserver スクリプトを使用すれば、より簡単に追加できます。方法については、第 4 章を参照してください。

---

nismkdir コマンドは、ルート以外の NIS+ ディレクトリを作成し、これをマスターサーバーに関連付けます。ルートディレクトリを作成するには、344 ページの「nisinit コマンド」で説明する nisinit -r コマンドを使います。nismkdir コマンドを使って、複製サーバーを既存のディレクトリに追加できます。

NIS+ ディレクトリを作成するには、いくつかの関連作業のほかに、いくつかの前提条件があります。

ディレクトリを作成するには、以下のように入力します。

nismkdir [**-m** *master-server*] *directory-name*

複製を既存のディレクトリに追加するには、以下のように入力します。

nismkdir -s *replica-server directory-name*
nismkdir -s *replica-server* org_dir.*directory-name*
nismkdir -s *replica-server* groups_dir.*directory-name*

ディレクトリを作成する

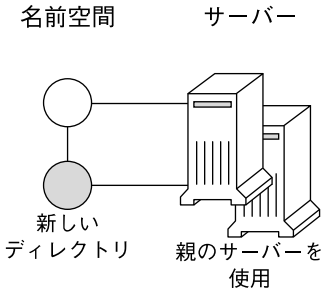
ディレクトリを作成するには、(ドメインのマスターサーバー上の) 親ディレクトリに対する作成権が必要です。まず -m オプションを使ってマスターサーバーを定義し、次に -s オプションを使って複製を定義します。

nismkdir -m *master directory*
nismkdir -s *replica directory*

注意 – nismkdir は必ず (複製サーバーではなく) マスターサーバー上で実行してください。複製サーバーで実行すると、マスターサーバーと複製サーバーの間で通信上の問題が発生します。

この例では、sales.doc.com. ディレクトリを作成してから、そのマスターサーバー smaster.doc.com. と複製サーバー rep1.doc.com. を作成します。複製サーバー rep1.doc.com. を作成します。これはルートマスターサーバーから入力します。

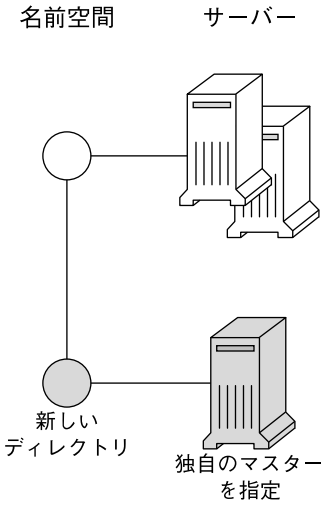
```
rootmaster% nismkdir -m smaster.doc.com. sales.doc.com.
rootmaster% nismkdir -m smaster.doc.com. org_dir.sales.doc.com.
rootmaster% nismkdir -m smaster.doc.com. groups_dir.sales.doc.com.
rootmaster% nismkdir -s rep1.doc.com. sales.doc.com.
rootmaster% nismkdir -s rep1.doc.com. org_dir.sales.doc.com.
rootmaster% nismkdir -s rep1.doc.com. groups_dir.sales.doc.com.
```



小規模またはテスト用の名前空間でないかぎりは推奨しませんが、nismkdir コマンドを使えば、独自のディレクトリを指定する代わりに、新しいディレクトリとして親ディレクトリのサーバーを使えます。次に 2 つの例を示します。

- 最初の例では、sales.doc.com. ディレクトリを作成し、これをその親ディレクトリのマスターと複製サーバーに関連付けます。

  ```
  rootmaster% nismkdir sales.doc.com
  ```



2 番目の例では、sales.doc.com. ディレクトリを作成し、独自のマスターサーバーである smaster.doc.com. を指定します。

```
rootmaster% nismkdir -m smaster.doc.com. sales.doc.com.
```

複製サーバーは指定されないため、nismkdir を再び使用して複製を割り当てるまでは、新しいディレクトリにはマスターサーバーしかありません。sales.doc.com. ドメインがすでに存在する場合は、上記のように nismkdir コマンドを実行すると、salesmaster.doc.com. が新しいマスター サーバーになり、 古いマスターサーバーは複製サーバーに格下げされます。

複製サーバーを既存のディレクトリに追加する

この章では nismkdir コマンドを使用して複製サーバーを既存のシステムに追加する方法を説明します。nisserver スクリプトを使用すると、より簡単に追加できます。

以下の点に注意してください。

- ルートドメインサーバーはルートドメイン (またはその一部) に置く
- サブドメインサーバーは、サブドメインの 1 つ上の階層の親ドメイン (またはその一部) に置く。たとえば、名前空間に prime という名前のルートドメインと sub1 という名前のサブドメインがある場合は、次のようになる
  - prime ドメインにサービスを提供するマスターサーバーと複製サーバーは、prime がルートドメインであるため、prime ドメインの一部になる
  - sub1 サブドメインにサービスを提供するマスターサーバーと複製サーバーも、prime が sub1 の親ドメインであるため、prime ドメインの一部になる

- マスターサーバーまたは複製サーバーから複数のドメインにサービスを提供することもできるが、そのような措置は避けた方がよい

新しい複製サーバーを既存のディレクトリに割り当てるには、-s オプションと、既存のディレクトリ名を使います。

```
nismkdir -s replica-server existing-directory-name
nismkdir -s replica-server org_dir. existing-directory-name
nismkdir -s replica-server groups_dir. existing-directory-name
```

nismkdir コマンドは、ディレクトリがすでに存在することを知っているため、再び作成しません。単に、追加の複製サーバーを割り当てるだけです。たとえば、新しい複製サーバーマシン名が rep1 である場合は、以下のように入力します。

```
rootmaster% nismkdir -s rep1.doc.com. doc.com.
rootmaster% nismkdir -s rep1.doc.com. org_dir.doc.com.
rootmaster% nismkdir -s rep1.doc.com. groups_dir.doc.com.
```

注意 – nismkdir は必ず、複製サーバーではなくマスターサーバー上で実行してください。nismkdir コマンドは必ずマスターサーバー上で実行してください。nismkdir を複製サーバー上で実行すると、マスターサーバーおよび複製サーバー間の通信に問題が発生します。

上記の説明に従って nismkdir を 3 回繰り返して実行した後は、以下の 3 つのディレクトリでマスターサーバーから nisping を実行する必要があります。

```
rootmaster# nisping doc.com.
rootmaster# nisping org_dir.doc.com.
rootmaster# nisping group_dir.doc.com.
```

次のような画面が表示されます。

```
rootmaster# nisping doc.com.
Pinging replicas serving directory doc.com. :
Master server is rootmaster.doc.com.
 Last update occurred at Wed Nov 18 19:54:38 1995
Replica server is rep1.doc.com.
 Last update seen was Wed Nov 18 11:24:32 1995
 Pinging ... rep1.doc.com
```

マスターサーバーの cron ファイルに、nisping コマンドが少なくとも 24 時間に一度 (この 3 つのディレクトリに対して) 実行されるよう設定しておくことをお勧めします。

nisrmdir コマンド

nisrmdir コマンドでは、ディレクトリを削除したり、ディレクトリと複製サーバーを切り離すことができます。ディレクトリを削除、またはディレクトリと複製サーバーを切り離すと、その NIS+ ドメインに対しては、マシンは NIS+ 複製サーバーとしては機能しなくなります。

ディレクトリの削除では、まずマスターサーバーと複製サーバーをディレクトリから切り離し、次にそのディレクトリを削除します。

- ディレクトリを削除するには、その親ディレクトリに対する削除権が必要です。
- ディレクトリから複製サーバーを切り離すには、そのディレクトリに対する変更権が必要です。

nisrmdir コマンドの実行で問題が生じる場合は、444 ページの「複製の失敗からの NIS+ ディレクトリの削除または分離」を参照してください。

ディレクトリを削除する

ディレクトリ全体を削除し、そのマスターサーバーと複製サーバーを切り離すには、nisrmdir コマンドをオプションなしで実行します。

```
nisrmdir directory-name
nisping  domain
```

次の例では、doc.com. ディレクトリの下の manf.doc.com. ディレクトリを削除します。

```
rootmaster% nisrmdir manf.doc.com.
rootmaster% nisping doc.com.
```

複製サーバーをディレクトリから切り離す

複製サーバーをディレクトリから切り離すには、まず、ディレクトリの org_dir と groups_dir サブディレクトリを削除します。その時は、nisrmdir コマンドに -s オプションを付けて実行します。サブディレクトリが削除されたら、親ドメインに戻って nisping を実行してください。

```
nisrmdir -s replicanameorg_dir.domain
nisrmdir -s replicanamegroups_dir.domain
nisrmdir -s replicaname domain
nisping domain
```

次の例では、manfreplica1 サーバーを manf.doc.com. ディレクトリから切り離します。

```
rootmaster% nisrmdir -s manfreplica1 org_dir.manf.doc.com.
rootmaster% nisrmdir -s manfreplica1 groups_dir.manf.doc.com.
rootmaster% nisrmdir -s manfreplica1 manf.doc.com.
rootmaster% nisping manf.doc.com.
```

nisrmdir -s コマンドを実行したが、切り離し対象の複製サーバーがダウンしていた、または通信が途絶していたという場合、「Cannot remove replica *name*: attempt to remove a non-empty table」というエラーメッセージが出されます。このような場合は、マスターサーバー上で nisrmdir -f -s *replicaname* コマンドを実行すれば、強制的に切り離すことができます。しかし、nisrmdir -f -s *replicaname* を使って通信不能な複製を分離する場合には、複製がオンライン状態に戻ったらすぐに、nisrmdir -f -s *replicaname* を再実行して、複製の /var/nis ファイルシステムを消去する必要があります。nisrmdir -f -s *replicaname* を再実行しないと、複製がサービスを再開した時に複製上に残された古い情報によって問題が発生します。

nisrm コマンド

nisrm コマンドは、標準の rm システムコマンドと似ています。このコマンドは、ディレクトリと空でないテーブルを除いて、名前空間からすべての NIS+ オブジェクトを削除します。nisrm コマンドを使うには、オブジェクトに対する削除権が必要です。しかし、この権利がない場合、-f オプションを使うことができます。このオプションは、アクセス権がなくても動作を強行しようと試みます。

nisgrpadm -d コマンド (329 ページの「NIS+ グループを削除する」を参照) を使えばグループオブジェクトを削除でき、nistbladm -r または nistbladm -R (363 ページの「テーブルを削除する」を参照) を使えばテーブルを空にできます。

ディレクトリ以外のオブジェクトは、次のコマンドで削除します。

nisrm [-if] *object-name*

表 18–2 nisrm 構文オプション

オプション	種類
-i	照会。オブジェクトを削除する前に確認を要求する。指定されたオブジェクト名が完全指定されていない場合、このオプションが自動的に使われる
-f	強制。たとえ適切なアクセス権がない場合でも、削除の強行を試みる。nischmod コマンドを使ってアクセス権の変更を試み、再度オブジェクトの削除を試みる

ディレクトリ以外のオブジェクトを削除する

ディレクトリ以外のオブジェクトを削除するには、nisrm コマンドを使い、オブジェクト名を指定します。

```
nisrm object-name...
```

次の例では、名前空間からグループとテーブルを削除します。

```
rootmaster% nisrm -i admins.doc.com. groups.org_dir.doc.com.
Remove admins.doc.com.? y
Remove groups.org_dir.doc.com.? y
```

rpc.nisd コマンド

rpc.nisd コマンドは、NIS+ デーモンを起動します。このデーモンは NIS 互換モードで実行でき、NIS クライアントからの要求にも応答できます。NIS+ デーモンを起動するにはアクセス権は不要ですが、そのすべての前提条件と関連作業を知っておく必要があります。前提条件については、115 ページの「rpc.nisd を実行するための前提条件」 を参照してください。

デフォルトでは、NIS+ デーモンはセキュリティレベル 2 で起動します。

デーモンを起動するには、以下のように入力します。

```
rpc.nisd
```

デーモンを NIS 互換モードで起動するには、以下のように入力します。

```
rpc.nisd -Y [-B]
```

DNS 転送機能によって NIS 互換デーモンを起動するには、以下のように入力します。

```
rpc.nisd -Y -B
```

表 **18–3** rpc.nisd 構文オプション

オプション	種類
-S *security-level*	セキュリティレベルを指定する。0 は「NIS+ セキュリティを使用しない」、2 は「最高レベルの NIS+ セキュリティを使用する」という意味 (レベル 1 はサポートされない)
-F	デーモンによって提供されるディレクトリのチェックポイント設定を強行する。この動作は、ディレクトリのトランザクションログを空にして、ディスク空間を解放することになる

任意のサーバーで NIS+ デーモンを起動するには、このコマンドをオプションなしで実行します。

```
rpc.nisd
```

デーモンは、デフォルトであるセキュリティレベル 2 で起動します。

セキュリティレベル 0 でデーモンを起動するには、-S フラグを使います。

```
rpc.nisd -S 0
```

NIS 互換の NIS+ デーモンを起動する

ルートマスターを含む任意のサーバーで、NIS+ デーモンを NIS 互換モードで起動できます。-Y (大文字) オプションを使います。

```
rpc.nisd -Y
```

サーバーが再起動された場合にデーモンを NIS 互換モードで再起動させるには、サーバーの /etc/init.d/rpc ファイル内で EMULYP=Y を含む行をコメント解除にしなければなりません。

DNS 転送 NIS 互換デーモンを起動する

NIS 互換モードで実行している NIS+ デーモンに DNS 転送機能を追加するには、rpc.nisd に -B オプションを追加します。

```
rpc.nisd -Y -B
```

サーバーが再起動された場合に、デーモンを DNS 転送 NIS 互換モードで再起動させるには、サーバーの /etc/init.d/rpc ファイル内の EMULYP=-Y を含む行をコメント解除し、次のように変更しなければなりません。

```
EMULYP -Y -B
```

NIS+ デーモンを停止する

NIS+ デーモンを停止するには、その実行モードが通常であろうと NIS 互換であろうと、他のデーモンと同じようにプロセスを終了させます。最初にそのプロセス ID を見つけ、次にプロセスを終了させます。次に例を示します。

```
rootmaster# ps -e | grep rpc.nisd
root 1081 1 61 16:43:33 ? 0:01 rpc.nisd -S 0
root 1087 1004 11 16:44:09 pts/1 0:00 grep rpc.nisd
rootmaster# kill 1081
```

nisinit コマンド

この節では、nisinit コマンドを使用してマシンクライアントを初期設定する方法について説明します。nisclient スクリプトを使用すると、より簡単に初期設定できます。110 ページの「NIS+ クライアントマシンの設定」 を参照してください。

nisinit コマンドは、NIS+ クライアントまたはサーバーとなるマシンを初期設定します。rpc.nisd コマンドと同様、nisinit コマンドを使うにはアクセス権は不要ですが、その前提条件と関連作業を知っておく必要があります。161 ページの「NIS+ クライアントを初期設定する」 を参照してください。

クライアントを初期設定する

クライアントを初期設定するには、次の 3 つの方法があります。

- ホスト名による方法
- ブロードキャストによる方法
- コールドスタートファイルによる方法

前提条件と関連作業は、方法によってそれぞれ異なります。たとえば、ホスト名によってクライアントを初期設定するには、その前にクライアントの /etc/hosts ファイルまたは /etc/inet/ipnodesファイルに、使用するホスト名を登録しなければなりません。また、nsswitch.conf ファイルの hosts の最初の選択肢として、files を指定する必要があります。IPv6 アドレスには、hosts の最初の選択肢としてipnodes を指定してください。nisinit コマンドを使う手順を次にまとめます。

ホスト名によってクライアントを初期設定するには、-c と -H のオプションを使い、クライアントがそのコールドスタートファイルを取得するサーバー名を指定します。

```
nisinit -c -H hostname
```

コールドスタートファイルによってクライアントを初期設定するには、-c と -C のオプションを使い、コールドスタートファイル名を指定します。

```
nisinit -c -C filename
```

ブロードキャストによってクライアントを初期設定するには、-c と -B のオプションを使います。

```
nisinit -c -B
```

ルートマスターサーバーを初期設定する

ルートマスターサーバーを初期設定するには、`nisinit -r` コマンドを使います。

```
nisinit -r
```

ここで次の情報が必要になります。

- ルートマスターサーバーとなるマシンのスーパーユーザーパスワード
- 新しいルートドメインの名前。ルートドメイン名は少なくとも 2 つの要素 (ラベル) で構成され、その末尾にドット (ピリオド) が打たれていなければならない (例: *something*`.com`)。最後の要素はインターネットの組織ドメイン (表 18–4 を参照) か、2〜3 文字の地域識別子 (日本であれば `.jp`) のどちらかにするように決められている

表 18–4 インターネットの組織ドメイン

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

nis_cachemgr コマンド

`nis_cachemgr` コマンドは、NIS+ キャッシュマネージャプログラムを起動します。このプログラムは、すべての NIS+ クライアント上で動作します。キャッシュマネージャは、名前空間で最もよく使われるディレクトリをサポートする NIS+ サーバーの位置情報キャッシュを管理します。位置情報はトランスポートアドレス、認証情報、生存期間値などです。

キャッシュマネージャが起動すると、クライアントのコールドスタートファイルから初期情報を取得し、それを `/var/nis/NIS_SHARED_DIRCACHE` ファイルにダウンロードします。

キャッシュマネージャは、クライアントマシンとして要求を行います。クライアントマシンには必ず適切な資格を与えてください。そうしないと、性能が向上するどころか、キャッシュマネージャが性能を低下させます。

キャッシュマネージャの起動と停止

キャッシュマネージャを起動するには、nis_cachemgr コマンドを入力します。

```
client% nis_cachemgr
client% nis_cachemgr -i
```

-i オプションがない場合、キャッシュマネージャは再起動されますが、情報を /var/nis/NIS_SHARED_DIRCACHE ファイルに入れておきます。コールドスタートファイル内の情報は、このファイル内の既存情報に追加されます。-i オプションは、キャッシュファイルをクリアし、クライアントのコールドスタートファイルの内容からこれを再び初期設定します。

キャッシュマネージャを停止させるには、他のプロセスと同様にプロセスを終了します。

nisshowcache コマンド

nisshowcache コマンドは、クライアントのディレクトリキャッシュの内容を表示します。

NIS+ キャッシュの内容を表示する

nisshowcache コマンドは、/usr/lib/nis 内にあります。このコマンドはキャッシュヘッダーとディレクトリ名だけを表示します。ルートマスターサーバーから入力した例を次に示します。

```
rootmaster# /usr/lib/nis/nisshowcache -v
Cold Start directory:
Name : doc.com.
Type : NIS
Master Server :
 Name : rootmaster.doc.com.
 Public Key : Diffie-Hellman (192 bits)
 Universal addresses (3)
 . .
Replicate:
 Name : rootreplica1.doc.com.
 Public Key : Diffie-Hellman (192 bits)
 Universal addresses (3)
 .
 .
 .
 Time to live : 12:0:0
 Default Access Rights :
```

ping とチェックポイントを実行する

NIS+ データセットに変更を加えると、その変更は、当該 NIS+ ドメイン (またはサブドメイン) のマスターサーバーのメモリに格納されます。変更の記録はマスターサーバーのトランザクションログ (`/var/nis/data/trans.log`) にも残されます。

通常、NIS+ データセットに変更が加えられると、その 120 秒 (2 分) 後に、マスターサーバーから当該ドメインの複製サーバーにその変更の内容が転送されます。この転送プロセスのことを「ping」といいます。マスターサーバーから複製サーバーへの ping が実行されると、通知された変更内容に従って複製サーバーのデータセットが更新されます。これにより、変更された NIS+ データがマスターサーバーと複製サーバーの両方のメモリに格納されることになります。

自動 ping プロセスが不調で複製サーバーのデータセットが更新されないこともあります。その場合は、348 ページの「`ping` を強制的に実行する」の説明に従って ping を強制的に実行する必要があります。複製サーバーが最新の NIS+ データどおりに正しく更新されているかどうか不安な場合は、348 ページの「最新更新時間を表示する」の説明に従って複製サーバーが最後に更新されたのはいつであるかを確認してください。

NIS+ データセットに対する変更は、サーバーのメモリーに格納され、トランザクションログに記録されたのち、ディスク上の NIS+ テーブルに書き込まれなければなりません。この NIS+ テーブルを更新することを「チェックポイントを実行する」といいます。

チェックポイントの実行は自動的には行われません。349 ページの「ディレクトリにチェックポイントを実行する」の説明に従ってチェックポイントコマンドを実行する必要があります。

`nisping` コマンド

`nisping` コマンドには次の用途があります。

- 複製サーバーが最後に ping されたのはいつであるかを表示する (詳細については、348 ページの「最新更新時間を表示する」を参照)
- 自動 ping サイクルが不調に終わった場合、マスターサーバーから複製サーバーへの ping を強制的に実行する (詳細については、348 ページの「`ping` を強制的に実行する」を参照)
- サーバーにチェックポイントを実行する (詳細については、349 ページの「ディレクトリにチェックポイントを実行する」を参照)

最新更新時間を表示する

-u オプションを指定して nisping コマンドを実行すると、ローカルドメインのマスターサーバーと複製サーバーが更新された時間が表示されます。

```
/usr/lib/nis/nisping -u [domain]
```

他のドメインで最後に更新が行われた時間を表示するには、コマンド行にそのドメイン名を指定します (-u オプションを指定して nisping コマンドを実行した場合、複製サーバーに対する ping は一切実行されない点に注意)。

たとえば、ローカルドメイン doc.com. で複製サーバーが最後に更新された時間を表示するには、次のように入力します。

```
rootmaster# /usr/lib/nisping -u
Last updates for directory doc.com.:
Master server is rootmaster.doc.com.
 Last update occurred at Wed Nov 25 10:53:37 1992
Replica server is rootreplica1.doc.com.
 Last update seen was Wed Nov 25 10:53:37 1992
```

ping を強制的に実行する

-u オプションを指定して nisping コマンドを実行した結果、複製サーバーが正しく更新されていないことがわかったとします。そのような場合は、nisping コマンドを実行することにより、マスターサーバーからドメイン内のすべての複製サーバーに対して、または特定の複製サーバーに対して、ping を強制的に実行できます。

すべての複製サーバーを ping の対象とする場合、次に示すように、nisping コマンドをオプションなしで実行します。

```
/usr/lib/nis/nisping
```

これにより、マスターサーバーからドメイン内のすべての複製サーバーへの ping が強制的に実行されます。ローカルドメイン doc.com. のすべての複製サーバーに対する ping を強制的に実行するには、次のように入力します。

```
rootmaster# /usr/lib/nis/nisping
Pinging replicas serving directory doc.com.:
Master server is rootmaster.doc.com.
 Last update occurred at Wed Nov 25 10:53:37 1992
Replica server is rootreplica1.doc.com.
 Last update seen was Wed Nov 18 11:24:32 1992
 Pinging ... rootreplica1.doc.com.
```

ローカルドメインでないドメイン内のすべての複製サーバーに対し ping を実行するには、ドメイン名を指定します。

```
/usr/lib/nis/nisping domainname
```

特定の 1 台のホスト上のすべてのディレクトリにあるすべてのテーブルに対し ping を実行することもできます。この場合、-a オプションを使用します。

/usr/lib/nis/nisping -a *hostname*

ディレクトリにチェックポイントを実行する

スワップの頻度やディスク領域との関連でトランザクションログが大きくなりすぎる場合、各ドメインおよびサブドメインに対しては、少なくとも 24 時間に 1 回はチェックポイントを実行する必要があります。

注 – 大きなドメイン、または大きなトランザクションログを持つドメインに対してチェックポイントを実行すると、終了するまでにかなりの時間がかかり、その間 NIS + サーバーが拘束され、NIS+ サービスの速度が低下します。サーバーは、チェックポイントを実行している間もサービス要求に応答しますが、更新できません。できるだけ、システムが混んでいない時間帯を選んでチェックポイントを実行することをお勧めします。チェックポイントの実行スケジュールは cron ファイルで調整できます。

チェックポイントを実行するには、対象のドメインのマスターサーバー上で nisping -C コマンドを実行します。先にすべての複製サーバーに対して ping を実行してから、チェックポイントを実行することをお勧めします。その場合には、複製サーバーに対して常に最新のデータでチェックポイントを実行できます。

- 特定のディレクトリに対してチェックポイントを実行するには、次に示すように、-C オプションを指定して nisping コマンドを実行します。たとえば、次のように指定します。

  ```
  rootmaster# /usr/lib/nis/nisping
  rootmaster# /usr/lib/nis/nisping -C org_dir
  ```

- ローカルドメイン内のすべてのディレクトリに対してチェックポイントを実行するには、次に示すように、-C -a オプションを指定して nisping コマンドを実行します。

```
rootmaster# /usr/lib/nis/nisping
rootmaster# /usr/lib/nis/nisping -C -a
```

サーバーによってそのトランザクションログの情報が NIS+ テーブルに転送されると、ログファイル内のトランザクションは、ディスク領域の節約のため消去されます。

doc.com. ドメイン内のすべてのディレクトリに対してチェックポイントを実行するには、次のように入力します。

```
rootmaster# /usr/lib/nis/nisping -C -a
Checkpointing replicas serving directory doc.com. :
Master server is rootmaster.doc.com.
 Last update occurred at Wed May 25 10:53:37 1995
Master server is rootmaster.doc.com.
checkpoint has been scheduled with rootmaster.doc.com.
Replica server is rootreplica1.doc.com.
```

```
 Last update seen was Wed May 25 10:53:37 1995
Replica server is rootreplica1.doc.com.
checkpoint has been scheduled with rootmaster.doc.com.
```

nislog コマンド

nislog コマンドは、トランザクションログの内容を表示します。

```
/usr/sbin/nislog
/usr/sbin/nislog -h [number]
/usr/sbin/nislog -t [number ]
```

表 **18–5** nislog コマンドのオプション

オプション	種類
-h [*num*]	ログの先頭からトランザクションを表示する。数字を指定しなければ、最初のトランザクションから表示が開始される。0 を指定すると、ログヘッダーだけが表示される
-t [*num*]	ログの末尾からトランザクションを表示する。数字を指定しなければ、最後のトランザクションから表示が開始される。0 を指定すると、ログヘッダーだけが表示される
-v	冗長モード

トランザクションログの内容を表示する

各トランザクションは、トランザクションの明細部とオブジェクト定義のコピー部の 2 つの部分から構成されます。

doc.com. ディレクトリが最初に作成されたときに作られたトランザクションログエントリを次の例に示します。「XID」はトランザクション ID を意味します。

```
rootmaster# /usr/sbin/nislog -h 1
NIS Log printing facility.
NIS Log dump:
 Log state : STABLE
Number of updates : 48
Current XID : 39
Size of log in bytes : 18432
***UPDATES***
@@@@@@@@@@@@@@@TRANSACTION@@@@@@@@@@@@@@@
#00000, XID : 1
Time : Wed Nov 25 10:50:59 1992
Directory : doc.com.
Entry type : ADD Name
```

```
Entry timestamp : Wed Nov 25 10:50:59 1992
Principal : rootmaster.doc.com.
Object name : org_dir.doc.com.
...................Object.....................
Object Name : org_dir
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : doc.com.
Access Rights : r---rmcdr---r---
Time to Live : 24:0:0
Object Type : DIRECTORY
Name : 'org_dir.doc.com.'
Type: NIS
Master Server : rootmaster.doc.com.
.
.
................................................
@@@@@@@@@@@@@@@TRANSACTION@@@@@@@@@@@@@@@
#00000, XID : 2
```

nischttl コマンド

nischttl コマンドは、名前空間内のオブジェクトまたはエントリの生存期間値を変更します。キャッシュマネージャは、この生存期間値を使って、キャッシュエントリをいつ期限切れにするかを決めます。生存期間を指定するには、合計秒数か、日、時間、分、秒の組み合わせのいずれかを使います。

オブジェクトまたはエントリに割り当てる生存期間値は、オブジェクトの安定性に左右されます。よく変化するオブジェクトの場合、生存期間値を低くします。安定したオブジェクトの場合、高い値を指定します。高い生存期間値であれば、1 週間などを指定し、低い値であれば 1 分未満を指定します。パスワードエントリの生存期間値は約 12 時間とし、1 日に 1 回パスワード変更ができるようにする必要があります。RPC テーブル内のエントリなど、あまり変化しないテーブルのエントリには、数週間の値を設定できます。

オブジェクトの生存期間を変更するには、そのオブジェクトに対する変更権が必要です。テーブルエントリの生存期間を変更するには、変更したいテーブル、エントリ、または列に対して変更権が必要です。

オブジェクトまたはテーブルエントリの現在の生存期間値を表示するには、第 15 章で説明する nisdefaults -t コマンドを使います。

オブジェクトの生存期間値を変更するには、以下のどちらかのように入力します。

nischttl *time-to-live object-name*

または

```
nischttl [-L] time-to-live object-name
```

エントリの生存期間値を変更するには、以下のどちらかのように入力します。

```
nischttl time-to-live \
  [column=value,...], \
  table-name
```

または

```
nischttl [-ALP] time-to-live \
  [column=value,...], \
  table-name
```

time-to-live には以下のものを指定します。

- 「秒」。数字だけで文字を指定しなければ、単位が秒であると解釈される (例 :1234 は 1234 秒)。数字の後に s をつけると、単位が秒であると解釈される (例 :987s は 987 秒)。日、時間、分などとともに指定する場合は、秒であることを明らかにするため s をつける
- 「分」。数字の後に m を指定すると、単位が分であると解釈される (例：90m は 90 分)
- 「時間」。数字の後に h をつけると、単位が時間であると解釈される (例：9h は 9 時間)
- 「日」。数字の後に d をつけると、単位が日であると解釈される (例：7d は 7 日)

以上の値は組み合わせて使うことができます。たとえば「4d3h2m1s」は、4 日 3 時間 2 分 1 秒を意味します。

nischttl コマンドでは以下のフラグを使用できます。

表 18–6 nischttl 構文のオプション

オプション	種類
A	すべて。[*column=value*] 指定に合致する全エントリに変更を適用する
L	リンク。リンクをたどり、リンク自体ではなく、リンクされたオブジェクトまたはエントリを変更する
P	パス。条件を満足するエントリが 1 つ見つかるまで、パスをたどる

オブジェクトの生存期間を変更する

オブジェクトの生存期間を変更するには、生存期間の値とオブジェクト名を指定して nischttl コマンドを入力します。-L コマンドを追加すれば、リンクされたオブジェクトにまで変更を拡張できます。

```
nischttl -L time-to-live object-name
```

生存期間は秒か、日、時間、分、秒の組み合わせかで指定できます。前者の場合、秒数だけを指定します。後者の場合、日数、時間数、分数と秒数に「s、m、h、d」をつけてください。たとえば:

```
client% nischttl 86400 sales.doc.com.
client% nischttl 24h sales.doc.com.
client% nischttl 2d1h1m1s sales.doc.com.
```

最初の 2 つでは、sales.doc.com. ディレクトリの生存期間が 86,400 秒、つまり 24 時間に変更されます。3 つ目では、hosts テーブル内の全エントリの生存期間が 176,461 秒、つまり 2 日と 1 時間 1 分 1 秒に変更されます。

テーブルエントリの生存期間を変更する

エントリの生存期間を変更するには、インデックス付きのエントリフォーマットを使います。-A、-L、または -P のどのオプションでも使えます。

```
nischttl [-ALP] time-to-live \
  [column=value,...], \
  table-name
```

注 – C シェル を使用している場合、[ ] がメタキャラクタとして解釈されないように引用符で囲みます。

次に示す例は上記の例と似ていますが、オブジェクトではなく、テーブルエントリの値を変更します。

```
client% nischttl 86400 '[uid=99],passwd.org_dir.doc.com.'
client% nischttl 24h `[uid=99],passwd.org_dir.doc.com.'
client% nischttl 2d1h1m1s `[name=fred],hosts.org_dir.doc.com'
```

第 19 章

NIS+ テーブルの管理

この章では、NIS+ テーブルとその管理方法について説明します。デフォルトの NIS+ テーブルの詳細については、表 10–1 を参照してください。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ テーブル

NIS+ で使用される情報は、 NIS+ テーブルに格納されます (Solaris オペレーティング環境で提供されるデフォルトの各 NIS+ テーブルについては、第 23 章を参照)。

NIS+ テーブル関連のコマンドと、その構文およびオプションの詳細については、`NIS+` のマニュアルページを参照してください。

nistbladm コマンド

注 – NIS+ テーブルに関連した作業のうちいくつかは、 Solstice AdminSuite™ツールを使用すると容易に行えます。

nistbladm コマンドは NIS+ テーブル管理コマンドの中でも最も重要なコマンドであり、NIS+ ディレクトリオブジェクトに格納されている NIS+ テーブル上で使います。nistbladm コマンドを使うと、NIS+ テーブルまたはそのエントリを作成、修正、削除できます。ただし、ディレクトリのないところにテーブルを作成はできません。同様に、テーブルと列が定義されていないところにエントリを追加できません。

テーブルを作成するには、そのテーブルが所属するディレクトリに対する作成権が必要です。テーブルを削除するには、そのディレクトリに対する削除権が必要です。テーブルの内容を変更 (エントリの追加、変更、または削除) するには、そのテーブルまたはエントリの変更権が必要です。

nistbladm 構文

nistbladm コマンドの一般的な構文は次のとおりです。

```
nistbladm options \
  [columspec | columnvalue] \
  [tablename | indexedname]
```

- *columnspec* には、テーブル内に作成する列を指定する (具体的な指定方法については、361 ページの「テーブルの列を指定する」 を参照)
- *columnvalue* には、*tablename* によって表されるテーブルの中の特定のセルを指定する (具体的な指定方法については、357 ページの「nistbladm と列の値」を参照)
- *tablename* には、テーブル名 (hosts.org_dir.doc.com. など) を指定する
- *indexedname* には、特定のテーブルの中の特定のセル値を指定する (具体的な指定方法については、357 ページの「nistbladm と列の値」を参照) 。*indexedname* は本質的に、*columnvalue* と *tablename* の役割を兼ね備えている

表 **19–1** nistbladm コマンドのオプション

オプション	説明
-a	-A	既存の NIS+ テーブルにエントリを追加する。-a を指定した場合、nistbladm コマンドを実行すると既存のエントリが上書きされる場合には、エラーが返される。-A を指定した場合、nistbladm コマンドが既存エントリを上書きする場合でも強制的に実行される (363 ページの「エントリをテーブルに追加する」を参照)
-D *defaults*	別のデフォルト特性を使ってオブジェクトを作成する (詳細については、デフォルトの nistbladm(1) のマニュアルページを参照)
-d	テーブルを破棄する (363 ページの「テーブルを削除する」を参照)
-c	テーブルを作成する (360 ページの「テーブルを作成する」を参照)
-r	-R	既存の NIS+ テーブルからエントリ (1 つまたは複数) を削除する。-r を指定した場合、複数のエントリの削除につながる nistbladm コマンドは実行されず、エラーが返される。-R を指定した場合、複数のエントリの削除につながる nistbladm コマンドであっても強制的に実行される (369 ページの「テーブルからエントリを削除する」を参照)
-m	テーブルエントリを修正するためのオプション。旧リリースとの互換性を維持するためにだけ残されている。エントリを修正するのであれば、-e オプションまたは -E オプションを使うほうが望ましい
-e	-E	既存の NIS+ テーブルのエントリを修正する。-e を指定した場合、複数のエントリの変更につながる nistbladm コマンドは実行されず、エラーが返される。-E を指定した場合、複数のエントリの変更につながる nistbladm コマンドであっても強制的に実行される (366 ページの「エントリを修正する」を参照)

nistbladm と列の値

列の値は、テーブル内の個々のエントリを識別するために使われます。列の値の書式は次のとおりです。

```
columname="value", \
 columnname="value", ...
```

引数の意味はそれぞれ以下のとおりです。

- *columname* はテーブルの列名を示す
- *value* は列内の特定のセルの内容 (値) を示す。この値により、テーブルの行を識別することができる (*column=value* を指定してテーブルデータを作成または修正する場合は、必ず *value* を引用符で囲むこと)

たとえば、マシン名と IP アドレスを登録した hosts という名前のテーブルがあるとします。

表 **19–2** hosts テーブルの例

IP アドレス	**name**	別名
129.146.168.4	altair	
129.146.168.119	deneb	mail
129.146.168.120	regulus	dnsmaster
129.146.168.121	regulus	dnsmaster
129.146.168.11	sirius	

このサンプルテーブルの altair エントリ (行) を識別するには、次の 3 通りの *column=value* の指定方法が考えられます。

- name=altair
- address=129.146.168.4
- name=altair,address=129.146.168.4.

上記のテーブルで注目すべき点としては、regulus という名前のマルチホームマシンに 2 つの IP アドレスが割り当てられていることです。この場合、ホストマシン regulus の *column=value* は 2 つの行を示します。そこで、最初の regulus 行だけを識別したいという場合は次のいずれかを入力します。

- address=129.146.168.120 または
- address=129.146.168.120.,name=regulus,dnsmaster

注 – nistbladm コマンドの一部のオプションには、テーブル内のすべての列に *column=value* を指定しなければならないものがあります。

---

nistbladm、検索可能列、キー、列の値

NIS+ テーブルを作成する際は、s フラグまたは i フラグを指定し、1 つまたは複数の列が検索可能になるようにします (361 ページの「テーブルの列を指定する」を参照)。なお、niscat -o *tablename* と入力すれば、テーブルの列とその特性を一覧表示できます。

テーブル内で行の「識別」に使われるのが検索可能列です。検索可能列の値 (値の組み合わせ) はすべて一意でなければなりません。したがって、検索可能列が 1 つしかないテーブルの場合、各行の検索可能列の値はすべて一意でなければならず、重複は一切許されません。

ここで、city という名前の検索可能列と country という名前の検索不可列からなるテーブルを作成する場合を想定します。次に示すのは、正しい (検索可能列の値の組み合わせに重複がない) テーブルの例です。

City	Country
San Francisco	United States
Santa Fe	United States
Santiago	Chile

次は正しくない (`City` 列に重複がある) テーブルの例です。

City	Country
London	Canada
London	England

1 つのテーブルに検索可能列が複数ある場合は、検索可能列どうしの値の組み合わせが一意であればかまいません。ここでは、`Lastname` および `Firstname` という 2 つの検索可能列と、`city` という検索不可列からなるテーブルを作成する場合を想定します。次に示すのは、正しい (検索可能列の値の組み合わせに重複がない) テーブルの例です。

Lastname	Firstname	City
Kuznetsov	Sergei	Odessa
Kuznetsov	Rima	Odessa
Sergei	Alex	Odessa

次は正しくない (検索可能列の値の組み合わせに重複がある) テーブルの例です。

Lastname	Firstname	City
Kuznetsov	Rima	Odessa
Kuznetsov	Rima	Chelm

NIS+ のコマンドはいずれも、検索可能列の値に基づいて特定の行を識別します。

`nistbladm` とインデックス名

NIS+ には、テーブル名と列の値の検索基準の組み合わせ (これを「インデックス名」という) によって特定のテーブルの特定のエントリを識別する、というテーブル管理の方法もあります。インデックス名の書式は次のとおりです。

[*search_criteria*] , *tablename* . *directory*

search_criteria には検索基準を指定しますが、このとき角カッコ ([]) で囲むのを忘れないでください。書式は次のとおりです。

columnname=value, \
columnname=value, . . .

columnname=value にはテーブルの検索可能列の値を指定します (357 ページの「nistbladm と列の値」を参照)。

たとえば、表 19–2 の `altair` エントリを識別するには、次のようにインデックス名を指定します。

```
[addr=129.146.168.4,cname=altair],hosts.org_dir.doc.com.
```

`nistbladm -R` コマンドを使用すると、間になにも入れない角カッコ `[ ]`をすべてのテーブル列を指定するワイルドカードとして使用し、テーブル中のすべてのエントリを一度に削除できます。

nistbladm とグループ

Solaris の NIS+ 環境には 3 種類のグループがあります。

- UNIXグループ - UNIX グループに関する情報は `groups.org_dir` テーブルに格納される。UNIX グループ情報の管理には `nistbladm` コマンドを使う
- ネットグループ - ネットグループに関する情報は、 `netgroups.org_dir` テーブルに格納される。ネットグループ情報の管理には `nistbladm` コマンドを使う
- NIS+ グループ - NIS+ グループに関する情報は `groups_dir` ディレクトリオブジェクトのテーブル (1 つまたは複数) に格納される。NIS+ グループ情報の管理には `nisgrpadm` コマンドを使う

注 – NIS+ グループの管理に `nistbladm` を使うことはできません。

その他のグループの詳細については、323 ページの「Solaris グループ」を参照してください。

テーブルを作成する

NIS+ テーブルには、少なくとも 1 つの列が必要で、その列のうち少なくとも 1 つが検索可能でなければなりません。NIS+ テーブルを作成するには、`nistbladm` コマンドに `-c` オプションを付けて使います。

```
nistbladm -c tabletype columnspec \
  ... tablename
```

引数の意味はそれぞれ以下のとおりです。

- *tabletype* は、単にテーブルをあるテーブルクラスに所属するものとして識別する文字列です。任意の名前を使用できます。
- *columnspec* 引数には、各列の名前と特性を指定します。1 つの *columnspec* には、新規テーブルに含める、それぞれの列を入力してください。複数の *columnspec* を指定する場合は、空白で区切ってください。

```
nistbladm -c tabletype columnspec columnspec   \
columnspec tablename
```

columnspec の書式については、361 ページの「テーブルの列を指定する」 (下記) を参照してください。

テーブルの列を指定する

各 *columnspec* エントリは、以下の形式のように、2 つから 4 つの要素から成っています。

name=type,rights:

表 19–3 テーブルの列の構成要素

構成要素	説明
name	列の名前
=	等号記号 (必須)
type	(オプション) `S`、`I`、または `C` で列の種類を指定する。表 19–4 参照。このコマンドは省略可能です。*type* を指定しないと、その列はデフォルトとなる
rights	(オプション) アクセス権を指定する。ここに指定したアクセス権は、テーブル全体や特定エントリに付与したアクセス権より優先される。*access* を指定しないと、その列のアクセス権は、テーブル全体や特定エントリに付与したアクセス権になる。アクセス権の構文については、281 ページの「コマンドによるアクセス権の指定」を参照

列には、次に挙げる種類のうち 1 つを指定できます。

表 **19–4** 列の種類

種類	説明
	等号 (=) のみ。列の種類は指定しない。その列は検索可能にもならなければ、暗号化されることもない
S	検索可能
I	大文字と小文字の区別をしない。NIS+ コマンドで列を検索する場合、大文字と小文字を区別しない
C	暗号化する

NIS+ の各コマンドは、列全体を調べ、検索可能列の値に基づいて個々の行を識別します。検索可能列は S フラグまたは I フラグで指定します (データベースの分野では、検索可能列のことをキー列といいます)。テーブルの最初の列は必ず検索可能にします。その他の列は検索可能にする必要はありません。NIS+ テーブルのキーは、検索可能列に対して作成されます。このため、検索可能列が複数存在する場合は、最初の列から連続した領域に割り当てなければなりません。検索可能列以外の列がその領域に存在してはなりません。たとえば、テーブルの検索可能列が 1 列だけの場合は、最初の列に配置しなければなりません。検索可能列が 2 列の場合は、最初の 2 列に配置しなければなりません。検索可能列の詳細については、358 ページの「nistbladm、検索可能列、キー、列の値」 を参照してください。

アクセス権だけを指定する場合、コンマを使用する必要はありません。-S、-I、-C フラグの 1 つか複数を指定する場合は、アクセス権の前にコンマを追加します。

以下の例では、上と同じテーブルが作成されますが、最初の 2 列にはその列に固有なアクセス権が付与されます。

```
master% nistbladm -c depts Name=I,w+m Site=w+m Name=C \
 divs.mydir.doc.com.
```

テーブル作成時の列のアクセス権の指定については、289 ページの「テーブル作成時の列権限設定」を参照してください。

注 – NIS+ では、すべての列エントリが NULL で終了するものと仮定しています。NIS+ テーブルに情報を書き込むアプリケーションやルーチンは、列エントリをすべて NULL で終了するように構成しなければなりません。

自動マウントテーブルを追加作成する

自動マウントテーブルには 2 つの列しか作ることができません。1 番目の列には key という名前を付け、2 番目の列には value という名前を付けます。たとえば、auto1 という名前の自動マウントテーブルを作成するには、次のように入力します。

```
master% nistbladm -c key-value key=S value= auto1.org_dir.doc.com.
```

テーブルを削除する

テーブルを削除するには、-d オプションを使ってテーブル名を入力します。

```
nistbladm -d tablename
```

テーブルを削除するには、その前にテーブルが空になっていなければなりません (369 ページの「テーブルからエントリを削除する」を参照)。次の例では、doc.com. ディレクトリから divs テーブルを削除します。

```
rootmaster% nistbladm -d divs.doc.com.
```

エントリをテーブルに追加する

エントリ (行) をテーブルに追加するには、nistbladm コマンドに -a オプションまたは -A オプションを指定し、続けて column=value のペア (1 つまたは複数)、テーブル名の順に指定します。あるいは、nistbladm コマンドに -a オプションまたは -A オプションを指定し、続けてインデックス名を指定します (359 ページの「nistbladm とインデックス名」を参照)。

```
nistbladm [-a | -A] indexedname
nistbladm [-a | -A] column="value" \
column="value" \
... tablename
```

-a オプションまたは -A オプションを指定して既存のテーブルに新しいエントリ (行) を追加する場合は、次の点に注意してください。

- *value* は必ず引用符で囲む。たとえば、新たに追加するエントリの cname 列の値を deneb にする場合は、*column=value* ペアを cname="deneb" と指定する
- テーブル内のすべての列の値を指定する
- 追加するエントリ (行) の列を空白にする場合は、*column*=" " と指定する。つまり、*value* には、引用符で囲んだ半角スペースを指定する

注 – NIS+ はネームサービスであり、設計上、そのテーブルにはオブジェクト本体ではなく、オブジェクトのリファレンスが格納されます。NIS+ は、最適化された状態では 10,000 におよぶオブジェクトをサポートし、すべてのテーブルのサイズを合計しても 10M バイトを超えることはありません。NIS+ では、1 つの列のフィールドサイズの合計が 7k を超えるようなテーブルはサポートされません。テーブルが大きすぎると、rpc.nisd は失敗します。

-a オプションを指定してエントリを追加する

nistbladm コマンドに -a オプションを指定しておくと、追加するエントリと同じエントリがすでに存在する場合は、エントリの上書きは行われずにエラーが返されます。あるエントリの検索可能列の値が、追加するエントリの検索可能列の値とまったく同じである場合、そのエントリは「すでに存在しているもの」と判断されます。このとき、検索不可列の値は考慮されません。

-a オプションを指定する場合、次のように、テーブル内のすべての列の値を指定する必要があります。

```
nistbladm -a column="value" \
 column="value" \
 ... tablename
nistbladm -a indexedname
```

テーブルのすべての列名とその特性を表示する場合は、niscat -o *tablename* コマンドを実行します。

たとえば、depts テーブルに新しい行を追加する場合は、次に示すように、列の数だけ *column=value* ペアを指定します。

```
rootmaster% nistbladm -a Name='R&D' Site='SanFran'  \
    Name='vattel' depts.doc.com.
```

なお、これと同じエントリをインデックス名を指定して追加する場合は、次のように入力します。

```
rootmaster% nistbladm -a [Name='R&D',Site='SanFran' \
Name='vattel'],depts.doc.com.
```

どちらの場合も、次のようなテーブルとなります。

Dept	Site	Name
R&D	SanFran	vattel

C シェルを使っている場合は、角カッコを含む数式を引用符でオフセットすることもできます。

nistbladm コマンドでは 1 回につき 1 つのエントリしか追加できません。つまり、追加する行の数だけ nistbladm コマンドを実行する必要があります。

追加するエントリ (行) の各列と同じ値を持つ行がすでに存在する場合、nistbladm -a はエラーを返します。1 つのテーブルの中に同じ行が存在することはできません。検索可能列の値が同一である場合、それらの行は同一であるとみなされます。このとき、検索不可列の値は考慮されません。

たとえば、Dept 列と Site 列が検索可能であって、Name 列は検索可能ではないテーブルに対して nistbladm を実行すると、次の 2 つの行は同一であるとみなされます。

Dept (検索可能)	Site (検索可能)	Name (検索不可)
Sales	Vancouver	Hosteen
Sales	Vancouver	Lincoln

この場合、nistbladm -a を実行して Sales Vancouver Lincoln という行を作成できません。

しかし、複数の検索可能列のどちらか一方の値が重複するだけであれば、nistbladm -a を実行して新しい行を作成できます。たとえば、以下の 2 つのコマンドを実行すると、一部の値が異なるだけの 2 つの列を作成できます。

```
rootmaster% nistbladm -a Dept='Sales' \
Site='Vancouver' Name='hosteen' staff.doc.com.
rootmaster% nistbladm -a Dept='Sales' \
Site='SanFran' Name='lincoln' staff.doc.com.
```

これらのコマンドを実行すると、検索可能列の一部が同じで、すべてが同じではない 2 つの行が作成されます。

Dept	Site	Name
Sales	Vancouver	hosteen
Sales	SanFran	lincoln

-A オプションを指定してエントリを追加する

-A オプションは、nistbladm コマンドで既存のエントリを上書きするアプリケーションに使用します。-a と同様に、-A もテーブルにエントリを追加するためのオプションです。しかし、追加するエントリがすでに存在する場合、エラーを返して終了するのではなく、既存のエントリを上書きします。

-A オプションを指定する場合、エントリ内のすべての列の値を指定する必要があります。

ここで、Dept 列と Site 列が検索可能である次のテーブルを想定します。

Dept (検索可能)	Site (検索可能)	Name
Sales	SanFran	Lincoln

このテーブルに対して次のコマンドを実行します。

```
rootmaster% nistbladm -A Name=Sales Site=SanFran \
Name=Tsosulu depts.doc.com.
```

`Name=Sales Site=SanFran` はすでに存在するので、オプションが `-a` であれば、上記のコマンドはエラーを返します。しかし、上記のコマンドでは `-A` が指定されているので、既存の行が上書きされます。

Dept	Site	Name
Sales	SanFran	Tsosulu

エントリを修正する

既存のエントリを修正 (編集) する場合は、`-e` オプションまたは `-E` オプションを指定します。Solaris オペレーティング環境では、旧リリースとの互換性を維持するため、`-m` オプションもサポートしています。新しいアプリケーションまたはコマンド行での操作には、`-e` オプションまたは `-E` オプションを使うことをお勧めします。

テーブル内の既存のエントリ (行) を編集するには、`nistbladm` コマンドに `-e` オプションまたは `-E` オプションを指定し、それに続けて、値を変更した *column=value* のペア (1 つまたは複数) とテーブル内の特定の行を示すインデックス名を指定します (359 ページの「`nistbladm` とインデックス名」を参照)。

```
nistbladm [-e | -E] column="value" \
 column="value" \
 ... indexedname
```

`-e` オプションまたは `-E` オプションを指定して既存エントリ (行) を修正する場合は、次の点に注意してください。

- *value* は必ず引用符で囲む。たとえば、`cname` 列の値を `deneb` に変更する場合は、*column=value* ペアを `cname="deneb"` と指定する
- 検索可能列の値は、1 行 (エントリ) 単位でしか変更できない
- エントリ (行) の列を空白にする場合は、*column*`=" "` と指定する。つまり、*value* には、引用符で囲んだ半角スペースを指定する

-e オプションを指定してエントリを編集する

-e オプションを指定した場合、複数のエントリの検索可能列値を変更することになる nistbladm コマンドは実行されず、エラーが返されます。このとき、検索不可列の値は考慮されません。

```
nistbladm column=" value" \
 column=" value" \
 ... indexedname
```

-e オプションを指定した場合、変更する列の値を指定するだけです。

ここで次のテーブルについて考えます。

Dept	Site	Name
Sales	SanFran	Tsosulu

Name 列の値を Chandar に変更するには、次のように入力します。

```
master% nistbladm -e Name="Chandar" [Dept='Sales',Site='SanFran'], \
    depts.doc.com.
```

修正後のテーブルは次のようになります。

Dept	Site	Name
Sales	SanFran	Chandar

上記の例では、インデックス名に Name 列が含まれていません。これは、Name 列が検索可能ではないためです。

C シェルを使っている場合は、角カッコを含む数式を引用符でオフセットすることもできます。

-e オプションを指定して検索可能列の値を編集するには、新たに指定した値が (インデックス名によって識別される) 1 行にだけ適用されることが条件になります。したがって、Dept 列の値を Manf に変更するには、次のように入力します。

```
master% nistbladm -e Dept="Manf" [Dept='Sales',Site='SanFran'], \
    depts.doc.com.
```

Dept (検索可能)	Site (検索可能)	名前
Manf	SanFran	Chandar

しかし、検索可能列に Manf と SanFran という値を持つエントリが既に存在する場合は、nistbladm -e はエラーを返します。

同一のエントリが対象である場合は、複数の列の変更を指定できます。たとえば、Dept 列と Name 列の値を変更するには、次のように入力します。

```
master% nistbladm -e Dept="Manf" Name=”Thi” \
    [Dept=’Sales’,Site=’SanFran’],depts.doc.com.
```

Dept (検索可能)	Site (検索可能)	Name
Manf	SanFran	Thi

-E オプションを指定してエントリを編集する

-E オプションは、nistbladm コマンドで既存のエントリを上書きする場合のアプリケーションに使用します。

ここで次のテーブルを想定します。

Dept (検索可能)	Site (検索可能)	名前
Sales	SanFran	Chandar
Sales	Alameda	Achmed

このテーブルに対して次のコマンドを実行します。

```
master% nistbladm -E Site="Alameda” Mgr="Chu"  \
[Div=’Sales’,Site=’SanFran’],depts.doc.com.
```

すると、Sales SanFran Chandar 行が Sales Alameda Chu に変わります。しかも、キーの値 Sales Alameda によって識別される Sales Alameda Achmed 行までもが変更の対象になります。その結果、コマンド実行前には 2 つあったはずの行が 1 つになります。

Dept (検索可能)	Site (検索可能)	名前
Sales	Alameda	Chu

これは複数の行を編集することになるので、オプションが -e であれば、上記のコマンドはエラーを返します。しかし、実際には -E が指定されているので、複数のエントリ (行) が編集されます。

テーブルからエントリを削除する

- テーブルから 1 つのエントリを削除する場合は、-r オプションを指定します (369 ページの「1 つのエントリを削除する」を参照)。
- テーブルから複数のエントリを削除する場合は、-R オプションを指定します (369 ページの「複数のエントリを削除する」を参照)。

1 つのエントリを削除する

テーブルから 1 つのエントリを削除するには、次に示すように -r オプションを指定します。

```
nistbladm -r indexed-name
```

次に示すコマンドでは、depts テーブルから Manf-1 エントリを削除します。

```
rootmaster% nistbladm -r [Dept=Manf-1,Site=Emeryville,Name=hosteen], \
depts.doc.com.
```

指定する列の値の数は最小限に減らすことができます。NIS+ では、列の値の重複が見つかると、行は 1 つも削除されることなく、エラーが返されます。次のテーブルから Manf-1 エントリを削除するには、次に示すように Site 列の値を指定するだけで済みます。

```
rootmaster% nistbladm -r [Site=Emeryville],depts.doc.com.
```

しかし、R&D エントリも Sales エントリと同じ値を持っているため、Site 列の値 (SanFran) を指定するだけでは Sales エントリを削除できません。

Dept	Site	Name
R&D	SanFran	kuznetsov
Sales	SanFran	jhill
Manf-1	Emeryville	hosteen
Manf-2	Sausalito	lincoln

複数のエントリを削除する

テーブルから複数のエントリを削除するには、次に示すように -R オプションを指定します。

```
nistbladm -R indexedname
```

オプションが -r であれば、最小限必要な列値を指定するだけで済みます。しかし、上記のコマンドでは -R が指定されているので、NIS+ が重複を見つけると、それに該当するすべてのエントリが削除されます。テーブル内の列の名前を確認したい場合は、niscat -o コマンドを実行します。次のコマンドを実行すると、Site 列の値が SanFran であるすべてのエントリが削除されます。

```
rootmaster% nistbladm -R [Site=SanFran],depts.doc.com.
```

Dept	Site	Name
Manf-1	Emeryville	hosteen
Manf-2	Sausalito	lincoln

-R オプションを指定すると、次に示すように、列の値を指定しなければテーブル内のすべてのエントリを削除できます。

```
rootmaster% nistbladm -R [],depts.doc.com.
```

nistbladm -R コマンドと共に使用すると、一対の空の角カッコはすべてのテーブル列を指定するワイルドカードとして解釈されます。

niscat コマンド

niscat コマンドは NIS+ テーブルの内容を表示します。このコマンドを使って、テーブルのオブジェクト属性を表示できます。表示するテーブル、エントリ、または列に対する読み取り権が必要です。

構文

テーブルの内容を表示するには、以下のように入力します。

```
niscat [-hM] tablename
```

テーブルのオブジェクト属性を表示するには、以下のように入力します。

```
niscat -o tablename
niscat -o entry
```

表 **19–5** niscat 構文のオプション

オプション	説明
-h	ヘッダー。テーブルエントリの上にあるヘッダー行を表示し、各列の名前を表示する
-M	マスター。マスターサーバーに収められているテーブルのエントリだけを表示する。これによって、最新情報を取得できる。デバッグにだけ使うようにする
-o	オブジェクト。列名、属性、サーバーなどの、テーブルについてのオブジェクト情報を表示する

テーブルの内容を表示する

テーブルの内容を表示するには、niscat にテーブル名を付けて実行します。

```
niscat tablename
```

次の例では、depts というテーブルの内容を表示します。

```
rootmaster% niscat -h depts.doc.com.
#  Name:Site:Name
R&D:SanFran:kuznetsov
Sales:SanFran:jhill
Manf-1:Emeryville:hosteen
Manf-2:Sausalito:lincoln
```

注 – *NP* は、ユーザーにそのエントリを表示するためのアクセス権がないことを示します。アクセス権は、テーブル、列、エントリ (行) ごとに与えられます。アクセス権の詳細については、第 15 章を参照してください。

テーブルまたはエントリのオブジェクト属性を表示する方法

テーブルのオブジェクト属性を表示するには、niscat -o にテーブル名を付けて実行します。

```
niscat -o tablename.org_dir
```

テーブルエントリのオブジェクト属性を表示するには、niscat -o を使い、インデックス付きの名前でエントリを指定します。

```
entry ::=column=value    ...  tablename  |  \
 [column=valu    ,...],  tablename
```

次に、テーブルの例とテーブルエントリの例を示します。

「テーブル」

```
rootmaster# niscat -o hosts.org_dir.doc.com.
Object Name : hosts
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : org_dir.doc.com.
Access Rights : ----rmcdr---r---
Time to Live : 12:0:0
Object Type :
TABLE Table Type : hosts_tbl
Number of Columns : 4
Character Separator :
Search Path :
Columns :
 [0] Name : cname
 Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INS
 Access Rights: ----------------
 [1] Name : name
 Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INS
 Access Rights: ----------------
 [2] Name : addr
 Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INS
 Access Rights: ----------------
 [3] Name : comment
 Attributes : (TEXTUAL DATA)
 Access Rights: ----------------
```

「テーブルエントリ」

```
rootmaster# niscat -o [name=rootmaster],hosts.org_dir.doc.com.
Object Name : hosts
Owner : rootmaster.doc.com.
Group : admin.doc.com.
Domain : org_dir.doc.com.
Access Rights : ----rmcdr---r---
Time to Live : 12:0:0
Object Type : ENTRY
Entry data of type hosts_tbl
Entry has 4 columns.
 .
#
```

nismatch と nisgrep コマンド

nismatch コマンドと nisgrep コマンドは、NIS+ テーブルを検索して、それぞれ特定の文字列または正規表現に一致するエントリを探します。これらのコマンドは、エントリ自体、またはエントリの検索できた回数のどちらかを表示します。nismatch コマンドと nisgrep コマンドの相違を表 19–6 に示します。

表 **19–6** nismatch と nisgrep の 比較

特性	nismatch	nisgrep
検索指定	テキストのみ指定可能	正規表現が指定可能
速度	高速	低速
検索対象	検索可能列のみ	すべての列 (検索可能かどうかとは無関係)
検索構文	*column= string ... tablename* [*column= string*, ...], *tablename*	*column=exp ... tablename*

この節の例では、両方のコマンドの構文を説明します。

どちらのコマンドを使用する場合にも、検索するテーブルに対する読み取りアクセス権が必要です。

この節の例は、次に示す depts.doc.com. というテーブルの値をベースにしています。最初の 2 つの列だけが検索可能です。

Name (検索可能)	**Site (検索可能)**	**Name**
R&D	SanFran	kuznetsov
Sales	SanFran	jhill
Manf-1	Emeryville	hosteen
Manf-2	Sausalito	lincoln
Shipping-1	Emeryville	tsosulu
Shipping-2	Sausalito	katabami
Service	Sparks	franklin

正規表現について

正規表現により、テキストと記号を組み合わせて使用し、特定の構成の列の値を検索できます。たとえば、正規表現 'Hello' は、Hello で始まる値を検索します。正規表現をコマンド行で使用するときは、必ず引用符で囲んでください。その理由は、正規表現記号の多くが Bourne シェルと C シェルでは特殊な意味をもつからです。たとえば :

```
rootmaster% nisgrep -h greeting='Hello' phrases.doc.com.
```

正規表現の記号を表 19–7 にまとめます。

表 **19–7** 正規表現記号

シンボル	説明
^*string*	*string* で始まる値を見つける
string $	*string* で終わる値を見つける
.	ピリオドの数と等しい数の文字をもつ値を見つける
[*chars*]	角かっこ内の文字のどれかを含む値を見つける
***expr**	*expr* についてゼロ回以上一致する値を見つける
+	1 回以上現われるものを見つける
?	任意の値を見つける
\'*s-char* '	? や $ などの特殊文字を見つける
x | y	x または y の値を見つける

構文

最初の列を検索するには、以下のように入力します。

```
nismatch string tablename
nisgrep  reg-exp tablename
```

特定の列を検索するには、以下のように入力します。

```
nismatch column=string tablename
nisgrep column=reg-exp tablename
```

複数の列を検索するには、以下のように入力します。

```
nismatch column=string tablename ...\
nismatch [column=string,...],tablename
nisgrep column=reg-exp ... \
    tablename
```

表 **19–8** nismatch 構文と nisgrep 構文のオプション

オプション	説明
c	カウント。エントリ自体ではなく、検索指定に一致したエントリのカウントを表示する
h	ヘッダー。エントリの上のヘッダー行を表示し、各列名を表示する
M	マスター。マスターサーバーに収められているテーブルのエントリだけを表示する。これによって、最新情報を取得できる。デバッグにだけ使うようにする

最初の列を検索する

テーブルの最初の列で特定の値を検索するには、最初の列の値と「*tablename*」を入力します。nismatch では、この値は文字列でなければなりません。nisgrep では、この値は正規表現でなければなりません。

```
nismatch [-h] string tablename
nisgrep [-h] reg-expression tablename
```

次の例では、depts テーブルを検索して、最初の列の値が R&D となっているすべてのエントリを探します。

```
rootmaster% nismatch -h 'R&D' depts.doc.com.
rootmaster% nisgrep -h 'R&D' depts.doc.com.
```

注 – & がシェルによってメタキャラクタとして解釈されないように、R&D が引用符で囲まれています。

特定の列を検索する

最初の列以外の特定の列を検索するには、次の構文を使います。

```
nismatch column=string tablename
nisgrep column=reg-expression tablename
```

次の例では、depts テーブルを検索して、第 2 列の値が SanFran となっているすべてのエントリを探します。

```
rootmaster% nismatch -h Site=SanFran depts.doc.com.
rootmaster% nisgrep -h Site=SanFran depts.doc.com.
```

複数の列を検索する

2 つ以上の列で一致するエントリを検索するには、次の構文を使います。

```
nismatch [-h] [column=string, ... \
 column=string,...],tablename
nisgrep [-h] column=reg-exp ... \
 tablename
```

以下の例では、第 2 列の値が SanFran で、第 3 列の値が jhill のエントリを検索します。

```
rootmaster% nismatch -h [Site=SanFran,Name=jhill], depts.doc.com.
rootmaster% nisgrep -h Site=SanFran Name=jhill depts.doc.com.
```

nisln コマンド

nisln コマンドは、NIS+ オブジェクトとテーブルエントリの間でシンボリックリンクを作成します。すべての NIS+ 管理コマンドで、NIS+ オブジェクト間のリンクをたどるように指示する -L フラグを使用できます。

注 – テーブルエントリはリンクしないでください。あるテーブルから他のテーブルへのリンクはできますが、あるテーブルのエントリから他のテーブルのエントリへのリンクはできません。

他のオブジェクトまたはエントリへのリンクを作成するには、ソースオブジェクトまたはエントリ、つまり他のオブジェクトまたはエントリを指すものに対する変更権が必要です。

注意 – cred テーブルをリンクしないでください。org_dir ディレクトリには、それぞれ専用の cred テーブルが必要です。org_dir cred テーブルもリンクしないでください。

構文

リンクを作成するには次のように入力します。

```
nisln source target
```

表 19–9 nisln 構文のオプション

オプション	説明
-L	リンクをたどる。ソース (*source*) 自体がリンクである場合、新しいリンクはこれにはリンクされず、そのリンク元のソースにリンクされる
-D	デフォルト。リンクされたオブジェクトに対して別のデフォルトセットを指定する。デフォルトについては、287 ページの「デフォルトを無効にする」を参照

リンクを作成する

オブジェクト間のリンク (テーブルとディレクトリの間のリンクなど) を作成するには、最初に「ソース (*source*)」、次に「リンク先 (*target*)」の順で、両方のオブジェクト名を指定します。テーブルエントリはリンクしないでください。

```
nisln source-object target-object
```

nissetup コマンド

nissetup コマンドは、org_dir ディレクトリと groups_dir ディレクトリ、それにフルセットの NIS+ テーブルを作成することによって、既存の NIS+ ディレクトリオブジェクトをドメインに展開します。しかし、データでテーブルを生成することはありません。これを行うには、378 ページの「nisaddent コマンド」で説明する nisaddent コマンドが必要です。ドメインにディレクトリを展開することは、ドメインの設定作業の一部です。

注 – 新しい NIS+ ドメインを設定するのであれば、nissetup コマンドを使うより、nisserver スクリプトを使った方が簡単です。nisserver の使用法の詳細については、98 ページの「NIS+ ルートサーバーの設定」を参照してください。

---

nissetup コマンドは、NIS クライアントをサポートするドメインにもディレクトリを展開できます。

nissetup を使用するには、テーブルを格納するディレクトリに対する変更権が必要です。

ディレクトリを NIS+ ドメインに展開する

nissetup コマンドは、ディレクトリ名を付けても付けなくても使えます。ディレクトリ名を付けない場合、このコマンドはデフォルトのディレクトリを使います。追加される各オブジェクトは、出力に表示されます。

```
rootmaster# /usr/lib/nis/nissetup doc.com.
org_dir.doc.com. created
groups_dir.doc.com. created
a uto_master.org_dir.doc.com. created
auto_home.org_dir.doc.com. created
bootparams.org_dir.doc.com. created
cred.org_dir.doc.com. created
ethers.org_dir.doc.com. created
group.org_dir.doc.com. created
hosts.org_dir.doc.com. created
mail_aliases.org_dir.doc.com. created
sendmailvars.org_dir.doc.com. created
netmasks.org_dir.doc.com. created
netgroup.org_dir.doc.com. created
networks.org_dir.doc.com. created
passwd.org_dir.doc.com. created
protocols.org_dir.doc.com. created
rpc.org_dir.doc.com. created
services.org_dir.doc.com. created
timezone.org_dir.doc.com. created
```

ディレクトリを NIS 互換ドメインに展開する方法

NIS+ と NIS のクライアント要求をサポートするドメインにディレクトリを展開するには、-Y フラグを使います。作成するテーブルは、NIS のクライアント要求がアクセスできるように、未認証カテゴリに対する読み取り権が与えられます。

```
rootmaster# /usr/lib/nis/nissetup -Y Test.doc.com.
```

nisaddent コマンド

nisaddent コマンドは、テキストファイルまたは NIS マップからの情報を NIS+ テーブルにロードします。また、NIS+ テーブルの内容をテキストファイルに逆にダンプできます。NIS+ テーブルを初めて生成する場合は、第 9 章を参照してください。すべての前提条件と関連作業が説明してあります。

nisaddent を使用して、情報をある NIS+ テーブルから別のテーブルに (たとえば、別のドメインの同じ種類のテーブルに) 転送できますが、直接には転送できません。まず、テーブルの内容をファイルにダンプし、次にそのファイルを他のテーブルにロードする必要があります。ただし、ファイル内の情報は正しくフォーマットされていなければなりません。各テーブルに必要なフォーマットについては第 10 章で説明しています。

情報をテーブルにロードするとき、置換 (replace)、追加 (append)、またはマージ (merge) の 3 つのオプションを自由に使用できます。追加オプションは、ソースエントリを NIS+ テーブルに単純に追加します。置換オプションの場合、NIS+ は、まずテーブル内のすべての既存エントリを削除し、次にソースからエントリを追加します。大規模なテーブルでは、これによって多くのエントリセット (1 セットは既存エントリの削除用、他のセットは新エントリの追加用) がテーブルの .log ファイルに追加され、/var/nis 内の領域を占有し、複製への伝達に長時間を要することになります。

マージオプションでは、置換オプションと同じ結果が得られますが、異なるプロセスを使っており、複製に送信する動作数が大幅に減少します。このオプションでは、NIS+ は次の 3 種類のエントリを異なる方法で処理します。

- ソース内にだけ存在するエントリは、テーブルに「追加」される
- ソースとテーブルの両方に存在するエントリは、テーブル内で「更新」される
- NIS+ テーブルにだけ存在するエントリは、テーブルから「削除」される

大きなテーブルを更新する場合で、内容の変更があまりないときは、マージオプションを使用するとサーバーは多くの動作を節約できます。ソース内で重複していないエントリだけを削除する (置換オプションは全エントリを無差別に削除する) ため、重複エントリごとに 1 回の削除と 1 回の追加動作を省略できます。

初めてテーブルに情報をロードする場合、テーブルオブジェクトに対する作成権が必要です。テーブル内の既存情報を上書きする場合、テーブルに対する変更権が必要です。

構文

テキストファイルから情報をロードするには、以下のように入力します。

```
/usr/lib/nis/nisaddent -f filename table-type\ [domain]
/usr/lib/nis/nisaddent -f filename \
 -t tablename table-type [domain]
```

NIS マップから情報をロードするには、以下のように入力します。

```
/usr/lib/nis/nisaddent -y NISdomain table-type\
   [domain]
/usr/lib/nis/nisaddent -y NISdomain -t tablename table-type [domain]
/usr/lib/nis/nisaddent -Y map table-type [domain]
/usr/lib/nis/nisaddent -Y map -t tablename table-type [domain]
```

NIS+ テーブルから情報をファイルにダンプするには、以下のように入力します。

```
/usr/lib/nis/nisaddent -d [-t tablename tabletype ] > filename
```

ファイルから情報をロードする

ファイルの内容を NIS+ テーブルに転送するには、いくつかの方法があります。

- `-f` を単独で指定すると、*table-type* の内容が *filename* の内容に置き換えられます。

```
nisaddent -f filename  table-type
```

- `-f` と `-a` を同時に指定すると、*filename* の内容が *table-type* に「追加」されます。

```
nisaddent -a -f filename table-type
```

- `-f` と `-m` を同時に指定すると、*filename* の内容が *table-type* の内容に「マージ」されます。

```
nisaddent -m -f filename  table-type
```

次に示す 2 つの例では、テキストファイル `/etc/passwd.xfr` の内容が NIS+ `passwd` テーブルにロードされます。最初の例ではローカルドメインに、2 番目の例では別のドメインのテーブルにロードされます。

```
rootmaster# /usr/lib/nis/nisaddent -f /etc/passwd.xfr passwd
rootmaster# /usr/lib/nis/nisaddent -f /etc/shadow.xfr shadow
rootmaster# /usr/lib/nis/nisaddent -f /etc/passwd.xfr passwd sales.doc.com.
rootmaster# /usr/lib/nis/nisaddent -f /etc/shadow.xfr shadow sales.doc.com.
```

注 – `/etc` ディレクトリのファイルから NIS+ passwd テーブルを作成する場合は、`nisaddent` を 2 回 (`/etc/passwd` ファイルと `/etc/shadow` ファイルに対して 1 回ずつ) 実行する必要があります。

`/etc/inet/ipnodes` ファイル (IPv6 アドレス) のエントリを `ipnodes.org_dir` テーブルにマージするには、`-v` オプションと `-f` オプションを使用します。

```
rootmaster# /usr/lib/nis/nisaddent -mv -f /etc/inet/ipnodes ipnodes
```

もう 1 つの方法では、`stdin` をソースとして使います。しかし、`stdin` では `-m` オプションを使えません。次に例を示します。リダイレクト (`>`) やパイプ (`|`) を使用することは可能です。ただし、別のドメインに出力が行われるような形でパイプを使用することはできません。

作業	コマンド
リダイレクト	`cat` *filename* `>` `nisaddent` *table-type*

作業	コマンド
リダイレクト処理後、追加する	`cat` *filename* `> nisaddent -a` *table-type*
リダイレクト処理後、別のドメインに追加する	`cat` *filename*`> nisaddent -a` *table-type NIS+ domain*
パイプ	`cat` *filename* `| nisaddent` *table-type*
パイプ処理後、追加する	`cat` *filename* `| nisaddent -a` *table-type*

NIS+ テーブルが、オートマウンタテーブルの 1 つであるか標準以外のテーブルである場合、`-t` オプションと NIS+ テーブルの完全な名前を追加します。

```
master# nisaddent -f /etc/auto_home.xfr \
  -t auto_home.org_dir.doc.com.key-value
master# nisaddent -f /etc/auto_home.xfr \
  -t auto_home.org_dir.doc.com. key-value sales.doc.com.
```

NIS マップからデータをロードする

NIS マップから情報を転送する方法は 2 つあり、NIS ドメインを指定するか、あるいは実際の NIS マップを指定します。ドメインを指定した場合、NIS+ は、*table-type* に基づいて、`/var/yp/`*nisdomain* 内のどのマップファイルをソースとして使うかを判断します。`/var/yp/`*nisdomain* は「ローカル」ファイルでなければなりません。

NIS+ テーブル形式	**NIS** マップ名
`Hosts`	`hosts.byaddr`
`Nodes`	`ipnodes.byaddr`
`Passwd`	`passwd.byname`
`Group`	`group.byaddr`
`Ethers`	`ethers.byname`
`Netmasks`	`netmasks.byaddr`
`Networks`	`networks.byname`
`Protocols`	`protocols.byname`
`RPC`	`rpc.bynumber`
`Services`	`services.byname`

NIS ドメインの指定によって転送するには、`-y` (小文字) オプションを使い、NIS+ テーブル形式に加えて NIS ドメインを指定します。

「テーブルの置換」

```
nisaddent -y nisdomain table-type
```

「テーブルの追加」

```
nisaddent -a -y nisdomain table-type
```

「テーブルのマージ」

```
nisaddent -m -y nisdomain table-type
```

デフォルトでは、nisaddent は NIS+ テーブルの内容を NIS マップの内容に置き換えます。追加またはマージを行うには、-a または -m のオプションを使います。次の例では、対応する NIS マップ (passwd.byname) からの NIS+ passwd テーブルを old-doc ドメインにロードします。

```
rootmaster# /usr/lib/nis/nisaddent -y old-doc passwd
```

2 番目の例でも同じことを行いますが、ローカルドメイン doc.com. の代わりに、sales.doc.com. ドメインに対して行います。

```
rootmaster# /usr/lib/nis/nisaddent -y old-doc passwd sales.doc.com.
```

NIS+ テーブルが、オートマウンタテーブルの 1 つであるか非標準テーブルである場合、そのソースがファイルであるかのように、-t オプションと NIS テーブルの完全な名前を追加します。

```
rootmaster# nisaddent -y old-doc \
  -t auto_home.org_dir.doc.com. key-value
rootmaster# nisaddent -y old-doc \
  -t auto_home.org_dir.doc.com. key-value sales.doc.com.
```

マップファイルをドメイン用に使うのではなく、特定の NIS マップを指定したい場合、-Y (大文字) オプションを使い、マップ名を指定します。オプションを見つけやすいように、次の例では太字にしています。

```
rootmaster# nisaddent -Y hosts.byname hosts
rootmaster#  nisaddent -Y hosts.byname hosts sales.doc.com.
```

NIS マップがオートマウンタマップの 1 つであるか非標準マップである場合、-Y オプションと -t オプションを組み合わせます。

```
rootmaster# nisaddent -Y auto_home \
 -t auto_home.org_dir.doc.com. key-value
rootmaster# nisaddent -Y auto_home \
 -t auto_home.org_dir.doc.com. key-value sales.doc.com.
```

NIS+ テーブルの内容をファイルにダンプする

NIS+ テーブルの内容をファイルにダンプするには、-d と -t のオプションを使います。-d オプションは、ダンプするようコマンドに指示します。-t オプションは、NIS+ テーブルを指定します。

```
rootmaster# nisaddent -d auto_home \
 -t auto_home.org_dir.doc.com. key-value
rootmaster# nisaddent -d auto_home \
 -t auto_home.org_dir.doc.com.  key-value sales.doc.com.
```

第 20 章

サーバー使用のカスタマイズ

この章では、NIS+ クライアントが使用するサーバーのカスタマイズおよび制御方法について説明します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ サーバーと NIS+ クライアント

クライアントマシン、ユーザー、アプリケーション、またはプロセスは、アクティブな NIS+ サーバー (マスターまたは複製) を検索して、そこから必要な情報を取り込みます。巨大なネットワーク、サブネットを多く持ったネットワーク、あるいは広域リンクにまたがったネットワークでは、サーバーの使用法をカスタマイズすることにより NIS+ のパフォーマンスを強化できます。

デフォルトでのクライアントの検索動作

カスタマイズ前の状態で、`nisprefadm` コマンドによるサーバーの優先順位設定がなにも設定されていなければ、クライアントは、まず自分自身のローカルサブネット上の NIS+ サーバーから情報を入手しようとします。そのサブネットにアクティブなサーバーが見つかれば、応答のあった最初のローカルサーバーから必要な情報を入手します。ローカルサブネットにサーバーがない場合、次にクライアントは、ローカルサブネットの外部を検索し、応答のあった最初の遠隔サーバーから必要な NIS+ 情報を入手します。

ネットワークが大規模で、混雑している場合、上記のデフォルトの検索動作では NIS + の性能を十分に発揮させることができないことがあります。それは次のような理由によります。

- サブネット上の複数のサーバーが多数のクライアントに情報の配布を行なっている場合、クライアントのデフォルト検索パターンがランダムなために、他にあまり使われていないサーバーがある一方で、いくつかのサーバーの負荷が高くなりすぎる可能性があります。
- ローカルサブネットの外で NIS+ サーバーを検索する際、クライアントは、オーバーワークしているサーバーであっても、低速な広域ネットワーク接続方式 (たとえば、モデム) や、すでにトラフィック多い専用回線によってリンクされているサーバーであっても、最初に応答したものから情報を入手します。

優先サーバーを指定する

Solaris オペレーティング環境には、新規機能 (サーバーの使用のカスタマイズ) が搭載されています。この機能を使用すると、クライアントが NIS+ サーバーを検索する順序をコントロールできます。この新規機能では、次のような方法でサーバーの使用の度合いをバランスよくカスタマイズできます。

- クライアントが特定のサーバーを優先的に選択する (検索する) ように指定します。
- 使用できるローカルサーバーがない場合、クライアントが遠隔サーバーを使用してもよいかどうかを指定します。

指定した検索基準は、ドメイン内のすべてのクライアント、サブネット上のすべてのクライアント、またはマシンごとに独立したクライアントに適用できます。

注 – サーバー使用の優先順位を特定のマシンに設定すると、この優先順位は、そのマシンで実行中のユーザー、アプリケーション、処理、または他のクライアントすべてに適用されます。同じマシン上の別のクライアントに、異なるサーバー使用のパターンを設定できません。

広域ネットワーク (WAN) 上での NIS+

使用サーバーのカスタマイズは、多数のサブネットを持つ大規模ネットワークや、モデムまたは専用回線で接続された複数の地理的サイトにまたがるネットワークで使用すると特に効果があります。ネットワークの性能を最大にするには、サブネット間やより低速な接続によってリンクされたサイト間のトラフィックを最小限にする必要があります。これは、クライアントが使用できる NIS+ サーバーとその優先順位を指定することによって実現できます。このようにして、可能な限り NIS+ ネットワークトラフィックがローカルサブネットから出ないようにします。

サーバー使用の最適化 - 概要

この節では、サーバー使用のカスタマイズの概要について説明します。

nis_cachemgr が必要

使用サーバーをカスタマイズするには、クライアントで nis_cachemgr を実行している必要があります。クライアントマシンが nis_cachemgr を実行していない場合は、使用サーバーのカスタマイズ機能は利用できません。クライアントマシン上で、nis_cachemgr を実行していない場合は、そのクライアントは、385 ページの「デフォルトでのクライアントの検索動作」で説明した方法で識別された最初のサーバーを使用します。

グローバルテーブルまたはローカルファイル

nisprefadm コマンドは、次のように、その使用方法により、ローカルの client_info ファイルまたはドメインの client_info テーブルのどちらかを作成します。

- 「ファイル」。nisprefadm を使用して、マシンの /var/nis ディレクトリに格納される、ローカルなマシン固有の client_info を作成できます。ローカルファイルは、サーバーの優先順位をそのマシンだけに指定するためのものです。マシンに、ローカルの /var/nis/client_info ファイルがある場合、そのマシンは、ドメインの client_info.org_dir テーブルに入っているサーバーの優先順位は無視します。ローカル client_info ファイルを作成するには、-L オプションを指定した nisprefadm を実行してください。
- 「テーブル」。nisprefadm を使用して、各ドメインの NIS+ ディレクトリオブジェクト org_dir 内に格納される NIS+ client_info テーブルを作成できます。このテーブルは、次のものに対してサーバーの優先順位を指定できます。
  - 個別のマシン。マシンにローカルの /var/nis/client_info ファイルがある場合、ドメインの client_info テーブルによって、そのマシンに対して指定された優先順位はすべて無視されます。
  - 1 つの特定のサブネット上にあるすべてのマシン。サブネット上のマシンにローカルの /var/nis/client_info ファイルがあるか、またはテーブル内にそのマシン固有の優先順位が設定されている場合、そのマシンは、サブネットに指定された優先順位を無視します。

  1 つのサブネット上のすべてのマシンに適用されるグローバル client_info テーブルを作成するには、-G および -C オプションを指定して nisprefadm を実行してください。これについては、394 ページの「グローバルサーバー優先順位を指定する」で説明しています。

マシンに、以下で説明するような、固有のローカル client_info ファイルがある場合、そのマシンに対してグローバル client_info テーブル内に設定されたサーバーの優先順位はすべて無視されるので注意してください。マシンに、ローカル client_info ファイルがあるか、またはグローバル client_info テーブルにそのマシン固有のエントリがある場合は、そのサブネットに設定された優先順位は無視されます。

注意 – client_info ファイルと client_info テーブルを変更する場合は、nistbladm コマンドだけを使用してください。nistbladm などの、他の NIS+ コマンドは絶対に使用しないでください。

client_info テーブルまたは client_info ファイルを処理する場合は、-G または -L オプションを使用して、グローバルテーブル (-G) またはコマンドを実行中のマシンのローカルファイル (-L) のどちらにそのコマンドを適用させるかを指定する必要があります。

優先順位の番号の指定

サーバーの優先順位は、各サーバーに「優先順位を表す番号」を指定することによって制御されます。クライアントは、数値で指定された優先順位にしたがって NIS+ サーバーを検索します。検索の順序は、まず、優先順位格付け番号が小さいサーバーを先に検索し、次に大きい数値の付いたサーバーを検索します。

つまり、クライアントは、まず初めにゼロの優先順位の付いた NIS+ サーバーから名前空間情報を入手しようとします。使用できる優先順位=0 のサーバーがない場合、クライアントは、次に優先順位=1 のサーバーに問い合わせを行います。1 のサーバーが使用できない場合は、2 のサーバーを検出しようと、次に 3 というように、この検索は必要な情報が入手できるか、または検索するサーバーがなくなるまで続きます。

優先順位格付け番号は、nisprefadm コマンドを使用してサーバーに割り当てます。これについては、394 ページの「グローバルサーバー優先順位を指定する」で説明しています。

サーバーの優先順位番号は、client_info テーブルと client_info ファイル内に格納されます。マシンに、固有の /var/nis/client_info ファイルがある場合は、このファイル内に格納された優先順位番号が使用されます。マシンに、固有の client_info ファイルがない場合は、ドメインの client_info.org_dir テーブル内に格納された優先順位番号が使用されます。このような client_info テーブルと client_info ファイルを、「優先サーバーのリスト」、または単に「サーバーリスト」と呼びます。

各クライアントのサーバーの優先順位を制御することで、サーバーの使用法をカスタマイズします。たとえば、ドメインに mailer という名前のクライアントマシンがあり、このマシンは名前空間情報を頻繁に利用していて、さらにこのドメインには、マスターサーバー (nismaster) と複製サーバー (replica1) の両方があるとします。

ここで、mailer マシン用に nismaster に優先順位番号 1 を割り当て、replica1 に優先順位番号 0 を割り当てると、mailer マシンは、名前空間情報を、必ず最初に replica1 から入手しようとし、その後で nismaster に移ります。次に、このサブネット上にある他のすべてのマシンに対して、優先順位番号を nismaster サーバーにはゼロを、replica1 には 1 を割り当てます。この結果、他のマシンは、必ず最初に nismaster に問い合わせを行います。

同じ優先順位番号を、ドメイン内の複数のサーバーに割り当てることができます。たとえば、nismaster1 と replica2 の両方に優先順位番号 0 を割り当て、replica3、replica4、replica5 に優先順位番号 1 を割り当てることができます。

サーバーの優先順位のデフォルト

client_info ファイルまたはテーブルがない場合は、キャッシュ管理プログラムが自動的に、ローカルサブネット上のすべてのサーバーにデフォルトの優先順位番号ゼロ (0) を割り当て、ローカルサブネットの外部のすべてのサーバーに無限大の優先順位を割り当てます。nisprefadm は、このデフォルトの優先順位番号を自由に変更するためのものです。

効率とサーバーの優先順位番号

クライアントは、すべてのサーバーを指定された優先順位番号を使用して昇順に検索しなければなりません。クライアントが指定された優先順位番号をもつすべてのサーバーを検索するには、5 秒以上必要です。つまり、ドメイン内に 1 つのマスターサーバーと 4 つの複製サーバーがあり、それぞれのサーバーに 0 から 4 の異なる優先順位番号を指定した場合、クライアントがこれらの優先レベルをすべて実行するには、25 秒以上かかるということです。

性能を最大にするために、サーバーの優先レベルを 2 つまたは 3 つ以上使用しないでください。たとえば、上記のような場合は、5 つのサーバーのうちの 1 つに優先順位=0 を割り当て、他すべてのサーバーには優先順位 1 を割り当てるか、または 5 つのサーバーのうち 2 つに優先順位 1 を、残りの 3 つのサーバーに 2 を割り当てるといった方法をとってください。

優先サーバー限定と全サーバー

サーバーリストには、クライアントが優先サーバーを検出できなかった場合の処置も指定できます。「優先サーバー」とは、優先順位にゼロが指定されたサーバー、または nisprefadm を使用して優先順位番号を割り当てたサーバーです。

デフォルトでは、クライアントは優先サーバーに到達できないと、検索モードを使用して、ネットワーク上のあらゆる場所を検索し、検出できるあらゆるサーバーを検索します。これについては、385 ページの「デフォルトでのクライアントの検索動作」

で説明しています。このデフォルト機能を、nisprefadm -o オプションを使用して変更し、クライアントが優先サーバーだけを使用し、使用できるサーバーがない場合でも、優先サーバー以外のサーバーには問い合わせしないように指定できます。詳細については、400 ページの「優先サーバー限定指定」を参照してください。

注 – マシンのドメインに優先サーバーが全くない場合、このオプションは無視されます。

優先順位の表示

現在特定のクライアントマシンに対して有効なサーバーの優先順位を表示するには、-l オプションを指定した nisprefadm を実行してください。これについては、393 ページの「現行のサーバー優先順位の表示」で説明しています。

サーバー名とクライアント名

サーバーまたはクライアントマシンを指定する場合、次の点に注意してください。

- サーバー名とクライアント名は、同じ NIS+ ドメイン内にあり、オブジェクトを個別に特定できれば、完全指定名である必要はありません。マシン名を単独で使用できます。
- サーバーまたはサブネットが別の NIS+ ドメインにある場合は、そのマシンを個別に特定できるだけのドメイン名を含める必要があります。たとえば、sales.doc.com ドメイン内にいて、manf.doc.com ドメイン内の nismaster2 マシンを指定する必要がある場合、nismaster2.manf と入力します。

サーバーの優先順位

以下のマシンにサーバーの優先順位を指定する方法は、それぞれ次のとおりです。

- 「個別のクライアントマシン」にサーバーの優先順位を指定するには、-L オプションを使用すると、nisprefadm を実行中のマシンにローカル client_info ファイルを作成します。-G -C マシンオプションを使用すると、グローバル client_info テーブル内にマシン固有の優先順位を作成します。
- 「サブネット上のすべてのマシン」にサーバーの優先順位を指定するには、-G -C サブネット番号オプションを使用してください。
- マシン固有のまたはサブネット固有の優先順位を持たない、現在のドメイン内にあるすべてのマシンにサーバーの優先順位を指定するには、-G オプションを使用してください。

サーバーの優先順位が有効になるタイミング

マシンまたはサブネットのサーバー優先順位の変更内容は、通常、指定したマシンがその nis_cachemgr データを更新するまでは有効になりません。マシンの nis_cachemgr がそのサーバー使用情報を更新するタイミングは、マシンがサーバーの優先順位をグローバル client_info テーブルまたはローカル /var/nis/client_info ファイルのどちらから入手するかによって決まります (387 ページの「グローバルテーブルまたはローカルファイル」を参照してください)。

- 「グローバルテーブル」。グローバルテーブルから入手する場合、サーバーの優先順位をグローバルテーブルから入手するマシンのキャッシュ管理プログラムは、マシンがブートされた時、または client_info テーブルの存続時間 (TTL) の値が満了した時に、マシン用のサーバーの優先順位を更新します。デフォルトでは、この TTL 値は 12 時間ですが、変更可能です。352 ページの「オブジェクトの生存期間を変更する」を参照してください。
- 「ローカルファイル」。ローカルファイルからサーバーの優先順位を入手するマシンのキャッシュ管理プログラムは、サーバーの優先順位を 12 時間ごとに更新するか、または nisprefadm を実行してサーバーの優先順位を変更した時に必ず更新します。マシンをリブートしても、キャッシュ管理プログラムのサーバーの優先順位情報は更新されません。

ただし、nisprefadm に -F オプションを指定して実行すると、サーバーの優先順位の変更内容を強制的にただちに有効にできます。-F オプションを使用すると、nis_cachemgr で、ただちにその情報が更新されます。詳細は、404 ページの「優先順位の変更内容をただちに実現する方法」を参照してください。

nisprefadm コマンドの使用方法

ここからの節では、nisprefadm コマンドを使用して、サーバーの優先順位を設定、変更、削除する方法について説明します。

nisprefadm コマンドは、クライアントが優先的に選択するサーバーを指定するために使用します。

nisprefadm コマンドの構文は次のとおりです。

```
nisprefadm -a|-m|-r|-u|-x|-l -L|-G [-o type] \
  [-d domain] \
  [-C machine] \
  servers
nisprefadm -F
```

表 **20–1** nisprefadm コマンドのオプション

オプション	説明
-G	ドメインの org_dir ディレクトリ内に格納されるグローバル client_info テーブルを作成する。つまり、グローバルな優先されるサーバーリストを作成する。このオプションは、指定したサブネット上のすべてのマシンに対して優先順位を指定する場合は -C *subnet*、個別のマシンに優先順位を指定する場合は -C *machine* のどちらかと同時に使用しなければならない
-L	ローカルマシンの /var/nis ディレクトリに格納されるローカル client_info ファイルを作成する。つまり、コマンドを実行中のマシンにだけ適用される優先サーバーのリストを作成する
-o *type*	オプションを指定する。有効なオプションには、クライアントが接続できる優先サーバーがない場合、優先サーバー以外のサーバーを使用できるように指定する pref_type=all と、指定した優先サーバーだけをクライアントが使用するように指定する pref_type=pref_only がある
-d *domain*	指定したドメインまたはサブドメインに、グローバルな優先サーバーの client_info テーブルを作成する
-C *subnet*	優先順位を適用するサブネットの番号
-C *machine*	クライアントマシン名
servers	1 つまたは複数の NIS+ サーバー。ここで指定されたサーバーは優先的に選択される
-a	サーバーリストに指定サーバーを追加する
-m	サーバーリストを変更する。たとえば、-m オプションを使用すると、1 つまたは複数のサーバーに指定された優先順位番号を変更できる
-r	サーバーリストから指定したサーバーを削除する
-u	サーバーリストを消去してから、指定したサーバーを追加する (つまり、現行のサーバーリストを優先サーバーの新規リストに置換する)
-x	サーバーリストを完全に削除する
-l	現行の優先サーバー情報を一覧表示 (表示) する
-F	優先サーバーのリストを強制的にただちに変更する

注 – -C *machine* オプションは、-L (ローカル) フラグとともに使用しても無効となるので、使用しないでください。たとえば、nisprefadm を altair マシン上で実行しているとします。ここで、-L フラグを使用して、指定した優先順位が altair のローカル client_info ファイルに書き込まれるように指定し、さらに、-C vega オプションを使用して、作成した優先順位が vega マシンに適用されるように指定します。すると、nisprefadm コマンドは、vega 用の優先順位を altair のファイルに書き込みますが、vega は、サーバーの優先順位を必ず自分のローカル client_info ファイルまたはドメインのグローバル client_info テーブルから入手するため、これらの情報を参照することはありません。そのため、-C オプションは、nisprefadm に -G (グローバル) フラグを指定して実行する場合にだけ意味をもちます。

現行のサーバー優先順位の表示

現行のサーバー優先順位を表示するには、nisprefadm に -l オプションを指定して実行してください。

マシンに指定された優先順位の表示方法

- そのマシン上で、nisprefadm に -L と -l を指定して実行します。

```
sirius# nisprefadm -L -l
```

このようにすると、マシンのローカル /var/nis/client_info ファイル内に定義されたサーバーの優先順位がすべて表示されます。ローカルファイルがない場合は、情報は表示されず、シェルプロンプトに戻ります。

1 台のマシンに設定されたグローバルな優先順位の表示方法

- nisprefadm に、-l、-G、-C *machinename* オプションを指定して実行します。

```
sirius# nisprefadm -G -l -C machinename
```

machinename には、マシンの IP アドレス (番号) が入ります。

このようにすると、このサブネット用にドメインのグローバル client_info テーブル内に設定された優先順位が表示されます。

サブネットに設定されたグローバル優先順位の表示方法

- `nisprefadm` に、`-l`、`-G`、`-C` *subnet* オプションを指定して実行します。

  ```
  sirius# nisprefadm -G -l -C subnet
  ```

  subnet には、サブネットの IP アドレス (番号) が入ります。

  このようにすると、このサブネット用にドメインのグローバル `client_info` テーブル内に設定された優先順位が表示されます。

優先順位格付け番号の指定方法

デフォルトでは、`-a` オプションの後ろにリストしたサーバーには、すべて優先順位番号ゼロが指定されます。別の優先順位番号を指定する場合は、次のように、サーバー名の直後に指定する番号をカッコで囲んで入れてください (`-a` *name*`(`*n*`)`)。*name* にはサーバー名が、*n* には優先順位番号が入ります。

たとえば、`replica2` サーバーに優先順位番号 3 を割り当てる場合は、次のように入力します。

```
# nisprefadm -G -a replica2(3)
```

注 – シェルの中には、次のように要素を 2 重引用符で囲まなければならないものがあります。`"name(n)"`

サーバーの優先順位格付け番号の基礎知識については、388 ページの「優先順位の番号の指定」を参照してください。

グローバルサーバー優先順位を指定する

ローカルまたは遠隔ドメインに、グローバルサーバー優先順位を設定できます。優先順位は、個別のマシンと 1 つのサブネット上のすべてのマシンに設定できます。

この節では、NIS+ ドメインのマスターサーバーに存在するグローバル `client_info` テーブル内にサーバーの優先順位を設定する方法を説明します。マスターサーバー上にテーブルがある場合、NIS+ は、そのテーブルをドメインの既存の複製サーバー上に複写します。

- 個別のマシンにローカル client_info ファイルを作成する方法については、396 ページの「ローカルサーバー優先順位を指定する」 を参照してください。
- グローバル client_info テーブルとローカル client_info ファイルの相違点については、387 ページの「グローバルテーブルまたはローカルファイル」を参照してください。

サーバーの優先順位番号を割り当てるには、次のどちらかを指定して nisprefadm を実行してください。

- 新規または別の優先サーバーを追加する場合は、-a オプションを指定します。
- 既存のサーバー優先順位を削除して、新規の優先順位を作成する場合は、-u オプションを指定します。

サブネットにグローバル優先順位を設定する方法

1 つのサブネット上のすべてのマシンのグローバルテーブルにサーバーの優先順位を割り当てるには、次のようにします。

- nisprefadm に -G および -C *subnet* オプションを指定して実行します。

  # nisprefadm -G -a -C *subnet servers* (*preferences*)

  引数の意味はそれぞれ以下のとおりです。

  - -C *subnet* には、優先順位を適用するサブネットの IP 番号を指定します。
  - *servers (preferences)* には、1 つまたは複数のサーバーが入ります。それらには優先順位格付け番号を指定できます。

    たとえば、サブネット 123.123.123.123 が、nismaster および replica3 サーバーにはデフォルトの優先順位番号ゼロを付け、manf.replica6 サーバーには優先順位番号 1 を付けて使用するように指定する場合は、次のように入力してください。

```
polaris# nisprefadm -a -G -C 123.123.123.123 nismaster1 \
replica3 "manf.replica6(1)"
```

個別のマシンにグローバル優先順位を設定する方法

- -G と -C マシンオプションを指定した nisprefadm を実行します。

  #nisprefadm -G -a -C *machine servers* (*preferences*)

  引数の意味はそれぞれ以下のとおりです。

  - -C マシンには、優先順位を適用するマシンを指定します。使用するシェルによっては、*machine* を引用符で囲む必要があります。

- *servers (preferences)* には、1 つまたは複数のサーバーが入ります。それらには優先順位格付け番号を指定できます。

  たとえば、マシン cygnus に指定された現行の優先順位を、replica7 と replica9 の両方にデフォルトの優先順位番号ゼロを指定したものに置き換える場合は、次のように入力します。

```
polaris# nisprefadm -u -G -C cygnus replica7 replica9
```

遠隔ドメインにグローバル優先順位を設定する方法

遠隔ドメイン内の個別のマシン、または遠隔ドメイン内の 1 つのサブネット上にあるすべてのマシンにサーバーの優先順位を割り当てるには、次のようにします。

- nisprefadm に -C、-G、-d オプションを指定して実行します。

```
#nisprefadm -a -G -C name \
 -d domain servers(preferences)
```

  - *name* には、サブネットの IP 番号またはマシン名が入ります。このコマンドを使用して行なった変更は、ここに指定したサブネットまたはマシンに適用されます。
  - *domainname* には、遠隔ドメイン名が入ります。
  - *servers (preferences)* には、1 つまたは複数のサーバーが入ります。それらには優先順位格付け番号を指定できます。

  たとえば、デフォルトの優先順位番号ゼロを指定した nismaster2 サーバーを、遠隔ドメイン sales.doc.com 内のサブネット 111.11.111.11 の優先サーバーのリストに追加する場合は、次のように入力します。

```
polaris# nisprefadm -a -G -C 111.11.111.11 -d sales.doc.com. nismaster2
```

ローカルサーバー優先順位を指定する

ここでは、ローカル client_info ファイルの作成および変更方法について説明します。ローカル client_info ファイルは、このファイルが存在するマシンにサーバーの優先順位を指定するためのものです。

マシンに /var/nis/client_info ファイルがある場合、このマシンは、NIS+ サーバー上のグローバル client_info テーブルではなく、そのマシンのローカルファイルからサーバーの優先順位を入手します。つまり、ローカルファイルはグローバルテーブルよりも優先されます。

- NIS+ サーバーのグローバル client_info テーブルを作成する方法については、394 ページの「グローバルサーバー優先順位を指定する」を参照してください。
- グローバル client_info テーブルとローカル client_info ファイルの相違点については、387 ページの「グローバルテーブルまたはローカルファイル」を参照してください。

サーバーの優先順位を割り当てるには、次のどちらかを指定した nisprefadm を実行してください。

- 新規または別の優先サーバーを追加する場合は、-a オプションを指定します。
- 既存のサーバー優先順位を削除して、新規の優先順位を作成する場合は、-u オプションを指定します。

ローカルマシン上に優先順位を設定する方法

nisprefadm コマンドを実行中のローカルマシンにサーバーの優先順位を割り当てるには、次のようにします。

- nisprefadm に -L オプションと -a または -u オプションを指定して実行します。

```
#nisprefadm -a -L servers(preferences)
```

ここで、*servers (preferences)* には、1 つまたは複数のサーバーが入ります。それらには優先順位格付け番号を指定できます。

たとえば、deneb マシンが NIS+ 情報を、まず初めにデフォルトの優先順位番号ゼロが指定された replica3 から検索し、次に manf.doc.com ドメイン内の (優先順位番号 1 が指定された) サーバー replica6 を検索するように指定する場合は、次のように入力します。

```
deneb# nisprefadm -a -L replica3 replica6.manf(1)
```

サーバーの優先順位を変更する

サーバーの優先順位番号は、変更することができ、また別のサーバーに優先順位番号の割り当てを移すこともできます

優先サーバーまたはサーバーに割り当てた優先順位番号を変更するには、nisprefadm に -m *oldserver-=newserver*(*n*) オプションを指定して実行してください。

サーバーの優先順位番号を変更する方法

- nisprefadm に -m *server=server(new)* オプションを指定して実行します。

```
#nisprefadm -L|-G -C name -m oldserver=newserver(n)
```

引数の意味はそれぞれ以下のとおりです。

- L|G には、ローカルファイルとグローバルテーブルのどちらを修正するかを指定します。
- -C *name* には、サブネットの IP 番号またはマシン名が入ります。このオプションは、-G オプションを使用している場合にだけ使用します。このコマンドを使用して行なった変更は、ここに指定したサブネットまたはマシンに適用されます。
- -m は、サーバーリストを変更するオプションです。
- *old server* には、その優先順位番号を変更したいサーバー名が入ります。
- *new server(n)* には、サーバー名とその新しい優先順位番号が入ります。

たとえば、deneb マシン上で、deneb のローカル client_info ファイルの replica6.manf サーバーに指定された番号を 2 に変更する場合は、次のように入力します。

```
deneb# nisprefadm -L -m replica6.manf=replica6.manf(2)
```

優先順位リスト内で 1 つのサーバーを別のサーバーに置換する方法

優先順位リスト内で、1 つのサーバーを別のサーバーに変更するには、次のようにします。

- nisprefadm に -m *oldserver=newserver* オプションを指定して実行します。

```
#nisprefadm -L|-G -C name -m \
 oldserver=newserver(prefnumber)
```

引数の意味はそれぞれ以下のとおりです。

- -L|-G には、ローカルまたはドメイン全体のどちらのサーバーリストを変更するのかを指定します。
- -C *name* には、サブネットの IP 番号またはマシン名が入ります。このオプションは、-G オプションを使用している場合にだけ使用します。このコマンドを使用して行なった変更は、ここに指定したサブネットまたはマシンに適用されます。
- -m は、サーバーリストを変更するオプションです。
- *oldserver* には、置換する対象となる旧サーバーが入ります。
- *newserver (prefnumber)* には、優先サーバーリスト内の旧サーバーの位置に入る新規サーバー (優先順位格付け番号を指定できる) を入力します。

-G オプションを使用して、グローバル client_info テーブル内のサーバーを置換する場合には、その置換内容は -C オプションで特定したサブネットまたはマシンにだけ適用されるので注意してください。それ以外にリストされている置換対象 (旧) サーバーは影響を受けません。

たとえば、3 つのサブネットを持つドメインを所有していて、この中の 2 つのサブネット用に replica1 サーバーが優先サーバーのリストに入っている場合を考えます。これ以上 replica1 を使用しないので、使用サーバーから除外する場合は、nisprefadm -m を実行して、最初のサブネットの replica1 を新規サーバーに置換します。この時、2 番目のサブネットに対しても同様の処理を行うまでは、replica1 はそのサブネットの優先サーバーのリストに入っています。個別のマシンの優先サーバーでも同様です。

たとえば、ドメインのグローバル client_info テーブル内で、サブネット 123.12.123.12 用に指定された replica3 サーバーを replica6 サーバーに置換し、replica6 に優先順位番号 1 を割り当てる場合は、次のように入力します。

```
nismaster# nisprefadm -G -C 123.12.123.12 -m replica3 replica6(1)
```

優先順位リストからサーバーを削除する方法

優先順位リストから 1 つまたは複数のサーバーを削除するには、次のようにします。

- nisprefadm に -r オプションを指定して実行します。

```
#nisprefadm -L|-G -C name -r servers
```

引数の意味はそれぞれ以下のとおりです。

- -L|-G には、ローカルまたはドメイン全体のサーバーリストのどちらを変更するのかを指定します。
- -C *name* には、サブネットの IP 番号またはマシン名が入ります。このオプションは、-G オプションを使用している場合にだけ使用します。このコマンドを使用した優先サーバーの削除は、ここに名前を指定したサブネットまたはマシンに適用されます。
- -r は、*servers* に名前を指定したサーバーをリストから削除するオプションです。

たとえば、ドメインのグローバル client_info テーブル内から、マシン polaris に指定された replica3 および replica6.manf サーバーを削除する場合は、次のように入力します。

```
polaris# nisprefadm -G -C polaris -r replica3 replica6.manf
```

優先サーバーリスト全体を置換する方法

グローバル client_info テーブルまたはマシンのローカル client_info ファイルどちらかの中にある、サブネットまたはマシンに指定された優先サーバーのリスト全体を置換するには、nisprefadm に -u オプションを指定して実行してください。

-u オプションは、マシンまたはサブネットに指定されたサーバーの優先順位を、まず初めにすべて削除してから指定した新規の優先順位を追加しますが、これ以外は、-a オプションと同じ処理を行います。既存の優先順位がある場合、-a オプションでは旧リストに新しい優先順位を追加します。

-u オプションの使用例については、395 ページの「個別のマシンにグローバル優先順位を設定する方法」を参照してください。

優先サーバー限定指定

優先サーバーが使用できない場合に、クライアントがとる動作を指定できます。

デフォルトでは、クライアントは、優先サーバーに到達できない場合、検出できたサーバーであればどれでも使用してしまいます。優先サーバー限定オプションを設定すると、クライアントが優先サーバーだけを使用するように指定できます。優先サーバー限定オプションと全サーバーオプションの基礎知識については、389 ページの「優先サーバー限定と全サーバー」を参照してください。

優先サーバーが使用できない場合のクライアントの動作を指定するには、nisprefadm に -o *value* オプションを指定して実行してください。

優先サーバーだけを指定する方法

サーバーリストを使用するクライアントが、リスト内に記述されたサーバーだけから NIS+ 情報を入手するように指定するには、次のようにします。

- nisprefadm に -o pref_only オプションを指定して実行してください。

```
# nisprefadm -L|-G -o pref_only
```

引数の意味はそれぞれ以下のとおりです。

- -L|-G には、ローカルまたはドメイン全体のサーバーリストのどちらを変更するかを指定します。
- -o pref_only により、クライアントがリスト内に記述されたサーバーだけから NIS+ 情報を入手するように指定されます。

注 – このオプションは、優先サーバーが全くないドメインでは無視されます。

たとえば、altair のローカル client_info ファイル内に、altair が必ず優先サーバーを使用し、altair の優先サーバーリストにないサーバーは使用しないように指定するには、次のように入力します。

```
altair# nisprefadm -L -o pref_only
```

優先サーバー以外のサーバーを使用する方法

サーバーリストを使用するクライアントが、優先サーバーが使用できない場合に、リストに記載されていないサーバーから NIS+ 情報を入手するように指定するには、次のようにします。

- nisprefadm に -o all オプションを指定して実行してください。

```
# nisprefadm -L|-G -o all
```

引数の意味はそれぞれ以下のとおりです。

- -L|-G には、ローカルまたはドメイン全体のサーバーリストのどちらを変更するかを指定します。
- -o all は、クライアントが、優先サーバーが使用できない場合に、リストに記載されていないサーバーから NIS+ 情報を入手するように指定するオプションです。

注 – これは、デフォルトの動作です。そのため、-o all オプションは、以前に -o pref_only オプションを使用して優先サーバーだけを指定していた場合に限り使用する必要があります。

たとえば、altair が優先サーバーを使用できない場合には、altair のローカル client_info ファイル内に、優先サーバー以外のサーバーを使用できるように指定し直すには、次のように入力してください。

```
altair# nisprefadm -L -o all
```

サーバーの優先順位の使用の終了

使用サーバーのカスタマイズ機能の使用を終了し、385 ページの「デフォルトでのクライアントの検索動作」で説明した NIS+ 情報の入手方法に戻すことができます。

サーバーの優先順位の使用を終了するには、nisprefadm に -x オプションを指定して実行してください。

注 – サーバーの優先順位の使用を終了する場合、クライアントは、391 ページの「サーバーの優先順位が有効になるタイミング」で説明していることがらが通常どおり進行するまではサーバーの優先順位の使用を終了しません。また、サーバーの優先順位の使用を強制的にただちに終了させることもできます。これについては、404 ページの「サーバーの優先順位の有効化」で説明しています。

グローバルサーバー優先順位を削除する方法

- nisprefadm に -G および -x オプションを指定して実行してください。

  ```
  # nisprefadm -G -x
  ```

  これにより、グローバルサーバーの優先順位が削除されます。

  - ローカルサーバーの優先順位が指定されていないクライアントマシンは、385 ページの「デフォルトでのクライアントの検索動作」で説明したように NIS+ 情報を入手します。
  - ローカル /var/nis/client_info ファイルで設定されたサーバーの優先順位があるクライアントマシンは、引き続きそのファイルに指定されたサーバーを使用します。

ローカルサーバー優先順位を削除する方法

ローカル優先順位を終了するということは、次の異なる 3 つの事項のいずれか 1 つを意味すると考えられます。

- 特定のマシンで、サーバーの優先順位にローカル client_info ファイルの使用を終了し、かわってドメインのグローバル client_info テーブル内にそのサブネット用に設定された優先順位を使用したい場合
- このマシンで、サーバーの優先順位にローカル client_info ファイルの使用を終了し、かわってドメインのグローバル client_info テーブル内にそのマシン固有に設定された優先順位を使用したい場合
- 特定のマシンで、サーバーの優先順位をまったく使用したくない場合。マシンがサーバーの優先順位を使用しない場合、そのマシンは、385 ページの「デフォルトでのクライアントの検索動作」に説明した方法で NIS+ 情報を入手する

ローカルからグローバルサブネットの優先順位に置換する方法

- マシンの /var/nis/client_info ファイルを削除します。

  ```
  # rm /var/nis/client_info
  ```

  この結果、マシンはドメインのグローバル client_info テーブル内でマシンのサブネットに指定された優先順位を使用するようになります。

ローカルからマシン固有のグローバル優先順位に置換する方法

1. マシンの /var/nis/client_info ファイルを削除します。

   ```
   # rm /var/nis/client_info
   ```

2. -G と -C オプションを使用して、グローバルテーブル内にそのマシン用の優先順位を指定します。

   詳細は、395 ページの「個別のマシンにグローバル優先順位を設定する方法」を参照してください。

マシンにサーバーの優先順位の使用を中止させる方法

1. マシンの /var/nis/client_info ファイルを削除します。

   ```
   # rm /var/nis/client_info
   ```

   マシンのドメインにグローバル client_info テーブルがない場合、必要な処理はこの手順だけです。ドメインに client_info テーブルがある場合は、手順 2 に進んでください。

2. 空の /var/nis/client_info ファイルを作成します。

   ```
   # touch /var/nis/client_info
   ```

   マシンに固有の /var/nis/client_info ファイルがある場合、そのマシンは client_info テーブルのグローバル優先順位は使用しません。マシンに空の /var/nis/client_info ファイルがある場合、そのマシンはどの優先順位も使用せず、385 ページの「デフォルトでのクライアントの検索動作」で説明した方法で NIS+ 情報を入手します。

サーバーの優先順位の有効化

使用サーバーの変更内容は通常、クライアントマシンをリブートするか、またはマシンのキャッシュ管理プログラムを更新した時に有効になります。

ローカル client_infoファイル (-L オプション) を使用するローカルマシン上で、nisprefadm を使用してサーバーの優先順位の設定または変更した場合は、その変更内容はただちに有効になります。

グローバル client_info テーブル (-G オプション) からサーバーの優先順位を入手しているマシンの場合、nisprefadm に -F オプションを指定して実行すると、サーバーの優先順位に加えた変更を強制的にただちに有効にできます。

```
# nisprefadm -F
```

-F オプションとは、マシンのキャッシュ管理プログラムに、そのサーバーの優先順位情報をドメインのグローバル client_info テーブルを使って強制的にただちに更新させるオプションです。nisprefadm -F を実行するマシンが、/var/nis 内にそのマシン固有のローカル client_info ファイルを持つ場合、そのマシン上で nisprefadm -F を実行しても無効になります。

注 – -F オプションは、他の nisprefadm オプションと同時に使用できません。nisprefadm -F コマンドは、必ず、このコマンドを適用するマシン上で、単独で使用してください。-G オプションを使用して、ドメイン内にあるすべてのマシンのキャッシュ管理プログラムを更新できません。nisprefadm -F コマンドは、各マシン上でそれぞれ実行する必要があります。

優先順位の変更内容をただちに実現する方法

新しく作成、または変更したサーバーリストを、指定したマシン上で強制的にただちに有効にするためには、次のようにします。

- マシン上で、nisprefadm に -F オプションを指定して実行します。

  ```
  # nisprefadm -F
  ```

  たとえば、vega の優先サーバーリストへの変更内容を、強制的にただちに実現させるには (ローカルまたはグローバルのどちらの場合でも)、次のように入力してください。

  ```
  vega# nisprefadm -F
  ```

第 21 章

NIS+ のバックアップと復元

この章では、NIS+ 名前空間のバックアップ方法と復元方法について説明します。

NIS+ のバックアップ機能と復元機能を使用すると、NIS+ 名前空間の保存と復元を素早く簡単に行うことができます。また、これらの機能を使用すると、新しい複製サーバーを簡単に作成でき、さらにそのサーバーをオンラインにするためにかかる時間を削減できます。これらのタスクは、次の 2 種類のコマンドを使用して実行します。

- `nisbackup` - NIS+ ディレクトリオブジェクトをバックアップする
- `nisrestore` - NIS+ ディレクトリオブジェクトを復元する

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

`nisbackup` を使用して名前空間をバックアップする

`nisbackup` コマンドは、1 つまたは複数の NIS+ ディレクトリオブジェクト、または 1 つの名前空間全体を、指定した UNIX ファイルシステムのディレクトリにバックアップします。

注 – `nisbackup` コマンドは、必ずマスターサーバー上で実行してください。複製サーバー上では絶対に実行しないでください。

---

nisbackup コマンドは、バックアップコマンドが動作するように設定された時点の NIS+ 名前空間をコピーします。この記録には、現行のすべての NIS+ データと、認証されたネットワーク管理者が NIS+ 名前空間に入力した変更が含まれます。ただし、まだ NIS+ テーブルにチェックポイントされていない (NIS+ テーブルに記入されていない) ものは除きます。このバックアップ処理は、NIS+ データのチェックや訂正は行いません。テーブル内のデータが破壊されると、破壊されたデータは、有効なデータと見分けがつかない状態にバックアップされます。

nisbackup コマンドは、マシンがそのマスターサーバーであるディレクトリオブジェクトだけをバックアップします。つまり、nisbackup は、マスターサーバー上でだけ使用でき、複製サーバー上では使用できません。

- マシンが、ドメインとサブドメイン両方の NIS+ ディレクトリオブジェクトのマスターサーバーである場合、nisbackup をこのマシン上で実行するとドメインとサブドメイン両方のディレクトリオブジェクトをバックアップできます。
- ただし、マシンが、1 つのディレクトリオブジェクトのマスターサーバーで、さらに別のディレクトリオブジェクトの複製サーバーである場合、nisbackup を実行してそのマシンがマスターサーバーであるディレクトリオブジェクトをバックアップできますが、複製サーバーのオブジェクトはバックアップされません。

バックアップ処理が、他の処理に割り込まれたり、またはその処理を正常に終了できない場合は、処理を停止し、転送先ディレクトリ内に格納された以前のバックアップファイルをすべて復元します。

nisbackup の構文

nisbackup コマンドで使用する構文は、次のとおりです。

```
nisbackup [-v][-a] backupdir objects
```

引数の意味はそれぞれ以下のとおりです。

- *backupdir* には、バックアップファイルを格納する転送先ディレクトリが入ります。たとえば、/var/master1_bakup です。
- *objects* には、バックアップする NIS+ ディレクトリオブジェクトが入ります。たとえば、org_dir.doc.com です。ここには、複数の NIS+ ディレクトリオブジェクトをスペースで区切って入力できます。

nisbackup コマンドには、次のようなオプションを指定できます。

表 21–1 nisbackup コマンドのオプション

オプション	種類
-v	冗長モード。このモードは、追加情報を出力する

表 **21–1** nisbackup コマンドのオプション　(続き)

オプション	種類
-a	すべて。サーバーがマスターである NIS+ ディレクトリオブジェクトをすべてバックアップする。これには、このサーバーがマスターであるサブドメインのディレクトリオブジェクトも含まれる。ただし、他のマスターサーバーを持つサブドメインのディレクトリオブジェクトはバックアップされない

nisbackup コマンドは、バックアップする NIS+ ディレクトリオブジェクトのマスターサーバー上で実行する必要があります。

バックアップする NIS+ ディレクトリオブジェクトを指定する場合、そのディレクトリ名には完全指定名、または部分指定名を使用できます。

マルチレベルディレクトリをバックアップする場合、下位ディレクトリのバックアップファイルは、自動的にバックアップ転送先ディレクトリのサブディレクトリ内に配置されます。

nisbackup によるバックアップの対象

nisbackup を使用する場合、nisbackup はサーバー固有のコマンドであることに注意してください。-a オプションを使用するかどうかに関係なく、nisbackup は、このコマンドを実行中のサーバーがマスターサーバーであるディレクトリだけをバックアップします。他にマスターサーバーを持つ NIS+ ディレクトリオブジェクトは、バックアップされません。

たとえば、submaster1 サーバーは sales.doc.com. ディレクトリオブジェクトのマスターサーバーで、west.sales.doc.com. ディレクトリオブジェクトの複製サーバーでもあると想定します。 この場合、submaster1 上で nisbackup を実行すると、sales.doc.com. ディレクトリオブジェクトだけがバックアップされます。

このサーバー固有の原則には次のものがあります。

- 「全 NIS+ 名前空間」。複数のドメインのすべての名前空間に、NIS+ バックアップを実行する場合で、ルートマスターサーバーが全サブドメインのマスターサーバーでもある場合は、ルートマスター上で nisbackup に -a オプションを指定して実行します。ただし、スーパーユーザーのマスターサーバーが、すべてのサブドメインのマスターサーバーでない場合は、すべての名前空間のバックアップを完全に実施するために、nisbackup を他のマスターサーバー上でも実行する必要があります。
- 「サブドメイン」。1 つまたは複数のサブドメインの NIS+ バックアップを実行する場合、サブドメインのマスターサーバー上で nisbackup を実行する必要があります。ルートマスターなどの 1 つのマシンが、1 つまたは複数のサブドメインのマスターでもある場合、そのマシン上で nisbackup に -a オプションを指定して実行します。

- 「FNS ctx_dir」。FNS を実行している場合は、nisbackup を ctx_dir のマスターサーバー上で実行し、ctx_dir をバックアップするように指定するか、または -a オプションを使用すると、ctx_dir ディレクトリだけがバックアップされます。通常は、ctx_dir と NIS+ ディレクトリオブジェクトが別々のマスターサーバーから提供されていますが、この場合は、すべてのディレクトリをバックアップするには、nisbackup を両方のマシン上で実行する必要があります。

バックアップ転送先ディレクトリ

バックアップ転送先ディレクトリは、バックアップ対象のサーバーが使用できるものでなければなりませんが、サーバー上に物理的にマウントしていない転送先ディレクトリを使用するのも良い方法です。この場合、サーバーがダメージを受けても、バックアップディレクトリは使用可能です。

独立した転送先ディレクトリは、バックアップ対象となるマスターサーバーごとに使用する必要があります。混乱を避けるために、マスターサーバーのマシン名を転送先ディレクトリ内に組み込むと良いでしょう。たとえば、master1 マシン上で実行した nisbackup の転送先ディレクトリは、/var/master1_bakup という名前にします。

注意 – 指定した 1 つの転送先ディレクトリに対して、複数のサーバーをバックアップしないでください。異なるマスターサーバーには、必ず、異なる転送先ディレクトリを使用してください。それは、指定した転送先ディレクトリに、1 つまたは複数の NIS+ ディレクトリオブジェクトをバックアップするたびに、このディレクト内のこれらの NIS+ ディレクトリオブジェクト用のそれ以前のバックアップファイルが上書きされるからです。

NIS+ のバックアップを日付順に保存する

バックアップファイルを日付順に保存するには、少なくとも次の 2 つの方法があります。

- 「別々の転送先ディレクトリとして保存」。異なる転送先ディレクトリは、バックアップした日付ごとに保持できます。たとえば、/var/master1_bakup/July14、/var/master1_bakup/July15、など。この方法は簡単ですが、ディスク領域がかなり必要です。
- 「ファイルシステムのバックアップ」。NIS+ バックアップを日付順に保存する最も一般的な方法は、使用する通常ファイルシステムのバックアップメソッドに、バックアップ転送先ディレクトリを単に組み込むだけの方法です。これを簡単に行うには、nisbackup コマンドを crontab ファイルから実行するか、または Solstice バックアップルーチン内から実行します。nisbackup のようなコマンドを、システムのバックアッププロシージャとして自動的に実行するように指定する方法については、Solstice の説明書を参照してください。

特定の NIS ディレクトリをバックアップする

特定の NIS+ ディレクトリオブジェクトをバックアップするには、これらのディレクトリをバックアップ転送先ディレクトリの後ろに入力します。

たとえば、ルート、sales ドメイン、manf ドメインの 3 つの org_dir ディレクトリオブジェクトを /master1_backup ディレクトリにバックアップするには、nisbackup を master1 マシン上で次のように実行します。

```
master1# nisbackup /var/master1_bakup org_dir org_dir.sales org_dir.manf
```

すべての NIS+ 名前空間をバックアップする

すべての NIS+ 名前空間をバックアップする場合は、ルートマスターサーバー上で、nisbackup コマンドに -a オプションを指定して実行します。

-a オプションを使用する場合は、バックアップする NIS+ ディレクトリオブジェクトは指定しません。サーバー上とそのサーバーの下にあるサブドメインのすべての NIS+ ディレクトリオブジェクトは、自動的にバックアップされます。

たとえば、doc.com. 名前空間を /master1_bakup ディレクトリにバックアップするには、ルートマスター上で、nisbackup を次のように実行してください。

```
rootmaster# nisbackup -a /var/master1_bakup
```

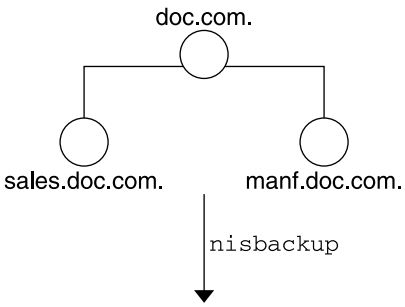
バックアップディレクトリの構造

ドメイン上でバックアップを実行すると、バックアップ転送先ディレクトリ内に、NIS+ ディレクトリオブジェクトごとにサブディレクトリが作成されます。これらのサブディレクトリ名は、完全指定の NIS+ ディレクトリオブジェクト名の末尾にピリオドが付いたものになります。

-a オプションを使用してすべての NIS+ オブジェクトをフルバックアップした場合は、3 つの関連ディレクトリオブジェクト (*domain.*、org_dir.*domain.*、および groups_dir.*domain.*) がすべてバックアップされ、3 つのサブディレクトリが作成されます。複数のオブジェクトをバックアップすると、サブディレクトリはバックアップしたそれぞれのオブジェクトごとに作成されます。

複数の NIS+ ディレクトリオブジェクトのバックアップサブディレクトリは、それがサブドメインであるかどうかに関係なく、親バックアップ転送先ディレクトリのサブディレクトリになるので注意してください。つまり、nisbackup は、親バックアップ転送先ディレクトリの下にドメインの階層を複製しません。その代わりに、バックアップサブディレクトリはすべて、転送先ディレクトリの単純なサブディレクトリになります。

たとえば、doc.com. のルート、sales、manf からディレクトリオブジェクトを /var/master1_bakup ディレクトリにバックアップする場合、図 21–1 に示すように、/var/master1_bakup ディレクトリ内には 9 個のサブディレクトリが作成されます。




```
/var/master1_bakup
     /master1_bakup/doc.com./
     /master1_bakup/groups_dir.doc.com./
     /master1_bakup/org_dir.doc.com.
     /master1_bakup/sales.doc.com./
     /master1_bakup/groups_dir.sales.doc.com./
     /master1_bakup/org_dir.sales.doc.com./
     /master1_bakup/manf.doc.com./
     /master1_bakup/groups_dir.manf.doc.com./
     /master1_bakup/org_dir.manf.doc.com./
```

図 **21–1** nisbackup によって作成されたディレクトリの例

バックアップファイル

バックアップ転送先ディレクトリには、この転送先ディレクトリにバックアップされた最新の NIS+ ディレクトリオブジェクトを表示する backup_list ファイルが入っています。

各サブディレクトリには、ファイルが 2 つと /data サブディレクトリが 1 つ組み込まれます。この 3 つのファイルを次に示します。

- data.dict。このディレクトリにバックアップされた NIS+ ディレクトリオブジェクトの NIS+ データ辞書の入った XDR コード化ファイル
- last.upd。このディレクトリにバックアップされた NIS+ ディレクトリオブジェクトに関する時刻スタンプ情報が入ったバイナリファイル

各 /data サブディレクトリには、1 つまたは複数の以下のファイルが入っています。

- root.object。NIS+ ルートディレクトリオブジェクトの説明の入った XDR コード化ファイル。たとえば、/master1_bakup/doc.com/data/root.object

- root_dir。これらのオブジェクトのルートディレクトリとサーバー情報内に組み込まれた NIS+ オブジェクトの説明が入った XDR コード化ファイル。たとえば、/master1_bakup/doc.com/data/root_dir
- *table.directory*。バックアップの実行時に、NIS+ テーブル内に表示されていたデータと関連するすべての NIS+ ログファイル内に含まれるデータがすべて入った XDR コード化ファイル。バックアップされた NIS+ ディレクトリオブジェクト内に NIS+ テーブルがある場合、対応する table.directory バックアップファイルが、そのディレクトリオブジェクトの /data サブディレクトリ内に作成される。

  たとえば、それぞれの NIS+ org_dir ディレクトリには、hosts テーブルが含まれるため、各 *target*/org_dir.*domain*/data サブディレクトリには、hosts.org_dir があります。たとえば、/master1_bakup/org_dir.doc.com./data/hosts.org_dir です。

  指定したディレクトリオブジェクト内に表示されたユーザー作成の NIS+ テーブルは、標準 NIS+ テーブルとして同じ方法でバックアップされます。
- groups_dir。NIS+ グループ情報の入った XDR コード化ファイル。このファイルは、対応する NIS+ groups_dir 転送先ディレクトリに格納される。

nisrestore を使用して NIS+ 名前空間を復元する

nisrestore コマンドによって、nisbackup を使用して作成したバックアップファイル内に格納されたデータと一致する NIS+ ディレクトリオブジェクトが再現されます。このコマンドを使用すると、NIS+ サーバーの復元、壊れたディレクトリオブジェクトの置換、または新しい NIS+ サーバーに NIS+ データを読み込めます。

nisrestore を実行するための前提条件

nisrestore を使用するには、nisrestore から NIS+ データを受け取るマシンは、NIS+ サーバーとして設定されている必要があります (NIS+ サーバーの設定の詳細については、第 4 章を参照)。つまり、次の設定が行われていなければなりません。

- マシンを、NIS+ クライアントとして初期化しておく必要があります。
- マシンを NIS 互換モードで実行し、ドメイン名システム (DNS) 転送をサポートする場合は、そのマシンには、適切に構成された /etc/resolv.conf ファイルが必要です。
- 名前空間内で他のサーバーが実行中の時、あるサーバーで nisrestore を使用する場合は、nisrestore は、他のサーバーを検証して、このサーバーがサーバーに復元するバックアップ NIS+ オブジェクトを配布するように構成されているかどうかを確認します。実行中のサーバーが他にない場合は、nisrestore に -f オプ

ションを指定して実行する必要があります。つまり、nisrestore がチェックする他のサーバーがある場合は、-f オプションを使用する必要はありません。また、使用できるサーバーが他にない場合、たとえば、1 台のマスターサーバーを復元するときに、機能複製サーバーが他にない場合は、-f オプションを使用する必要があります。

注意 – 上記の 3 つの前提条件への追加条件として、マシン上で rpc.nisd デーモンを実行しないでください。rpc.nisd デーモンを実行する場合は、rpc.nisd を消去してから nisrestore を実行してください。

nisrestore の構文

nisrestore コマンドでは、次の構文を使用します。

```
nisrestore [-fv][-a][-t] backupdir [directory_objects]
```

引数の意味はそれぞれ以下のとおりです。

- *backupdir* には、NIS+ オブジェクトの復元に使用するバックアップファイルの入ったディレクトリを入力します (/var/master1_bakup)。
- *directory_objects* には、復元する NIS+ ディレクトリオブジェクトを入力します (org_dir.doc.com)。複数の NIS+ ディレクトリオブジェクトを、スペースで区切って入れることができます。nisrestore に -a オプションを指定して実行する場合は、特定のディレクトリオブジェクトは指定しません。

nisrestore コマンドには、以下のオプションを指定できます。

表 **21–2** nisbackup コマンドのオプション

オプション	種類
-a	すべて。バックアップディレクトリ内に入っている NIS+ ディレクトリオブジェクトをすべて復元する
-f	サーバーが、ディレクトリオブジェクトのサーバーリストに記載されているかどうかを検査せずに、強制的に復元を行う。このオプションは、ルートマスターサーバーを復元する時、または「オブジェクトを検出できません」といった種類のエラーを受け取った場合に使用する必要がある
-v	冗長モード。このモードは、追加情報を出力する
-t	このオプションを使用すると、バックアップディレクトリ内に格納された NIS+ ディレクトリオブジェクトがすべて表示される。オブジェクトの復元は行われない

nisrestore を使用する

NIS+ バックアップファイルから NIS+ データを復元するには、nisrestore コマンドを使用します。

たとえば、org_dir.doc.com. ディレクトリオブジェクトを replica1 サーバーに復元する場合は、スーパーユーザーになって replica1 にログインします。で説明した前提条件が満たされていることを確認してから、以下のように nisrestore を実行します。

```
replica1# nisrestore /var/master1_bakup org_dir.doc.com.
```

nisrestore には、以下の項目が適用されます。

- 「損傷した名前空間」。損傷した、または破壊された NIS+ 名前空間を復元するには、復元する NIS+ ディレクトリオブジェクトのすべてのサーバー上で nisrestore コマンドを実行する必要があります。
- 「検出エラー」。nisrestore が必要なデータを確認できないか、または検出できないというエラーメッセージを受け取った場合は、-f オプションを使用する必要があります。

  たとえば、master1 という名前のルートマスターサーバー上に NIS+ データをロードし直す場合は、次のように入力します。

```
master1# nisrestore -f -a /var/master1_bakup
```

- 「ディレクトリ名」。復元する NIS+ ディレクトリオブジェクトを指定する場合は、完全指定または部分指定ディレクトリ名を使用します。

バックアップと復元を使用して複製サーバーを設定する

NIS+ バックアップおよび復元機能を使用すると、NIS+ データを新しい複製サーバーに速く読み込むことができます。名前空間が広い場合は、この方法の方が、nisping を使用するよりもマスターサーバーからのデータを非常に速く入手できます。

nisbackup と nisrestore を使用して新しい複製サーバーを設定するには、次の手順を行います。

1. マスター上で nisserver を実行し、新しい複製サーバーを作成します。
2. 複製サーバー上の rpc.nisd を終了させます。

   この処理は、nisping コマンドを使用した名前空間データのマスターから複製への自動転送に割り込んで実行されます。
3. マスターサーバー上で、nisbackup を実行します。

4. 新しい複製サーバー上で nisrestore を実行し、**NIS+** データを読み込みます。

5. 新しい複製サーバー上で rpc.nisd を再実行します。

サーバーマシンを置換する

nisbackup と nisrestore を使用すると、サーバーとして使用中のマシンと別のマシンをすぐに置換できます。たとえば、旧サーバーを新しい高速のサーバーと交換すると、ネットワークのパフォーマンスを向上させることができます。

マシンを置換する場合の必要条件

NIS+ サーバーとして使用中のマシンを他のマシンに置き換える場合には、次の条件が必要です。

- 新しいマシンには、置換する旧マシンと同じ IP アドレスを割り当てます。
- 新しいマシンには、置換する旧マシンと同じマシン名を割り当てます。
- 新しいマシンは、置換する旧マシンと同じサブネットに接続します。

サーバーマシンの置換方法

サーバーマシンを置換する場合は、次の手順に従ってください。

1. 旧サーバーが管理するドメインのマスターサーバー上で nisbackup を実行します。
   詳細は、409 ページの「すべての NIS+ 名前空間をバックアップする」を参照してください。置換する旧サーバーがマスターサーバーである場合もあるので注意してください。この場合は、この旧マスターサーバー上で nisbackup を実行します。

2. 旧サーバーの /var/nis/NIS_COLD_START ファイルをバックアップディレクトリにコピーします。

3. 旧サーバーの /etc/.rootkey ファイルをバックアップディレクトリにコピーします。

4. 旧サーバーをネットワークから切り離します。

5. 新しいサーバーをネットワークに接続します。

6. 新しいサーバーに旧サーバーと同じ **IP** アドレス **(**番号**)** を割り当てます。

7. 新しいサーバーに旧サーバーと同じマシン名を割り当てます。

8. 必要な場合は、新しいサーバー上で rpc.nisd を消去します。

9. 新しいサーバー上で nisrestore を実行し、**NIS+** データを読み込みます。

   詳細は、411 ページの「nisrestore を使用して NIS+ 名前空間を復元する」を参照してください。

10. .rootkey ファイルを、バックアップディレクトリから新しいサーバーの /etc にコピーします。

11. NIS_COLD_START ファイルを、バックアップディレクトリから新しいサーバーの /var/nis にコピーします。

12. 新しいサーバーを再起動します。

第 22 章

NIS+ の削除

この章では、NIS+ ディレクトリ管理コマンドを使用して、クライアントまたはサーバーから NIS+ を削除する方法と、NIS+ 名前空間全体を削除する方法について説明します。

NIS+ 複製サーバーを、ディレクトリから分離し、そのドメインの複製サーバーとして機能しないようにする場合は、340 ページの「nisrmdir コマンド」を参照してください。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。詳細については、`http://www.sun.com/directory/nisplus/transition.html` を参照してください。

クライアントマシンから NIS+ を削除する

この節では、クライアントマシンから NIS+ を削除する方法について説明します。ただし、クライアントマシンから NIS+ を削除しても、ネットワークから NIS+ ネームサービスを削除したことにはならないので注意してください。ネットワークから NIS+ ネームサービスを削除して、NIS または `/etc` ディレクトリのファイルをネームサービスとして使用する状態に戻す場合は、420 ページの「NIS+ 名前空間を削除する」を参照してください。

nisclient を使用してインストールした NIS+ を削除する

第 4 章の説明に沿って nisclient - i スクリプトに使用して、NIS+ クライアントとして設定したクライアントマシンから NIS+ を削除するには、 nisclient を -r オプションで実行します。

```
client# nisclient -r
```

nisclient -r では、nisclient -i 1 回分の処理が取り消されます。つまり、nisclient -i 実行以前にクライアントによって使用されていたネーミングシステム (NIS や /etc ディレクトリのファイルなど) が再び使用されるようになります。

NIS+ コマンドでインストールした NIS+ を削除する

第 4 章の説明に沿って nisaddcred、domainname、および nisinit コマンドを使用して、NIS+ クライアントとして設定されたクライアントマシンから NIS+ を削除するには、 次の手順を行います。

1. ファイル .rootkey を削除します。

```
client# rm -f /etc/.rootkey
```

2. keyserv、nis_cachemgr、nscd のプロセス **ID** を確認して、終了します。

```
client# ps -ef | grep keyserv
    root 714 1 67 16:34:44 ? keyserv
client# kill -9 714
client# ps -ef | grep nis_cachemgr
    root 123 1 67 16:34:44 ? nis_cachemgr
client# kill -9 123
client# ps -ef | grep nscd
    root 707 1 67 16:34:44 ? nscd
client# kill -9 707
```

3. /var/nis ディレクトリとその下のファイルを削除します。

```
clientmachine# rm -rf /var/nis/*
```

サーバーから NIS+ を削除する

この節では、NIS+ サーバーから NIS+ を削除する方法を示します。

ただし、サーバーから NIS+ を削除しても、ネットワークから NIS+ ネームサービスを削除したことにはならないので注意してください。ネットワークから NIS+ ネームサービスを削除して、NIS または /etc ディレクトリのファイルをネームサービスとして使用する状態に戻す場合は、420 ページの「NIS+ 名前空間を削除する」を参照してください。

注 – NIS+ サーバーとして使用しているマシンを、別のマシンに置換できます。414 ページの「サーバーマシンを置換する」を参照してください。

サーバーから NIS+ を削除する手順は以下のとおりです。

1. クライアントから **NIS+** を削除する作業を行います。

   NIS+ サーバーも NIS+ クライアントの一種なので、まずクライアントに関連する部分を削除する必要があります。このためには `nisclient -r` (418 ページの「`nisclient` を使用してインストールした NIS+ を削除する」を参照) か、NIS+ コマンド (418 ページの「NIS+ コマンドでインストールした NIS+ を削除する」を参照) を使用します。

2. サーバーの `groups_dir` ディレクトリと `org_dir` ディレクトリを削除する。

   ```
   server# nisrmdir -f groups_dir. domainname
   server# nisrmdir -f org_dir. domainname
   ```

3. `keyserv`、`rpc.nisd`、`nis_cachemgr`、`nscd` のプロセス **ID** を確認し、終了します。

   ```
   server# ps -ef | grep rpc.nisd
       root 137 1 67 16:34:44 ? rpc.nisd
   server# kill -9 137
   server# ps -ef | grep keyserv
       root 714 1 67 16:34:44 ? keyserv
   server# kill -9 714
   server# ps -ef | grep nis_cachemgr
       root 123 1 67 16:34:44 ? nis_cachemgr
   server# kill -9 123
   server# ps -ef | grep nscd
       root 707 1 67 16:34:44 ? nscd
   server# kill -9 707
   ```

4. `/var/nis` ディレクトリとその下のファイルを削除します。

   ```
   rootmaster# rm -rf /var/nis/*
   ```

NIS+ 名前空間を削除する

NIS+ 名前空間を削除し、NIS または /etc ディレクトリのファイルをネームサービスとして使用する状態に戻す手順は以下のとおりです。

1. ルートマスターから .rootkey ファイルを削除します。

```
rootmaster# rm -f /etc/.rootkey
```

2. ルートマスターのルートドメインから groups_dir サブディレクトリと org_dir サブディレクトリを削除します。

```
rootmaster# nisrmdir -f groups_dir.domainname
rootmaster# nisrmdir -f org_dir.domainname
```

domainname には、ルートドメイン名 (doc.com など) が入ります。

3. ルートドメインを削除します。

```
rootmaster# nisrmdir -f domainname
```

domainname には、ルートドメイン名 (doc.com など) が入ります。

4. keyserv、rpc.nisd、nis_cachemgr、nscd のプロセス **ID** を確認し、終了します。

```
rootmaster# ps -ef | grep rpc.nisd
    root 137 1 67 16:34:44 ? rpc.nisd
rootmaster# kill -9 137
rootmaster# ps -ef | grep keyserv
    root 714 1 67 16:34:44 ? keyserv
rootmaster# kill -9 714
rootmaster# ps -ef | grep nis_cachemgr
    root 123 1 67 16:34:44 ? nis_cachemgr
rootmaster# kill -9 123
rootmaster# ps -ef | grep nscd
    root 707 1 67 16:34:44 ? nscd
rootmaster# kill -9 707
```

5. 新しいドメインを作成します

```
rootmaster# domainname name
```

name には、新しいドメイン名 (NIS+ インストール前のドメイン名など) が入ります。

6. 既存の /etc/defaultdomain ファイルを削除します。

```
rootmaster# rm /etc/defaultdomain
```

7. /etc/defaultdomain ファイルを、新しいドメイン名を使用して作成し直します。

```
rootmaster# domainname > /etc/defaultdomain
```

8. nsswitch.conf ファイルを元のファイルに戻します。

サーバーを nisserver -r を使用して設定した場合は、以下のコマンドを使用します。

```
rootmaster# cp /etc/nsswitch.conf.no_nisplus /etc/nsswitch.conf
```

また、デフォルトスイッチテンプレートファイルの 1 つをコピーする方法もあります。NIS スイッチのデフォルトファイルテンプレートを使用する場合は、以下のコマンドを入力します。

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

/etc ファイルのデフォルトスイッチファイルテンプレートを使用する場合は、以下のコマンドを入力します。

```
rootmaster# cp /etc/nsswitch.files etc/nsswitch.conf
```

9. keyserv プロセスを再起動します。

```
rootmaster# keyserv
```

10. /var/nis ディレクトリとその下のファイルを削除します。

```
rootmaster# rm -rf /var/nis/*
```

11. この状態で、別のネームサービス **(NIS** または /etc ファイル**)** を再起動できます。

第 23 章

NIS+ テーブルの情報

この章では、Solaris オペレーティング環境で提供されているデフォルトの NIS+ テーブルについて概要を説明します。対応するマニュアルページも参照してください。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ テーブル

NIS+ 環境では、ほとんどの名前空間の情報は、NIS+ テーブルに格納されます。

ネームサービスがないと、ほとんどのネットワーク情報は /etc ファイルに格納され、ほとんどすべての NIS+ テーブルが対応する /etc ファイルを持ちます。NIS サービスでは、/etc ファイルにほとんど対応する NIS マップにネットワーク情報を格納しました。

注 – この章では、NIS+ の一部として配布されたものだけを説明します。ユーザーとアプリケーション開発者は、目的に応じて NIS+ との互換性があるテーブルを頻繁に作成します。ユーザーと開発者によって作成されたテーブルの詳細については、マニュアルを参照してください。

すべての NIS+ テーブルは、groups_dir ディレクトリオブジェクトに格納される admin とグループテーブルを除いて、ドメインの org_dir NIS+ ディレクトリオブジェクトに格納されます。

注 – テーブルエントリはリンクしないでください。テーブルは他のテーブルにリンクされますが、あるテーブルのエントリを別のテーブルのエントリにリンクしないでください。

NIS+ テーブルと他のネームサービス

Solaris の環境では、ネームサービスのスイッチファイル (nsswitch.conf) によって、1 つ以上のソースを異なる名前空間の情報に対して指定できます。NIS+ テーブルの他に、ソースは NIS マップ、DNS ゾーンファイル、/etc テーブルにすることができます。これらをスイッチファイルで指定する順番によって、異なるソースから情報が組み合わされる方法が決まります。スイッチファイルの詳細については 第 1 章を参照してください。

NIS+ テーブル入力ファイルフォーマット

これらテーブルのどれかに対して入力ファイルを作成している場合、ほとんどのテーブルは、次の 2 つのフォーマットの条件を共有します。

- エントリごとに 1 行を使う
- 1 つ以上のスペースまたはタブで列を分ける

特定のテーブルに、異なるまたは追加のフォーマットの条件がある場合には、「入力ファイルフォーマット」の項で説明します。

auto_home テーブル

auto_home テーブルは間接オートマウンタマップです。このマップによって NIS+ クライアントは、ドメイン内の任意のユーザーのホームディレクトリをマウントできます。このテーブルは、各ユーザーのホームディレクトリのマウントポイント、各ホームディレクトリの位置と、マウントオプションがあればそれを指定してマウントします。これは間接マップであるため、マウントポイントの最初の部分は auto_master テーブルに指定され、デフォルトでは /home となります。マウントポイントの 2 番目の部分 (つまり、/home の下にあるサブディレクトリ) は auto_home マップ内のエントリによって指定され、ユーザーごとに異なります。

auto_home テーブルには 2 つの列があります。

表 23–1 auto_home テーブル

列	コメント	説明
キー	マウント先	ドメイン内のユーザーのログイン名
値	オプションと位置	ユーザーごとのマウントオプションがあればそれと、ユーザーのホームディレクトリの位置

たとえば:

```
costas barcelona:/export/partition2/costas
```

ユーザー costas のホームディレクトリは、サーバー barcelona 上にあり、ディレクトリ /export/partition2/costas 内にあります。またこのホームディレクトリは、クライアントの /home/costas ディレクトリの下にマウントされます。このエントリにはマウントオプションはありません。

auto_master テーブル

auto_master テーブルは、ドメイン内のすべてのオートマウンタマップを含みます。直接マップの場合、auto_master テーブルは、単にマップ名を提供するだけです。間接マップの場合、このテーブルは、そのマウントポイントの先頭ディレクトリとマップ名の両方を提供します。auto_master テーブルには次の 2 つの列があります。

表 **23–2** auto_master テーブル

列	コメント	説明
キー	マウント先	マップがマウントされる先頭ディレクトリ。マップが直接マップの場合、これは /– で表されるダミーディレクトリである
値	マップ名	オートマウンタマップの名前

たとえば、auto_master テーブルに次のエントリがあるとします。

```
/home auto_home
/-auto_man
/programs auto_programs
```

最初のエントリは auto_home マップを指定します。このエントリによって、auto_home マップのすべてのエントリに対して、マウント先の最上位ディレクトリ /home を指定します。auto_home マップは、間接マップです。2 番目のエントリには、auto_man マップを指定します。このマップは直接マップであるため、このエントリはマップ名だけを与えます。auto_man マップはそれ自体で、その各エントリに対するマウントポイントの完全パス名だけではなく、最上位ディレクトリも提供します。3 番目のエントリは auto_programs マップを指定します。このエントリはマウントポイントの先頭ディレクトリを与えるため、auto_programs マップは間接マップです。

すべてのオートマウンタマップは NIS+ テーブルとして格納されます。デフォルトでは、Solaris 環境は必須の auto_master マップ、および非常に便利な auto_home マップを提供します。

1 つのドメインに対してさらに多くのオートマウンタマップを作成できますが、これらは必ず NIS+ テーブルとして格納し、auto_master テーブルに登録してください。その他のオートマウントマップを作成して、auto_master (ユーザーに対して作成された) に追加するときは、列名は「キー」と「値」にする必要があります。オートマウンタの詳細については、オートマウンタ、または NFS ファイルシステムの関連マニュアルを参照してください。

bootparams テーブル

bootparams テーブルは、ドメイン内のすべてのディスクレスマシンに関する構成情報をもっています。ディスクレスマシンとは、ネットワークに接続されているが、ハードディスクのないマシンのことです。ディスクレスマシンにはディスク装置がないため、自分のファイルとプログラムをネットワーク上のサーバーのファイルシステムに持っています。また、自分の構成情報 (つまり「ブートパラメタ」) をサーバーに持っています。

このような構成になっているため、すべてのディスクレスマシンには、この情報がどこに格納されているかを認識する初期設定プログラムがあります。ネットワークにネームサービスがない場合、このプログラムはサーバーの /etc/bootparams ファイル内でこの情報を探します。ネットワークが NIS+ ネームサービスを使う場合、このプログラムは、代わりに bootparams テーブル内でこの情報を探します。

bootparams テーブルは、ディスクレスマシンに関するどんな構成情報でも格納できます。このテーブルには 2 つの列があり、構成キーとその値を格納します。デフォルトでは、このテーブルは各マシンのルート、スワップ、およびダンプの各パーティションの位置を格納するように設定されます。

デフォルトの bootparams テーブルには 2 つの列しかありませんが、列を使用して、次に示す情報を提供します。

表 23–3 bootparams テーブル

列	コメント	説明
キー	ホスト名	ディスクレスマシンの正式ホスト名、hosts テーブルで指定
値	構成	ルートパーティション :マシンのルートパーティションの位置 (サーバー名とパス)
		スワップパーティション :マシンのスワップパーティションの位置 (サーバー名とパス)
		ダンプパーティション :マシンのダンプパーティションの位置 (サーバー名とパス)
		インストールパーティション
		ドメイン

「入力ファイルのフォーマット」

列はタブ文字で区切ります。バックスラッシュ (\) は、1 つのエントリを複数の行で指定するのに使用します。ルート、スワップ、およびダンプの各パーティション用のエントリのフォーマットを次に示します。

client-name root=*server:path* \
swap=*server:path* \
dump=*server:path* \
install=*server:path* \
domain=*domainname*

次に例を示します。

```
buckaroooroot=bigriver:/export/root1/buckaroo \
 swap=bigriver:/export/swap1/buckaroo \
 dump=bigriver:/export/dump/buckaroo \
 install=bigriver:/export/install/buckaroo \
 domain=sales.doc.com
```

x86 ベースのマシンでは、使用できるパラメータが増えます。詳細については、bootparams(4) のマニュアルページを参照してください。

client_info テーブル

client_info テーブルは、それが存在するドメインのサーバー設定を格納するために使われるオプションの内部 NIS+ テーブルです。このテーブルは、nisprefadm コマンドで作成および保管されます。

注意 – nisprefadm だけを使って、このテーブルで作業してください。このテーブルでは、他の NIS+ コマンドは使わないでください。

cred テーブル

NIS+ 主体の資格に関する情報を格納するテーブルです。ドメインにつき 1 つ存在し、ドメインに属するクライアントマシンおよびマシンにログインできるクライアントユーザー (つまり、ドメイン中の主体) の資格に関する情報を持っています。cred テーブルはドメイン中の org_dir サブディレクトリにあります。

注 – cred テーブルはリンクしないでください。org_dir ディレクトリにはそれぞれ固有の cred テーブルがあり、他の org_dir ディレクトリの cred テーブルへのリンクを使用できません。

cred テーブルには以下の 5 つの列があります。

表 **23–4** cred テーブル

			公開データ	非公開データ
NIS+ 主体名	認証タイプ	認証名		
主体ユーザー名	LOCAL	ユーザー ID	グループ ID リスト	

表 **23–4** cred テーブル　(続き)

NIS+ 主体名	認証タイプ	認証名	公開データ	非公開データ
主体ユーザー名またはマシン名	DES	Secure RPC ネット名	公開鍵	暗号化非公開鍵

2 番目の「認証タイプ」という列の値により、他の 4 つの列の値のタイプが決定されます。

- 「LOCAL」。認証タイプが LOCAL の場合、残りの列には主体ユーザー名、UID、GID が入ります (最後の列は空白)。
- 「DES」。認証タイプが DES の場合、残りの列には主体名、Secure RPC ネット名、公開鍵、暗号化非公開鍵が入ります。これらの鍵は他の情報と組み合わされ、DES 資格の暗号化および復号化に使用されます。

資格および cred テーブルの詳細については、第 12 章を参照してください。

ethers テーブル

ethers テーブルは、インターネット上のマシンの 48 ビット Ethernet アドレスに関する情報を格納しています。次の 3 列があります。

表 **23–5** ethers テーブル

列	コメント	説明
アドレス	Ethernet アドレス	マシンの 48 ビット Ethernet アドレス
名前	公式なホスト名	マシンの名前、hosts テーブルで指定
コメント	コメント	エントリに関するコメント (必要に応じて)

Ethernet アドレスの形式は次のとおりです。

n:*n* :*n*:*n* :*n*: *n hostname*

ここで *n* は 0 ～ FF の 16 進数で、1 バイトを表します。このアドレスのバイト列は常に最上位のバイトから始まるネットワーク順になっています。

group テーブル

group テーブルは、UNIX ユーザーグループについての情報を格納します。group テーブルには 4 つの列があります。

表 **23–6** group テーブル

列	説明
名前	グループ名
パスワード	グループのパスワード
グループ ID	グループの数値 ID
メンバー	グループメンバー名、コンマで区切る

Solaris の以前のリリースでは、ローカル /etc/group ファイルで +/- 構文を使用して、NIS グループマップ内のエントリを結合または上書きしました。Solaris 環境ではネームサービススイッチファイルを使用してマシンの情報ソースを指定するため、この仕様は不要になります。Solaris 2.x システムで必要なのは、クライアントの /etc/nsswitch.conf ファイルを編集して files を指定し、そのあとグループ情報のソースとして nisplus を指定することです。これによって、group テーブルの内容がクライアントの /etc/group ファイルの内容に効果的に追加されます。

hosts テーブル

hosts テーブルは、ドメイン内の全マシン名とそれらの IP アドレスを関連付けます。マシンは通常、 NIS+ クライアントでもありますが、その必要はありません。bootparams、group、および netgroup などのほかのテーブルは、このテーブルに収められたネットワーク名に依存しています。これらのテーブルは、ネットワーク名を使用して、ホームディレクトリやグループメンバーなどのほかの属性を個々のマシンに割り当てます。hosts テーブルには 4 つの列があります。

表 **23–7** hosts テーブル

列	説明
アドレス	マシンの IP アドレス (ネットワーク番号とマシン ID 番号)
正式名	マシンの正式名

表 23–7 hosts テーブル　(続き)

名前	マシンを識別するために、ホスト名の代わりに使われる任意の名前
コメント	エントリに関するコメント (必要に応じて)

mail_aliases テーブル

mail_aliasesテーブルは、sendmail によって認識されるドメインのメール別名を含んでいます。これには 4 つの列があります。

表 23–8 mail_aliases テーブル

列	説明
別名	別名の名前
メンバーリスト	この別名に送信されたメールを受信するメンバーが収められているリスト。メンバーはユーザー、マシン、またはほかの別名とすることができる
コメント	エントリに関するコメント (必要に応じて)
オプション	(マニュアルページを参照)

「入力ファイルのフォーマット」

各エントリのフォーマットは次のとおりです。

alias-name:*member*[,*member*]...

1 つのエントリが複数の行にまたがるときには、バックスラッシュを使用します。

netgroup テーブル

netgroup テーブルは、リモートマウント、ログインとシェルのアクセス権をチェックするために使用される、ネットワーク全域にわたるグループを定義します。リモートマウントに使用されるネットグループのメンバーは、マシンです。リモートログインとシェルの場合、これらのメンバーはユーザーです。

注 – サービスを提供している NIS+ サーバーが互換モードで動作している場合、クライアントマシンのユーザーは `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」という結果が出ます。

`netgroup` テーブルには 6 つの列があります。

表 23–9 `netgroup` テーブル

列	コメント	説明
名前	グループ名	ネットワークグループ名
グループ	グループ名	このグループを構成するグループ名
ホスト	ホスト名	ホスト名
ユーザー	ユーザー名	ユーザーのログイン名
ドメイン	ドメイン名	ドメイン名
コメント	コメント	エントリに関するコメント (必要に応じて)

「入力ファイルのフォーマット」

入力ファイルは、1 つのグループ名と任意の数のメンバーから構成されます。

groupname member-list . . .

メンバー指定は、他のネットグループ名、または 3 つのフィールドを順序正しく並べたものとすることができます。ネットグループ名、3 つのフィールドを両方指定することもできます。

member-list : : =*groupname* | (*hostname*, *username*, *domainname*)

最初のフィールドには、グループに属するマシン名を指定します。2 番目のフィールドに、グループに属するユーザー名を指定します。3 番目のフィールドには、メンバー指定が有効となっているドメインを指定します。

指定のないフィールドは、ワイルドカードを示します。たとえば、次の `netgroup` には、すべてのドメインのユーザーとすべてのマシンが含まれます。

```
everybody ( , , )
```

フィールド内のダッシュはワイルドカードの反対で、このグループに所属するマシンやユーザーがないことを示します。次に 2 つの例を示します。

```
(host1, -,doc.com.) (-,joe,doc.com.)
```

最初の指定では、1 台のマシン host1 を doc.com. ドメインに入れますが、すべてのユーザーを排除します。2 番目の指定では、1 人のユーザーを doc.com. ドメインに入れますが、すべてのマシンを排除します。

netmasks テーブル

netmasks テーブルには、標準のインターネットサブネットに使われるネットワークマスクが収められています。このテーブルには 3 つの列があります。

表 **23–10** netmasks テーブル

列	説明
アドレス	ネットワークの IP 番号
マスク	ネットワーク上で使うネットワークマスク
コメント	エントリに関するコメント (必要に応じて)

ネットワーク番号には、マシンアドレスで使用される従来の IP ドット表記を使用できますが、マシンアドレスの代わりにゼロを残します。たとえば、

```
128.32.0.0 255.255.255.0
```

というエントリでは、クラス B ネットワーク 128.32.0.0 が、サブネットフィールドに 24 ビット、ホストフィールドに 8 ビットを持つことを意味します。

networks テーブル

networks テーブルにはインターネットのネットワークが含まれます。このテーブルは、Network Information Control Center (NIC) で管理される公式のネットワークテーブルから作成されるのが普通ですが、これにローカルネットワークを追加しなければならないこともあります。これには 4 つの列があります。

表 **23–11** networks テーブル

列	説明
正式名	ネットワークの正式名、インターネットが提供

表 **23–11** networks テーブル　(続き)

列	説明
アドレス	ネットワークの公式 IP 番号
名前	ネットワークの非公式名
コメント	エントリに関するコメント (必要に応じて)

passwd テーブル

passwd テーブルには、ドメイン内のユーザーのアカウントに関する情報が入っています。これらのユーザーは、一般的には NIS+ 主体ですが、その必要はありません。ただし、ユーザーが NIS+ 主体である場合、それらの資格はここには収められず、ドメインの cred テーブルに格納されることに注意してください。passwd テーブルは、その他 (または未認証) に対して通常読み取り権を与えます。

注 – スーパーユーザー (ユーザー ID = 0) のエントリをこのテーブルに入れることはできません。root のパスワード情報は、/etc ディレクトリ上のファイルに格納してください。

---

passwd テーブル内の情報は、ユーザーのアカウントが作成されたときに追加されます。

passwd テーブルには次の列があります。

表 **23–12** passwd テーブル

列	説明
名前	ユーザーのログイン名であり、ユーザーのアカウントが作成されたときに割り当てられる。この名前は大文字を含む必要はない。最大 8 文字まで
パスワード	ユーザーの暗号化されたパスワード
ユーザー ID	ユーザーの ID 番号であり、ユーザーのアカウントが作成されたときに割り当てられる
グループ ID	ユーザーのデフォルトグループの ID 番号
GCOS	ユーザーの実名と、ユーザーがメールメッセージ見出しの「From:」フィールドに入れたい情報。この列が「&」の場合、単にユーザーのログイン名を使用する

表 **23–12** passwd テーブル　(続き)

列	説明
ホーム	ユーザーのホームディレクトリのパス名
シェル	ユーザーの初期シェルプログラム。デフォルトは B シェル :/usr/bin/sh
シャドウ	表 23–13 を参照してください。

passwd テーブルには、さらにシャドウ列があります。この列には、ユーザーアカウントに関して、次に示すような制限情報が格納されています。

表 **23–13** passwd テーブルのシャドウ列

項目	説明
最終変更日	1970 年 1 月 1 日からパスワードの最終変更日までの日数
最小値	推奨されるパスワード変更間隔の最小日数
最大値	パスワードが有効な最大日数
警告	ユーザーパスワードが期限切れになる前にユーザーが警告を受ける日数
間隔	ユーザーに許される休止日数
期限	ユーザーのアカウントが無効となる絶対日付
フラグ	将来のために獲保。現在は 0 に設定されている

Solaris の以前のリリースでは、ローカル /etc/passwd ファイル内で +/- 構文を使って、NIS パスワードマップ内のエントリを統合または上書きしました。Solaris 2.x 環境ではネームサービススイッチを使ってマシンの情報ソースを指定するため、この仕様は不要になります。Solaris 2.x システムで必要なのは、クライアントの /etc/nsswitch.conf ファイルを編集して files を指定し、それに続けて passwd 情報のソースとして nisplus を指定することです。これによって、passwd テーブルの内容が /etc/passwd ファイルの内容に効果的に追加されます。

しかし、それでも +/- 方式を使用したい場合は、クライアントの nsswitch.conf ファイルを編集します。NIS を使用しているなら passwd ソースに compat を指定し、NIS+ を使用しているなら passwd_compat: nisplus を追加してください。

protocols テーブル

protocols テーブルにはインターネットで使われるプロトコルが含まれます。これには 4 つの列があります。

表 **23–14** protocols テーブル

列	説明
正式名	プロトコル名
名前	プロトコルの識別に使われる非公式な別名
番号	プロトコルの番号
説明	プロトコルに関するコメント

rpc テーブル

rpc テーブルには、RPC プログラム名が含まれます。これには 4 つの列があります。

表 **23–15** rpc テーブル

列	説明
正式名	プログラム名
名前	プログラムの起動に使用できる別の名前
番号	プログラムの番号
説明	RPC プログラムに関するコメント

rpc テーブルの入力ファイルの例を次に示します。

```
#
# rpc file
#
rpcbind    00000    portmap    sunrpc    portmapper
rusersd    100002    rusers
nfs    100003    nfsprog
mountd    100005    mount    showmount
walld    100008    rwall    shutdown
sprayd    100012    spray
llockmgr    100020
nlockmgr    100021
status    100024
bootparam    100026
keyserv    100029    keyserver
nisd    100300    rpc.nisd
#
```

services テーブル

services テーブルは、インターネット上で使用できるインターネットサービスに関する情報を格納しています。これには 5 つの列があります。

表 **23–16** services テーブル

列	説明
正式名	サービスの公式インターネット名
名前	サービスを要求できる代替名のリスト
プロトコル	サービスを提供するためのプロトコル (たとえば 512/tcp)
ポート	ポート番号
コメント	サービスに関するコメント

timezone テーブル

timezone テーブルは、ドメイン内の全マシンのデフォルトの時間帯を含んでいます。デフォルトの時間帯はインストール中に使用されますが、インストーラがこれを無効にすることができます。このテーブルには 3 つの列があります。

表 **23–17** timezone テーブル

列	説明
名前	ドメイン名
時間帯	時間帯の名前 (たとえば US/Pacific)
コメント	時間帯に関するコメント

その他のデフォルトのテーブル

以下のテーブルも、デフォルトで用意されています。

- audit_user
- auth_attr

- exec_attr
- prof_attr
- user_attr

詳細については、『man pages section 4』を参照してください。

第 24 章

NIS+ の問題解決

この章では、問題をいくつかの種類に分類しています。各問題について、一般的な症状を示し問題について説明してから、推奨する対策を 1 つまたは複数挙げています。

また、付録 A では、NIS+ の詳細な共通エラーメッセージをアルファベット順に掲載しています。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。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+ のデバッグオプション

NIS+ のデバッグオプション

`NIS_OPTIONS` の環境変数を設定して、NIS+ デバッグオプションを制御できます。

オプションは、二重引用符で囲まれたオプションセットと共にスペースで区切られた `NIS_OPTIONS` コマンドの後に指定されます。各オプションに *name=value* のフォーマットがあります。値は、特定のオプションに従って整数、文字列、ファイル名になります。整数値のオプションに対して、値が指定されていない場合には、デフォルト値は 1 になります。

`NIS_OPTIONS` は次のオプションを認識します。

表 **24–1** NIS_OPTIONS オプションと値

オプション	値	アクション
debug_file	*filename*	デバッグの指定ファイルへの出力を指示する。このオプションが指定されていない場合は、デバッグは stdout に出力する
debug_bind	*Number*	サーバーの選択プロセスに関する情報を表示する
debug_rpc	*1* または *2*	値が 1 の場合には、NIS+ に対する RPC コールを表示する。値が 2 の場合には、RPC コールと RPC の内容、引数、結果の両方を表示する
debug_calls	*Number*	NIS+ API へのコールと、アプリケーションに戻される結果を表示する
pref_srvr	*String*	nisprefadm コマンドによって生成されるのと同じフォーマットで、優先サーバーを指定する (表 20–1 を参照)。これは nis_cachemgr で指定された優先サーバーを上書きする
server	*String*	特定のサーバーを設定する
pref_type	*String*	現在実装されていない

例を以下に示します。(C シェルの使用を前提)

- デバッグメッセージを表示するには、以下を入力します。

```
setenv NIS_OPTIONS "debug_calls=2 debug_bind debug_rpc"
```

- API コールの簡単なリストを得て、それをファイル /tmp/CALLS に格納するには、以下を入力します。

```
setenv NIS_OPTIONS "debug_calls debug_file=/tmp/CALLS"
```

- 特定のサーバーに送られた API コールの簡単なリストを得るには、以下を入力します。

```
setenv NIS_OPTIONS "debug_calls server = sirius"
```

NIS+ の管理上の問題

この節では、日常的な NIS+ 名前空間の管理作業を行なっているときに発生する可能性のある問題について説明します。一般的な症状には次のようなものがあります。

- 「Illegal object type for operation」メッセージ
- その他の「オブジェクトの問題」のエラーメッセージ
- 初期設定のエラー
- チェックポイントのエラー
- ユーザーをグループに追加するときのエラー
- ログが大きすぎる場合、ディスク容量が不足した場合、ログの切り捨てのエラー

- groups_dir や org_dir が削除できない

無効なオブジェクトの問題

「症状」

- 「Illegal object type for operation」メッセージ
- その他の「オブジェクトの問題」のエラーメッセージ

このエラーメッセージには、いくつもの原因が考えられます。

- 検索可能な列がない状態で、テーブルを作成しようとしました。
- データベース処理が DB_BADOBJECT の状態を返しました (db のエラーコードの詳細は、nis_db(3N) のマニュアルページを参照してください)。
- 長さを 0 に指定して、データベースオブジェクトを追加または変更しようとしました。
- 所有者を指定せずに、オブジェクトを追加しようとしました。
- その処理はディレクトリオブジェクトを想定していますが、指定したオブジェクトは、ディレクトリオブジェクトではありませんでした。
- ディレクトリを LINK オブジェクトにリンクしようとしました。
- テーブルエントリにリンクしようとしました。
- グループオブジェクトではないオブジェクトを、nisgrpadm コマンドに渡しました。
- グループオブジェクトの処理が想定されていましたが、指定したオブジェクトのタイプはグループオブジェクトではありませんでした。
- テーブルオブジェクトの処理が想定されていましたが、指定したオブジェクトはテーブルオブジェクトではありませんでした。

nisinit のエラー

次の点をチェックしてください。

- NIS+ が動作していることを、ping によって確認できるか
- -H オプションで指定した NIS+ サーバーが、適切なサーバーであるか、また現在も動作しているか
- rpc.nisd がサーバー上で動作しているか
- その他クラスが、このドメインの読み取り権を持っているか
- このマシンで、ネットマスクが正しく設定されているか

チェックポイントのエラーが続く

チェックポイント処理 (たとえば、`nisping -C` コマンドによる操作) が、継続してエラーを起こしている場合は、十分なスワップ空間やディスク容量があるかどうか確認してください。`syslog` 内のエラーメッセージもチェックします。`core` ファイルがディスク空間を使い果たしていないかどうかチェックします。

ユーザーをグループに追加することができない

ユーザーをそのドメイン内のグループのメンバーとして追加する前に、そのユーザーは、ドメインの `cred` テーブルの中で LOCAL の資格を持つ NIS+ 主体クライアントになっていなければなりません。DES の資格を持っているだけでは、十分ではありません。

ログが大きくなりすぎた

`nisping -C` を使用して定期的にチェックポイントを実行していないと、ログファイルが極端に大きくなることがあります。マスターサーバーのすべての複製サーバーが更新されるまでは、マスター上にあるログはクリアされません。複製がダウンしている場合や、サービスが行われていない場合、複製サーバーと通信できない場合は、その複製サーバーに対応するマスターをクリアできません。このため複製がダウンしているか、または一定時間使用できない場合には、340 ページの「ディレクトリを削除する」で説明するとおり、マスターから複製を削除する必要があります。ディレクトリ `org_dir` とサブディレクトリ `groups_dir` を最初に削除してから、ディレクトリ自体を削除してください。

ディスク容量の不足

十分なディスク容量が確保できない場合は、様々なエラーメッセージが表示されます (詳細は、467 ページの「ディスク容量の不足」を参照)。

トランザクションログファイルを切り捨てることができない

まずはじめに、問題のファイルが存在するか、読み込み可能か、そのファイルに書き込み権が割り当てられているかどうかチェックしてください。

- トランザクションログは、`ls /var/nis/trans.log` で表示できます。
- ファイルの存在、アクセス権、読み取り権については、`nisls -l` と `niscat` で確認できます。
- 関係するメッセージは、`syslog` でチェックできます。

最も可能性のある原因は、適切なアクセス権を割り当てられているものの、ディスク容量が不足していることです。チェックポイント処理では、ログを切り捨て一時ファイルを削除する前に、ますログ一時ファイルのコピーを作成します。このため、一時ファイルを作成するためのディスク容量がない場合は、チェックポイント処理を進めることができません。使用可能なディスク容量をチェックし、必要であれば容量を確保してください。

ドメイン名の混同

NIS+ の多くのコマンドや操作にとって、ドメイン名は重要な役割を果たします。ルートサーバーを除いて、NIS+ のすべてのマスターと複製は、それ自身がサービスを提供するドメインより上にあるドメインのクライアントであるということに特に注意してください。サーバーまたは複製サーバーを、それ自身がサービスを提供するドメインのクライアントとして誤って扱った場合、「`Generic system error`」や「`Possible loop detected in namespace` *directoryname*`:`*domainname*」というエラーメッセージが表示されます。

たとえば、`altair` というマシンが、`subdoc.doc.com.` ドメインのクライアントだとします。サブドメイン `subdoc.doc.com.` のマスターサーバーが `sirius` というマシンだとすると、`sirius` は `doc.com.` ドメインのクライアントになります。したがって、ドメインの指定や変更を行うときは、混同しないように次のルールに注意してください。

1. クライアントマシンは、特定のドメインかサブドメインに所属します。
2. 特定のサブドメインにサービスを提供するサーバーや複製サーバーは、そのドメインより上にあるサブドメインのクライアントです。
3. 2 の規則の唯一の例外は、ルートマスターサーバーとルート複製サーバーです。これらは、それ自身がサービスを提供するドメインのクライアントになります。つまり、ルートドメインのクライアントになるのは、ルートマスターとルート複製だけです。

したがって、上の例では、マシン `altair` の完全指定名は、`altair.subdoc.doc.com.` です。マシン `sirius` の完全指定名は、`sirius.doc.com.` です。`sirius.subdoc.doc.com.` という名前は正しくありません。`sirius` は、`doc.com.` のクライアントで `subdoc.doc.com.` のクライアントではないため、エラーの原因となります。

`org_dir` や `groups_dir` を削除できない

親ディレクトリを削除する前に、必ず `org_dir` と `groups_dir` を削除してください。ドメインの `groups_dir` と `org_dir` を削除する前に、`nisrmdir` を使用してドメインを削除すると、これらの 2 つのサブディレクトリは、どちらも削除できなくなります。

複製の失敗からの NIS+ ディレクトリの削除または分離

複製サーバーからディレクトリを削除または分離する場合には、最初にディレクトリの `org_dir` と `groups_dir` のサブディレクトリを削除してから、ディレクトリ自体を削除します。各サブディレクトリが削除された後に、削除しようとするディレクトリの親ディレクトリで `nisping` を実行する必要があります (340 ページの「ディレクトリを削除する」 を参照)。

`nisping` の操作に失敗すると、ディレクトリは完全に削除または分離されません。

この状態が発生したら、次の手順を実行して、修正する必要があります。

1. 複製上の `/var/nis/rep/org_dir` を削除します。
2. `org_dir.` *domain* が、複製上の `/var/nis/rep/serving_list` に表示されないことを確認してください。
3. *domain* で `nisping` を実行します。
4. マスターサーバーから `nisrmdir -f` *replica_directory* を実行します。

分離しようとしている複製サーバーがダウンしているか、または通信不能である場合に `nisrmdir -s` コマンドは「`Cannot remove replica` *name* `: attempt to remove a non-empty table`」というエラーメッセージが出されます。

このような場合は、マスターサーバー上で `nisrmdir -f -s` *replicaname* コマンドを実行すれば、強制的に切り離すことができます。しかし、`nisrmdir -f -s` *replicaname* を使って通信不能な複製を分離する場合には、複製がオンライン状態に戻ったらすぐに、`nisrmdir -f -s` *replicaname* を再実行して、複製の `/var/nis` ファイルシステムを消去する必要があります。`nisrmdir -f -s` *replicaname* を再実行しないと、複製がサービスを再開した時に複製上に残された古い情報によって問題が発生します。

NIS+ データベースの問題

この節では、名前空間のデータベースとテーブルに関連する問題を説明します。一般的な症状には次のようなものがあります。処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- `Abort_transaction: Internal database error` メッセージ
- `Abort_transaction: Internal Error, log entry corrupt`
- `Callback: select failed`
- `CALLBACK_SVC: bad argument`

`rpc.nisd` が失敗します。

451 ページの「NIS+ の所有権とアクセス権の問題」を参照してください。

`rpc.nisd` の複数の親プロセス

「症状」

次の表現が含まれている様々なエラーメッセージが表示されます。これらのメッセージは、データベースやトランザクションログが壊れていることを意味しています。

- `Log corrupted`
- `Log entry corrupt`
- `Corrupt database`
- `Database corrupted`

「考えられる原因」

複数の独立した `rpc.nisd` デーモンを実行させています。通常の動作では、`rpc.nisd` は他の `rpc.nisd` デーモンを子プロセスとして生成できます。このこと自体は問題はありません。しかし、1 台のマシン上で 2 つの `rpc.nisd` デーモンが親として動作している場合は、互いのデータを上書きし、ログとデータベースを壊してしまいます。このような現象が発生するのは、`rpc.nisd` が手作業で起動された場合です。

「診断」

`ps -ef | grep rpc.nisd` を実行します。親の `rpc.nisd` プロセスは、1 つしか動作していないことを確認します。

「対策」

「親」としての `rpc.nisd` のエントリが複数ある場合は、1 つを残して、他のデーモンの動作を終了させなければなりません。`kill -9` *process-id* を実行し、もう一度 `ps` コマンドを実行して、他のデーモンが終了したかどうか確認します。

注 – `-B` オプションを指定して `rpc.nisd` を起動した場合は、`rpc.nisd_resolv` デーモンも終了させる必要があります。

NIS+ データベースが壊れている場合は、壊れていないデータベースを、最新のバックアップから復元します。次に、バックアップの時点よりあとで、名前空間に加えられた変更を反映します。しかし、ログも壊れている場合は、バックアップを行なったあとで名前空間に加えられた変更を、再度手作業で行わなければなりません。

`rpc.nisd` の失敗

NIS+ テーブルが大きすぎると、`rpc.nisd` は失敗します。

「診断」

`nisls` を使って、NIS+ テーブルの大きさをチェックします。7K バイト以上のテーブルでは、`rpc.nisd` は失敗します。

「対策」

NIS+ テーブルの大きさを小さくします。ネームサービスとして NIS+ は、オブジェクト自体ではなく、オブジェクトへのリファレンスを格納するために設計されています。

NIS+ と NIS の互換性の問題

この節では、NIS と NIS+ や以前のシステムとの間の互換性に関連する問題、またスイッチ構成ファイルに関連する問題を説明します。一般的な症状には次のようなものがあります。

- `nsswitch.conf` ファイルが正しく実行されない
- 処理内容に関係する次の表現が含まれているエラーメッセージが表示される

表示されるエラーメッセージの中には、次のようなものがあります。

- `Unknown user`
- `Permission denied`
- `Invalid principal name`

ユーザーがパスワードを変更したあと、ログインできない

「症状」

新しいユーザーや、最近パスワードを変更したユーザーが、ログインできません。または、特定のマシンからログインできますが、他のマシンからログインできません。そのようなユーザーに対して、次の表現が含まれているエラーメッセージが表示されることがあります。

- `Unknown user` *username*
- `Permission denied`
- `Invalid principal name`

「最初に考えられる原因」

NIS マシン上でパスワードが変更されました。

NIS+ の名前空間サーバーがサービスを提供しているドメインの中で、NIS を実行している Solaris オペレーティング環境のマシン上で、ユーザーまたはシステム管理者が `passwd` コマンドを使用してパスワードを変更した場合、そのユーザーのパスワードは、そのマシンの `/etc/passwd` ファイルの中だけで変更されています。そのユーザーが、ネットワーク上の他のマシンを使用してログインしても、そのマシン上では新しいパスワードは認識されません。NIS+ のパスワードテーブルに格納されている、古いパスワードを使用しなければなりません。

「診断」

そのユーザーの古いパスワードが、NIS+ の他のマシンでまだ有効かどうかチェックしてください。

「対策」

NIS+ を実行しているマシンで、passwd コマンドを使用して、ユーザーのパスワードを変更します。

「2 番目に考えられる原因」

パスワードの変更を行なっても、システム全体に反映されるまでに時間がかかります。

「診断」

名前空間の変更がドメインやシステム全体に反映されるまでに、ある程度の時間がかかります。ドメインのサイズや複製サーバーの台数により、数秒間で済むこともあれば、何十分もかかることがあります。

「対策」

変更結果がドメインに伝わるまで、常識的に受け入れられる程度の時間、待つだけです。または、nisping org_dir コマンドを使用して、システムを同期させることもできます。

nsswitch.conf ファイルが正しく実行されない

変更した (または、新しくインストールした) nsswitch.conf ファイルが正しく実行されません。

「症状」

新しい nsswitch.conf ファイルをインストールしたか、または既存のファイルを変更しましたが、システムがその変更結果を反映していません。

「考えられる原因」

nsswitch.conf ファイルのインストールや変更を行なった後は、必ずマシンを再起動して、変更結果を有効にしなければなりません。nscd が nsswitch.conf ファイルをキャッシュに書き込むためです。

「対策」

nsswitch.conf(4) のマニュアルページの情報を参照して、現在の nsswitch.conf ファイルをチェックしてください。必要に応じてファイルを修正し、マシンを再起動します。

NIS+ オブジェクトが見つからない問題

この節では、NIS+ がオブジェクトや主体を見つけることができない問題について説明します。一般的な症状には次のようなものがあります。

処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- `Not found`
- `Not exist`
- `Can't find suitable transport for` *name*
- `Cannot find`
- `Unable to find`
- `Unable to stat`

構文やスペリングの誤り

NIS+ のオブジェクトが見つからない場合、最も可能性のある原因は、名前の入力を間違えたことです。構文をチェックし、正しい名前を使用しているかどうか確認してください。

正しくないパス名

「オブジェクト」が見つからない場合、考えられる原因として、正しくないパスを指定していることが挙げられます。指定したパスが正しいことを確認してください。また、`NIS_PATH` 環境変数が正しく設定されているかどうかも確認してください。

ドメインレベルが正しく指定されていない

すべてのサーバーは、それ自身がサービスを提供しているドメインではなく、その上にあるドメインのクライアントであることに注意してください。この規則には、2 つの例外があります。

- ルートマスターサーバーとルート複製サーバーは、ルートドメインのクライアントです。
- *NIS+* のドメイン名は、ピリオドで終わります。完全指定名を使用する場合は、ドメイン名の終わりにピリオドをつけなければなりません。ドメイン名の終わりにピリオドをつけないと、NIS+ は、それが部分指定名であると想定します。この規則の例外としては、`/etc/defaultdomain` ファイルの中では、マシンのドメイン名の終わりにピリオドをつけません。`/etc/defaultdomain` ファイルの中で、マシンのドメイン名にピリオドをつけると、起動時に「`Could not bind to server serving domain` *name*」というエラーメッセージが表示され、ネットワークへの接続に問題が生じます。

オブジェクトが存在しない

NIS+ のオブジェクトが削除されたため、または作成されていないために、現在は存在していなくて、見つからないこともあります。`nisls -l` を使用して、そのドメインに目的のオブジェクトが存在するかどうかチェックしてください。

複製サーバーの同期遅延

NIS+ のオブジェクトの作成や変更を行うと、処理が完了して、特定の複製サーバーの情報が更新されるまでに、ある程度の遅れが発生します。通常の動作では、マスターかその複製から名前空間情報の照会が行われます。クライアントは、照会を複数のサーバー (マスターと複製) に自動的に分散し、システムの負荷のバランスを取ります。これは、どの時点でも、名前空間の情報をどのマシンが返してくるかわからないということを意味します。新しく作成されたオブジェクトや変更されたオブジェクトに関係するコマンドが、更新された情報をまだマスターから受け取っていない複製に送られた場合、「オブジェクトが見つかりません」というタイプのエラーか、同期していない古い情報を受け取ることになります。同様に、`nisls` のような、全体に関係するコマンドを使用して、まだ更新されていない複製サーバーに照会を行なった場合、新しく作成されたオブジェクトが含まれていないリストを受け取る可能性があります。

`nisping` を使用すると、同期していない状態にある複製サーバーを同期させることができます。

代わりに、NIS+ のコマンドで `-M` オプションを指定して、そのコマンドがドメインのマスターサーバーから名前空間の情報を受け取るように指定することも可能です。この方法を使用すると、確実に最新の情報を取得し、使用できます。ただし、`-M` オプションは、必要なときにだけ使用してください。複製サーバーを使用して名前空間のサービスを行う最大の理由は、負荷を分散して、ネットワークの効率を向上させることにあります。

ファイルが見つからないか壊れている

`/var/nis/data` ディレクトリの中の 1 つまたは複数のファイルが、壊れているか削除されています。最新のバックアップから、これらのファイルを復元してください。

旧バージョンの `/var/nis` について

Solaris 2.4 以前では、`/var/nis` ディレクトリに *hostname*`.dict`、*hostname*`.log` という 2 つのファイルが含まれていました。またサブディレクトリ `/var/nis/`*hostname* もありました。Solaris 2.5 においては、2 つのディレクトリ名は `trans.log`、`data.dict`、サブディレクトリ名は `/var/nis/data` となります。

`nisinit` など NIS+ 設定プロシージャーによって作成された `/var/nis`、`/var/nis/data` といったディレクトリ、およびその下のファイルは、名前を変更しないでください。

Solaris 2.5 ではこれらのファイルの内容も変更されており、Solaris 2.4 以前との互換性はなくなっています。したがって、これらのファイルやディレクトリを Solaris 2.4 での名前にしてしまうと、Solaris 2.4、および 2.5 以降の `rpc.nisd` で機能しなくなりますので名前の変更をしないようにしてください。ディレクトリ名もファイル名も変更しないでください。

名前の中の空白

「症状」

ある時にはオブジェクトが存在し、別の時には存在しないことがあります。NIS+ や UNIX の特定のコマンドが NIS+ のオブジェクトが存在しない、または見つからないと報告しますが、別のコマンドは同じオブジェクトを見つけることができます。

「診断」

`nisls` を使用して、オブジェクト名を探します。オブジェクト名を注意深くチェックして、その名前が実際は空白で始まっていないかどうか確認してください。NIS+ のコマンド行を使用して NIS+ のオブジェクトを作成するときに、フラグの後に誤って空白文字を 2 回入力すると、NIS+ のコマンドの中には、2 番目の空白文字をオブジェクト名の一部と解釈するものがあります。

「対策」

NIS+ のオブジェクト名が空白で始まっている場合は、名前を変更して空白を取り除くか、いったん削除して初めから作成し直します。

オートマウンタを使用できない問題

「症状」

他のホスト上のディレクトリに移動できません。

「考えられる原因」

NIS+ 環境では、NIS+ の必要条件に合わせて、オートマウンタの名前を変更しなければなりません。`/etc/auto*` テーブル名の一部としてピリオドが使用されている場合、NIS+ はそれらのテーブルをアクセスすることができません。たとえば、NIS+ は `auto.direct` というファイルにアクセスすることができません。

「診断」

`nisls` と `niscat` を使用して、オートマウンタのテーブル名が正しく割り当てられているかどうか確認します。

「対策」

ピリオドを下線 (_) に変更します。たとえば、`auto.direct` を `auto_direct` という名前に変更します。これらのテーブルを参照している可能性のある他のマップも変更してください。

テーブルエントリ間でのリンクが機能しない

`nisln` コマンド (またはその他のコマンド) を使って、テーブルエントリ間でのリンクは作成できません。NIS+ コマンドはエントリレベルでのリンクは追跡しません。

NIS+ の所有権とアクセス権の問題

この節では、ユーザーの所有権とアクセス権に関連する問題を説明します。一般的な症状には次のようなものがあります。

処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- `Unable to stat name`
- `Unable to stat NIS+ directory name`
- `Security exception on LOCAL system`
- `Unable to make request`
- `Insufficient permission to . . .`
- `You name do not have secure RPC credentials`

一般的に観察されるその他の現象

- ユーザーまたはスーパーユーザーが、名前空間に関係する作業を行うことができない

アクセス権がない

アクセス権に関連して最も頻繁に発生する問題は、最も単純な問題です。行おうとしている業務に必要なアクセス権が、割り当てられていません。対象としているオブジェクトを指定して `niscat -o` を使用し、どのアクセス権が割り当てられているか確認します。他のアクセス権も必要な場合は、ユーザー自身、オブジェクトの所有者、システム管理者のうちの誰かが、そのオブジェクトのアクセス権の変更を行うことができます。詳細については、第 15 章 および 第 17 章を参照してください。

資格がない

ユーザーやマシンに適切な資格がない場合は、ほとんどの操作を行なったときにエラーが発生します。ホームドメインの `cred` テーブルを対象にして `nismatch` を使用し、正しい資格を割り当てられているかどうか確認します (資格に関連した問題の詳細については、458 ページの「無効になった資格」を参照)。

サーバーがセキュリティレベル 0 で動作している

セキュリティレベル 0 で動作しているサーバーは、NIS+ の主体の資格の作成や管理を行いません。

セキュリティレベル 0 で動作しているサーバーで `nispasswd` を試みると、次のエラーメッセージが表示されます。「`You` *`name`* `do not have secure RPC credentials in NIS+ domain` *`domainname`*」

セキュリティレベル 0 は、管理者が名前空間の初期設定やテストを行う際にだけ使用されます。一般のユーザーがアクセスするような環境で使用すべきではありません。

ユーザーのログインがマシン名と同じ

マシン名と同じものを、ユーザーのログイン ID とすることはできません。ユーザー名と同じ名前をマシンに割り当てる (またはその逆) と、最初の主体は、セキュリティに関係するアクセス権を必要とする動作を行うことができなくなります。2 番目の主体の鍵が、cred テーブルの中にある、最初の主体の鍵を上書きするからです。さらに、2 番目の主体は、最初の主体に割り当てられていたアクセス権を持つようになります。

たとえば、saladin というログイン名を持つユーザーが、名前空間の中で読み込み専用のアクセス権を割り当てられていたとします。次に、saladin という名前を持つマシンをドメインに追加します。ユーザー saladin は、何らかの種類のアクセス権を必要とする名前空間の操作を行うことができなくなります。そして、マシン saladin のスーパーユーザーは、名前空間の中で、読み込み専用のアクセス権だけを割り当てられます。

「症状」

- ユーザーやマシンが「permission denied」エラーメッセージを受け取ります。
- ユーザーまたはスーパーユーザーが、keylogin を正しく実行できなくなります。
- 「Security exception on LOCAL system. UNABLE TO MAKE REQUEST.」というエラーメッセージが表示されます。
- 最初の主体が読み込みアクセスを割り当てられていない場合、2 番目の主体が、本来表示できるはずのオブジェクトを見ることができなくなります。

注 – nisclient や nisaddcred を実行したときに、「Adding Key」ではなく「Changing Key」というメッセージが表示された場合は、そのドメインの中で、ユーザー名またはホスト名がすでに重複しています。

「診断」

nismatch を実行して、hosts テーブルや passwd テーブル内のホストとユーザーを表示し、各テーブルの中に、同じホスト名やユーザー名が存在しないか確認します。以下に例を示します。

```
nismatch username passwd.org_dir
```

次に、ドメインの cred テーブルを対象にして nismatch を実行し、重複しているホスト名やユーザー名に、どのようなタイプの資格が割り当てられているかを調べます。LOCAL と DES 両方の資格が割り当てられている場合、cred テーブルのエントリはユーザーを表わしています。DES の資格だけが割り当てられている場合、エントリはマシンを表わしています。

「対策」

マシン名を変更します (ユーザー名を変更するより、マシン名を変更することをお勧めします)。次に、cred テーブルからそのマシンのエントリを削除し、nisclient を使用して、マシンを NIS+ のクライアントとして初期設定します (必要に応じて、nistbladm を使用し、そのマシンの別名を hosts テーブルの中に作成し、元のマシン名を別名として使用することもできます)。必要に応じて、cred テーブル内のユーザーの資格を変更します。

正しくない資格

458 ページの「無効になった資格」を参照してください。

NIS+ のセキュリティの問題

この節では、パスワード、資格、暗号、その他セキュリティに関係した一般的な問題を取り上げます。

セキュリティ問題の症状

処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- `Authentication error`
- `Authentication denied`
- `Cannot get public key`
- `Chkey failed`
- `Insufficient permission to`
- `Login incorrect`
- `Keyserv fails to encrypt`
- `No public key`
- `Permission denied`
- `Password [problems]`

ユーザーまたはスーパーユーザーは、名前空間に関係する作業を行うことができません (451 ページの「NIS+ の所有権とアクセス権の問題」を参照)。

「Login Incorrect」というメッセージが表示された

原因としてもっとも多いのが、パスワードの誤入力です。もう一度入力するようユーザーに指示してください。「記憶しているパスワードが正しいか」、「パスワードでは大文字と小文字が区別されることを理解しているか」、「アルファベットの o と数字の 0、アルファベットの l と数字の 1 などを混同していないか」といったことについても確認してください。

「login incorrect」メッセージの原因としては他に以下のようなものが考えられます。

- パスワードが管理者によってロックされている (310 ページの「パスワードのロック」、311 ページの「パスワードロックの解除」を参照)。
- 指定の日数以上ログインが行われなかったため、パスワードがロックされた (318 ページの「ログインの間隔の最大値の指定」 を参照)。
- パスワードが有効期限を過ぎた (316 ページの「パスワード使用権の有効期限」 を参照)。

パスワードについては、第 16 章を参照してください。

パスワードがロック状態、期限切れ、または無効である

「`Permission denied, password expired`」といったタイプのメッセージは、「ユーザーのパスワードが有効期限を過ぎた」、「またはパスワード使用権が無効になった」といった理由で表示されることが最も多くなっています。詳細については、第 16 章を参照してください。

- 313 ページの「パスワードの有効期間の設定」参照
- 316 ページの「パスワード使用権の有効期限」参照

資格情報が古い

資格、アクセス権ともに正しいにもかかわらず、クライアントの要求が拒否されるという場合もあります。これは、名前空間のどこかに古い情報が存在することが原因である可能性があります。

資格情報の保存と更新

公開鍵など、資格に関する情報は、名前空間内の様々な場所に保存されています。NIS+ は、この情報を保存先のオブジェクトの生存期間の値にもとづいて定期的に更新しますが、更新と更新との間にときおり情報のずれが起こります。その結果、一部の操作が行えなくなります。表 24–2 は、資格に関する情報を保存するすべてのオブジェクト、テーブル、ファイルと、そのリセットの方法を示したものです。

表 **24–2** 資格に関連する情報が格納されている場所

項目	保存対象	リセットおよび更新の方法
`cred` テーブル	NIS+ 主体の非公開鍵と公開鍵。これらの鍵のマスターコピーとなる	`nisaddcred` を使用して新しい資格を作成する。これによって既存の資格が更新される。`chkey` を使用しても同様のことが行える
ディレクトリオブジェクト	個々のサーバーの公開鍵のコピー	ディレクトリオブジェクトに対して `/usr/lib/nis/ nisupdkeys` コマンドを実行する

表 **24–2** 資格に関連する情報が格納されている場所　(続き)

項目	保存対象	リセットおよび更新の方法
キーサーバー	その時点でログインされている NIS + 主体の非公開鍵	主体ユーザーに対して `keylogin` を実行する。または主体マシンに対して `keylogin -r` を実行する
NIS+ デーモン	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	デーモンおよびキャッシュマネージャを終了した後、再起動する
ディレクトリキャッシュ	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	NIS+ キャッシュマネージャを終了した後、`nis_cachemgr -i` コマンドを使用して再起動する。`-i` オプションを指定すると、コールドスタートファイルからディレクトリキャッシュがリセットされた後、キャッシュマネージャが再起動される
コールドスタートファイル	ディレクトリオブジェクトのコピー (そのサーバーの公開鍵のコピーが含まれる)	ルートマスターで NIS+ デーモンを終了した後、再起動する。デーモンは、既存の `NIS_COLD_START` ファイルに新しい情報を再ロードする。 まず、主体の `/var/nis` からコールドスタートファイルと共有ディレクトリファイルを削除し、キャッシュマネージャを終了する。次に `nisinit - c` を使用して主体を再初期化する。主体に対して「trusted」と指定したサーバーは、新しい情報を主体の既存のコールドスタートファイルに再読み込みする
`passwd` テーブル	ユーザーのパスワードまたはマシンのスーパーユーザーのパスワード	`passwd -r nisplus` コマンドを使用する。これによって、NIS+ `passwd` テーブル、`cred` テーブルの中でパスワードが更新される
`passwd` ファイル	ユーザーのパスワードまたはマシンのスーパーユーザーのパスワード	`passwd -r nisplus` コマンドを使用する。スーパーユーザー、一般ユーザーのどちらでログインしてもよい
`passwd` マップ (NIS)	ユーザーのパスワードまたはマシンのスーパーユーザーのパスワード	`passwd -r nisplus` を使用する

古くなったキャッシュに保存された鍵の更新

情報が古くなる原因として最も多いのが、サーバーの公開鍵のバージョンが古くなることです。一般に、この問題を解決するには、アクセスするドメインに対して `nisupdkeys` を実行します (`nisupdkeys` コマンドの使用方法の詳細については、第 12 章を参照)。

鍵の中にはファイルやキャッシュに保存されているものもあるため、nisupdkeys ですべての問題を解決できるわけではありません。鍵を手作業で更新しなければならない場合もあります。この場合は、「サーバーの公開鍵の内容は、公開鍵が作成された後どのように名前空間オブジェクトに伝えられるのか」ということについて理解する必要があります。サーバーの公開鍵の伝播には、一般に以下の 5 つの段階があります。それぞれの詳細について説明します。

1：サーバーの公開鍵が作成される

NIS+ サーバーはまず第 1 に、NIS+ クライアントです。つまり、NIS+ サーバーの公開鍵は、NIS+ クライアントの公開鍵と同様に、 nisaddcred コマンドによって生成されます。公開鍵はその後、サーバーが実際にサポートするドメインではなく、サーバーのホームドメインの cred テーブルに保存されます。

2：公開鍵の内容がディレクトリオブジェクトに伝えられる

NIS+ ドメインと NIS+ サーバーの設定後は、サーバーとドメインを関係づけることができます。この「関係づけ」は、nismkdir コマンドで行います。nismkdir コマンドによってサーバーとディレクトリの関係づけが行われる際、サーバーの公開鍵も cred テーブルからドメインのディレクトリオブジェクトにコピーされます。たとえば、サーバーが doc.com. ルートドメインのクライアントで、sales.doc.com. ドメインのマスターサーバーになっているという場合を考えてみましょう。

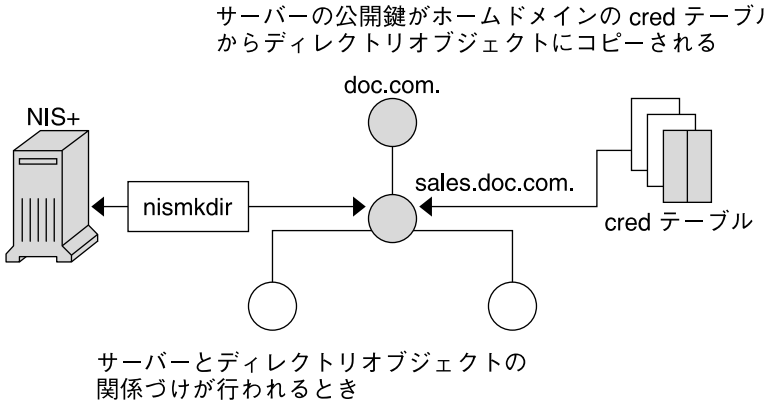


図 **24–1** ディレクトリオブジェクトへの公開鍵の伝播

公開鍵は cred.org_dir.doc.com. ドメインから、ディレクトリオブジェクト sales.doc.com. にコピーされます。以上のことは、niscat -o sales.doc.com. というコマンドを使用して確認できます。

3：ディレクトリオブジェクトの内容がクライアントファイルに伝えられる

nisinit ユーティリティまたは nisclient スクリプトを使用すれば、すべての NIS+ クライアントを初期化できます。

他の類似のコマンドと同様、nisinit (または nisclient) では、コールドスタートファイル /var/nis/NIS_COLDSTART が作成されます。コールドスタートファイルは、クライアントのディレクトリキャッシュ /var/nis/NIS_SHARED_DIRCACHE の初期化に使用されます。コールドスタートファイルには、クライアントのドメイン中のディレクトリオブジェクトのコピーが含まれています。ディレクトリオブジェクトには、すでにサーバーの公開鍵のコピーが含まれているため、これで公開鍵の内容はクライアントのコールドスタートファイルに伝えられたことになります。

また、クライアントがホームドメインの外のサーバーに対して要求をした場合、リモートドメインのディレクトリオブジェクトのコピーが、クライアントの NIS_SHARED_DIRCACHE ファイルに保存されます。クライアントのキャッシュの内容は、nisshowcache コマンドを使用して調べることができます。

複製サーバーがドメインに追加されるか、サーバーの鍵が更新されるまでは、鍵の伝播はこの段階にとどまります。

4：複製サーバーがドメインに追加された場合の処理

複製サーバーがドメインに追加されると、nisping コマンド (347 ページの「nisping コマンド」 を参照) によって NIS+ テーブル (cred テーブルを含む) が新しい複製サーバーにダウンロードされます。これによって、元のサーバーの公開鍵も複製サーバーの cred テーブルに保存されます。

5：サーバーの公開鍵が更新された場合の処理

サーバーの DES 資格 (サーバーのルート ID) を変更すると、公開鍵も変更されます。その結果、サーバー用に cred テーブルに保存される公開鍵が、以下の場所に保存されるものと矛盾します。

- 複製サーバーの cred テーブル (数分の間)
- サーバーがサポートするドメイン中の、メインディレクトリオブジェクト (生存期間終了まで)
- サーバーがサポートするドメイン中の、クライアントの NIS_COLDSTART ファイルおよび NIS_SHARED_DIRCACHE ファイル (生存期間終了まで、通常 12 時間)
- サーバーがサポートするドメインに要求をしたクライアントの、NIS_SHARED_DIRCACHE ファイル (生存期間終了まで)

サーバーの鍵は、ほとんどの場所において数分～ 12 時間で自動的に更新されます。すぐに更新するには、以下のコマンドを使用します。

表 **24–3** サーバーの鍵の更新

保存場所	コマンド	参照する項目
複製サーバーの cred テーブル (nisping を使用しなくても、テーブルは数分で自動的に更新される)	nisping	347 ページの「nisping コマンド」

表 **24–3** サーバーの鍵の更新　(続き)

保存場所	コマンド	参照する項目
サーバーがサポートするドメインのディレクトリオブジェクト	nisupdkeys	259 ページの「nisupdkeys コマンド」
クライアントの NIS_COLDSTART ファイル	nisinit -c	344 ページの「nisinit コマンド」
クライアントの NIS_SHARED_DIRCACHE ファイル	nis_cachemgr	345 ページの「nis_cachemgr コマンド」

注 – nis_cachemgr の再起動は、既存の nis_cachemgr を終了してから行います。

無効になった資格

主体 (ユーザーかマシン) の資格が無効になっている場合は、その主体は nisls のようなコマンドも含め、名前空間の操作や処理を行うことができなくなってしまいます。資格が無効になると、未認証クラスに割り当てられるアクセス権も含め、すべてのアクセス権が失われるからです。

「症状」

ユーザーまたはスーパーユーザーが、名前空間に関係する作業を行うことができなくなります。名前空間のどのような操作を行なっても、「permission denied」というタイプのエラーメッセージが表示されます。ユーザーまたはスーパーユーザーは、nisls を実行することも不可能になります。

「考えられる原因」

鍵の破損、物理的な破損、古い資格、その他何らかの不適切な点が、/etc/.rootkey ファイルの中にあります。

「診断」

snoop を使用して、不適切な資格を識別します。

あるいは、その主体がリスト表示できる場合は、その主体としてログインを行い、主体が間違いなく承認されているはずのオブジェクトを対象として、NIS+ コマンドを実行します。たとえば、ほとんどの場合、未認証クラスにオブジェクトの読み込みは承認されているはずです。そこで、cred テーブルの中にリストされている主体は、nisls コマンドを正しく実行できるはずです。このコマンドを実行しても「permission denied」エラーが発生する場合は、おそらく、その主体の資格は無効になっています。

「対策」

- 「一般のユーザー」の場合、keylogout を実行してから、次に keylogin を実行して、その主体のログインを試みます。
- 「root」すなわち「スーパーユーザー」の場合、`keylogout -f` を実行してから、`keylogin -r` を実行します。

Keyserv のエラー

keyserv が、セッションを暗号化できません。このタイプの問題には、いくつかの原因が考えられます。

「考えられる原因と対策」

- クライアントが `keylogin` を実行していません。`keylogin` を実行したかどうか確認します。クライアントが正しく `keylogin` を実行したかどうかを確かめるには、(そのクライアントで) `nisdefaults -v` を実行します。`Principal Name` という行に「`not authenticated`」というメッセージが返れば、クライアントが正しく `keylogin` を実行していないことになります。
- クライアント (またはホスト) が、LOCAL や DES のような適切な資格を持っていません。クライアントの `cred` テーブルを対象にして `niscat` を実行し、クライアントが適切な資格を持っているかどうか確認します。必要に応じて、247 ページの「NIS+ 主体の資格情報を作成する方法」に従って資格を追加します。
- `keyserv` デーモンが動作していません。`ps` コマンドを使用して、`keyserv` が動作しているかどうか確認します。動作していない場合は、このデーモンを起動してから `keylogin` を実行します。
- `keyserv` が動作している場合は、Secure RPC や NIS+ のコールを行う終始動作しているはずのプロセスが、まだ動作していません。たとえば、`automountd`、`rpc.nisd`、`sendmail` などが動作していません。これらのプロセスが正しく動作しているかどうか確認します。動作していない場合は再起動します。

マシンが以前は NIS+ のクライアントだった

このマシンが、同じドメインの中で NIS+ のクライアントとして初期設定されている場合は、試みに `keylogin -r` を実行し、Secure RPC パスワードプロンプトでスーパーユーザーのログインパスワードを入力します。

cred テーブルにエントリがない

`cred` テーブルの中に主体 (ユーザーまたはホスト) の NIS+ のパスワードが存在することを確認するために、主体のホームドメインの中で次のコマンドを実行します。

```
nisgrep -A cname=principal cred.org_dir.domainname
```

`nisgrep` を別のドメインで実行する場合は、*domainname* には完全な名前を指定する必要があります。

ドメイン名が変更されている

ドメイン名は変更すべきではありません。

既存のドメイン名を変更すると、認証の問題が発生するはずです。ネットワーク全体で、オブジェクトの中に完全指定のドメイン名が埋め込まれているからです。

ドメイン名を変更しないでください。既にドメイン名を変更してしまい、認証の問題や、ドメイン名に関係する「malformed」や「illegal」などの表現が含まれているメッセージが表示されている場合は、ドメイン名を元の名前に戻します。ドメイン名を変更したい場合は、次の手順に従ってください。「新しい名前」を使用して「新しいドメイン」を作成し、新しいドメインでマシンをサーバーやクライアントとして設定し、これらが正しく動作していることを確認した上で、古いドメインを削除します。

マシンを新しいドメインに変更する

NIS+ のクライアントとなっているマシンを、他のドメインのクライアントに変更したい場合は、`/etc/.rootkey` ファイルを削除し、ネットワーク管理者から受け取ったパスワードか `nisclient` スクリプトから取り出したパスワードを使用して、`nispopulate` スクリプトを実行し直します。

/etc/passwd ファイルの中にある NIS+ のパスワードとログインパスワード

NIS+ のパスワードは、NIS+ の `passwd` テーブルの中に格納されています。ログインパスワードは、NIS+ の `passwd` テーブルか、各ユーザーの `/etc/passwd` ファイルの中に格納されています。ユーザーパスワードと NIS+ のパスワードは、同じでも違っていてもかまいません。`/etc/passwd` ファイル内のパスワードを変更するには、`nsswitch.conf` ファイルでの設定を「files」にするか、「`-r files`」というフラグを指定するかして `passwd` コマンドを実行する必要があります。

`nsswitch.conf` ファイルは、どの目的にどのファイルを使用するか指定します。`nsswitch.conf` ファイルが、システムの照会に対して誤った場所を指示している場合は、パスワードやアクセス権のエラーが発生するはずです。

Secure RPC パスワードとログインパスワードが異なる

主体のログインパスワードと Secure RPC パスワードが一致しないと、ログイン時に `keylogin` はパスワードを復号化できません。`keylogin` はデフォルトで主体のログインパスワードを使用することになっていて、また非公開鍵は主体の Secure RPC パスワードを使用して暗号化されているからです。

この場合、主体はシステムにログインすることはできますが、NIS+ においては未認証クラスとして扱われます。キーサーバーが、そのユーザーの非公開鍵を復号化されていない状態のまま持っているからです。NIS+ 環境では多くの場合、未認証クラス

による NIS+ オブジェクトへのアクセス (作成、削除、変更) が拒否されるため、この状態でユーザーが NIS+ オブジェクトにアクセスしようとしても「permission denied」といったタイプのエラーになってしまいます。

注 – 「ネットワークパスワード」と「Secure RPC パスワード」は、このコンテキストでは同義語として扱われる場合があります。ネットワークパスワードの入力を求められたら Secure RPC パスワードを入力します。

認証クラスを「未認証」以外にするには、ユーザーがこの状況で keylogin プログラムを実行し、パスワード入力を求める keylogin プロンプトが表示されたときに主体の Secure RPC パスワードを入力する必要があります (254 ページの「キーログイン」を参照)。

しかし keylogin を実行しても、現在のログインセッション以外には無効で、一時的な解決にしかなりません。キーサーバーはこの方法によって、復号化された形でユーザーの非公開鍵を持つようになるのですが、ユーザーの cred テーブル中の非公開鍵が Secure RPC パスワード (ユーザーのログインパスワードとは異なっている) を使用して暗号化されているという点に変わりはないからです。ログインし直してしまえば、状況はまったく元どおりになってしまいます。根本的に問題を解決するためには、cred テーブルの非公開鍵を、Secure RPC パスワードではなくログイン ID に基づいたものに変更する必要があります。この作業は、chkey プログラムを使用して行います (255 ページの「NIS+ 主体の鍵の変更」を参照)。

Secure RPC パスワードとログインパスワードが異なっているという問題を根本的に解決するには、具体的には以下の作業を行います。

1. ログインパスワードを使用してログインします。
2. keylogin プログラムを実行して、キーサーバーに保存される非公開鍵を一時的に復号し、一時的な NIS+ アクセス権を得ます。
3. chkey -p 実行して、cred テーブル中の暗号化された非公開鍵を、ユーザーのログインパスワードに基づいたものに固定的に変更します。

/etc/.rootkey ファイルがすでに存在している

「症状」

「insufficient permission」や、「permission denied」などの、様々なエラーメッセージ

「考えられる原因」

サーバーやクライアントの設定や初期設定を行なったときに、/etc/.rootkey ファイルがすでに存在していました。以前そのマシンに NIS+ をインストールしたことがあり、NIS+ を削除したとき、または NIS や /etc への変更を行なったときに、.rootkey ファイルを削除しなかったためにこのような状態が起こります。

「診断」

/etc ディレクトリで ls -l と nisls -l org_dir を実行し、/etc/.rootkey の日付を、cred テーブルの日付と比較します。/etc/.rootkey の日付が明らかに cred テーブルより古い場合は、ファイルがあらかじめ存在していたことが考えられます。

「対策」

問題のあるマシンで、root として keylogin -r コマンドを実行し、そのマシンをもう一度クライアントとして設定し直します。

root のパスワードを変更したための問題

「症状」

マシンの root のパスワードを変更した結果、変更結果が反映されなかったか、スーパーユーザーとしてログインできなくなりました。

「考えられる原因」

注 – セキュリティ上の理由から、passwd テーブルの中に、UserID 0 という項目を、設けるべきではありません。

---

root のパスワードを変更した際、root に対して chkey -p を実行していなかったり、何らかの問題が発生したことにより、変更が正しく行われませんでした。

「対策」

管理特権を持つユーザー (つまり、管理特権が割り当てられているグループに所属するメンバー) としてログインし、passwd を使用して、元のパスワードに戻します。元のパスワードが正しく機能するかどうか確かめてください。正しく機能すれば、passwd を使用してパスワードを変更した後、chkey -p を実行します。

注意 – NIS+ の名前空間の設定が終わり、すでに動作している状態でも、ルートマスターのマシンを使って root のパスワードを変更することは可能です。しかし、ルートマスターの鍵は変更しないでください。これらは、サブドメイン内のすべてのクライアント、複製サーバー、サーバーの中のすべてのディレクトリオブジェクトに埋め込まれているからです。chkey を root で実行するとき、必ず -p オプションを指定するようにすれば、ルートマスターの鍵を変更する必要はなくなります。

NIS+ の性能の低下とシステムのハングアップの問題

この節では、性能の低下とシステムのハングアップの問題を取り上げます。

性能の低下の症状

処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- 「Busy try again later」
- 「Not responding」

次の一般的な現象が観察されることもあります。

- コマンドを実行しても、長時間、何も行われていないように見える
- システムやシェルが、キーボードやマウスの操作に全く反応しない
- NIS+ の動作が通常よりも遅い

チェックポイントの実行

誰かが nisping か nisping -C どちらかのコマンドを実行しました。または、rpc.nisd デーモンが、チェックポイントを実行しています。

注意 – 再起動しないでください。nisping を実行しないでください。

nisping やチェックポイントを実行すると、サーバーは反応が遅くなったり、他のコマンドにすぐに応答しなくなったりすることがあります。名前空間のサイズに応じて、このようなコマンドが完了するまでに、かなりの時間を要します。これらのコマンドの実行中に、誰かが同様のコマンドを実行すると、所要時間は何倍にもなります。再起動も行わないでください。この種の問題は自然に解決します。サーバーが nisping やチェックポイントの実行を完了するまで待つだけです。

マスターサーバーと複製サーバーの同期が完全に行われている場合、同期が完了するまで、複製サーバーはサービスを停止します。再起動しないで待機してください。

変数 NIS_PATH

NIS_PATH 変数が、明快かつ単純な値に設定されているかどうか確認します。たとえば、デフォルトでは org_dir.$:$ に設定されています。複雑な NIS_PATH、特に他の変数を含んでいる変数は、システムの速度を低下させ、特定の操作にエラーを引き起こす可能性があります (詳細は、84 ページの「環境変数 NIS_PATH」 を参照)。

デフォルト以外のテーブルパスを指定するために nistbladm を使用することは避けてください。デフォルト以外のテーブルパスを指定すると、性能は低下します。

テーブルパス

テーブルパスは使用しないでください。性能を低下させることになります。

複製サーバーが多すぎる

1 つのドメインの中に複製サーバーが多すぎる場合は、複製作業を行なっている間はシステムの性能が低下します。1 つのドメインまたはサブドメインの中に、10 台より多くの複製サーバーを置くことは避けてください。ドメインの中の複製サーバーが 5 台より多い場合は、何台かを取り除いて性能が改善されるかどうか調べます。

再帰的なグループ

再帰的なグループとは、他のグループを包含するグループのことです。グループの中に他のグループを含めると、システム管理者として行う作業は減少しますが、システムの速度は低下します。再帰的なグループを使うべきではありません。

起動時の NIS+ データベースのログが大きい

`rpc.nisd` は、起動時に各ログを参照します。ログが大きい場合は、起動時の参照に時間がかかることがあります。`rpc.nisd` を起動する前に、`nisping -C` を使用して、チェックポイントを実行することをお勧めします。

マスターの `rpc.nisd` デーモンが終了している

「症状」

`-M` オプションを指定して要求をマスターサーバーに送ったときに、マスターのマシンで `rpc.nisd` デーモンが終了していると、「`server not responding`」タイプのエラーメッセージが発生し、更新を行うことは認められません。`-M` オプションを指定しなかったときは、要求は機能している複製サーバーに自動的に転送されます。

「考えられる原因」

ホームディレクトリ名やホスト名の一部として大文字を使うと、`rpc.nisd` デーモンが終了することがあります。

「診断」

最初に、サーバー自体が起動されて、動作しているかどうか確認します。動作している場合は、`ps -ef | grep rpc.nisd` を実行し、デーモンが動作しているかどうか調べます。

「対策」

デーモンが終了している場合は、起動し直します。たびたびデーモンが終了する場合は、ご購入先にご連絡ください。

nis_cachemgr がない

「症状」

あるマシンが他のドメインにある名前空間オブジェクトを探すのに、非常に長い時間がかかっています。

「考えられる原因」

nis_cachemgr を実行していません。

「診断」

ps -ef | grep nis_cachemgr を実行して、nis_cachemgr が動作しているかどうか確認します。

「対策」

そのマシンで、nis_cachemgr を起動します。

NIS+ のインストール後、サーバーの起動が非常に遅い

「症状」

NIS+ のスクリプトを使用して NIS+ をインストールした後、サーバーの動作が非常に遅く、緩慢です。

「考えられる原因」

nispopulate スクリプトを動作させた後、nisping -C -a を実行していません。

「対策」

nisping -C -a を実行して、できるだけ早くチェックポイントを実行します。

niscat が エラーメッセージ「Server busy. Try Again 」を返す

「症状」

niscat を実行すると、サーバーがビジーであるというエラーメッセージが表示されます。

「考えられる原因」

- 再同期を実行しているときのように重い負荷がかかっていて、サーバーがビジーです。
- サーバーがスワップ空間を使い果たしました。

「診断」

`swap -s` を実行して、サーバーのスワップ空間をチェックします。

「対策」

NIS+ を実行するには、適度のスワップ空間とディスク容量を用意すべきです。必要に応じて、空間を増やします。

ホスト名を変更した後で、NIS+ の照会がハングする

「症状」

NIS+ サーバーのホスト名を完全指定することは、推奨されていません。このような指定を行なった後、NIS+ の照会を実行すると、エラーメッセージを表示することなくハングアップが発生しました。次の可能性をチェックします。

「考えられる原因」

完全指定のホスト名は、次の条件を満たしていなければなりません。

- ホスト名のドメイン部は、`domainname` コマンドの結果と完全に一致している
- ホスト名を完全指定名に変更した後、`/etc` や `/etc/inet` 内のファイルに、新しいホスト名の情報を反映させる
- ホスト名の終わりにピリオドをつける

「対策」

ハングアップした NIS+ プロセスを終了させ、ホストやサーバーの `rpc.nisd` も終了させます。上の 2 つの条件に合わせて、ホスト名を変更します。`nisinit` を使用してサーバーを初期設定します。ホスト名が正しいことを確認した後もハングアップが発生する場合は、この節の他の考えられる原因をチェックしてください。

NIS+ のシステムリソースの問題

この節では、メモリーやディスク容量のようなシステムリソースが不足したときの問題を取り上げます。

リソースの問題

処理内容に関係する次の表現が含まれているエラーメッセージが表示されます。

- `No memory`
- `Out of disk space`

- 「Cannot [do something] with log」などのメッセージ
- Unable to fork

メモリーの不足

システムのメモリーやスワップ空間が不足すると、NIS+ の様々な問題やエラーメッセージに遭遇します。一時的な解決策として、不必要なウィンドウやプロセスを終了させることが考えられます。必要に応じて、ウィンドウシステムを終了し、端末のコマンド行を使用します。それでもメモリー不足を知らせるメッセージが表示されるときは、スワップ空間やメモリーを追加するか、十分なスワップ空間やメモリーを持つシステムに切り替えます。

特定の状況では、アプリケーションやプロセスがメモリーを無駄に消費し、使用メモリーサイズが極端に大きくなることがあります。次のコマンドを実行すると、アプリケーションやプロセスの現在のサイズを知ることができます。

```
ps -el
```

sz (サイズ) の列には、各プロセスの現在のメモリーサイズが表示されています。必要に応じて、サイズが同程度と思われるアプリケーションやプロセスを比較し、メモリーサイズが極端に大きいものが存在しないかどうか調べます。

ディスク容量の不足

ディスク容量が不足すると、様々なエラーメッセージが表示されます。ディスク領域の不足に共通する原因は、定期的に行われている tmp ファイルの削除やログファイルの切り捨てをしていないことです。切り捨てを行わないと、ログファイルと tmp ファイルは常時増加します。これらのファイルの増大速度はシステムごとに異なり、同じシステムでも状態によって変化します。非効率的なシステムや、名前空間に問題を抱えているシステムでは、ログファイルは非常に急速に増大します。

注 – 多くの問題が発生しているときは、ログファイルと /tmp のファイルを頻繁にチェックします。ディスク容量が不足して他の問題が発生する前に、ログファイルを切り捨て、/tmp ファイルを削除します。また、ルートディレクトリとホームディレクトリで core ファイルを探して削除します。

ログファイルを切り捨てるには、システムのチェックポイントを設けます。チェックポイントのプロセスがある程度の時間を要し、完了するまではシステムの速度を低下させることに注意してください。

システムのチェックポイントを実行するには、nisping -C を実行します。

プロセス数の不足

過度に負荷の大きいマシンでは、マシンの構成の中で指定されている、同時に実行できる最大のプロセス数に達する可能性があります。この結果、「unable to fork」のようなメッセージが表示されます。この問題を解決するために推奨されている方法は、不必要なプロセスを終了させることです。それでも問題が持続する場合は、システムの管理マニュアルの説明に従って、より多くのプロセスを扱えるようにシステムを構成し直してください。

NIS+ のユーザーの問題

この節では、一般ユーザーが遭遇する NIS+ の問題を取り上げます。

ユーザーの問題

- ユーザーがログインできない
- ユーザーが他のドメインにリモートログインできない

ユーザーがログインできない

ユーザーがログインできない原因としては、以下のように様々なものが考えられます。

- パスワードを忘れた。新しいパスワードを設定します。別のマシンのユーザーの場合は、nispasswd を使用します。この作業は NIS+ 管理者だけが行えます。
- パスワードを間違えて入力した。ユーザーが、正しいパスワードを覚えているかどうか、また「パスワードでは大文字と小文字が区別される」、「アルファベットの o 、l と数字の 0 、1 は、まったく別のものである」という点を理解しているかどうかを確認します。
- 「Login incorrect」というタイプのメッセージが表示される。単純に「パスワードの入力を間違えた」という以外の原因については、453 ページの「「Login Incorrect」というメッセージが表示された」を参照してください。
- ユーザーのパスワード使用権が有効期限を過ぎている (316 ページの「パスワード使用権の有効期限」を参照)。
- 指定された期間を超えてログインが行われなかった (318 ページの「ログインの間隔の最大値の指定」を参照)。
- nsswitch.conf ファイルの設定が正しくない。passwd エントリの設定は以下の 5 つのうちのいずれかにします。
  - passwd: files
  - passwd: files nis
  - passwd: files nisplus
  - passwd: compat

- `passwd: compatpasswd_compat: nisplus`

これ以外の設定をすると、ユーザーがログインできなくなります。

詳細は、300 ページの「`nsswitch.conf` ファイルの必要条件」を参照してください。

ユーザーが新しいパスワードを使ったときにログインできない

「症状」

最近パスワードを変更したユーザーが、ログインできません。または、特定のマシンからログインできますが、他のマシンからログインできません。

「考えられる原因」

- パスワードの変更を行なっても、システム全体に反映されるまでに時間がかかります。試しに、古いパスワードを使ってログインしてみてください。
- NIS+ が動作していないマシンで、パスワードを変更しました (469 ページの「ユーザーが新しいパスワードを使ったときにログインできない」を参照)。

ユーザーがリモートドメインにログインできない

「症状」

ユーザーが `rlogin` を使用して、他のドメインへのログインを試みましたが、「`Permission denied`」メッセージが表示されて、拒絶されました。

「考えられる原因」

他のドメインにリモートログイン (`rlogin`) するには、ユーザーはそのドメインで LOCAL の資格を持っていなければなりません。

「診断」

そのドメインで、`nismatch` *username*`.` *domainname*`. cred.org_dir` を実行し、LOCAL の資格を持っているかどうか調べます。

「対策」

リモートドメインから `nisaddcred` を使用し、そのドメインでの LOCAL の資格をユーザーに割り当てます。

ユーザーがパスワードを変更できない

原因として最も多いのが、古いパスワードの入力を間違えた (または忘れた) ということです。

他には以下のような原因が考えられます。

- パスワードの最小値に、最大値よりも大きい値が設定されている (305 ページの「nistbladm と シャドウ列のフィールド」参照)
- パスワードがロックされている、あるいは有効期限を過ぎている (453 ページの「「Login Incorrect」というメッセージが表示された」、454 ページの「パスワードがロック状態、期限切れ、または無効である」を参照)

NIS+ に関するその他の問題

この節では、上記の問題に当てはまらない問題を取り上げます。

NIS+ の動作

特定のホストが NIS+ を実行しているかどうか知りたい場合があります。NIS+ が動作しているか知るには、スクリプトが必要です。

次のどれかに一致する場合は、NIS+ が動作していると考えられます。

- `nis_cachemgr` が動作している
- ホストに `/var/nis/NIS_COLD_START` ファイルがある
- `nisls` の実行に成功した

複製サーバーの更新のエラー

「症状」

更新が成功しなかったことを示すエラーメッセージが表示されます。次のメッセージが更新の成功を示すことに注意してください。「`replica_update:`*number* `updates` *number* `errors` 」

「考えられる原因」

次のエラーメッセージのどれかが表示された場合、サーバーがビジーであり、更新を延期すべきことがわかります。

- `Master server busy, full dump rescheduled`
- `replica_update: error result was Master server busy full dump rescheduled, full dump rescheduled`
- `replica_update: master server busy, rescheduling the resync`
- `replica_update: master server busy, will try later`
- `replica_update: nis dump result Master server busy, full dump rescheduled`
- `nis_dump_svc: one replica is already resyncing`

これらのメッセージは、NIS+ のエラーコード定数 NIS_DUMPLATER (ある複製がすでに再同期を実行している) によって、または同時に生成されます。

次のメッセージは、他の問題が起こっていることを示します。

- replica_update: error result was ...
- replica_update: nis dump result nis_perror error string
- rootreplica_update: update failednis dump result nis_perror string-variable: could not fetch object from master

-C (診断チャンネルのオープン) オプションを指定した rpc.nisd が動作している場合は、マスターサーバーか複製サーバーのシステムログに、詳細な情報が記録されることもあります。

これらのメッセージは、次のような潜在的な問題が起こっていることを示しています。

- サーバーが割り当てることができる子プロセスを使い果たした
- 読み込み専用の子プロセスがダンプを行うよう要求された
- 他の複製サーバーが、現在同期を実行している

「診断」

複製サーバーとマスターサーバー両方のシステムログから、詳細な情報を探します。情報が記録されている場合でも、詳細の程度は、システムのエラー報告レベルと、-C オプション (診断) を指定して rpc.nisd を実行したかどうかによって異なります。

「対策」

ほとんどの場合、これらのメッセージは、システムが修正することのできる、ソフトウェアの小さな問題が発生したことを意味しています。あるコマンドを実行した結果、これらのメッセージが表示された場合は、しばらく待って、もう一度同じコマンドを実行します。これらのメッセージが頻繁に表示される場合は、/etc/syslog.conf ファイル内のしきい値レベルを変更します。詳細は、syslog.conf のマニュアルページを参照してください。

パート IV　FNS の設定、構成、および管理

このパートでは、フェデレーテッド・ネーミング・サービス (FNS) の設定、構成、および管理について説明します。

第 25 章

フェデレーテッド・ネーミング・サービス (FNS)

この章では、NIS+、NIS、またはファイルを使用したネームサービス環境にフェデレーテッド・ネーミング・サービス (FNS) を設定および構成する方法について説明します (「ファイルを使用した」ネームサービスでは、NIS+ または NIS ではなく、/etc ファイルからデータを取得します)。また、Solaris オペレーティング環境上の FNS に関する管理方法および問題解決について説明します。

FNS の手引き

「FNS」は、1 つの Solaris オペレーティング環境でさまざまなネームサービスを独立して動作させるための機能です。FNS を使用すれば、ネットワーク上のさまざまなネームサービスすべてに、1 つの簡単なネームシステムインタフェースで対応できます。FNS は、X/Open federated naming (XFN) 規格に適合しています。

FNS は、NIS+、NIS、DNS、/etc ファイルの代わりとして使用することはできません。FNS はむしろこれらのサービスの一番上に位置しており、通常の名前をデスクトップ上のアプリケーションで使用できるようにします。

XFN (X/Open Federated Naming)

FNS がサポートするプログラミングインタフェースとそのポリシーは、XFN (X/Open Federated Naming) に述べられています。

FNS を使用する理由

FNS は、次の点で有用です。

- 単一の一貫したネーミングおよびディレクトリインタフェースがクライアントに対して用意されている。これを使用してネーミングおよびディレクトリサービスにアクセスできる。したがって、新しいネーミングおよびディレクトリサービスを追加しても、アプリケーションや既存のサービスに変更を加える必要はない
- 名前を一貫した方法で作成できる。FNS は、異なる複数のネーミングシステムから一貫した名前を作成する方法を定義する。これにより、アプリケーションは、これらの異なる複数のネーミングシステム内のオブジェクトを一定の方法でアドレス指定できる
- 共有コンテキストと共有名の使用によって、一貫したネーミングを行うことができる。異なる複数のアプリケーションでこれらの共有名と共有コンテキストを使用すると、作業を重複して行う必要がなくなる

複合名と複合コンテキスト

FNS の基本概念として、複合名と複合コンテキストがあります。

複合名

複合名とは、複数のネーミングシステムで使用される名前のことをいいます。

複合名は、複数の構成要素の順序付きリストからなります。各構成要素は、単一のネーミングシステムの名前空間からとられた名前です。各構成要素の構文は、個々のネーミングシステムにより異なります。FNS は、複数のネーミングシステムからとられた名前を使用して複合名を作成するときの構文を定義します。複合名は、スラッシュ (/) を構成要素区切り文字として使用して、左から右へ作成されます。

たとえば、複合名 `.../doc.com/site/bldg-5.alameda` は、`...`、`doc.com`、`site`、`bldg-5.alameda` という 4 つの構成要素から構成されます。

コンテキスト

コンテキストは、次の操作を行います。

- 名前をオブジェクトに関連付ける (割り当てる)
- 名前をオブジェクトとして解釈処理する
- 割り当てを削除する
- 名前を表示する
- 名前を変更する
- 名前付きオブジェクトに属性を関連付ける

- 名前付きオブジェクトに関連付けられた属性を取り出して更新する
- 属性を使用してオブジェクトを検索する

コンテキストには、一連の名前とリファレンスの割り当てが含まれます。各リファレンスには、通信の終端またはアドレスのリストが含まれます。フェデレーテッド・ネーミング・システムは、別のネーミングシステムのコンテキストで割り当て中の、あるネーミングシステムのコンテキストによって形成されます。複合名の解釈処理は、その名前全体が解釈処理されるまで、あるネーミングシステム内のコンテキストから、次のネーミングシステム内のコンテキストへと進みます。

属性

名前付きオブジェクトには、属性を適用できます。属性はオプションです。名前付きオブジェクトには、属性を付けないことも、1 つまたは複数の属性を付けることもできます。

属性はユニークな属性識別子、属性構文、および 0 個以上の明確な属性値のセットからなります。

XFN で、既存の名前付きオブジェクトに対応する属性値の検索や変更のための、基本的な属性インタフェースが定義されます。これらのオブジェクトは、コンテキストまたは他の任意のタイプのオブジェクトにできます。コンテキストに関連しているのは、構文属性で、これはコンテキストがどのように複合名を解析するかを示します。

拡張属性インタフェースには、特定の属性を検索したり、オブジェクトやそれに対応する属性を作成するオペレーションが含まれます。

FNS とネームサービススイッチ

FNS (XFN API を Solaris 上に実装したもの) は、クライアントがネーム情報を照会するためのネームサービスを指定するときにも使用できます。XFN API は、X、Y 両次元において、スイッチファイルを使用する最新の `getXbyY()` インタフェースよりも一般的です。たとえば、XFN API を使用して、NIS+ と NIS の両方からホストとユーザーに関する情報を調べることができます。アプリケーションは、`getXbyY()`、XFN、あるいはその両方のクライアントとして使用できます。

FNS とスイッチファイルに一貫性を持たせる

FNS による名前空間データの変更を、スイッチファイルを使用して名前空間情報を入手しているクライアントが常に把握できるようにするために、スイッチと FNS には常に同じネームサービスを設定してください。

名前空間の更新

XFN API によるデータ更新は、`getXbyY()` インタフェースによる更新よりも優れています。ほとんどの名前空間は複数ソースのデータで構成されています。たとえば、`groups` の名前空間には、`/etc/group` ファイルと NIS+ `group.org_dir` オブジェクトの両方の情報があるかもしれません。しかし、スイッチファイルは、グループデータの特定の部分のソースや更新するソースを識別するための、アプリケーションの更新関数に十分な情報を提供しません。

各 FNS の従属名前空間は、すべてが 1 つのネームサービスから取られます。更新がどのネームサービスに行われるかというような混乱がないため、更新は簡単明瞭なものになります。

エンタープライズネームサービス

エンタープライズレベルのネームサービスは、組織内のオブジェクトをネーミングするために使用されます。現在 FNS は、3 つのエンタープライズレベルのネームサービスをサポートしています。サポートされているネームサービスは、NIS+、NIS およびファイルの 3 つです。

NIS+

NIS+ は、Solaris オペレーティング環境で推奨されるエンタープライズ規模の情報サービスです。FNS の組識単位は、NIS+ のドメインとサブドメインに対応します。各ドメインとサブドメインごとに 1 つの `orgunit` コンテキストがあります。

NIS+ のもとで、FNS のコンテキストと属性データは、NIS+ テーブルに格納されます。これらのテーブルは、`ctx_dir` という名前の NIS+ ディレクトリオブジェクトに格納されます。各 NIS+ ドメインとサブドメインごとに 1 つの `ctx_dir` ディレクトリオブジェクトがあり、ドメインの `groups_dir` および `org_dir` の各ディレクトリオブジェクトと同じレベルにあります。したがって、ディレクトリオブジェクト `ctx_dir.sales.doc.com.` には、`sales.doc.com` ドメインの FNS コンテキストと属性データを格納する FNS テーブルが含まれます。

NIS+ のもとでは、FNS と NIS+ のコマンドを使用して、FNS テーブル内の情報を処理します。これらのテーブルを直接編集したり、UNIX コマンドを使用して処理したりしないでください。

NIS

NIS は、Solaris オペレーティング環境上に実装されたエンタープライズ規模の情報サービスです。各エンタープライズは 1 つの NIS ドメインにあたります。1 つの NIS ドメインに対応する 1 つの FNS 組織単位があります。

NIS のもとで、FNS のコンテキストと属性データは NIS マップに格納されます。これらのマップは、NIS サーバー上の `/var/yp/`*domainname* ディレクトリに格納されます。NIS では、スーパーユーザーは、FNS コマンドを使用して、FNS マップ内の情報を処理できます。

SKI が実行されている場合に NIS クライアントが FNS でコンテキストを更新する

一定の条件が満たされれば、NIS クライアント (マシン、プロセス、またはユーザー) はすべて、`fncreate_fs`、`fncreate_printer` などの FNS コマンドを使用して、クライアント独自のコンテキストを更新できます。これにより、NIS クライアントは、FNS コマンドを使用して、Printer Administrator、CDE カレンダマネージャ、admintool などのアプリケーションを更新できます。

スーパーユーザー以外のユーザーが、FNS コマンドを使用して独自のコンテキストを更新するには、次の条件が必要です。

- NIS マスターサーバー上で SKI (Secure Key_management Infrastructure) が使用可能
- `fnsypd` デーモンが、NIS マスターサーバー上で実行されている。このデーモンは、スーパーユーザー特権を持つユーザーが開始する
- クライアントユーザーまたはマシンは、独自のコンテキストの更新だけできる
- クライアントは、要求された更新を実行するための権限を持っている

注 – SKI は 64 ビットモードをサポートしていません。したがって NIS クライアントは 64 ビットモードでのコンテクストの更新は行うことができません。

ファイルベースのネーミング

「ファイル」とは、通常、マシンの `/etc` ディレクトリにあるネーミングファイルを指します。これらのマシンベースファイルには、UNIX のユーザーとパスワード情報、ホスト情報、メール別名などが入っています。これらのファイルは、自動マウントマップなどの Solaris 特定のデータもサポートしています。

ファイルベースのネーミングシステムでは、FNS コンテキストと属性データはファイルに格納されます。これらの FNS ファイルは、マシンの `/var/fn` ディレクトリに格納されます。`/var/fn` ディレクトリを各マシンに置く必要はありません。このディレクトリは、NFS ファイルサーバーからエクスポートされます。

ファイルネーミングシステムでは、FNS コマンドを使用して FNS ファイルの情報を処理します。

グローバルネームサービス

FNS は、DNS と X.500 による NIS+ と NIS のフェデレートもサポートします。つまり、エンタープライズレベルの名前空間をグローバル名前空間に接続し、エンタープライズオブジェクトをグローバルな有効範囲でアクセス可能にできるということです。

FNS は現在、次のグローバルネームサービスをサポートしています。

- DNS
- X.500 (DAP または LDAP を経由)

FNS ネーミングポリシー

FNS は、ネーミングポリシーを定義して、ユーザーとアプリケーションが共有名前空間に依存し、それを使用できるようにします。

エンタープライズ内には、組織単位、サイト、ホスト、ユーザー、ファイルとサービスごとの名前空間があり `orgunit`、`site`、`host`、`user`、`fs` (ファイルシステムを示す)、および `service` という名前で呼ばれます。これらの名前空間は、各名前の前に下線 (_) を付けることもできます。たとえば、`host` と `_host` は同じものと見なされます。

表 25–1 は、エンタープライズレベルの名前空間に関する FNS ポリシーを要約しています。

表 **25–1** FNS ポリシーの要約

コンテキストタイプ	従属コンテキスト	親コンテキスト
`orgunit _orgunit`	`site user host fs service`	`enterprise root`
`site _site`	`user host fs service`	`enterprise root orgunit`

表 **25–1** FNS ポリシーの要約　*(続き)*

コンテキストタイプ	従属コンテキスト	親コンテキスト
user _user	service fs	enterprise root
		orgunit
host _host	service fs	enterprise root
		orgunit
service _service	プリンタなどのアプリケーション	enterprise root
		orgunit site user host
fs _fs (ファイルシステム)	(なし)	enterprise rootorgunit site user host

組織名

FNS orgunit の割り当ては、基本となるネームサービスによって決まります。

- NIS+ では、組織単位は NIS+ のドメインまたはサブドメインに対応する。たとえば、NIS+ のルートドメインが doc.com. で、sales が doc.com. のサブドメインと仮定する。この場合、FNS 名 org/sales.doc.com. および org/sales が、NIS+ ドメイン sales.doc.com. に対応する組織単位になる (sales.doc.com. の最後のドットは、完全指定 NIS+ 名に必要であることに注意)
- NIS では、組織単位は NIS ドメインであり、必ず FNS 名 org// または org/*domainname* によって識別される。ここで、*domainname* は、doc.com. などの完全指定ドメイン名を示す。NIS の組織単位名には階層はない
- ファイルベースのネーミングシステムで、組織単位は、必ず FNS 名 org// によって識別されるシステムである

組織単位名に対応して命名されるオブジェクトのタイプは、user、host、service、fs、site です。次に例を示します。

- org/sales/site/conference1.bldg-6 は、組織単位 sales に関連するサイトの 6 号ビルにある会議室 conference1 を指定する。この例では、org/sales が sales.doc.com. に対応している場合、このオブジェクトは org/sales.doc.com. /site/conference1.bldg-6 という方法で指定することもできる (sales.doc.com. の末尾のドットに注意)
- org/finance/user/mjones は、組織単位 finance のユーザー mjones を指定する
- org/finance/host/inmail は、組織単位 finance に属するマシン inmail を指定する
- org/accounts.finance/fs/pub/reports/FY92-124 は、組織単位 accounts.finance に属するファイル pub/reports/FY92-124 を指定する

- org/accounts.finance/service/calendar は、組織単位 accounts.finance のカレンダサービスを指定する。これは、組織単位のミーティングのスケジュールを管理する

サイト名

サイト名は、必要に応じて作成されます。サイト名に対応して指定されるオブジェクトのタイプは、 user, host, service, fs です。次に例を示します。

- site/alameda/user/mjones は、サイト alameda にあるユーザー mjones を指定する
- site/alameda/host/sirius は、サイト alameda にあるマシン sirius を指定する
- site/alameda/service/printer/Sparc-2 は、サイト alameda にあるプリンタ Sparc-2 を指定する
- site/alameda/fs/usr/dist は、サイト alameda で使用可能なファイルディレクトリ usr/dist を指定する

ユーザー名

ユーザー名は、NIS+ の対応する passwd テーブル、NIS の passwd マップ、またはファイルの /etc/passwd ファイルにある名前に対応します。ユーザーのファイルコンテキストは、そのユーザーの passwd エントリから獲得されます。

ユーザー名に対応して指定されるオブジェクトのタイプは、 service と fs です。次に例を示します。

- user/chou/service/fax は、ユーザー chou のファックスサービスを指定する
- user/esperanza/fs/projects/conf96.doc は、ユーザー esperanza のファイルシステムの projects サブディレクトリにあるファイル、conf96.doc を指定する

ホスト名

ホスト名は、NIS+ の対応する hosts、NIS の hosts マップ、またはファイルの /etc/hosts ファイルにある名前に対応します。ホストのファイルコンテキストは、ホストによってエクスポートされるファイルシステムに対応します。

ホスト名に対応して指定されるオブジェクトのタイプは、service と fs です。次に例を示します。

- host/smtp-1/service/mailbox は、マシン smtp-1 に関連するメールボックスサービスを指定する

- host/deneb/fs/etc/.cshrc は、ホスト deneb 上の /etc ディレクトリにあるファイル .cshrc を指定する

サービス名

サービス名は、サービスアプリケーションに対応し、それによって決定されます。service コンテキストは、組織、ユーザー、ホスト、またはサイトコンテキストに対応して指定する必要があります。次に例を示します。

- org//service/printer は、組織のプリンタサービスを指定する
- host/deneb/service/printer は、マシン deneb に関連するプリンタサービスを指定する
- host/deneb/service/printer/Sparc-2 は、マシン deneb に関連するプリンタを指定する
- user/charlie/service/calendar は、ユーザー charlie のカレンダサービスを指定する
- site/conf_pine.bldg-7.alameda/service/calendar は、Alameda サイトの 7 号ビルにある conf_pine 会議室のカレンダサービスを指定する

ファイル名

ファイルシステム名は、ファイル名に対応します。例を次に示します。

- host/altair/fs/etc/.login は、マシン altair 上の .login ファイルを指定する
- user/prasad/fs/projects/96draft.doc は、ユーザー prasad の projects ディレクトリにあるファイル 96draft.doc を指定する

開始前に必要な処置

各自のネームサービスで FNS を開始するには、fncreate コマンドを実行します。

fncreate コマンドは、FNS コンテキストが作成されるネームサービス (NIS +、NIS、ファイルベースなど) を認識します。特定のネームサービスを指定するには、下記の 483 ページの「デフォルト以外のネームサービスの指定」で説明する、fnselect コマンドを実行する必要があります。

デフォルト以外のネームサービスの指定

デフォルトでは次のようになります。

- `fncreate` を NIS+ クライアントまたはサーバーであるマシン上で実行すると、FNS 名前空間が NIS+ に設定されます。FNS NIS+ マスターサーバーとして別のマシンを指定する必要がある場合は、513 ページの「NIS+ の下での FNS の複製」 を参照してください。
- マシンが NIS クライアントの場合、名前空間は NIS で設定されます。
- マシンが上記のどちらでもない場合、名前空間はマシンで `/var/fn` ディレクトリに設定されます。基本となるネームサービスがファイルベースの場合、各マシンで `fncreate` を実行することによって `/var/fn` を作成する点は共通です。ただし、一方のマシンで `/var/fn` を作成し、NFS によってそれをエクスポートして、別のクライアントによってマウントできます。

`fnselect` コマンドを使用すると、デフォルト以外のターゲットネームサービスを明確に指定することもできます。たとえば、次のコマンドは、ターゲットネームサービスに NIS を選択します。

```
# fnselect nis
```

FNS 名前空間の作成

デフォルトポリシー、または `fnselect` によって明示的にネームサービスを指定したら、次のコマンドを使用して、FNS 名前空間を作成できます。

```
# fncreate -t org org//
```

このコマンドは、対応するネームサービス内のユーザーとホストに必要なすべてのコンテキストを作成します。

NIS+ についての考慮事項

各自のエンタープライズレベルの基本ネームサービスが NIS+ の場合は、次の点を考慮に入れてください。

NIS+ のドメインとサブドメイン

上記のコマンド構文は、ルート NIS+ ドメインの FNS 名前空間を作成します。ルート以外のドメインを指定するには、次のように、2 つのスラッシュの間にドメイン名を追加します。

```
# fncreate -t org org/sales.doc.com./
```

完全指定の `sales.doc.com.` の最後のドットに注意してください。

空間とパフォーマンスの考慮事項

`fncreate` コマンドは、`ctx_dir` ディレクトリに NIS+ のテーブルとディレクトリを作成します。`ctx_dir` ディレクトリオブジェクトは、ドメインの NIS+ の `groups_dir` と `org_dir` ディレクトリオブジェクトと同じレベルにあります。

- 大きなドメインでは、NIS+ サーバー上で必要な追加スペースが多く、大きなインストール環境では、FNS と標準 NIS+ の各テーブルに個別のサーバーを使用することによって、パフォーマンスを改善できる場合があります。FNS および NIS+ のサーバーを個別に使用する方法については、各サーバーの説明を参照してください。
- 大きいドメイン、または重要なドメインでは、FNS サービスを複製する必要があります。FNS サービスの複製方法については、512 ページの「 FNS サービスの複製」 を参照してください。

NIS+ のセキュリティ要件

`fncreate` などの FNS コマンドを実行するユーザーは、必要な NIS+ 資格を持つことが前提とされます。

環境変数 `NIS_GROUP` は、`fncreate` によって作成された NIS+ オブジェクトのグループ所有者を指定します。NIS+ オブジェクトの管理を容易にするために、`NIS_GROUP` には、そのドメインの FNS 管理を担当する NIS+ グループ名を設定してから、`fncreate` などの FNS コマンドを実行する必要があります。

デフォルトのアクセス制御権を含む NIS+ 関連属性は、コンテキストの作成後、NIS+ の管理ツールやインタフェースを使用して変更できます。FNS 複合名に対応する NIS+ オブジェクト名は、後で説明する `fnlookup` および `fnlist` によって獲得できます。

NIS についての考慮事項

`fncreate` コマンドは、FNS マップの NIS マスターサーバーとして機能する NIS システム上のスーパーユーザーが実行してください。

FNS によって使用される NIS マップは、`/var/yp/`*domainname* に保存されています。

FNS NIS マスターサーバー上のスーパーユーザーだけが、FNS コマンドを使用して FNS 情報を変更できます。

ファイルについての考慮事項

`fncreate` に `-t org` オプションを付けて FNS 名前空間を作成する場合は、`/var` が存在するファイルシステムを所有するマシン上のスーパーユーザーが、コマンドを実行してください。FNS が使用するファイルは、`/var/fn` ディレクトリに格納されています。

ユーザーのコンテキストを作成すれば、ユーザーは、各自の UNIX 資格に基いて、各自のコンテキストを変更できます。

ファイルシステム `/var/fn` は、エクスポートすることにより、別のシステムでマウントして、FNS 名前空間にアクセスできます。

FNS 名前空間の表示

名前空間を設定したら、次のコマンドを使用して表示できます。

- `fnlist` - コンテキストの内容を表示 (486 ページの「コンテキストの内容の表示」を参照)
- `fnlookup` - 複合名の割り当てを表示 (487 ページの「複合名の割り当ての表示」を参照)
- `fnattr` - 複合名の属性を表示 (487 ページの「複合名の属性の表示」を参照)

コンテキストの内容の表示

次の `fnlist` コマンドは、*name* のコンテキスト名とリファレンスの割り当てを表示します。

```
fnlist [-lvA] [name]
```

表 25–2 `fnlist` コマンドのオプション

オプション	説明
name	複合名。*name* のコンテキストでの名前の割り当てを表示する
`-v`	詳細表示。割り当てに関する詳細な情報を表示する
`-l`	指定のコンテキストで割り当てられた名前の割り当ても表示する
`-A`	`fnlist` で、権限のあるサーバーからの情報を強制的に獲得する。NIS と NIS + で、これは、ドメインのマスターサーバーになる。`-A` オプションは、基本ネームサービスがファイルベースの場合、無効である

たとえば、

初期コンテキストの名前をリスト表示する場合

```
% fnlist
```

現在の組織単位内のユーザーすべての詳細をリスト表示する場合

```
% fnlist -v user
```

ユーザー pug の service コンテキストの内容をリスト表示する場合

```
% fnlist user/pug/service
```

権限のあるサーバーからの名前と割り当てをリスト表示する場合

```
% fnlist -l -A
```

複合名の割り当ての表示

fnlookup コマンドは、指定の複合名の割り当てを表示します。

```
fnlookup [-vAL] [name]
```

表 **25–3** fnlookup コマンドのオプション

オプション	説明
name	コンテキスト名。*name* の割り当てと XFN リンクを表示する
-v	詳細表示。割り当てに関する詳細な情報を表示する
-L	名前に割り当てられている XFN リンクを表示する
-A	fnlist で、権限のあるサーバーからの情報を強制的に獲得する。NIS と NIS + で、これは、ドメインのマスターサーバーになる。-A オプションは、基本ネームサービスがファイルベースの場合、無効である

たとえば、user/ana/service/printer の割り当てを表示するには、次のコマンドを実行します。

```
# fnlookup user/ana/service/printer
```

複合名の属性の表示

fnattr コマンドは、指定の複合名の属性を表示、更新します。

たとえば、ユーザー ada に関連する属性を検索するには、次のコマンドを実行します。

```
# fnattr user/ada
```

プリンタ laser-9 に関連する属性を検索するには、次のコマンドを実行します。

```
# fnattr thisorgunit/service/printer/laser-9
```

詳細については、495 ページの「属性の処理」を参照してください。

FNS 情報の検索

fnsearch コマンドは、属性が所定の検索基準を満たす複合名以下で割り当てられたオブジェクト名を表示し、必要に応じてその属性とリファレンスを表示します。

たとえば 、

realname という属性を持つユーザーとその属性を表示するには、次のコマンドを実行します。

```
% fnsearch user realname
```

値が Ravi Chattha の属性 realname を持つユーザーをリスト表示するには、次のコマンドを実行します。

```
% fnsearch user "realname == 'Ravi Chattha'"
```

fnsearch コマンドは、共通のブール型演算子を使用します。上記の例では、二重引用符と一重引用符、2 つの等号の使用に注意してください。

名前空間の更新

名前空間を設定したら、次のコマンドを使用して、要素の追加、削除、および変更できます。

- fnbind - 複合名に新しいリファレンスを割り当てる (489 ページの「複合名へのリファレンスの割り当て」を参照)
- fnunbind - 割り当てを削除する (491 ページの「割り当ての削除」を参照)
- fncreate - 新しい組織、ユーザー、ホスト、サイト、サービスコンテキストを作成する (491 ページの「新しいコンテキストの作成」を参照)
- fncreate_fs - 新しいファイルシステムコンテキストを作成する (492 ページの「ファイルコンテキストの作成」を参照)
- fncreate_printer - 新しいプリンタコンテキストを作成する (493 ページの「プリンタコンテキストの作成」を参照)
- fndestroy - コンテキストを削除する (494 ページの「コンテキストの削除」を参照)
- fnattr - 属性を表示、作成、変更、削除する (495 ページの「属性の処理」を参照)

- `fncopy` - あるネームサービスから別のネームサービスへ FNS コンテキストと属性をコピーする (496 ページの「FNS コンテキストのコピーと変換」を参照)

FNS 管理特権

FNS システム管理は、基本となるネームサービスによって異なります。

- 「NIS+」。「NIS+」では、FNS システム管理タスクは、その権限を持つユーザーだけが実行できます。システム管理特権を付与する通常の方法では、NIS+ グループを作成し、そのドメインに必要な特権をそのグループに割り当てます。これにより、そのグループのメンバーはすべて、システム管理機能を実行できるようになります。
- 「NIS」。「NIS」では、FNS 管理タスクは、NIS マスターサーバー上の `root` で実行しなければなりません。
- 「ファイル」。「ファイルベース」のネーミングシステムでは、FNS 管理タスクは、`/var/fn` ディレクトリへの `root` アクセス権を持つユーザーが実行しなければなりません。

ユーザーが各自のサブコンテキストを変更できるかどうかは、基本となるネームサービスによって異なります。

- 「NIS+」。「NIS+」では、ユーザーのコンテキストと関連サブコンテキストは、ユーザーが所有します。NIS+ ポリシーでログインした場合、適切な資格と特権を持つユーザーは、`fncreate`、`fnbind`、`fnunbind` などのコマンドを使用して、各自のコンテキストを変更できます。
- 「NIS」。「NIS」では、ユーザーはどの FNS データも変更できません。NIS マスターサーバーへの `root` アクセス権を持つユーザーだけが、FNS データを変更できます。
- 「ファイル」。「ファイルベース」のネーミングシステムでは、ユーザーは独自のコンテキストを所有します。標準の UNIX アクセス制御が FNS ファイルに適用されます。

複合名へのリファレンスの割り当て

`fnbind` コマンドは、既存のリファレンス (名前) を新しい複合名に割り当てるために使用されます。

`fnbind -r [-s][-v][-L]` *name* `[-O|-U]` *newname reftype addrtype* `[-c|-x]` *address*

表 25–4 `fnbind` コマンドのオプション

オプション	説明
name	すでに存在している複合名

表 **25–4** fnbind コマンドのオプション　(続き)

オプション	説明
newname	新しい割り当ての複合名
addrtype	使用するアドレスのタイプ。onc_cal_str のようにアプリケーション特定
address	使用するアドレスの内容。たとえば、tsvi@altair
reftype	使用するリファレンスのタイプ。one_calendar のようにアプリケーション特定
-s	すでに割り当てられている場合でも、*newname* に割り当てられる。これは、以前の *newname* の割り当てを置き換える。-s を指定しないと、*newname* がまだ割り当てられていない場合、fnbind は失敗する
-v	*newname* に割り当てられるリファレンスを表示する
-L	*oldname* を使用して XFN リンクを作成し、それを *newname* に割り当てる
-r	コマンド行引数によって作成されたリファレンスに *newname* を割り当てる
-c	入力された形式で *address* の内容を格納する。XDR コードは使用しない
-x	*address* を XDR コードに変換しないで、16 進文字列に変換する
-O	識別子の形式は FN_ID_ISO_OID_STRING。これは、ASN.1 のドットで区切られた整数リスト文字列である
-U	識別子の形式は FN_ID_DCE_UUID。これは、文字列形式での DCE UUID である

次に例を示します。

ユーザー jamal にカレンダ割り当てを追加する場合

```
# fnbind -r user/jamal/service/calendar onc_calendar onc_cal_str
 jamal@cygnus
```

org//service/Sparc-4 の既存の割り当てを、org//service/printer の割り当てによって置換する場合

```
# fnbind -s org//service/printer org//service/Sparc-4
```

site/bldg-5/service/printer のリファレンスを user/ando/service/printer にコピーする場合

```
# fnbind site/bldg-5/service/printer user/ando/service/printer
```

シンボリックリンクを使用して、site/bldg-5/service/printer のリファレンスを user/ando/service/printer に割り当てる場合

```
# fnbind -L site/bldg-5/service/printer user/ando/service/printer
```

staff@altair が onc_cal タイプのリファレンスであり、かつタイプ onc_cal _str のアドレスである場合、thisens/service/calendar の名前をアドレス staff@altair に割り当てる場合

```
# fnbind -r thisens/service/calendar onc_calendar onc_cal_str staff@altair
```

コマンド行 *address* によって作成されたリファレンスとして *newname* を割り当てる場合

```
# fnbind -r [-sv] newname [-O|-U] reftype {[-O|-U] addrtype [-c|-x] address}
```

割り当ての削除

fnunbind コマンドは、割り当てを削除するために使用されます。

たとえば、user/jsmith/service/calendar の割り当てを削除するには、次のコマンドを実行します。

```
# fnunbind user/jsmith/service/calendar
```

新しいコンテキストの作成

fncreate コマンドは、コンテキストを作成するために使用されます。

```
fncreate -t context [-f file] [-o] [-r reference] [-s] [-v] [-D] name
```

表 25–5 fncreate コマンドオプション

オプション	説明
-t *context*	*context* タイプのコンテキストを作成する。*context* タイプは、org、hostname 、host、username、user、service、fs、 site、nsid、generic
-f *file*	入力ファイルを使用して、コンテキストを作成するユーザーとホストを一覧表示する
-r *reference*	リファレンスのタイプ。-r *reference* オプションは、-t generic とともにしか使用できない
name	複合名
-o	*name* によって識別されるコンテキストだけを作成する
-s	既存の割り当てすべてを上書き (置換) する。-s を使用しないと、名前がすでに割り当てられている場合、fncreate は失敗する
-D	各コンテキストの作成時に、そのコンテキストと、対応するテーブル、ディレクトリ、ファイルに関する情報を表示する

表 **25–5** fncreate コマンドオプション　(続き)

オプション	説明
-v	詳細表示。各テキストの表示時にその情報を表示する

たとえば:

ルート組織のコンテキストとサブコンテキストを作成する場合

```
# fncreate -t org org//
```

ホスト deneb のコンテキストとサブコンテキストを作成する場合

```
# fncreate -t host host/deneb
```

コンテキスト、サービス、ファイルサブコンテキストを作成して、ユーザー sisulu のカレンダ割り当てを追加する場合

```
# fncreate -t user user/sisulu
# fnbind -r user/sisulu onc_calendar onc_cal_str sisulu@deneb
```

sales 組織のサイトコンテキストを作成する場合

```
# fncreate -t site org/sales/site/
```

サイトコンテキストは、ドットで区切られた右から左へという順番の名前によって、階層名前空間をサポートします。これにより、地理上のカバレージ関係によってサイトを区分化できます。たとえば、サイトコンテキスト alameda とサイトサブコンテキスト bldg-6.alameda を作成するには、次のコマンドを実行します。

```
# fncreate -t site org/sales/site/alameda
# fncreate -t site org/sales/site/bldg-6.alameda
```

ファイルコンテキストの作成

- fncreate_fs コマンドは、コマンド行から入力された割り当て記述によって、組織とサイトのファイルコンテキストを作成します。

  fncreate_fs [-r] [-v] *name* [*options*] [*mount*]

- fncreate_fs コマンドは、入力ファイルから提供された割り当ての記述によって、組織とサイトのファイルコンテキストを作成します。

  fncreate_fs [-r] [-v] -f *file name*

表 **25–6** fncreate_fs コマンドのオプション

オプション	説明
name	ファイルコンテキスト名

表 25–6 fncreate_fs コマンドのオプション　(続き)

オプション	説明
options	マウントオプション
mount	マウントする位置
-f *file*	入力ファイル
-v	詳細。作成中のコンテキストに関する情報を表示
-r	コンテキスト *name* の割り当てを、入力で指定された割り当てによって置換する

次に例を示します。

FNS サーバー server4 の /export/data パスに割り当てられた sales 組織に関して、ファイルシステムコンテキスト data を作成する場合

```
# fncreate_fs org/sales/fs/data server4:/export/data
```

2 つの異なるサーバーにマウントされた buyers および buyers/orders という名前の sales 組織に関して、ファイルシステムコンテキストの階層を作成する場合

```
# fncreate_fs org/sales/fs/buyers server2:/export/buyers
# fncreate_fs org/sales/fs/buyers/orders server3:/export/orders
```

input_a という名前の入力ファイルによって指定されたサーバーとパスに割り当てられた sales 組織に関して、leads という名前のファイルシステムコンテキストを作成する場合

```
# fncreate_fs -f input_a org/sales/fs/leads
```

入力ファイル形式については、fncreate_fs(1M) のマニュアルページを参照してください。

プリンタコンテキストの作成

fncreate_printer コマンドは、組織、ユーザー、ホスト、サイトの各コンテキストのプリンタコンテキストを作成します。プリンタコンテキストは、対応する複合名のサービスコンテキストのもとで作成されます。

```
fncreate_printer [-vs] name printer [prntaddr]

fncreate_printer [-vs] [-f [file]] name
```

表 **25–7** fncreate_printer コマンドのオプション

オプション	説明
name	プリンタの組織、ホスト、ユーザー、またはサイトの名前
printer	プリンタ名
prntaddr	プリンタアドレス。<addresstype>=<address> という形式をとる
-f *file*	指定の ファイル を、作成するプリンタリストへの入力として使用。入力ファイルは /etc/printers.conf ファイルの形式をとる。プリンタ名も -f ファイル も指定されていない場合、fncreate_printer は、fncreate_printer がデフォルト入力ファイルとして実行されるマシン上の /etc/printer.conf ファイルを使用する
-s	既存のアドレスを、同じアドレスタイプによって置換する
-v	詳細。割り当ての詳細を表示する

たとえば :

fncreate_printer が実行されるマシンの /etc/printers.conf ファイルに示されたプリンタに基いて sales 組織のプリンタを作成する場合

```
# fncreate_printer -s org/sales/
```

マシン altair が、プリンタ Sparc-5 のサーバーであると想定します。ユーザー nguyen に対して、実際に Sparc-5 プリンタである invoices という名前のプリンタを作成する場合、次のように入力します。

```
# fncreate_printer user/nguyen invoices bsdaddr=altair,Sparc-5
```

プリンタを階層構造で構成することもできます。たとえば、fncreate_printer コマンドは、プリンタ color、color/inkjet、color/Sparc のプリンタコンテキストを作成できます。コンテキストは次のようになります。

```
org/doc.com/service/printer/color
org/doc.com/service/printer/color/inkjet
org/doc.com/service/printer/color/Sparc
```

上記のコンテキストを作成するには、次のコマンドを実行します。

```
# fncreate_printer org/doc.com color bsdaddr=colorful,color
# fncreate_printer org/doc.com color/inkjet bsdaddr=colorjet,inkjet
# fncreate_printer org/doc.com color/Sparc bsdaddr=colorprt,Sparc
```

コンテキストの削除

fndestroy コマンドは、空のコンテキストを削除するために使用されます。

たとえば、ユーザー patel の service コンテキストを削除するには、次のコマンドを実行します。

```
# fndestroy user/patel/service
```

属性の処理

fnattr コマンドは、名前に関連する属性の追加、削除、または変更に使用できます。変更は一度に 1 つずつ行うことも、同じコマンド内で何回かバッチ処理することもできます。

- fnattr [-l] *namename* の属性をリスト表示
- fnattr *name* -a-s -U -O *attrib values* 属性の追加
- fnattr *name* -m -O -U *attrib oldvalue newvalue* 属性の変更
- fnattr *name* -d -O | -U [*values attrib*] 属性の削除

表 25–8 fnattr コマンドのオプション

オプション	説明
name	複合名
attrib	属性の識別子
values	1 つまたは複数の属性値
oldvalue	新しい値によって置換される属性値
newvalue	古い値を置換する属性値
-a	属性の追加
-d	属性の削除
-l	属性のリスト表示
-m	属性の変更
-s	指定された属性の新しい値によって、すべての古い属性値を置換する
-O	識別子の形式は FN_ID_ISO_OID_STRING。これは、ASN.1 のドットで区切られた整数リスト文字列である
-U	識別子の形式は FN_ID_DCE_UUID。これは、文字列形式での DCE UUID である

たとえば:

ユーザー名 rosa に関連する属性すべてを表示する場合

```
# fnattr user/rosa
```

ユーザー uri に関連する size 属性を表示する場合

```
# fnattr user/uri/ size
```

devlin という名前のユーザーについて、値が small の属性 shoesize を追加するには、hatsize 属性を削除して、dresssize 属性の値を 12 から 8 に変更します。

```
# fnattr user/devlin -a shoesize small -d hatsize -m dresssize 12 8
```

グローバル名前空間のフェデレート

NIS+ または NIS は、DNS や X.500 などのグローバルネームサービスにフェデレートできます。

NIS+ または NIS の名前空間を DNS または X.500 のもとでフェデレートするには、まず NIS+ 階層または NIS ドメインのルートリファレンスを獲得する必要があります。

グローバルネームサービスでは、ルートリファレンスは、「次のネーミングシステムリファレンス」として知られています。これは、このリファレンスが、DNS ドメインまたは X.500 エントリの下にある次のネーミングシステムを指すためです。NIS+ または NIS をグローバルネームサービスによってフェデレートするには、そのグローバルサービスに、ルートリファレンス情報を追加します。

ルートリファレンス情報をグローバルサービスに追加すると、各自の NIS+ 階層または NIS ドメインの外部のクライアントは、その NIS+ 階層または NIS ドメインのコンテキストにアクセスして、操作を実行できます。外部 NIS+ クライアントは、未認証 NIS+ クライアントとしてその階層にアクセスします。

たとえば:

NIS+ が、DNS ドメイン doc.com. の下でフェデレートされる場合は、次のコマンドを使用して NIS+ エンタープライズのルートをリスト表示できます。

```
# fnlist .../doc.com/
```

NIS+ が、X.500 エントリ /c=us/o=doc の下でフェデレートされる場合は、次のコマンドを使用して NIS+ エンタープライズのルートをリスト表示できます。

```
# fnlist .../c=us/o=doc/
```

どちらの例でも、最後にスラッシュが必要であることに注意してください。

FNS コンテキストのコピーと変換

fncopy コマンドは、FNS のコンテキストと属性を新しい FNS コンテキストにコピー、または変換するために使用できます。

-i および -o オプションを使用すると、あるエンタープライズレベルのネームサービスに基く FNS コンテキストを、異なるネームサービスに基くコンテキストにコピーできます。たとえば、NIS の上部で実行される FNS インストール環境があって、NIS サービスを NIS+ にアップグレードする場合、fncopy を使用すると、NIS+ を使って新しいコンテキストを作成できます。

以下の点に注意してください。

- 古いコンテキストをコピー中の新しい FNS コンテキストが、そのターゲットネームサービスにすでに存在する場合は、新しいコンテキストと割り当てだけがコピーされる。これらのコンテキストは上書きも変更もされない
- fncopy は、リンクに従わないが、名前に割り当てられた FNS リンクを、新しいコンテキストの名前空間にコピーする

表 25–9 fncopyコマンドオプション

オプション	説明
-i *oldservice*	エンタープライズレベルの古い (入力) 基本ネームサービス。たとえば、-i nis は、古いサービスが NIS であることを指定する。指定できる値は、files、nis、nisplus
-o *newservice*	エンタープライズレベルの新しい (出力) ネームサービス。たとえば、-o nisplus は、新しいサービスが NIS+ であることを指定する。指定できる値は、files、nis、nisplus
-f *filename*	コピーされる FNS コンテキストのリストが入っているテキストファイル。-i および -o オプションを指定しない場合、コンテキストは、グローバル名を使用する識別子でなければならない
oldcontext	コピーされるコンテキスト名
newcontext	作成先またはコピー先のコンテキスト名

たとえば、doc.com プリンタコンテキスト (およびサブコンテキスト) と割り当てを orgunit/east/doc.com にコピーするには、次のコマンドを実行します。

```
# fncopy .../doc.com/service/printer .../doc.com/orgunit/east/service/printer
```

ファイル user_list に指定された NIS FNS ユーザーのコンテキストを、orgunit west/doc.com の NIS+ FNS ユーザーのコンテキストにコピーするには、次のコマンドを実行します。

```
# fncopy -i nis -o nisplus -f /etc/user_list thisorgunit/user org/doc.com/user
```

名前空間ブラウザのプログラムの例

この節のプログラミング例は、次の操作を実行するための XFN API の使用法を示しています。

コンテキストに割り当てられた名前のリスト表示

次の例は、コンテキストをリスト表示するための XFN 操作を示しています。

```
#include <stdio.h>
#include <xfn/xfn.h>
#include <string.h>
#include <stdlib.h>
/*
 このルーチンは与えられたコンテキスト (ctx_name) の下で
 割り当てられた名前のリストを返す。
 ctx_name の例としては "user"、"thisorgunit/service"、
    host/alto/service、user/jsmit/service/calendar などがある
*/
typedef struct fns_listing {
    char *name;
    struct fns_listing *next;
} fns_listing;
fns_listing *
fns_list_names(const char *ctx_name)
{
    FN_status_t *status;
    FN_ctx_t *initial_context;
    FN_composite_name_t *context_name;
    FN_namelist_t *name_list;
    FN_string_t *name;
    unsigned int stat;
    fns_listing *head = 0, *current, *prev;
    int no_names = 0;
    status = fn_status_create();
    /* 初期コンテキストの獲得 */
    initial_context = fn_ctx_handle_from_initial(0, status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to obtain intial context\n");
        return (0);
    }
    context_name = fn_composite_name_from_str((unsigned char *)
                  ctx_name);
```

```
    /* FNS によるリスト名の呼び出し */
    name_list = fn_ctx_list_names(initial_context, context_name,
                status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to list names\n");
        return (0);
    }
    /* 名前を個々に呼び出す */
    while (name = fn_namelist_next(name_list, status)) {
        no_names++;
        current = (fns_listing *) malloc(sizeof(fns_listing));
        current->name = (char *)
         malloc(strlen((char *) fn_string_str(name, &stat)) + 1);
        strcpy(current->name, (char *) fn_string_str(name, &stat));
        current->next = 0;
        if (head) {
            prev->next = current;
            prev = current;
        } else {
            head = current;
            prev = current;
        }
        fn_string_destroy(name);
    }
    fn_namelist_destroy(name_list);
    fn_status_destroy(status);
    fn_ctx_destroy(initial_context);
    return (head);
```

割り当ての作成

例 25–1 は、割り当ての作成方法を示しています。

例 25–1 割り当ての作成

```
#include <stdio.h>
#include <xfn/xfn.h>
#include <string.h>
/*
 このルーチンは "name" によって提供される名前を使用して
 割り当てを作成する。提供される名前のリファレンスのタイプは
"reference_type" で、アドレスのタイプは "address_type" である。
 関数の使用例として、
     fns_create_bindings(
     "user/jsmith/service/calendar",
     "onc_calendar",
     "onc_cal_str",
     "jsmith&calserver");
 がある
*/
int fns_create_bindings(
 char *name,
```

例 **25–1** 割り当ての作成　(続き)

```
 char *reference_type,
 char *address_type,
 char *data)
{
    int return_status;
    FN_composite_name_t *binding_name;
    FN_identifier_t ref_id, addr_id;
    FN_status_t *status;
    FN_ref_t *reference;
    FN_ref_addr_t *address;
    FN_ctx_t *initial_context;
    /* 初期コンテキストの獲得 */
    status = fn_status_create();
    initial_context = fn_ctx_handle_from_initial(0, status);
    /* エラーメッセージ状態のチェック */
    if ((return_status = fn_status_code(status)) != FN_SUCCESS) {
        fprintf(stderr, "Unable to obtain the initial context\n");
        return (return_status);
    }
    /* プリンタ名に付ける複合名の獲得 */
    binding_name = fn_composite_name_from_str((unsigned char *) name);
    /* アドレスのコンストラクト */
    addr_id.format = FN_ID_STRING;
    addr_id.length = strlen(address_type);
    addr_id.contents = (void *) address_type;
    address = fn_ref_addr_create(&addr_id,
        strlen(data), (const void *) data);
    /* リファレンスのコンストラクト */
    ref_id.format = FN_ID_STRING;
    ref_id.length = strlen(reference_type);
    ref_id.contents = (void *) reference_type;
    reference = fn_ref_create(&ref_id);
    /* リファレンスにアドレスを追加する */
    fn_ref_append_addr(reference, address);

    /* 割り当ての作成 */
    fn_ctx_bind(initial_context, binding_name, reference, 0, status);
    /* エラー状態をチェックして返す */
    return_status = fn_status_code(status);
    fn_composite_name_destroy(binding_name);
    fn_ref_addr_destroy(address);
    fn_ref_destroy(reference);
    fn_ctx_destroy(initial_context);
    return (return_status);
}
```

オブジェクト属性のリスト表示と処理

次の例は、オブジェクト属性をリスト表示して処理する方法を示しています。

オブジェクト属性のリスト表示

次の例は、オブジェクト属性をリスト表示する方法を示しています。

```
#include <stdio.h>
#include <xfn/xfn.h>
/*
 このルーチンは名前付きオブジェクトに関連付けられたすべての属性を
 標準出力に出力する。関数の使用例として
     fns_attr_list("user/jsmith"); や
    fns_attr_list("thisorgunit/service/printer/color"); がある
*/
void fns_attr_list(const char *name)
{
    FN_composite_name_t *name_comp;
    const FN_identifier_t *identifier;
    FN_attribute_t *attribute;
    const FN_attrvalue_t *values;
    char *id, *val;
    FN_multigetlist_t *attrset;
    void *ip;
    FN_status_t *status;
    FN_ctx_t *initial_context;
    name_comp = fn_composite_name_from_str((unsigned char *) name);
    status = fn_status_create();
    /* 初期コンテキストの獲得 */
    initial_context = fn_ctx_handle_from_initial(0, status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to obtain intial context\n");
        return;
    }
    /* 全属性の獲得 */
    attrset = fn_attr_multi_get(initial_context, name_comp, 0, 0,
            status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to obtain attributes\n");
        return;
    }
    /* 全属性の表示 */
    while (attribute = fn_multigetlist_next(attrset, status)) {
        identifier = fn_attribute_identifier(attribute);
        switch(identifier->format) {
        case FN_ID_STRING:
            id = (char *) malloc(identifier->length + 1);
            memcpy(id, identifier->contents, identifier->length);
            id[identifier->length] = '\0';
            printf("Attribute Identifier: %s", id);
            free(id);
            break;
        default:
            printf("Attribute of non-string format\n\n");
            continue;
        }
        for (values = fn_attribute_first(attribute, &ip);
         values != NULL;
```

```
         values = fn_attribute_next(attribute, &ip)) {
            val = (char *) malloc(values->length + 1);
            memcpy(val, values->contents, values->length);
            val[values->length] = '\0';
            printf("Value: %s", val);
            free(val);
        }
        fn_attribute_destroy(attribute);
        printf("\n");
    }
    fn_multigetlist_destroy(attrset);
    fn_ctx_destroy(initial_context);
    fn_status_destroy(status);
    fn_composite_name_destroy(name_comp);
}
```

オブジェクト属性の追加、削除、変更

次の例は、オブジェクト属性の追加、削除、変更を示しています。

```
#include <stdio.h>
#include <xfn/xfn.h>
/*
 このルーチンは名前付きオブジェクトに関連付けられた属性を変更する。
 変更としては、
 FN_ATTR_OP_ADD
 FN_ATTR_OP_ADD_EXCLUSIVE
 FN_ATTR_OP_REMOVE
 FN_ATTR_OP_ADD_VALUES
 FN_ATTR_OP_REMOVE_VALUES
 が有効である。この関数は属性値が文字列であることを前提とする。
 関数の使用例として、以下に "James Smith" という値を持つ識別子 "realname"
 の属性を、ユーザーオブジェクト "user/jsmith" に追加する。
       fns_attr_modify(
            "user/jsmith",
            "realname",
            "James Smith",
            FN_ATTR_OP_ADD);
 次の関数は識別子 "location" の属性をプリンタオブジェクト
 "thisorgunit/service/printer/color" から削除する。
       fns_attr_modify(
           "thisorgunit/service/printer/color",
            "location",
          NULL,
          FN_ATTR_OP_REMOVE);
*/
static const char *attr_id_syntax = "fn_attr_syntax_ascii";
void fns_attr_modify(const char *name,
     const char *attr_id,
     const char *attr_value,
     unsigned int operation)
{
      FN_composite_name_t *name_comp;
```

```
        FN_identifier_t identifier, syntax;
        FN_attrvalue_t *values;
        FN_attribute_t *attribute;
        FN_status_t *status;
        FN_ctx_t *initial_context;
        name_comp = fn_composite_name_from_str((unsigned char *) name);
        status = fn_status_create();
         /* 初期コンテキストの獲得 */
         initial_context = fn_ctx_handle_from_initial(0, status);
        if (!fn_status_is_success(status)) {
             fprintf(stderr, "Unable to obtain intial context\n");
             return;
        }
        /* 追加する属性の作成 */
        /* 最初に識別子 */
        identifier.format = FN_ID_STRING;
        identifier.length = strlen(attr_id);
        identifier.contents = (void *) strdup(attr_id);
        /* 次に構文 */
        syntax.format = FN_ID_STRING;
        syntax.length = strlen(attr_id_syntax);
        syntax.contents = (void *) strdup(attr_id_syntax);
        /* 次に属性値 */
        if (attr_value) {
             values = (FN_attrvalue_t *)      malloc(sizeof(FN_attrvalue_t));
             values->length = strlen(attr_value);
             values->contents = (void *) strdup(attr_value);
        } else
               values = NULL;
        /* 次に属性の作成 */
        attribute = fn_attribute_create(&identifier, &syntax);
        /*次に属性値の追加 */
        if (values)
             fn_attribute_add(attribute, values, 0);

        /* XFN オペレーションの実行 */
        fn_attr_modify(initial_context, name_comp, operation, attribute, 0,
        status);
        if (!fn_status_is_success(status))
             fprintf(stderr, "Unable to perform attribute operation\n");
        fn_ctx_destroy(initial_context);
        fn_status_destroy(status);
        fn_composite_name_destroy(name_comp);
        fn_attibute_destroy(attribute);
        free(identifier.contents);
        free(syntax.contents);
        if (values) {
             free(values->contents);
             free(values);
 ]
]
```

コンテキスト内のオブジェクトの検索

次の例は、特定の属性識別子と値によって、コンテキスト内のオブジェクトを検索する方法を示しています。

```
#include <stdio.h>
#include <xfn/xfn.h>
#include <string.h>
#include <stdlib.h>
/*
 このルーチンは、指定された属性識別子と値を持つ
 コンテキスト内のオブジェクトを検索します。
*/
typedef struct fns_search_results {
    char *name;
    struct fns_search_results *next;
} fns_search_results;
static const char *attr_id_syntax = "fn_attr_syntax_ascii";
fns_search_results *
fns_attr_search(const char *name,
    const char *attr_id,
    const char *attr_value)
{
    FN_status_t *status;
    FN_ctx_t *initial_context;
    FN_composite_name_t *context_name;
    FN_searchlist_t *search_list;
    FN_string_t *search_name;
    FN_attribute_t *attribute;
    FN_attrset_t *attrset;
    FN_identifier_t identifier, syntax;
    FN_attrvalue_t *values;
    unsigned stat;
    fns_search_results *head = 0, *current, *prev;
    int no_names = 0;
    context_name = fn_composite_name_from_str((unsigned char *) name);
    status = fn_status_create();
    initial_context = fn_ctx_handle_from_initial(0, status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to obtain intial context\n");
        return (0);
    }
    /* 検索する属性を持つ attrset のコンストラクト */
    /* 最初に識別子 */
    identifier.format = FN_ID_STRING;
    identifier.length = strlen(attr_id);
    identifier.contents = (void *) strdup(attr_id);
    /* 次に構文 */
    syntax.format = FN_ID_STRING;
    syntax.length = strlen(attr_id_syntax);
    syntax.contents = (void *) strdup(attr_id_syntax);
    /* 次に属性値 */
    values = (FN_attrvalue_t *) malloc(sizeof(FN_attrvalue_t));
    values->length = strlen(attr_value);
    values->contents = (void *) strdup(attr_value);
```

```
    /* 次に属性の作成 */
    attribute = fn_attribute_create(&identifier, &syntax);
    /* 次に属性値の作成 */
    fn_attribute_add(attribute, values, 0);
    /* 次に attrset の作成と属性の追加 */
    attrset = fn_attrset_create();
    fn_attrset_add(attrset, attribute, 0);
    search_list = prelim_fn_attr_search(initial_context,
     context_name, attrset, 0, 0, status);
    if (!fn_status_is_success(status)) {
        fprintf(stderr, "Unable to list names\n");
        return (0);
    }
    while (search_name = prelim_fn_searchlist_next(search_list,
     0, 0, status)) {
        no_names++;
        current = (fns_search_results *)
    malloc(sizeof(fns_search_results));
        current->name = (char *)
         malloc(strlen((char *) fn_string_str(search_name, &stat)) + 1);
        strcpy(current->name, (char *) fn_string_str(search_name, &stat));
        current->next = 0;
        if (head) {
            prev->next = current;
            prev = current;
        } else {
            head = current;
            prev = current;
        }
        fn_string_destroy(search_name);
    }
    fn_searchlist_destroy(search_list);
    fn_status_destroy(status);
    fn_ctx_destroy(initial_context);
    fn_attrset_destroy(attrset);
    fn_attribute_destroy(attribute);
    free(identifier.contents);
    free(syntax.contents);
    free(values->contents);
    free(values);
    return (head);
}
```

FNS の設定 - 概要

Solaris オペレーティング環境ソフトウェアのインストール後に次の作業を実行して、FNS を設定する必要があります。

1. サーバーが FNS を取り扱えることを確認します。506 ページの「リソース条件の決定」 を参照してください。

2. FNS 用の名前空間を準備します。507 ページの「FNS 用の名前空間の準備」 を参照してください。
3. FNS 名前空間のコンテキストを設定します。次の 2 つの方法があります。
   a. 1 つのプロセスで、すべてのコンテキストをグローバルに作成します。510 ページの「グローバルな FNS の名前空間コンテキストの作成」 を参照してください。
   b. FNS コンテキストを個別に作成します。
4. FNS の複製サーバーを設定します。512 ページの「 FNS サービスの複製」 を参照してください。

組織の大きさによっては、FNS の設定が完了するまでに数時間と、名前空間の準備にもある程度の時間が必要になります。

リソース条件の決定

インストールの手順に進む前に、FNS をサポートしているサーバーに十分なメモリーとディスク領域があることを最初に確認する必要があります。企業レベルのネームサービス (NIS+、NIS、ファイル) で必要な領域の他に FNS 用の領域が必要です。

一般的に確実な方法として、ユーザーとホストごとに、約 17K バイトのディスク領域とスワップ領域が必要になります。このディスク領域が配置される場所と、その計算方法は、使用している企業レベルのネームサービスによって異なります。

- 「NIS+」。ドメインまたはサブドメインに対して FNS サーバーとして機能するマシンに、ディスク領域が必要です。NIS+ 環境では、FNS の `ctx_dir` ディレクトリを提供するサーバーは、`org_dir` など標準の NIS+ ディレクトリを提供するのと同じサーバーである必要はありません。サーバーの負荷を均等にするために、大規模な構成で使用しているサイトの多くでは、NIS+ と FNS のサーバーに別個のマシンを使用しています。NIS+ 環境で FNS サーバーに必要な領域は、サーバーがネームサービスを提供するドメインまたはサブドメイン内のユーザーとホストの数で決まります。
- 「NIS」。ドメインに対して FNS サーバーとして機能するマシンにディスク領域が必要です。NIS 環境では、FNS をホストするサーバーは NIS をホストするのと同じサーバーである必要はありません。サーバーの負荷を均等にするために、大規模な構成で使用しているサイトの多くでは、NIS と FNS のサーバーに別個のマシンが使用されています。NIS 環境で FNS サーバーに必要な領域は、ドメイン内のユーザーとホストの数で決まります。
- 「ファイル」。企業レベルのネームサービスがファイルのときは、FNS に必要なディスク領域は、`/var/fn` をマウントするマシンの`/etc/users` と `/etc/hosts` のファイル内のユーザーとホストの数で決まります。マシンごとに `/var/fn` ディレクトリがある場合には、必要な領域は各マシンのユーザーとホストのファイルで決まります。`/var/fn` がマシンにマウントされ、FNS によってネットワーク上の

残りのマシンにエクスポートされる場合には、/var/fn を提供するマシンに必要な領域は、マシンの /etc/users と /etc/hosts のファイル内のユーザーとホストの数で決まります。

たとえば、1,200 のユーザーとホストを持つ NIS+ ドメインで FNS 環境をサポートするには、以下が必要になります。

- 基礎となる名前空間で必要な領域 (NIS+、NIS、ファイル) の他に、少なくとも 20M バイトのディスク領域
- FNS 用に 40M バイト のスワップ領域

FNS 用の名前空間の準備

この節では、fncreate を実行して FNS コンテキストを設定する前の準備について説明します。準備は該当する企業レベルのネームサービスによって異なります。以下を参照してください。

FNS 用の名前空間の準備 — タスクマップ

表 **25–10** FNS 用の名前空間の準備

作業	説明	指示の参照先
FNS 用の名前空間の準備	NIS マップへのファイルの変換	『*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「NIS マップに関する作業」
FNS 用の名前空間の準備	NIS サービスの準備	509 ページの「FNS 用の NIS サービスの準備」
FNS 用の名前空間の準備	ファイルベースのネーミングの準備	509 ページの「FNS 用のファイルを使用したネームサービスの準備」

▼ FNS 用の NIS+ サービスの準備

FNS 名前空間を設定する前に、以下の作業を行います。

1. **NIS+** ドメインが正確に設定されていることを確認します。

   NIS+ ドメインとそのサブドメインは、FNS を構成する前に設定されていなければなりません。つまり hosts と passwd など NIS+ の標準のテーブルが既に存在し、生成されている必要があります。

2. ドメインの hosts.org_dir と passwd.org_dir のテーブルが、すべてのホスト名とユーザー名で生成されていることを確認します。

   niscat と nismatch のコマンドを使用して、これらテーブルの内容を確認できます。

3. NIS_GROUP 環境変数を、**FNS** オブジェクトを管理するグループの名前に設定します。

   この変数を最初に設定しないと、fncreate コマンドで FNS 設定を完了できません。fncreate コマンドがユーザーとホストのコンテキストを作成するとき、それらはコマンドを実行したシステム管理者ではなく、そのホストとユーザーの所有となります。NIS_GROUP の設定によって、グループのメンバーになっているシステム管理者は、オブジェクトを所有していなくてもコンテキストを修正できます。

   Cシェルの場合、次のように NIS_GROUP に fns_admins.doc.com を設定します。

   ```
   rootmaster# setenv NIS_GROUP fns_admins.doc.com
   ```

4. 必要に応じて、**NIS+** マスターサーバー以外のマシンで **FNS** を実行するように指定します。

   FNS が使用する、すべての NIS+ オブジェクトは、NIS+ ドメインの ctx_dir ディレクトリの下に保管されます。5,000 以上のユーザーやホストを有する大規模なドメインでは、FNS が使用する ctx_dir が、groups_dir のような標準の NIS+ ディレクトリをサポートするサーバーとは別のサーバーによってサポートされるようにしてください。これは必須ではありませんが、推奨されています。別個のサーバーを使用することで、1 つのサーバーに過剰な負荷がかからなくなります。またこれによって、FNS による NIS+ の使用の管理と NIS+ 自体の管理を分離できます。

   ドメインに対する NIS+ マスターサーバーではないマシンによって FNS が提供されるように指定するには、ドメインに対する FNS ホストとして機能するマシン上に ctx_dir ディレクトリオブジェクトを手作業で作成する必要があります。この手順を省略すると、FNS はドメインの NIS+ ルートマスターサーバーにインストールされます。

   FNS のマスターサーバーになるマシンを指定するには次のようにします。

   a. **NIS+** ドメインに ctx_dir ディレクトリを作成します。

      たとえば、doc.com ドメイン内で fns_server と名前のついたマシン上に ctx_dir ディレクトリを作成するには、ドメインのマスターサーバーで次のコマンドを実行します。ドメイン名の最後にドットが付いていることに注意してください。

      ```
      nismaster# nismkdir -m fns_server ctx_dir.doc.com.
      ```

      nismkdir を使用した NIS+ ディレクトリの作成方法の詳細については、336 ページの「nismkdir コマンド」 を参照してください。

注 – 「サブドメイン」用の FNS の ctx_dir を作成する場合、ctx_dir を提供する FNS サーバーとして指定するマシンはサブドメイン内に存在する必要があります、親ドメイン内のマシンは指定できません。これとは対照的に、サブドメインの NIS+ のマスターサーバーは常に、それが機能する 1 つ上のドメインに存在します。つまり、NIS+ のサブドメインに対して FNS を構成しているときに、NIS+ と FNS の両方で同じサーバーを使用する場合には、そのサーバーはサブドメインの上のドメインに存在します。しかし、NIS+ と FNS で異なるサーバーを使用する場合には、NIS+ のマスターサーバーはその上のドメインに存在し、FNS サーバーはそれが機能するサブドメインに存在します。

b. nisls コマンドを使用して、ctx_dir ディレクトリが作成されたことを検証します。

```
rootmaster # nisls doc.com.ctx_dir
```

c. nisping を実行して、ディレクトリでチェックポイントを実行します。

```
# /usr/lib/nis/nisping -C ctx_dir.doc.com.
```

▼ FNS 用の NIS サービスの準備

FNS 名前空間を設定する前に、以下の作業を行います。

- hosts.byname、user.byname、printer.conf.byname のマップが完全、正確、最新であることを確認します。

注 – 他の任意の NIS マップに対して異なるマスターサーバーを割り当てるのと同じ手順で、FNS マップに異なるマスターサーバーを割り当てることができます。

FNS 用のファイルを使用したネームサービスの準備

「ファイルを使用した」ネームサービスとは、NIS+ または NIS ではなく、/etc ファイルからデータを得るネームサービスのことです。

/var/fn ディレクトリをマシンごとにインストールする場合には、一般的に次の手順をマシンごとに実行する必要があります。1 つのマシンから /var/fn ディレクトリのマウントとエクスポートをする場合には、次の手順を /var/fn をエクスポートするマシンで実行する必要があります。

- /etc/hosts と /etc/passwd のファイルが完全で、すべてのユーザー名とホスト名を含むことを確認してください。

グローバルな FNS の名前空間コンテキストの作成

この節では、企業または NIS+ ドメインに対する名前空間をグローバルに作成する方法を説明します。

FNS の名前空間は、`fncreate` コマンドを使用して作成されます。

```
# fncreate -t org org//
```

あるいは、次のコマンドを使用します。

```
# fncreate -t org org/domain/
```

この場合 *domain* 名は、NIS+ドメイン名またはサブドメイン名です。

`fncreate` コマンドは、指定された編成と、そのすべてのサブコンテキスト用のデフォルトのコンテキストを作成します。これには編成内のユーザーとホストに対するコンテキストとサブコンテキストが含まれます。

グローバルな FNS の名前空間コンテキストの作成 — 作業マップ

表 25–11 グローバルな FNS の名前空間コンテキストの作成

作業	説明	指示の参照先
グローバルな FNS の名前空間コンテキストの作成	NIS+ の下での FNS の名前空間コンテキストの作成	510 ページの「NIS の下での名前空間コンテキストの作成」
グローバルな FNS の名前空間コンテキストの作成	NIS の下での FNS の名前空間コンテキストの作成	511 ページの「NIS の下での名前空間コンテキストの作成」
グローバルな FNS の名前空間コンテキストの作成	ファイルの下での FNS の名前空間の作成	512 ページの「ローカルファイルの下での名前空間コンテキストの作成」

▼ NIS の下での名前空間コンテキストの作成

企業レベルの主なネームサービスが NIS+ であるときは、NIS+ ドメインまたはサブドメインごとに名前空間のコンテキストを個別に作成する必要があります。

- NIS+ ドメインまたはサブドメインがすでに存在している必要があります。

- NIS+ と FNS の両方で同じサーバーを使用する場合には、ドメイン (サブドメイン) のマスターサーバーで `fncreate` コマンドを実行する必要があります。NIS+ と FNS で異なるサーバーを使用する場合には、FNS サーバーとして機能するマシンで `fncreate` コマンドを実行する必要があります。異なるマシンを使用する場合には、前述の「FNS 用の NIS+ サービスの準備」の手順 4 に従って FNS サーバーを最初に準備する必要があります。
- NIS+ の完全な管理権限が必要です。

たとえば、そのドメイン用の NIS+ マスターサーバーである `submaster` マシン上の `manf.doc.com` サブドメイン用にコンテキストを作成するには、次のようにします。

- サブドメインのマスターで、`fncreate` を実行します。

```
submaster# fncreate -t org org/manf.doc.com./
```

これで、NIS+ `manf.doc.com.`サブドメイン用の編成コンテキスト、サブドメインの `passwd.org_dir` テーブルのすべてのユーザーとサブドメインの `hosts.org_dir` テーブルのすべてのホストに対するコンテキストとサブコンテキストが作成されます。

NIS+ と FNS のサーバーに異なるマシンを使用するには、FNS サーバーとして使用するマシン上で前述のコマンドを実行します。NIS+ ではないサーバーを FNS サーバーとして設定する方法についての説明は、前記の「FNS 用の NIS+ サービスの準備」の手順 4 を参照してください。

- `nisping` コマンドを使用して `ctx_dir` ディレクトリでチェックポイントを実行します。

```
# /usr/lib/nis/nisping -C ctx_dir.manf.doc.com.
```

注 – 数千のユーザーやホストを有する大規模な組織では、最初の `fncreate` 操作に数時間、それ以降のチェックポイントにも数時間かかる場合があります。

▼ NIS の下での名前空間コンテキストの作成

企業レベルの主なネームサービスが NIS であるときは、企業に対して 1 つのドメインだけが存在します。その企業全体のドメインに対して名前空間コンテキストが作成されます。

- NIS ドメインが既に存在している必要があります。
- FNS マスターサーバー上で `root` によって `fncreate` コマンドが実行される必要があります。通常、これは NIS マスターサーバーになりますが、別のサーバーを選択することもできます。

たとえば、NIS マスターサーバーでもある `fns_master` というマシンで `doc.com` ドメイン用にコンテキストを作成するには、次のようにします。

- ドメインマスターで fncreate を次のように実行します。

```
fns_master# fncreate -t org org//
```

これで NIS ドメイン doc.com 用の編成コンテキストと、NIS サーバーの passwd マップのすべてのユーザーとサーバーの hosts マップの、すべてのホストに対するコンテキストと関連のサブコンテキストが作成されます。

注 – コンテキストマップを作成したら、他の任意の NIS マップに異なるマスターを割り当てるのと同じ手順で、同じマシンをマスターサーバーに割り当てることができます。FNS マップはすべて、.ctx または .attr のいずれかで終わる名前を持ちます。

▼ ローカルファイルの下での名前空間コンテキストの作成

企業レベルの主なネームサービスがファイルであるときは、コンテキストはシステムに対して作成されます。

- /var/fn ディレクトリが存在するマシン上の /etc/passwd と /etc/hosts のファイルは誤りがなく完全に生成されている必要があります。
- fncreate コマンドは、/var/fn ディレクトリが存在するマシン上で root によって実行される必要があります。

たとえば、システム用のコンテキストを作成するには次のようにします。

- /var/fn ディレクトリのあるマシンで、fncreate を次のように実行します。

```
server1# fncreate -t org org//
```

これで、マシン内の /etc/passwd ファイルのすべてのユーザーとマシン内の /etc/hosts ファイルのすべてのホスト用にシステム、コンテキスト、関連のサブコンテキストに対する編成コンテキストが作成されます。

FNS サービスの複製

FNS のネームサービスの性能と信頼性が重要な大規模で重要性の高いネットワークでは、FNS サービスを複製してください。

FNS サービスの複製 — 作業マップ

表 **25–12** FNS サービスの複製

作業	説明	指示の参照先
FNS サービスの複製	NIS+ の下での FNS サービスの複製	513 ページの「NIS+ の下での FNS の複製」
FNS サービスの複製	NIS の下での FNS サービスの複製	513 ページの「NIS の下での FNS の複製」
FNS サービスの複製	ファイルの下での FNS の複製	514 ページの「ファイルを使用したネームサービスの下での FNS の複製」

▼ NIS+ の下での FNS の複製

マスターサーバーで FNS 名前空間が設定されたら、その他の複製サーバーを各ドメインに追加して、ドメインの `ctx_dir` を提供するサーバーにします。複製サーバーによって、サーバーの可用性と性能を拡張できます。

1. **FNS** マスターサーバー上で `nismkdir` コマンドを実行して、`ctx_dir` ディレクトリ用の複製サーバーを追加します。

   たとえば、マシン `fnsrserver` を `doc.com.` ドメインの FNS の複製サーバーにします。

   ```
   # nismkdir -s fnsrserver ctx_dir.doc.com.
   ```

2. `nisping` コマンドを使用して `ctx_dir` ディレクトリでチェックポイントを実行します。

   ```
   # /usr/lib/nis/nisping -C ctx_dir.doc.com.
   ```

   FNS の複製では一定の間隔でチェックポイントを実行してください。間隔は、数日に一回程度をお勧めします。選択する間隔は、FNS 名前空間への変更頻度によって異なります。

▼ NIS の下での FNS の複製

FNS 名前空間がドメインのマスターサーバーで設定された後に、スレーブサーバーを追加して、サーバーの可用性と機能を拡張できます。

1. `root` として、スレーブサーバーの `/etc/hosts` 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。
2. 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。
3. 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化します。

```
# /usr/sbin/ypinit -c
```

ypinit コマンドが、NIS サーバーのリストの入力を促します。作業中のローカルスレーブの名前を最初に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番で入力します。

注 – NIS クライアントとして、新しいスレーブサーバーを最初に構成して、マスターサーバーから最初に NIS マップを得ることができるようにします。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS サービスの設定と構成」を参照してください。

---

4. ypbind が実行中かどうかを判断するには、次のように入力します。

   ```
   # ps -ef | grep ypbind
   ```

   リストが表示されたら、ypbind は実行中です。

5. ypbind が実行中なら、次のように入力して停止します。

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

6. 次のように入力して、ypbind を再開します。

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

7. 次のように入力して、このマシンをスレーブとして初期化します。

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

   この場合 *master* は、既存の NIS マスターサーバーのマシン名です。

8. スレーブサーバーの yp プロセスを停止します。

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

9. yp サービスを再起動します。

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

   これとは別に、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。

▼ ファイルを使用したネームサービスの下での FNS の複製

主なネームサービスがファイルになっているときはサーバーの複製はありません。

FNS の管理、問題解決、エラーメッセージ

FNS エラーメッセージ

FNS エラーメッセージは、FN_status_t オブジェクトにステータスコードとしてカプセル化されます。対応するステータスコードについては、FN_status_t のマニュアルページを参照してください。

エラーが発生すると、FNS コマンドは、操作が失敗した名前の残りの部分を表示します。表示されなかった名前の部分の処理は成功しています。

たとえば、ユーザーが、org//service/trading/bb に対するコンテキストの作成を試みたとします。名前 org//service/ の解決には成功しましたが、trading は org//service/ によって名前を付けられたコンテキストで見つかりませんでした。このため trading/bb は、操作が失敗したときに残る名前の一部として表示されます。

```
Error in creating 'org//service/trading/bb': Name Not Found: 'trading/bb'
```

別の例では、ユーザーがコンテキスト org//service/dictionary/english の削除を試みましたが、名前を付けられたコンテキストが空であったので、操作を実行できませんでした。引用符 ('') の組み合わせは、与えられた完全な名前を FNS は解決できましたが、要求されたとおりに操作を完了できなかったことを示します。

```
Error in destroying 'org//service/dictionary/english': Context Not Empty: ''
```

XFN リファレンスに準拠した DNS 文書レコードの書式

Solaris の環境は、DNS 内で広域ネーミングシステムをフェデレートさせるための XFN 規格に準拠しています。DNS でネーミングシステムをフェデレートさせるには、DNS TXT リソースレコードに情報を入力する必要があります。この情報は、従属ネーミングシステム用に XFN リファレンスを作成するために使用されます。この章では、これらの DNS TXT レコードの書式について説明します。

- DNS のフェデレートに必要な手順については、483 ページの「開始前に必要な処置」 を参照してください。

- DNS 内のレコードを操作する一般的な方法の詳細については、『*DNS and BIND*』(Paul Albitz & Cricket Liu 著 浅羽登志也／上水流由香 監訳、アスキー出版局、1995年) を参照してください。

XFN リファレンスのリファレンスタイプは、XFNREF というタグで始まる TXT レコードから作成します。書式は以下の通りです。

```
TXT "XFNREF rformat reftype"
```

TXT の後に続く文字列の中にスペースが存在する場合、そのスペースを削除するか、文字列全体を引用符 (" ") で囲む必要があります。XFNREF、*rformat*、*reftype* という 3 つのフィールドは、スペース (スペースとタブ) で区切ります。*rformat* は、リファレンスタイプの識別子の書式を指定します。種類は次のとおりです。

- STRING – FN_ID_STRING 書式のマップ
- OID – FN_ID_ISO_OID_STRING 書式のマップ
- UUID – FN_ID_DCE_UUID 書式のマップ

reftype は、リファレンスタイプの識別子の内容を指定します。

XFNREF レコードが存在しない場合、リファレンスタイプはデフォルト設定により FN_ID_STRING を持つ XFN_SERVICE 識別子になります。2 つ以上の XFNREF TXT レコードが存在する場合、どのレコードが処理されるかは不定です。以下の TXT レコードはデフォルト設定の XFNREF と同じ働きをします。

```
TXT "XFNREF STRING XFN_SERVICE"
```

XFN リファレンスのアドレス情報は、XFN 文字列を接頭語にしたタグを持つ TXT レコードを使用して作成します。1 つのリファレンスに複数のアドレスを指定することもできます。同じタグを持つレコードはグループにまとめられ、グループごとにハンドラに渡されます。各ハンドラは渡された TXT レコードからアドレス (複数あるいは、ない場合もある) を作成し、リファレンスに付加します。XFNREF タグの場合は特別で、リファレンスタイプを作成するためにだけ使用されるため、アドレス作成の過程からは除外されます。

TXT レコードのアドレスを指定する構文は次のとおりです。

```
XFNaddress_type_tag address_specific_data
```

XFN_address_type_tag と *address_specific_data* という 2 つのフィールドは、スペース (スペースとタブ) で区切ります。*address_type_tag* は、*address_specific_data* に使用するハンドラを指定します。

TXT レコードには 1 レコードにつき 2 K バイトという文字の制限があります。特定のアドレスのデータが長すぎて 1 つの TXT レコードに格納できない場合は、以下のように複数の TXT レコードを使用できます。

```
TXT "XFNaddress_type_tag address_specific_data1"
TXT "XFNaddress_type_tag address_specific_data2"
```

特定のタグのハンドラが呼び出され、両方のデータが渡されます。ハンドラはこれら 2 行を解釈する順番を決定します。

TXT レコードの順番はあまり重要ではありません。異なるタグを持つ行がある場合、特定のタグのハンドラが呼び出される前に、同じタグを持つ行はグループにまとめられます。以下の例では、*tag1* のハンドラは 2 つの文書行で呼び出され、*tag2* のハンドラは 3 つの文書行で呼出されます。

```
TXT "XFNtag1  address_specific_data1"
TXT "XFNtag2  address_specific_data2"
TXT "XFNtag1  address_specific_data3"
TXT "XFNtag2  address_specific_data4"
TXT "XFNtag2  address_specific_data5"
```

XFN リファレンスに使用できる TXT レコードの例を以下に示します。

「例 1」

```
TXT "XFNREF STRING XFN_SERVICE"
TXT "XFNNISPLUS doc.com. nismaster 129.144.40.23"
```

「例 2」

```
TXT "XFNREF OID 1.3.22.1.6.1.3"
TXT "XFNDCE (1 fd33328c4-2a4b-11ca-af85-09002b1c89bb...)"
```

以下は、従属ネーミングシステムが割り当てられた DNS テーブルの例です。

```
$ORIGIN test.doc.com
@       IN SOA foo root.eng.doc.com          (
                      100    ;; Serial
                      3600   ;; Refresh
                      3600   ;; Retry
                      3600   ;; Expire
                      3600   ;; Minimum
              )
           NS    nshost
           TXT   "XFNREF STRING XFN_SERVICE"
           TXT   "XFNNISPLUS doc.com. nismaster 129.144.40.23"
nshost IN  A 129.144.40.21
```

XFN リファレンス用 X.500 属性の構文

この節では、XFN リファレンス用の X.500 属性の使用についての補足情報を述べます。XFN リファレンスを X.500 における属性として保存できるようにするためには、ディレクトリスキーマを、この章で定義されているオブジェクトクラスと属性をサポートするように変更する必要があります。

- X.500 のフェデレートに必要な手順については、483 ページの「開始前に必要な処置」 を参照してください。
- X.500 のディレクトリスキーマの変更については、『 *Managing the X.500 Client Toolkit* 』を参照してください。

オブジェクトクラス

XFN リファレンスをサポートするために、XFN と XFN 補助の 2 つの新しいオブジェクトクラスが導入されています。XFN オブジェクトクラスは FNS に適切ではありません。これは、米国 Sun Microsystems, Inc. の X.500 ディレクトリ製品が、新しく導入された複合 ASN.1 構文をサポートしていないためです。その代わりに、FNS では XFN 補助オブジェクトクラスを使用します。

ASN.1 では、次のような 2 つの新しいオブジェクトクラスが定義されています。

```
xFN OBJECT-CLASS ::= {
         SUBCLASS OF         { top }
         KIND               auxiliary
         MAY CONTAIN        { objectReferenceId |
                              objectReference |
                              nNSReferenceId |
                              nNSReference }
          ID                 id-oc-xFN
     }
     id-oc-xFN OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) ansi(840) sun(113536)
        ds-oc-xFN(24)
     }
xFNSupplement OBJECT-CLASS ::= {
         SUBCLASS OF           { top }
         KIND                  auxiliary
         MAY CONTAIN           { objectReferenceString |
                               nNSReferenceString }
            ID                 id-oc-xFNSupplement
      }
      id-oc-xFNSupplement OBJECT IDENTIFIER ::= {
          iso(1) member-body(2) ansi(840) sun(113536)
          ds-oc-xFNSupplement(25)
      }
```

XFN 補助オブジェクトクラスは、補助オブジェクトクラスとして定義されているため、X.500 のオブジェクトクラスすべてに引き継がれる可能性があります。XFN 補助オブジェクトクラスは、以下の 2 つの任意属性によって定義されます。

- `objectReferenceString` は、XFN リファレンスをオブジェクト自体で保存するときに使用される
- `nNSReferenceString` は、XFN リファレンスを次のネーミングシステムで保存するときに使用される

ASN.1 では、この 2 つの属性を以下のように定義しています。

```
        objectReferenceString ATTRIBUTE ::= {
            WITH SYNTAX             OCTET STRING
            EQUALITY MATCHING RULE  octetStringMatch
            SINGLE VALUE            TRUE
            ID                      { id-at-objectReferenceString }
        }
        id-at-objectReferenceString OBJECT IDENTIFIER ::= {
```

```
        iso(1) member-body(2) ansi(840) sun(113536)
        ds-at-objectReferenceString(30)
    }
    nNSReferenceString ATTRIBUTE ::= {
        WITH SYNTAX             OCTET STRING
        EQUALITY MATCHING RULE  octetStringMatch
        SINGLE VALUE            TRUE
        ID                      { id-at-nNSReferenceString }
    }
    id-at-nNSReferenceString OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) ansi(840) sun(113536)
        ds-at-nNSReferenceString(31)
    }
```

objectReferenceString と nNSReferenceString は、共に XFN リファレンスを文字列形式で保存します。それぞれのオクテット列構文は、さらに以下の BNF 定義に準拠するように制約を加えられます。

```
<ref>          ::= <id> '$' <ref-addr-set>
<ref-addr-set> ::= <ref-addr> | <ref-addr> '$' <ref-addr-set>
<ref-addr>     ::= <id> '$' <addr-set>
<addr>         ::= <hex-string>
<id>           ::= 'id'   '$' <string> |
                   'uuid' '$' <uuid-string> |
                   'oid'  '$' <oid-string>
<string>       ::= <char> | <char> <string>
<char>         ::= <PCS> | '\' <PCS>
<PCS>          ::= // Portable Character Set:
                   // !"#$%&'()*+,-./0123456789:;<=>?
                   // @ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
                   // `abcdefghijklmnopqrstuvwxyz{|}~
<uuid-string>  ::= <uuid-char>  | <uuid-char> <uuid-string>
<uuid-char>    ::= <hex-digit> | '-'
<oid-string>   ::= <oid-char>  | <oid-char> <oid-string>
<oid-char>     ::= <digit> | '.'
<hex-string>   ::= <hex-octet> | <hex-octet> <hex-string>
<hex-octet>    ::= <hex-digit> <hex-digit>
<hex-digit>    ::= <digit> |
                   'a' | 'b' | 'c' | 'd' | 'e' | 'f' |
                   'A' | 'B' | 'C' | 'D' | 'E' | 'F'
<digit>        ::= '0' | '1' | '2' | '3' | '4' | '5' |
                   '6' | '7' | '8' | '9'
```

以下は、文字列形式の XFN リファレンスの例です。

```
id$onc_fn_enterprise$id$onc_fn_nisplus_root$0000000f77697a2e636fd2e2062696762696700
```

この例では、onc_fn_enterprise というタイプの XFN リファレンスを使用しています。この中には、onc_fn_nisplus_root というアドレスタイプと 1 つのアドレス値があります。アドレス値は XDR によって符号化された文字列で、ドメイン名、doc.com の後にホスト名 cygnus が続く形で構成されています。

XFN リファレンスを X.500 エントリに追加するには、次の例のように fnattr という FNS コマンドを使用します。

```
# fnattr -a .../c=us/o=doc object-class top organization xfn-supplement
```

この例では、c=us/o=doc という新しいエントリを作成し、top、organization、XFN-supplement という値のオブジェクトクラス属性を追加しています。

fnbind という FNS コマンドは、NIS+ リファレンスを命名されたエントリに割り当て、X.500 を NIS+ 名前空間のルートにリンクします。fnbind の名前因数内の文字の後にスラッシュ (/) が使用されていることに注意してください。

```
# fnbind -r .../c=us/o=doc/ onc_fn_enterprise onc_fn_nisplus_root "doc.com. cygnus"
```

FNS コンテキストを個別に作成する

FNS コンテキストは、fncreate コマンドを使用して作成します。この節では、組織全体の FNS コンテキストではなく、FNS コンテキストを個別に 作成する方法について説明します。fncreate コマンドを使用して、特定のタイプのコンテキストを作成し、指定した複合名にそのコンテキストを割り当てます。また、コンテキストのサブコンテキストを作成することもできます。

fncreate コマンドの構文は次のとおりです。

fncreate -t *context_type* [-f *input_file*] [-o][-r *reference_type*][-s][-v] [-D] *composite_name*

表 **25–13** fncreate コマンドオプション

オプション	説明
-t *context*	作成するコンテキストのタイプを指定する。*context* には、org、hostname、username、host、user、service、site、nsid、generic、fs のうちいずれかを使用する
-f	*input_file* に指定された個々のホストまたはユーザーのコンテキストを作成する。このオプションは、-t username オプションまたは -t hostname オプションと併用される場合にだけ有効で、対応する NIS+ passwd テーブル (または hosts テーブル) 中のユーザー (またはホスト) のサブセットのコンテキストを個々に作成する場合に使用できる
-o	指定されたコンテキストだけを作成する。-o オプションを指定しないと、サブコンテキストは FNS ポリシーに基づいて作成される
-r	作成される汎用コンテキストの *reference_type* を指定する。これは -t generic オプションと併用される場合にだけ有効
-s	すでに使用されている複合名で、新しいコンテキストを作成する。このオプションを使用しなければ、すでに割り当てられている名前で新しいコンテキストを作成できない

表 25–13 fncreate コマンドオプション　(続き)

オプション	説明
-D	コンテキストが作成されるたびに、コンテキストに関連付けられた NIS+ オブジェクトの情報を表示する。このオプションはデバッグに役立つ
-v	コンテキスト作成時に、作成に関する情報を表示する

注 – 組織コンテキスト作成時に -o オプションを指定した場合、host、user、service のコンテキストは作成されてもサブコンテキストは生成されません。

名前空間識別子に割り当てられるコンテキストを作成する場合、下線のない名前 (user など) はコンテキストの作成に使用され、下線の付いた名前 (_user など) は新しく作成されたコンテキストのリファレンスに割り当てられます。この点は、コマンド行で指定された名前が下線のついてものであるか否かに関わらず常に同じです。

たとえば、以下のコマンドでは、

```
fncreate -t username org/sales/_user
```

org/sales/user のコンテキストを作成し、org/sales/_user の割り当てを org/sales/user のコンテキストに追加しています。

組織コンテキストを作成する

組織コンテキストを作成するには、コンテキストタイプに org を指定します。複合名には、使用しているネームサービスによって以下のいずれかにする必要があります。

- 「NIS+」。すでに存在している NIS+ ドメイン (またはサブドメイン) の名前。NIS+ ドメインとは、org_dir サブディレクトリを含む NIS+ ディレクトリオブジェクトのことです。ドメインの org_dir サブディレクトリには、そのドメインの生成された host テーブルと passwd テーブルが必要です。
- 「NIS」。NIS ドメインの名前。host マップと passwd マップが必要です。
- /etc ファイル。/etc ファイルを使用するのは、org// 組織コンテキストだけです。

NIS+ の場合の組織コンテキストの例

NIS+ のルートドメインが doc.com で、sales.doc.com というサブドメインがあるとします。sales というサブドメインに対応する sales という組織コンテキストを作成するには、以下のコマンドを入力します。

```
fncreate -t org org/sales/
```

新しいコンテキストの作成時には、ドメインである sales.doc.com の下に ctx_dir というディレクトリが作成されます。すでに ctx_dir が存在する場合は作成されません。

上記の例では -t オプションだけを使用し、-o オプションは使用しません。これによって複合名 org/sales/ の組織コンテキストが作成されると同時に、サブコンテキストとして hostname、username、service が作成されます。また host コンテキストと user コンテキスト、ホストとユーザーの service サブコンテキストも作成されます。実際には以下のコマンドが実行されます。

```
fncreate -t hostname org/sales/host/
fncreate -t username org/sales/user/
fncreate -t service org/sales/service/
```

代わりに fncreate -o -t org を使用すると、org コンテキストだけが作成されます。hostname、username、service のコンテキストは作成されますが、host コンテキストと user コンテキストの生成はされません。

org コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。この点は、hostname、username、service といったサブコンテキストについても同様です。ただし、host コンテキスト、user コンテキストとそのサブコンテキストの所有者となるのは、コンテキストの作成対象となったホストおよびユーザーです。管理者が引き続き host コンテキストや user コンテキストを処理するには、fncreate を実行する時に NIS_GROUP 環境変数を適宜設定する必要があります。たとえば、C シェルの場合、NIS_GROUP を以下のように fns_admins.doc.com に設定します。

```
rootmaster# setenv NIS_GROUP fns_admins.doc.com
```

すべてのホストのコンテキスト

fncreate で hostname をコンテキストのタイプとして指定すると hostname コンテキストが作成されます。hostname コンテキストでは、host コンテキストの作成、割り当てができます。この場合、-o オプションを指定しなければ、host コンテキスト (およびそのサブコンテキスト) も NIS+ の host.org_dir テーブル中のホストごとに作成されます。-o オプションを指定した場合は、コンテキストだけが作成されます。

たとえば、以下のコマンドを実行すると、

```
fncreate -t hostname org/sales/host/
```

hostname コンテキストが作成されます。

```
fncreate -t host org/sales/host/hname
```

hname とは、各マシンの hosts.org_dir テーブル中にある名前です。また org/sales/_host/ の、org/sales/host/ のリファレンスへの割り当ても行われます。

hostname コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。また host コンテキスト (およびそのサブコンテキスト) の所有者となるのは、コンテキストの作成対象となったホストです。つまり、個々のホストがそれぞれ自分のホストのコンテキスト (およびそのサブコンテキスト) を持つことになります。

-f オプションを使用すると、NIS+ の hosts.org_dir テーブル中のホストの、サブセットのコンテキストを作成できます。このオプションでは、*input_file* に指定されたホストのコンテキストが作成されます。

1 台のホストのコンテキスト

fncreate で host をコンテキストのタイプとして指定すると、ホストのコンテキストとサブコンテキストが作成されます。-o オプションを指定しなければ、コマンドによってホストの service コンテキストと fs の割り当てが自動的に作成されます。-o オプションを指定した場合は、host コンテキストだけが作成されます。

たとえば、以下のコマンドでは、

```
# fncreate -t host org/sales/host/antares/
```

antares というホストのコンテキストが作成されます。実際には以下のコマンドが実行されます。

```
fncreate -t service org/sales/host/antares/service/
fncreate -t fs org/sales/host/antares/fs/
```

host コンテキスト (およびそのサブコンテキスト) の所有者となるのはホストです。上記の例では antares というホスト (NIS+ 主体名 antares.sales.doc.com) が以下のコンテキストの所有者となります。

- org/sales/host/antares/
- org/sales/host/capsule/service/
- org/sales/host/capsule/fs

ただしこの場合、ホストの属する hostname コンテキスト (上記の例では、org/sales/host/) が必要です。また NIS+ テーブル hosts.org_dir に、ホスト名を指定する必要があります。

ホストの別名

NIS+ の hosts.org_dir テーブルには、別名を持つホストが存在することもあります。これは、テーブル中では「正式名が同じで別名が異なる」ホストの集合として存在します。

FNS においては、たとえ複数の別名を持っていても、1 つのホストが持つ host コンテキストは 1 つだけです。hostname コンテキスト中のホストの別名が割り当てられるのは、正式名と同じコンテキストのリファレンスです。

すべてのユーザーのコンテキスト

fncreate で username をコンテキストのタイプとして指定すると、username コンテキストが作成されます。username コンテキストでは、個々の user コンテキストの作成と割り当てが行われます。-o オプションを指定しないと、NIS+ テーブル passwd.org_dir に存在するユーザー名ごとに、user コンテキスト (およびそのサブコンテキスト) が作成されます。-o オプションを指定した場合は、username コンテキストだけが作成されます。

たとえば、以下のコマンドを実行すると、

```
# fncreate -t username org/sales/user/
```

username コンテキストが作成されます。

```
fncreate -t user org/sales/user/uname
```

実際には、passwd.org_dir テーブルに存在するユーザー *uname* ごとに、以下のコマンドが実行されます。また、org/sales/_user/ の org/sales/user/ への割り当ても行われます。

username コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。個々の user コンテキスト (およびそのサブコンテキスト) の所有者となるのは、コンテキストの作成対象となったユーザーです。ユーザーがそれぞれ独自の user コンテキスト (およびそのサブコンテキスト) を所有することになります。

-f オプションを使用すると、NIS+ テーブル passwd.org_dir に存在するユーザーのサブセットのコンテキストを作成できます。このオプションでは、*input_file* に指定されたホストのコンテキストが作成されます。

1 人のユーザーのコンテキスト

fncreate で user をコンテキストタイプとして指定すると、ユーザーの user コンテキスト (およびそのサブコンテキスト) が作成されます。-o オプションを指定しなければ、user コンテキストの下に service サブコンテキストと fs の割り当てが作成されます。-o オプションを指定した場合は、user コンテキストだけが作成されます。

たとえば、以下のコマンドでは、

```
# fncreate -t user org/sales/user/jjones/
```

jjones という名前のユーザーの user コンテキストが作成されます。実際には、以下のコマンドが実行されます。

```
fncreate -t service org/sales/user/jjones/service/
fncreate -t fs org/sales/user/jjones/fs/
```

user コンテキスト (およびそのサブコンテキスト) の所有者となるのは、コンテキストの作成対象となったユーザーです。上記の場合、作成されたコンテキストの所有者となるのは、jjones というユーザー (NIS+ 主体名 jjones.sales.doc.com) です。

ただしこの場合、ユーザーの属する username コンテキスト (上記の org/sales/user) が必要です。また、指定されたユーザー名が NIS+ テーブル passwd.org_dir 中に存在している必要があります。

サービスのコンテキスト

fncreate でコンテキストタイプとして service を指定すると、service コンテキストが作成されます。service コンテキストでは、サービス名の割り当てが行えます。service コンテキスト中で割り当てられるリファレンスのタイプに制約はありません。ただし、service コンテキストを使用するアプリケーションによっては制約が設けられます。たとえば、デスクトップアプリケーションのグループなら、service コンテキスト中で割り当てられるリファレンスのタイプは、カレンダ、カードファイル、ファックスサービス、プリンタなどになるでしょう。

たとえば、以下のコマンドでは、

```
# fncreate -t service org/sales/service/
```

sales という組織の service コンテキストが作成されます。末尾の名前 (service) は名前空間識別子なので、fncreate では org/sales/_service/ の org/sales/service/ のリファレンスへの割り当ても行われます。このコマンドの実行後は、org/sales/service/calendar、org/sales/service/fax といった名前をこのサービスコンテキストで割り当てることができます。

service コンテキストでは、階層型の名前空間 (左から順に名前が並べられ、名前と名前の間はスラッシュで区切られる) がサポートされています。これによってアプリケーションは、名前空間をサービスごとに区切ることができます。デスクトップアプリケーションの場合、plotter コンテキストを作成した後に以下のコマンドを使用すれば、service コンテキストの下に plotter のグループを作成することができます。

```
# fncreate -t service org/sales/service/plotter
```

さらにこの下には、org/sales/service/plotter/speedy、org/sales/service/plotter/color といった名前を割り当てることができます。

注 – ただし、末尾の名前が名前空間識別子ではないので、service と _service のような割り当てが行われることはありません。

作成される service コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です 。

プリンタコンテキスト

printer コンテキストは、複合名の service コンテキストの下に作成されます。

汎用コンテキスト

fncreate でコンテキストタイプに generic を使用すると、割り当て名のコンテキスト (アプリケーションによって使用される) が作成されます。

generic コンテキストは、リファレンスのタイプを変更できるという点以外は service コンテキストと同じです。generic コンテキストのリファレンスのタイプを指定するには、-r オプションを使用します。このオプションが省略された場合は、親コンテキストが generic タイプであればそれがそのまま使用され、別のタイプであればデフォルトとして指定されたものが使用されます。

service コンテキストと同様、generic コンテキストに割り当てられるリファレンスのタイプには制約がありません。ただし generic コンテキストを使用するアプリケーションによって制約が設けられる場合があります。

たとえば、以下のコマンドでは、

```
# fncreate -t generic -r WIDC_comm org/sales/service/extcomm
```

sales という組織の service コンテキストの下に、リファレンスタイプ WIDC _comm の generic コンテキストが作成されます。この generic コンテキストでは、org/sales/service/extcomm/modem などの名前を割り当てることができます。

generic コンテキストでは、階層型の名前空間 (左から順に名前が並べられ、名前と名前の間はスラッシュで区切られる) がサポートされています。これによってアプリケーションは、名前空間をサービスごとに区切ることができます。上の例の場合、以下のコマンドを使用すれば、modem 用の generic サブコンテキストが作成できます。

```
# fncreate -t generic org/sales/service/extcomm/modem
```

さらにこの下には、
org/sales/service/extcomm/modem/secure、
org/sales/service/extcomm/modem/public といった名前を割り当てることができます。

作成された generic コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。

サイトコンテキスト

site コンテキストでは、サイト名の割り当てが行えます。

たとえば、以下のコマンドでは、

```
# fncreate -t site org/sales/site/
```

サイトコンテキストを作成します。ここでは末尾の名前 (site) が名前空間識別子なので、fncreate によって org/sales/site/ のリファレンスに org/sales/_site/ が割り当てられます。

site コンテキストでは、階層型の名前空間 (左から順に名前が並べられ、名前と名前の間はドットで区切られる) がサポートされています。これによって、地理的な位置関係にもとづいてサイトを区分できます。

たとえば、以下のコマンドでは、

```
# fncreate -t site org/sales/alameda
# fncreate -t site org/sales/site/alameda.bldg-5
```

サイトコンテキスト alameda とサイトサブコンテキスト alameda.bldg-5 を作成します。

注 – ただし末尾の名前が名前空間識別子ではないので、site と _site の場合のような割り当てが行われることはありません。

作成された site コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。

ファイルのコンテキスト

fncreate でコンテキストタイプに fs を指定すると、ユーザーまたはホストのファイルシステムコンテキスト (ファイルコンテキストとも呼ばれる) が作成されます。たとえば、以下のコマンドでは、

```
# fncreate -t fs org/sales/user/petrova/fs/
```

ユーザー petrova の fs コンテキストを作成します。ここでは末尾の名前 (fs) が名前空間識別子なので、org/sales/user/petrova/fs/ のリファレンスに org/sales/user/petrova/_fs/ が割り当てられます。

ユーザーの fs コンテキストは、NIS+ テーブル passwd.org_dir に保存されているユーザーのホームディレクトリと同じものです。一方ホストの fs コンテキストは、ホストによってエクスポートされる NFS ファイルシステムをいくつか集めたものです。

組織 およびサイトの file コンテキストを作成する場合、あるいはユーザーおよびホストのデフォルトでない file コンテキストを作成する場合は、fncreate_fs コマンドを使用します。詳細については、550 ページの「ファイルコンテキストの管理」を参照してください。

作成された fs コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。

名前空間識別子のコンテキスト

nsid (namspace identifier = 名前空間識別子) タイプのコンテキストでは、名前空間識別子を割り当てることができます。

たとえば、以下のコマンドでは、

```
# fncreate -t nsid org/sales/site/alameda.bldg-5/
```

サイト alameda.bldg-5 の nsid コンテキストを作成します。このコンテキストに対して、service/ などのサブコンテキストを作成できます。この例に続いて、以下のコマンドを実行して、

```
# fncreate -t service org/sales/site/alameda.bldg-5/service/
```

alameda.bldg-5 の service コンテキストを作成できます。

作成された nsid コンテキストの所有者となるのは、fncreate コマンドを実行した管理者です。

エンタープライズレベルのコンテキストを管理する

FNS コンテキストを管理する (コンテキストの内容を確認する) ためのツールには様々な種類があります。以下の表は、そのツール (コマンド) の構文を示したものです。

割り当てに関する情報を表示する

fnlookup は、複合名の割り当てに関する情報を表示するのに使用します。

fnlookup [-v][-L] *composite_name*

表 **25–14** fnlookup コマンドのオプション

オプション	説明
-v	割り当てに関する詳細な情報を表示する
-L	「XFN リンクの割り当て先となるのはどのリファレンスか」を表示する

たとえば、以下の例では、ユーザー darwin の割り当てに関する情報が表示されます。

```
# fnlookup -v user/darwin/
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 length: 52
 context type: user
 representation: normal
 version: 0
 internal name: fns_user_darwin.ctx_dir.sales.doc.com.
```

ここで user/Charles.Darwin が user/darwin にリンクされているとすると、下の例の最初のコマンドでは、user/Charles.Darwin の割り当て先 (XFNリンク) が表示されます。2 番目のコマンドでは、XFNリンク user/jjones が調べられ、user/darwin の割り当て先 (user コンテキスト) が表示されます。

```
# fnlookup user/Charles.Darwin
Reference type: fn_link_ref
Address type: fn_link_addr
 Link name: user/darwin
# fnlookup -L user/Charles.Darwin
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 context type: user
```

コンテキストの内容を表示する

fnlist は、コンテキストの内容を表示するのに使用されます。

fnlist [-lv] [*name*]

表 **25–15** fnlist コマンドのオプション

オプション	説明
-v	割り当てに関する詳細な情報を表示する
-l	コンテキスト中で割り当てられている名前に関する情報を表示する

以下の例では、user コンテキストに割り当てられている名前が表示されます。

```
# fnlist user/
Listing 'user/':
```

```
jjones
julio
chaim
James.Jones
```

名前を指定しないと、初期コンテキストの内容が表示されます。

```
# fnlist
Listing ’’:
_myorgunit
...
_myself
thishost
myself
_orgunit
_x500
_host
_thisens
myens
thisens
org
orgunit
_dns
thisuser
_thishost
myorgunit
_user
thisorgunit
host
_thisorgunit
_myens
user
```

-l オプションを指定すると、指定されたコンテキスト中で割り当てられている名前に関する情報が表示されます。

```
# fnlist -l user/
Listing bindings ’user/’:
name: julio
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 context type: user
name: chaim
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 context type: user
name: James.Jones
Reference type: fn_link_ref
Address type: fn_link_addr
 Link name: user/jjones
name: jjones
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 context type: user
```

-l、-v オプションを同時に指定すると、割り当てられた名前に関するさらに詳細な情報が表示されます。

```
# fnlist -lv user/
Listing bindings 'user/':
name: julio
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 length: 52
 context type: user
 representation: normal
 version: 0
 internal name: fns_user_julio.ctx_dir.sales.doc.com.
name: chaim
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 length: 52
 context type: user
 representation: normal
 version: 0
 internal name: fns_user_chaim.ctx_dir.sales.doc.com.
name: James.Jones
Reference type: fn_link_ref
Address type: fn_link_addr
 length: 11
 data: 0x75 0x73 0x65 0x72 0x2f 0x6a 0x6a 0x6f 0x6e 0x65
user/jjones
name: jjones
Reference type: onc_fn_user
Address type: onc_fn_nisplus
 length: 52
 context type: user
 representation: normal
 version: 0
 internal name: fns_user_jjones.ctx_dir.sales.doc.com.
```

複合名をリファレンスに割り当てる

fnbind は、複合名をリファレンスに割り当てるためのコマンドです。

このコマンドには、2 種類の構文があります。

- すでに存在する名前のリファレンスを新しい名前に割り当てるもの (以下参照)
- コマンド行の引数にもとづいて作成されたリファレンスを、別の名前に割り当てるもの (532 ページの「コマンド行にリファレンスを作成する」 を参照)

既存の名前を新しい名前に割り当てる

既存の名前を新しい名前に割り当てるための fnbind の構文は以下のようになります。

`fnbind [-s][-v][-L]` *oldname newname*

表 **25–16** fnbind コマンドのオプション (割り当て名)

オプション	説明
oldname	すでに存在している複合名
newname	すでに存在している複合名に割り当てる新しい名前
`-s`	複合名がすでに割り当てられている場合、割り当てを上書きする
`-v`	割り当てに使用されたリファレンスに関する情報を表示する
`-L`	*name* を使用して XFN リンクを作成し、それを *new_name* に割り当てる

たとえば、以下のコマンドでは、`user/julio/service/printer` という名前が `myorgunit/service/printer` のリファレンスに割り当てられます。

```
# fnbind myorgunit/service/printer user/julio/service/printer
```

newname がすでに割り当てられている場合は、`fnbind -s` を使用しないと処理が正しく行われません。つまり上記の例で `user/julio/service/printer` がすでに割り当てられていれば、以下のように `-s` オプションを使用して `myorgunit/service/printer` との割り当てで上書きする必要があるということになります。

```
# fnbind -s myorgunit/service/printer user/julio/service/printer
```

`-v` オプションは、割り当てに使用されたリファレンスに関する情報を表示するのに使用されます。

```
# fnbind -v myorgunit/service/printer user/julio/service/printer
Reference type: onc_printers
Address type: onc_fn_printer_nisplus
```

以下の例では、`user/jjones` によって XFN リンクが作成され、`user/James.Jones` という名前に割り当てられます。

```
# fnbind -L user/jjones user/James.Jones
```

同様に以下の例では、`user/julio/service/printer` から `myorgunit/service/printer` へのリンクが作成されます。

```
# fnbind -sL myorgunit/service/printer user/julio/service/printer
```

コマンド行にリファレンスを作成する

コマンド行にリファレンスを作成するための `fnbind` の構文は次のようになります。

`fnbind -r [-s] [-v]` *newname* `[-O | -U]` *reftype* `{[-O | -U] |` *addresstype* `[-c|-x]` *addresscontents*`}+`

表 **25–17** fnbind コマンドオプション (リファレンス作成)

オプション	説明
newname	リファレンスを作成する新しい名前
reftype	作成するリファレンスのタイプ。-O、-U オプションを使用しない場合は、*reftype* の識別子として FN_ID_STRING を使用する
addresstype	作成するアドレスのタイプ。-O、-U オプションを使用しない場合は、*addresstype* の識別子として FN_ID_STRING を使用する
addresscontents	作成するリファレンスのアドレス。-c あるいは -x オプションを使用しない場合、アドレスは XDR によって符号化された文字列として保存される
-s	複合名がすでに割り当てられている場合、割り当てを上書きする
-v	割り当てに使用されたリファレンスに関する情報を表示する
-c	アドレス内容を、XDR による符号化を行わずに保存する
-x	アドレス内容を 16 進数で入力された文字列であると解釈し、そのまま保存する
-r	指定タイプのリファレンスを作成し、コマンド行で指定された名前に割り当てる
-O	タイプを指定する文字列を、ASN.1 の形式 (整数をいくつか並べてドットで区切ったもの) として解釈して保存する
-U	タイプを指定する文字列を、DCE UUID として解釈して保存する

たとえば、以下の例では、thisorgunit/service/calendar という名前が、「タイプ onc_calendar、アドレスタイプ onc_cal_str、アドレス staff@cygnus」というリファレンスに割り当てられます。

```
# fnbind -r thisorgunit/service/calendar onc_calendar
onc_cal_str staff@cygnus
```

コマンド行で指定されたアドレスは、デフォルトでは XDR で符号化された後、リファレンスに保存されます。-c オプションが指定された場合は、XDR による符号化は行われず、そのままの形で保存されます。-x オプションが指定された場合は、16 進文字列として解釈されて保存されます (XDR による符号化は行われない)。

リファレンス (およびそのアドレス) のタイプの指定には、デフォルトでは FN_ID_STRING 識別子の形式が使用されます。-O オプションでは FN_ID_ISO_OID_STRING (ASN.1、10進数を並べてドットで区切る) の形式が使用されます。-U オプションでは FN_ID_D CE_UUID (DCE UUID、文字列を使用する) の形式が使用されます。

注 – ASN.1 の詳細については、『*ISO 8824: 1990, Information Technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1)*』を参照してください。DCE UUID の詳細については、『*X/Open Preliminary Specification, October 1993, X/Open DCE: Remote Procedure Call (ISBN: 1-872630-95-2)*』を参照してください。

以下の例では、thisorgunit/service/nx という名前に割り当てられるリファレンス (およびそのアドレス) の、タイプが OIDs の形式で、アドレスが 16 進の文字列で指定されています。

```
# fnbind -r thisorgunit/service/nx -O 1.2.99.6.2.1
-O 1.2.99.6.2.3 -x ef12eab67290
```

複合名を削除する

fnunbind は、複合名を名前空間から削除するのに使用されます。ただし、名前に対応するオブジェクトは削除されず、オブジェクトと名前の割り当てを解除するだけであるという点に注意してください。

以下の例では、user/jjones/service/printer/color という複合名が削除されます。

```
# fnunbind user/jjones/service/printer/color
```

コンテキストに割り当てられた名前を変更する

fnrename は、コンテキストに割り当てられている名前を変更するのに使用されます。

以下の例では user/jjones/service/ というコンテキストに割り当てられた、clndr という名前が calendar に変更されます。

```
# fnunbind user/jjones/service/printer/color
```

コンテキストを削除する

fndestroy は、指定された複合名を名前空間から削除し、その名前を持つコンテキストを削除するのに使用されます。

たとえば、user/jones/ という名前を名前空間との割り当てから解除し、user/jjones/ という名前のコンテキストを削除するには、以下のように入力します。

```
# fndestroy user/jjones/
```

削除の対象となるコンテキスト (指定された複合名を持つもの) にサブコンテキストが存在する場合、コマンドは正常に動作しません。

FNS の管理 - 属性の概要

属性は、名前付きオブジェクトに使用することができます。属性はオプションです。名前付きオブジェクトには、属性を付けないことも、1 つまたは複数の属性を付けることもできます。

属性はユニークな属性識別子、属性構文、および 0 個以上の明確な属性値のセットからなります。

XFN で、既存の名前付きオブジェクトに対応する属性値の検索や変更のための、基本的な属性インタフェースが定義されます。これらのオブジェクトはコンテキストの場合もあれば、別のタイプのオブジェクトの場合もあります。コンテキストに関連しているのは、構文属性で、これはコンテキストがどのように複合名を解析するかを示します。

拡張属性インタフェースには、特定の属性を検索したり、オブジェクトやそれに対応する属性を作成するオペレーションが含まれます。

属性を検索する

`fnsearch` コマンドで、属性を検索します。

`fnsearch` コマンドの構文を以下に示します。

```
fnsearch [-ALlv] [-n max] [-s scope] name [-a ident]... [-O|-U] filter_expr [filter_arg]
```

表 25–18 `fnsearch` コマンドのオプション

オプション	説明
`-n` *max*	オブジェクトの最大数だけを表示する
`-s` *scope*	検索の範囲を設定する
`-a` *ident*	*ident* に一致する属性だけを表示する

表 **25–18** fnsearch コマンドのオプション　(続き)

オプション	説明
name	複合名
filter_expr	ブール、論理、グルーピング、リレーショナル、比較演算子 (表 25–19 参照)
filter_arg	フィルタ表現の引数 (表 25–19 参照)
`-A`	信頼できるソースだけを参照する
`-L`	XFN リンクに従う
`-l`	一致するオブジェクトのオブジェクトリファレンスを表示する
`-v`	詳細表示。一致するオブジェクトの詳細なオブジェクトリファレンスを表示する
`-O`	識別子として OSI OID を使用する
`-U`	識別子として DCE UUID を使用する

属性に対応するオブジェクトを検索する

fnsearch コマンドを使って、選択した属性に対応するオブジェクトを検索できます。

たとえば、orgunit/sales/site/ の中の for_sale という属性に対応している属性をすべて見つけ出す場合、以下のコマンドを入力します。

```
% fnsearch orgunit/sales/site/ for_sale
```

属性検索をカスタマイズする

検索パターンにおいて、フィルタ表現のうち以下のすべてを使用することもできます。

表 **25–19** fnsearch フィルタ表現の演算子

フィルタ表現演算子	シンボル、あるいは形式
論理演算子	or, and , not
グルーピングのための丸括弧	( )

表 **25–19** fnsearch フィルタ表現の演算子　(続き)

フィルタ表現演算子	シンボル、あるいは形式
関係演算子：ある属性を入力した値と比較する	== 1 つ以上の属性値が入力値に等しい場合、True (真)。!= 入力値に等しい属性値がない場合、True (真)。< 1 つ以上の属性値が入力値未満の場合、True (真)。<= 1 つ以上の属性値が入力値以下の場合、True (真)。> 1 つ以上の属性値が入力値を超える場合、True (真)。>= 1 つ以上の属性値が入力値以上の場合、True (真)。~= コンテキスト固有のあいまい照合条件に基づいて、1 つ以上の属性値が入力値と一致している場合、True (真)。この条件は厳密な一致を含む必要がある
次に例を示します。	`% fnsearch` *name* `"not (make == 'olds' and year == 1983)"`
置換トークン：	`%a` は、属性に置き換えられる
シェルスクリプトを書くときに役に立ち、`-O` および `-UO` を指定すると、OSI OID および DCE UUID を使用できる	`%s` は、文字列に置き換えられる `%i` は、識別子に置き換えられる `%v` は、属性値に置きかえられる (現在は、`fn_attr_syntax_ascii` しかサポートされていない)
次に例を示します。	以下の 3 つの例はどれも同じ % fnsearch *name* "color == 'red'" % fnsearch *name* "%a == 'red'" color % fnsearch *name* "%a == %s" color red
ワイルドカード文字列	`*`, `*`*string* , *string*`*`, *str*`*`*ing*, `%s*`
拡張演算子	`'name'`(ワイルドカードを使った文字列), `'reftype'`(識別子), `'addrtype'`(識別子)
次に例を示します。	名前が "Bill" で始まり、IQ 属性が 80 以上のオブジェクトを検索する `% fnsearch name "'name'('bill'*) and IQ> 80"`

検索パターン作成の詳細については、`fnsearch(1)` マニュアルページを参照してください。

属性を更新する

`fnattr` コマンドにより、FNS 名前付きオブジェクトに関連する属性を更新したり検索できます。`fnattr` コマンドを使って、以下の 4 つの属性に関する操作が行えます。

- 属性を追加する

`fnattr -a [-s]` *name* `[-O|-U]` *identifier values*

- 属性を削除する

```
fnattr -d name [[-O|-U] identifier [values]]
```

- 属性を変更する

```
fnattr -m name [-O|-U identifier oldvalue newvalue
```

- 属性を表示する

```
fnattr -l name [[-O|-U] identifier
```

表 **25–20** `fnattr` コマンドのオプション

オプション	説明
name	複合名
identifier	属性名
values	1 つあるいは複数の属性値
oldvalue	変更する属性値
newvalue	新しい属性値
`-a`	新しい属性値を追加 (作成) する
`-d`	属性を削除する
`-m`	属性を変更 (修正) する
`-l`	属性値を一覧表示する
`-s`	「代用」モードで追加する。*identifier* 属性に対する既存の値をすべて削除し、新しい属性値を作成する
`-l`	属性と値を一覧表示する
`-O`	識別子として OSI OID を使用する
`-U`	識別子として DCE UUID を使用する

各オプションでの識別子の形式は、`-O` あるいは `-U` のオプションを使用していない限り、`FN_ID_STRING` になります。

属性を追加する

属性を追加したり、属性に値を追加したりするには、`-a` オプションを使用します。この場合、複合名、属性識別子、追加する値も指定します。

```
fnattr -a [-s] name [-O | -U] identifier value1 [value2+]
```

以下の例では、属性識別子 `model` と値 `hplaser` が `thisorgunit/service/printer` に追加されます。

```
# fnattr -a thisorgunit/service/printer model hplaser
```

-s は、「代用モードで追加する」という意味のオプションです。指定の識別子を持つ属性がすでに存在する場合、-s はその値をすべて削除し、それらを追加した値に置き換えます。このオプションを指定しないと、指定した属性の値には既存の値と追加した新しい値の両方が含まれることになります。

```
# fnattr -as thisorgunit/service/printer model hplaser
```

上の例では最初に model に対応する値をすべて削除し、hplaser を値として追加します。

属性を削除する

FNS 名前付きオブジェクトの属性を削除するには、-d オプションを使用します。

```
fnattr -d name [[-O | -U] identifier value1 [value2+]]]
```

削除対象の指定には以下の規則があります。

- 名前のみ。複合名だけを指定し、属性識別子を指定しないと、名前付きオブジェクトの属性はすべて削除される
- 名前および識別子のみ。複合名と属性識別子だけを指定し、属性値を指定しない場合、「*identifier*」によって識別された属性全体が削除される
- 名前、識別子、および値。複合名、属性識別子、および 1 つあるいは複数の属性値を指定した場合、これらの値だけが属性から削除される。属性の最後に残っている値が削除されると、属性自体が削除されたのと同じ結果になる

以下の例では、thisorgunit/service/printer のすべての属性が削除されます。

```
# fnattr -d thisorgunit/service/printer
```

属性の内容を表示する

属性の内容を表示するには、-l オプションを使用します。構文は以下のとおりです。

```
fnattr -l name [[-O | -U] identifier]
```

以下の例では、thisorgunit/service/printer の model 属性の値が表示されます。

```
# fnattr -l thisorgunit/service/printer model
laser
postscript
```

識別子を指定しない場合、オブジェクトのすべての属性の内容が表示されます。

属性を変更する

属性値を変更するには、-m オプションを使用します。構文は以下のとおりです。

```
fnattr -m name [-O | -U] identifier old_value new_value
```

以下の例では、postscript という値が laser に変更されます。

```
# fnattr -m thisorgunit/service/printer model postscript laser
```

指定した値だけが変更されます。その属性に関連付けられたその他の属性と値は変更されません。

その他のオプション

-O オプションを指定した場合、属性識別子の形式には、ASN.1 の整数を並べてドットで区切る形式 (FN_ID_ISO_OID_STRING、10 進数を並べてドットで区切る形式) を使用します。

-U オプションを指定した場合、属性識別子の形式には、DCE UUID 文字列 (FN_ID_DCE_UUID) を使用します。

FNS とエンタープライズレベルのネームサービス

エンタープライズレベルのネームサービスは、組織内のオブジェクトをネーミングするために使用されます。現在 FNS は、NIS、NIS+、およびローカルファイルに対して、3 つのエンタープライズレベルのネームサービスをサポートしています。

エンタープライズレベルのネームサービスを選択する

fncreate コマンドを使用して FNS 名前空間を初めて設定および構成するときは、名前空間の設定方法について 507 ページの「FNS 用の名前空間の準備」 を参照してください。正しいデフォルトのネームサービスが、各マシンに対して自動的に選択されます。

マシンの主要なエンタープライズレベルのネームサービスを後で変更する場合は、そのマシンで fnselect コマンドを実行する必要があります。詳細については、542 ページの「ネームサービスを選択する」を参照してください。

FNS とネームサービスとの整合性

タスクの 1 つにシステム管理者の機能が割り当てられていて、FNS とその下層のネームサービス間の整合性を維持しています。これは、ネームサービスのファイル、マップ、またはテーブルと FNS コンテキストが正しく対応しているかどうかをチェックするということです。

507 ページの「FNS 用の名前空間の準備」の説明に従って、fncreate コマンドを使用して FNS 名前空間を初めて設定および構成すると、 fncreate により FNS コンテキストが適切に作成され、配下のネームサービスデータと整合性が確保されます。FNS コンテキストが設定された後、ユーザー、ホスト、プリンタなどがシステムに追加または削除されるときに、この対応関係は維持される必要があります。以下の項では、FNS とネームサービスとの整合性を維持する方法について説明します。

FNS と Solstice AdminSuite

Solstice AdminSuite 製品で、基本的なネームサービスでのユーザーとホストの情報を追加、変更、削除できます。AdminSuite ツールでは対応する FNS 名前空間が自動的に更新されるため、この方法をお勧めします。

ネーミングの不一致をチェックする

Solstice AdminSuite 製品を使用せずに FNS または主要なネームサービスを更新するときに、不一致が発生した場合は、FNS ツールの fncheck を使用して解決します。fncheck コマンドは、FNS の hostname と user のコンテキストと、次のものとの不一致をチェックします。

- 「NIS+」。NIS+ の hosts.org_dir および passwd.org_dir システムテーブル
- 「NIS」。NIS の hosts.byname および passwd.byname マップ
- 「ファイル」。etc/hosts および etc/passwd ファイル

fncheck コマンドは、FNS 名前空間にあってネームサービスのデータにないホストとユーザー名およびネームサービスのデータにあって FNS 名前空間にないホストとユーザー名を表示します。

コマンド構文は以下の通りです。

```
fncheck [-r] [-s] [-u] [-t hostname|username] [domain_name]
```

表 **25–21** fncheck コマンドオプション

オプション	説明
domain	コマンドを実行しているドメイン以外の NIS+ ドメインにコマンドを適用する
-t	チェックするコンテキストのタイプを指定する。許容されるタイプは、hostname または username
-s	FNS 名前空間にない名前空間のデータセットからホスト名とユーザー名を表示する
-r	対応する名前空間のデータセットにないエントリを持たない FNS 名前空間からホスト名またはユーザー名を表示する
-u	関連した名前空間のデータセットにある情報に基づいて FNS 名前空間を更新する

-t オプションは、チェックするコンテキスト (ホストまたはユーザー) を指定するために使用します。-t オプションを省略した場合は、hostname と username のコンテキストの両方がチェックされます。

-r オプションを -u オプションとともに使用すると、FNS コンテキストにしか表示されない項目が FNS コンテキストから削除されます。-s オプションを -u オプションとともに使用すると、名前空間のデータセットにしか表示されない項目が FNS コンテキストに追加されます。-r と -s のどちらも指定しない場合は、項目が FNS コンテキストに追加および削除され、対応する名前空間のデータとの整合性が保たれます。

ネームサービスを選択する

FNS がマシンに対する初期コンテキストに割り当てを構成するときには、特定のネームサービスに基づいて行います。

FNS では fnselect コマンドで使用するネームサービスを選択できます。fnselect で指定したネームサービスの設定は、マシン全体、そのマシンで動作するすべてのアプリケーション、およびそのマシンにログインしたすべてのユーザーに影響します。

スーパーユーザーだけが fnselect を実行できます。コマンド構文は以下の通りです。

```
fnselect [-D] [namesvc]
```

表 **25–22** fnselect コマンドオプション

オプション	説明
namesvc	選択するネームサービス。必ず次のどれかになる。default、nisplus、nis または files
-D	FNS 初期コンテキストを生成するために使用されるネームサービスを表示する

たとえば、マシンのネームサービスとして NIS+ を選択するには以下のコマンドを使用します。

```
#fnselect nisplus
```

たとえば、マシンのネームサービスとして default を選択し、FNS 初期コンテキストを生成するために使用されるサービスの名前を印刷するには次のように入力します。

```
#fnselect -D default
```

デフォルトのネームサービス

fnselect でネームサービスを指定しない場合、FNS はデフォルトのネームサービスを使用します。デフォルトのネームサービスは、マシンの使用するネームサービスに基づいて FNS で決定されます。マシンが NIS+ クライアントである場合は、FNS は NIS+ をネームサービスとして使用します。マシンが NIS クライアントである場合は、FNS は NIS を使用します。マシンが NIS+ および NIS クライアントのどちらでもない場合は、FNS は /etc ファイルをマシンのネームサービスとして使用します。

NIS+ と NIS が共存する場合

ごくまれに、NIS+ と NIS ベースのコンテキストの両方にアクセスする必要が生じることがあります。たとえば、それ自体が NIS+ クライアントである NIS サーバーを稼動している場合です。この場合、fnselect コマンドを使用して、使用するエンタープライズレベルのネームサービスを選択します。

FNS と NIS+ の詳細情報

この節では、NIS+ オブジェクトと FNS オブジェクトの関係の詳細について説明します。この情報は、FNS オブジェクトのアクセス制御を変更するときに有効です。

注 – 以下を参照してください。

- 547 ページの「NIS から NIS+ への変更」
- 549 ページの「ファイルベースのネーミングから NIS または NIS+ への変更」

FNS コンテキストを NIS+ オブジェクトにマップする

FNS コンテキストは、NIS+ オブジェクトに格納されます。組織に関連するすべてのコンテキストは、関連する NIS+ ドメインの `ctx_dir` ディレクトリに格納されます。`ctx_dir` ディレクトリは、同じドメインの `org_dir` と同じレベルにあります。すなわち、FNS と同時に実行される時には、それぞれの NIS+ ドメインまたはサブドメインには対応する `org_dir`、`groups_dir`、および `ctx_dir` というディレクトリオブジェクトが存在します。

`fnlookup` または `fnlist` のコマンドで `-v` オプションを使用して、リファレンスについての詳細な説明を表示します。内部名フィールドに、対応する NIS+ オブジェクトの名前が表示されます。

NIS+ コマンドを使用して FNS 構造を表示する

NIS+ コマンドの `nisls` を使用して、FNS で使用される NIS+ オブジェクトを表示できます。たとえば、次のコマンドでは、NIS+ ドメインのディレクトリおよび `ctx_dir` サブディレクトリの内容が表示されます。

```
# nisls doc.com.
doc.com.:
manf
sales
groups_dir
org_dir
ctx_dir

# nisls ctx_dir.doc.com.
ctx_dir.DOC.COM.:
fns
fns_user
fns_host
fns_host_alto
fns_host_mladd
fns_host_elvira
fns_user_jjones
fns_user_jsmith
fns_user_aw
```

`niscat` コマンドを使用して、`fns_hosts` テーブルの内容を表示します。

```
# niscat fns_host.ctx_dir
altair *BINARY* *BINARY*
cygnus *BINARY* *BINARY*
centauri *BINARY* *BINARY*
```

アクセス制御をチェックする

niscat コマンドを -o オプションつきで使用して、コンテキストのアクセス制御を確認します。特定の割り当てのアクセス制御を確認するには、親コンテキストの割り当てテーブルにある割り当てエントリの名前 (つまり、fnlookup -v および fnlist -v の出力の内部名フィールドに表示される名前) を使用します。

```
# niscat -o fns_host.ctx_dir
Object Name          :    fns_host
Owner                :    alto.doc.com.
Group                   :       admin.doc.com.
Domain                     :    ctx_dir.doc.com.
Access Rights        :    r-c-rmcdrmcdr-c-
Time to Live         :    53:0:56
Object Type          :    TABLE
Table Type           :    H
Number of Columns    :    3
Character Separator
Search Path          :
Columns              :
[0]    Name          :    atomicname
    Attributes     :    (検索可能、テキストデータ、大文字・小文字の区別なし)
    Access Rights     :    r-c-rmcdrmcdr-c-
[1]    Name          :    reference
    Attributes     :    (2 進データ)
    Access Rights     :    r-c-rmcdrmcdr-c-
[2]    Name          :    flags
    Attributes     :    (2 進データ)
    Access Rights     :    r-c-rmcdrmcdr-c-

# niscat -o "[atomicname=altair],fns_host.ctx_dir"
Object Name    :    fns_host
Owner        :    altair.doc.com.
Group        :       admin.doc.com.
Domain       :    ctx_dir.doc.com.
Access Rights    :    r-c-rmcdrmcdr-c-
Time to Live    :    12:0:0
Object Type    :    ENTRY
 Entry data of type H
 [1] - [5 bytes] 'alto'
 [2] - [104 bytes] '0x00 ...'
 [3] - [1 bytes] 0x01
```

niscat コマンドに関する追加情報については、370 ページの「niscat コマンド」を参照してください。

特定のコンテキストのアクセス権または所有権を変更するには、次のコマンドを使用します：

- nischown
- nischmod
- nischgrp

操作が影響するオブジェクトに応じて、割り当てエントリまたは割り当てテーブルのどちらかを引数として与えます。

FNS と NIS の詳細情報

ここでは、NIS と FNS の関係についての特定情報を説明します。

NIS と FNS のマップと Makefile

FNS は、NIS マスターおよびスレーブのサーバー上の /var/yp/*domainname* ディレクトリに格納されている 6 つのマップを使用します。

- fns_host.ctx。ホスト属性とサブコンテキストのデータを格納する。これが最初に作成されると、hosts.byname マップから情報が得られる
- fns_host.ctx。ユーザー属性とサブコンテキストのデータを格納する。これが最初に作成されると、passwd.byname マップから情報が得られる
- fns_org.ctx。エンタープライズ属性とサブコンテキストのデータを格納する
- fns_host.attr。属性による検索のためのホスト属性を格納する
- fns_user.attr。属性による検索のためのユーザー属性を格納する
- fns_org.attr。属性による検索のためのエンタープライズ属性を格納する

ホスト、ユーザー、およびエンタープライズのサービスとファイルのコンテキスト情報は、それぞれ fns_host.ctx、fns_user.ctx、および fns_org.ctx のマップに格納されます。プリンタのコンテキスト情報は、他のサービスのコンテキスト情報と同じマップに格納されます。しかし、古い printers.conf.byname マップはここでもサポートされます。

サイトは、エンタープライズのサブコンテキストであり、サイトのコンテキスト情報は fns_org.ctx マップに格納されます。

注 – これらの FNS マップは、直接編集しないでください。fncreate、fndestroy、fnbind、fnunbind、fnrename、fnattr、fnlookup、および fnlist などの適切な FNS コマンドを実行して、これらのマップで変更または作業を行います。これらのコマンドは、必ず NIS マスターサーバーで実行します。スレーブサーバーまたはクライアントマシンでこれらを実行できません。

FNS マップファイルは、/var/yp/*domainname* ディレクトリにあります。/var/yp にある NIS Makefile は変更され、/etc/fn/*domainname* にある FNS Makefile が認識されます。

大型 FNS コンテキスト

NIS では、NIS マップに含まれるエントリ数に 64K という制限があります。サービスおよびプリンタのコンテキストだけが各オブジェクト (ホストまたはユーザー) に作成された場合、ユーザーまたはホストの数が 7K を超えると、エントリ数がその制限に達します。通常行われるように、ホストまたはユーザーに追加のコンテキストが作成された場合、ずっと少ないホストまたはユーザーでエントリ数が 64,000 の上限に達します。

FNS は、古いマップが最大サイズに達したら新規マップを自動的に作成して、この問題を解決します。各新規マップは、マップ名に数字の接尾辞を追加して識別されます。たとえば、2 つめの fns_user.ctx マップが作成されると、そのマップには名前 fns_user_0.ctx という名前が与えられます。3 つめのマップが必要になると、そのマップには名前 fns_user_1.ctx という名前が与えられます。追加のマップが作成されるにつれて、数字は毎回増加していきます。

プリンタの下位互換

Solaris 2.5 では、FNS は、printers.conf.byname という名前のマップを使用して、組織コンテキストに対して NIS のプリンタのネーミングをサポートします。現在の Solaris では、組織コンテキストのプリンタサポートは、fns_org.ctx マップに保持されています。つまり、ここでの fncreate_printer コマンドでは fns_org.ctx マップが変更され、printers.conf.byname マップは変更されません。

NIS から NIS+ への変更

fncopy コマンドは、エンタープライズレベルのネームサービスを NIS から NIS+ に変更するときに、FNS 関連の側面を処理します。このコマンドは、NIS ベースの FNS コンテキストを NIS+ ベースのコンテキストにコピーして変換します。

コマンド構文は以下の通りです。

fncopy [-i *oldsvc* -o *newsvc*] [-f *filename*] *oldctx newctx*

表 **25–23** fncopyコマンドオプション

オプション	説明
`-i` *oldsvc*	ソースのネームサービス。`nis` または `files` だけが指定される
`-o` *newsvc*	目標のネームサービス。 `nisplus` または `nis` だけが指定される
`-f` *filename*	コピーされる FNS コンテキストを表示するファイル名
oldctx	コピーされる古い FNS コンテキスト
newctx	目標の新規 FNS コンテキスト

たとえば、ファイル `/etc/sales_users` に表示されるコンテキストを、NIS ベースのネームサービスの `doc.com` ドメインから NIS+ ネームサービスの `sales.doc.com` ドメインにコピーするには、次のように入力します。

```
fncopy -i nis -o nisplus -f /etc/sales_users org/sales.doc.com/user
```

FNS とファイルベースのネーミングの詳細情報

この節では、ファイルベースのネーミングと FNS の関係についての特定の情報を説明します。

FNS ファイル

FNS では、各マシンの `/var/fn` ディレクトリに格納された新規ファイルが使用されます。通常 `/var/fn` ディレクトリは各マシンに格納されていますが、NFS 経由で中央の `/var/fn` ディレクトリをマウントしたり、エクスポートしたりできます。

新規 FNS ファイルを以下に示します。

- `fns_host.ctx`。ホスト属性とサブコンテキストのデータを格納する。これが最初に作成されると、`/etc/hosts` ファイルから情報が得られる
- `fns_user.ctx`。ユーザー属性とサブコンテキストのデータを格納する。これが最初に作成されると、`/etc/passwd` ファイルから情報が得られる
- `fns_org.ctx`。エンタープライズ属性とサブコンテキストのデータを格納する
- `fns_host.attr`。属性による検索のためのホスト属性を格納する
- `fns_user.attr`。属性による検索のためのユーザー属性を格納する
- `fns_org.attr`。属性による検索のためのエンタープライズ属性を格納する

- ユーザーのサブコンテキストと属性の情報は、各ユーザーが所有する別個の /var/fn ファイルに格納される。これにより、ユーザーは FNS コマンドを使用して、自分のデータを変更できる。これらのユーザー固有のファイルは、fns_user_*username*.ctx とネーミングされる。ここでの *username* は、個々のユーザーのログイン ID になる

ホスト、ユーザー、およびエンタープライズのサービスとファイルのコンテキスト情報は、それぞれ fns_host.ctx、fns_user.ctx、および fns_org.ctx のファイルに格納されます。プリンタのコンテキスト情報は、他のサービスのコンテキスト情報と同じファイルに格納されます。

サイトは、エンタープライズのサブコンテキストであり、サイトのコンテキスト情報は fns_org.ctx ファイルに格納されます。

注 – これらの FNS ファイルは、直接編集しないでください。fncreate、fndestroy、fnbind、fnunbind、fnrename、fnattr、fnlookup、および fnlist などの適切な FNS コマンドを実行して、これらのファイルで変更または作業を行います。スーパーユーザーとしてこれらのコマンドを実行すると、ホスト、サイト、および組織単位などの、コマンドが適用されるコンテキストが影響されます。ユーザーとしてこれらのコマンドを実行すると、自分のユーザーのサブコンテキストだけが影響されます。

ファイルベースのネーミングから NIS または NIS+ への変更

fncopy コマンドは、エンタープライズレベルのネームサービスをファイルから NIS または NIS+ に変更するときに、FNS 関連の側面を処理します。このコマンドは、ファイルベースの FNS コンテキストを NIS または NIS+ ベースのコンテキストにコピーして変換します。

コマンド構文は以下の通りです。

```
fncopy [-i oldsvc -o newsvc] [-f filename] oldctx newctx
```

たとえば、ファイル /etc/host_list に表示される内容を NIS+ ネームサービスの doc.com ドメインにコピーするには、次のように入力します。

```
fncopy -i files -o nisplus -f /etc/host_list //doc.com/host
```

プリンタの下位互換

Solaris 2.5 では、FNS は、printers.conf.byname という名前のファイルを使用して、組織コンテキストに対してファイルへのプリンタのネーミングをサポートします。現在の Solaris では、組織コンテキストのプリンタサポートは、fns_org.ctx マップに保持されています。つまり、ここでの fncreate_printer コマンドでは fns_org.ctx マップが変更され、printers.conf.byname マップは変更されません。

ファイルコンテキストの管理

ファイルコンテキストには、以下のようなものがあります。

- fncreate_fs コマンドを使用して作成したもの (550 ページの「fncreate_fs を使ってファイルコンテキストを作成する」を参照)
- fnlist コマンド (529 ページの「コンテキストの内容を表示する」を参照)、あるいは fnlookup コマンド (528 ページの「割り当てに関する情報を表示する」を参照) を使用して、検索されたもの
- fnunbind コマンド (534 ページの「複合名を削除する」を参照)、あるいは fndestroy コマンド (534 ページの「コンテキストを削除する」を参照) を使用して、切り取られたものあるいは削除されたもの

fncreate_fs を使ってファイルコンテキストを作成する

fncreate_fs コマンドにより、組織やサイト用にファイルコンテキストが作成できます。fncreate コマンドによって作成されたホストやユーザーのためのデフォルトのファイルコンテキストを無効にするために使用されることもあります。

fncreate_fs コマンドの使用には、以下のような 2 つの方法があります。

- 「入力ファイル」入力ファイルによって、ファイルコンテキストの割り当てが提供されます (551 ページの「入力ファイルを使って、ファイルコンテキストを作成する」を参照)。
- 「コマンド行」コマンド行によって、ファイルコンテキストの割り当てが作成されます (553 ページの「コマンド行の入力によりファイルコンテキストを作成する」を参照)。

fncreate_fs の 2 つの方法には、以下のような構文があります。

```
fncreate_fs [-v] [-r] -f file composite_name
fncreate_fs [-v] [-r] composite_name [options] [location...]
```

表 **25–24** `fncreate_fs` コマンドのオプション

オプション	説明
composite_name	ファイルコンテキストの複合名
`-f` *file*	*file* という名前の入力ファイルを使用する
options	マウントのオプション
location	マウントする位置
`-v`	詳細出力に設定。作成および修正されるコンテキストに関する情報を表示
`-r`	*composite_name* で名前付きコンテキスト、およびそのサブコンテキストのすべての割り当てを、入力で指定したものだけで置き換える。これは、コンテキスト (およびそのサブコンテキスト) を削除し、このオプションなしで、`fncreate_fs` を実行するのと同じ。`-r` オプションは、注意して使用する必要がある

`fncreate_fs` コマンドは、FNS コンテキスト、および `onc_fn_fs` リファレンスタイプの割り当てを操作します。これは、`onc_fn_fs_mount` タイプのアドレスを使って、それぞれのリモートのマウント先を表わします。このタイプのアドレスに関連するデータは、XDR で符号化された 1 つの文字列で示した、対応するマウントのオプションやマウント先になっています。

入力ファイルを使って、ファイルコンテキストを作成する

入力ファイルには、`composite_name` のコンテキストで割り当てられる名前や値が入っています。この形式は、間接自動マウントマップの形式に基づいてはいますが、全く同じではありません。入力ファイルには、以下のような形式のエントリが 1 つあるいは複数含まれています。

name [*options*] [*location...*]

引数の意味はそれぞれ以下のとおりです。

- *name* には、リファレンス名が入ります。*name* フィールドは、単純な名前のこともあれば、あるいはスラッシュで区切った階層構造を示す名前の場合もあります。さらに (`.`) のようにドットを使って示すこともあり、この場合リファレンスは、*composite_name* に直接割り当てられます。
- *options* には、マウントのオプションがあれば、それが入ります。*options* フィールドは、ハイフン (-) で始まります。この後に、マウントオプションのコンマで区切ったリスト (スペースなし) が続き、これはディレクトリをマウントするのに使用されます。またこれらのオプションは、*composite_name/name* サブコンテキスト

にも適用されます。なお、*composite_name/name* は、それ自身のマウントオプションは指定しません。

- *location* には、マウントの位置が入ります。*location* フィールドでは、*composite _name/name* のファイルを提供する 1 つあるいは複数のホストを指定します。簡単な NFS マウントでは、*location* は、以下のような形式をとります。

  host : *path*

- *host* には、ファイルシステムをマウントするサーバー名が入ります。*path* には、マウントするディレクトリのパス名が入ります。

各エントリについて、マウントの位置や対応するマウントのオプションのリファレンスは、*composite_name*/*name* に割り当てられます。

options と *location* の両方が省略されると、リファレンスは *composite_name/name* に割り当てられません。既存のリファレンスもすべて割り当てられません。

たとえば、kuanda のファイルシステムを図 25–3 に示すように、ホスト `altair` からディレクトリ `/export/home/kuanda` を NFS マウントするとしましょう。コマンドは以下のように実行されます。

```
% fncreate_fs -f infile user/kuanda/fs
```

ここでは、以下を含む `infile` を使用します。

```
. altair:/export/home/kuanda
```

図 25–4 に示されるような、複数のサーバーに分散された複雑なファイルシステムを設定する場合、以下のコマンドを実行します。

```
% fncreate_fs -f infile org/sales/fs
```

ここでは、以下を含む `infile` を使用します。

```
tools/db         altair:/export/db
project             altair:/export/proj
project/lib             altair:/export/lib
project/src         deneb:/export/src
```

プロジェクトやそのサブコンテキストの `src` や `lib` の NFS マウントを、読み取り専用に変更する場合、`infile` を以下のように変更します。

```
tools/db                  svr1:/export/db
project          -ro            svr1:/export/projproject/lib
altair:/export/lib
project/src                 svr2:/export/src
```

`-ro` は、3 行目と 4 行目では必要ありません。なぜならば、`src` および `lib` は、`project` のサブコンテキストであり、これらは、上から `-ro` マウントオプションを継承するからです。

以下の入力ファイルは、`org/sales/fs/project/src` を除いて、すべてのマウントを読み取り専用にします。

```
.       -ro
tools/db    svr1:/export/db
project     svr1:/export/proj
project/lib altair:/export/lib
project/src      -rw      svr2:/export/src
```

コマンド行の入力によりファイルコンテキストを作成する

`fncreate_fs` コマンドにより、以下のように割り当ての説明をコマンド行で入れることもできます。

`fncreate_fs` *composite_name* [*mmount_options*] [*mount_location ...*]

これは、コマンドの入力ファイル書式を使用し、キーボードから個々の割り当てを入力するのと同じです。先の `kuanda` のファイルシステムを設定した例は、コマンド行からは、以下のように設定できます。

```
% fncreate_fs user/kuanda/fs altair:/export/home/kuanda
```

同様に、図 25–4 に示した階層構造は、以下のような一連のコマンドを実行することによって、設定できます。

```
% fncreate_fs org/sales/fs/tools/db altair:/export/db
% fncreate_fs org/sales/fs/project altair:/export/proj
% fncreate_fs org/sales/fs/project/lib altair:/export/lib
% fncreate_fs org/sales/fs/project/src deneb:/export/src
```

これら 3 つすべてのマウントを読み取り専用にするには、以下のコマンドを実行します。

```
% fncreate_fs org/sales/fs -ro
```

詳細入力フォーマット

以下の 2 つの項で述べる内容は、入力ファイル、およびコマンド行入力の両方の形式に適用されます。

多重マウントの位置

複数の *location* フィールドが、機能的には同じである複数の位置からエクスポートされた NFS ファイルシステムに対して指定できます。

```
% fncreate_fs org/sales/fs altair:/sales cygnus:/sales
```

オートマウンタが、選択肢の中から最適のサーバーを選択します。リストの中のいくつかの位置が同じパス名を共有している場合、これらは以下のようにコンマで区切ったホスト名のリストを使用して、結合されます。

```
% fncreate_fs org/sales/fs altair,cygnus:/sales
```

ホストは、丸括弧で囲った整数の形でホストに追加された重み係数により加重されることがあります。この場合、数字が小さければ小さいほど、望ましいサーバーであることを示します。デフォルトの重み係数は、ゼロ (もっとも望ましい) です。マイナスの数字は使用できません。

以下の例では、cygnus が望ましいサーバーになっていることを示す 1 つの方法を表わしています。

```
% fncreate_fs org/sales/fs altair(2),cygnus(1):/sales
```

変数の置き換え

前に $ の付く変数名を、fncreate_fs の *options* や *location* のフィールドで使用できます。たとえば、マウントの位置は、以下のように指定できます。

```
altair:/export/$CPU
```

対応するファイルシステムにマウントする際、オートマウンタにより、これらの変数がクライアントに固有の値に置き換えられます。上の例の場合、$CPU は、uname -p の出力結果、たとえば sparc で置き換えられます。

下位互換入力フォーマット

さらにオートマウントマップとの互換として、以下の入力ファイル書式も fncreate_fs で使用できます。

name [*mount_options*] [*mount_location ...*] \
/offset1 [*mount_options1*] *mount_location1 ...* \
/offset2 [*mount_options2*] *mount_location2 ...* \
. . .

ここでは、各 *offset* フィールドは、スラッシュで区切った階層構造になっています。バックスラッシュ (\) は、1 つの長い行の続きであることを示します。これは、以下を行うのと同じことです。

name [*mount_options*] [*mount_location ...*] \
name/offset1 [*mount_options1*] *mount_location1 ...* \
name/offset2 [*mount_options2*] *mount_location2 ...* . . .

最初の行は、*mount_options* と *mount_location* の両方が省略されている場合、省略します。この書式は、互換のためだけのものです。これ以外の機能はないので、あまり使用しないでください。

FNS および XFN ポリシーの概要

XFN は、フェデレートされた名前空間にあるオブジェクトをネーミングするためのポリシーを定義します。これらのポリシーは、次のような目的を持っています。

- 統一性のある複合名を簡単に作成する
- アプリケーションおよびサービスでネーミングの一貫性を持たせる
- 簡単で十分な機能をもつポリシーを提供し、アプリケーションで特定環境に特別のポリシーを作成したり、実行したりする必要をなくす
- アプリケーションの移植性を拡張する
- 異機種システム混在コンピュータ環境でのプラットフォーム同士の相互運用性を高める

FNS ポリシーで指定されるもの

FNS ポリシーには、すべての XFN ポリシーと Solaris 環境用の拡張機能が含まれます。

現在、コンピュータ環境は世界的な規模となり、広範囲なサービスを提供しています。ユーザーは、どのレベルのコンピュータ環境のサービスにもアクセスできます。FNS ポリシーは、グローバル、エンタープライズ、アプリケーションの 3 つのレベルのサービスに共通の枠組みを提供します。

FNS は、ネームサービスをどのように調整して使用するかについて次のようなポリシーをアプリケーション側に提供します。

- エンタープライズ名前空間をフェデレートして、グローバル名前空間でアクセス可能にする方法を指定する
- 各プロセスの初期コンテキストにある名前および割り当てを指定する
- 組織、ホスト、ユーザー、サイト、ファイル、サービスなどのエンタープライズのオブジェクトに対するネームサービス
- 組織、ホスト、サイト、ファイル、サービスのエンタープライズのオブジェクト間の関係を定義する
- これらのエンタープライズのオブジェクトのリファレンスに使用される名前の構文を指定する

FNS ポリシーで指定されないもの

FNS ポリシーでは、次のものは指定されません。

- ネームサービスで使用される実際の名前
- アプリケーション内のネーミング。アプリケーションレベルのネーミングは、個別のアプリケーションまたは関連アプリケーションのグループで行われる
- オブジェクトがネーミングされた後に使用する属性

エンタープライズ名前空間のポリシー

FNS ポリシーでは、エンタープライズ内の名前空間のタイプおよび配列と、アプリケーションが名前空間を使用する方法が指定されます。たとえば、名前空間が他のどの名前空間と関連するかが指定されます。ここで説明する FNS ポリシーには、XFN ポリシーへの拡張機能があります。これらは、注釈によって明確に定義されます。

デフォルトの FNS エンタープライズ名前空間

FNS エンタープライズのポリシーでは、名前空間内でのエンタープライズのオブジェクトの配列が処理されます。各エンタープライズのオブジェクトは、独自の名前空間を持っています。

デフォルトでは、FNS には 7 つのエンタープライズオブジェクトと対応する名前空間があります。

- 「組織」(`orgunit`)。部、センター、課などのエンティティ。サイト、ホスト、ユーザー、サービスには、組織に関連する名前を付けることができる。組織に対する XFN 用語は、「組織単位」となる。初期コンテキストで使用されたときは、識別子 `org` は、`orgunit` の別名として使用できるFNS組織の名前空間
- 「サイト」(`site`)。建物、建物内のマシン、建物内の会議室などの物理的な位置。サイトは、それらと関連するファイルおよびサービスを持つ
- 「ホスト」(`host`)。マシン。ホストは、それらと関連するファイルおよびサービスを持つFNS ホストの名前空間
- 「ユーザー」(`user`)。人間のユーザー。ユーザーは、それらと関連するファイルおよびサービスを持つ
- 「ファイル」(`fs`)。ファイルシステム内のファイル
- 「サービス」(`service`)。プリンタ、FAX、メール、電子カレンダなどのサービス
- 「プリンタ」(`service/printer`)。プリンタの名前空間は、サービスの名前空間に従属する

これらの名前空間に適用されるポリシーは、表 25–26 に要約しています。

エンタープライズ名前空間の識別子

エンタープライズ名前空間は、フェデレートされた名前空間の原子名によって参照されます。

XFN では、先行する下線 (_) 文字を使用して、エンタープライズ名前空間の識別子が示されます。たとえば、`_site` となります。FNS では、下線 (_) なしの識別子の使用もサポートされます。下線なしの名前は、XFN ポリシーの範疇には入りません。`site` および `printer` のコンテキストも、XFN ポリシーの範疇には入りません。これらの名前は、表 25–25 に挙げられています。

表 25–25 エンタープライズでのエンタープライズ名前空間の識別子

名前空間	**XFN** 識別子	**FNS** 識別子	解釈処理
組織	_orgunit	orgunit または org	組織単位をネーミングするためのコンテキスト
サイト	_site	site	サイトをネーミングするためのコンテキスト
ホスト	_host	host	ホストをネーミングするためのコンテキスト
ユーザー	_user	user	ユーザーをネーミングするのためのコンテキスト
ファイルシステム	_fs	fs	ファイルをネーミングするためのコンテキスト
サービス	_service	service	サービスをネーミングするためのコンテキスト
プリンタ		printer	プリンタをネーミングするためのコンテキスト (サービスの名前空間に従属)

注 – XFN の用語では、先行する下線のある名前は、「正規」名前空間識別子です。下線のない名前は、Solaris オペレーティング環境用に「カスタマイズ」された名前空間識別子です。これらのカスタマイズされた名前空間識別子に `printer` を追加しても、Solaris 以外の環境では認識されません。正規の名前空間識別子は、他の環境でも常に認識され、互換性があります。

構成要素の区切り文字

XFN 構成要素の区切り文字 (/) で、名前空間識別子を区切ります。たとえば、名前空間識別子 `orgunit` を組織単位名 `west.sales` とともに作成すると、複合名は `orgunit/west.sales` となります。

デフォルトの FNS 名前空間

FNS では、次の 7 つの名前空間が提供されます。

- 組織
- サイト
- ホスト
- ユーザー
- ファイル
- サービス
- プリンタ

組織単位の名前空間

組織単位の名前空間では、エンタープライズのサブ単位をネーミングするために階層になった名前空間が提供されます。各組識単位名は組織単位を表す「組織単位コンテキスト」と結合されています。組織単位名は、接頭辞 `org/`、`orgunit/`、または `_orgunit/` によってネーミングされます。短縮形別名の `org/` は、内部コンテキストだけで使用され、複合名の途中では使用されません。および、562 ページの「複合名の例」を参照してください。

NIS+ 環境

NIS+ 環境では、組織単位は NIS+ のドメインおよびサブドメインに対応します。

NIS+ では、組織単位は必ずドメインおよびサブドメインにマップされます。それぞれの NIS+ ドメインおよびサブドメインに組織単位を与える必要があります。ドメインまたはサブドメインに論理、組織ユニットを与えることはできません。つまり、NIS+ ドメインまたはサブドメインを小規模な組織単位に分割することはできません。1 つの NIS+ ドメイン (`doc.com.`) と 2 つのサブドメイン (`sales.doc.com.` と `manf.doc.com.`) が存在する場合は、ドメインごとに FNS 組織単位を割り当てる必要があります。

組織単位は、ドットで区切られた右から左への複合名を使用してネーミングされます。ここでは、各原子要素がより大きな単位内の組織単位をネーミングします。たとえば、`org/sales.doc.com.` という名前は、`doc.com.` という名前のより大きな単位内にある `sales` をネーミングします。この例では、`sales` は、`doc.com.` の NIS+ サブドメインです。

組織単位名は、完全指定の NIS+ ドメインまたは相対名の NIS+ 名のどちらかになります。完全指定名には終端にドットが付きますが、相対名には付きません。そのため、終端のドットが組織名にある場合は、その名前は完全指定の NIS+ ドメイン名として処理されます。終端にドットがない場合は、その組織名は最上部の組織階層に対して相対的に解釈処理されます。たとえば、`orgunit/west.sales.doc.com.` は、`west` 組織単位をネーミングする完全指定名で、`_orgunit/west.sales` は、同じサブドメインをネーミングする相対的な名前です。

NIS 環境

NIS 環境では、NIS ドメインに対応する組織単位は各エンタープライズ 1 つずつあります。この `orgunit` には、`orgunit/`*domainname* という名前が付けられ、ここでの *domainname* は NIS ドメイン名です。たとえば、NIS ドメイン名が `doc.com` である場合は、組織単位は `org/doc.com` です。

NIS 環境では、空の文字列を組織単位の省略形として使用できます。したがって、`org//` は、`org/`*domainname* に相当します。

ファイルベースの環境

エンタープライズレベルのネームサービスがファイルベースであるときには、1 つだけの FNS 組織単位が存在し、サブ単位はありません。ファイルベースのネーミングで許容される唯一の組織単位は `org//` です。

サイトの名前空間

サイトの名前空間では、物理的位置でそのまま識別される地域名前空間が提供されます。たとえば、これらのオブジェクトには、キャンパスにある建物、ある階のマシンやプリンタ、建物内の会議室とその使用スケジュール、隣接するオフィスにいるユーザーなどがあります。サイト名は、接頭辞 `site/` または `_site/` で識別されます。

Solaris オペレーティング環境では、サイトは複合名を使用してネーミングされます。複合名では、各原子名がより大きなサイト内のサイトを表わします。サイト名の構文は、ドット区切りの右から左であり、もっとも一般的な位置記述からもっとも特定の位置記述に配列された構成要素が含まれます。たとえば、`_site/pine.bldg5` は、建物 5 にある Pine 会議室を表わし、`site/bldg7.alameda` は、あるエンタープライズの Alameda にある建物 7 を表わします。

ホストの名前空間

ホストの名前空間では、コンピュータをネーミングするための名前空間が提供されます。ホスト名は、接頭辞 `host/` または `_host/` で識別されます。たとえば、`host/deneb` は、`deneb` という名前のマシンを識別します。

ホストは、「ホスト名」コンテキストでネーミングされます。ホストコンテキストには、フラットな名前空間があり、ホストコンテキストへのホスト名の割り当てが含まれます。「ホストコンテキスト」では、そのホストで見つかったファイルやプリンタなどの、マシンに相対的なオブジェクトをネーミングできます。

Solaris オペレーティング環境において、ホスト名は Solaris のホスト名に相当します。単一のマシンに対する別名で同じコンテキストが共有されます。たとえば、`mail_server` という名前がマシン `deneb` および `altair` の別名である場合、`deneb` と `altair` の両方で、`mail_server` に作成されたコンテキストが共有されます。

ネットワーク資源は、必要に応じてホストに関連付けてネーミングする必要があります。たいていの場合、組織、ユーザー、サイトなどの構成要素に関連して資源をネーミングした方がわかりやすくなります。ホスト名に依存すると、ユーザーは、あいまいで変更される可能性のある情報を覚えなくてはならなくなります。たとえば、ハードウェアの変更、ファイルスペースの使用、ネットワークの再構成などのために、ユーザーのファイルがあるサーバーから他のサーバーに移動されてしまう場合もあります。さらに、ユーザーは、同じファイルサーバーを共有することが多くなり、ファイルがホストに関連付けてネーミングされた場合に混乱を招きます。しかし、ファイルがユーザーに関連付けてネーミングされた場合は、このような変更はファイルのネーミング方法に影響しません。

ホスト名を使用した方が適切な場合もあります。たとえば、資源が特定のマシンだけで使用可能であり、他の構成要素に関連付けてその資源をネーミングする論理的方法がない場合には、ホストに関連付けてネーミングする意味があります。または、ファイルシステムでは、ファイルが多くのユーザーに共有されている場合、格納されているマシンに関連付けてファイルをネーミングする意味があります。

ユーザーの名前空間

ユーザーの名前空間では、コンピュータ環境内のユーザーをネーミングするための名前空間が提供されます。ユーザー名は、接頭辞 `user/` または `_user/` で識別されます。

ユーザーは、「ユーザーコンテキスト」でネーミングされます。ユーザーコンテキストには、単一レベルの名前空間があり、「ユーザーコンテキスト」へのユーザー名の割り当てが含まれます。ユーザーコンテキストでは、ユーザーに関連するファイル、サービス、資源などのオブジェクトをユーザーに関連付けてネーミングすることができます。

Solaris オペレーティング環境において、ユーザー名は Solaris のログイン ID に相当します。たとえば、`_user/inga` は、ログイン ID が `inga` のユーザーを識別します。

ファイルの名前空間

ファイルの名前空間 (またはファイルシステム) では、ファイルをネーミングするための名前空間が提供されます。ファイル名は、接頭辞 `fs/` または `_fs/` で識別されます。たとえば、`fs/etc/motd` という名前では、`/etc` ディレクトリに格納されているファイル `motd` が識別されます。

ファイルの名前空間については、479 ページの「ファイルベースのネーミング」 で詳しく説明し、ファイルのコンテキストについては、550 ページの「ファイルコンテキストの管理」 で説明します。

サービスの名前空間

サービスの名前空間では、エンタープライズ内のオブジェクトに使用される、または関連するサービスのための名前空間が提供されます。このようなサービスには、電子カレンダ、FAX、メール、印刷などがあります。サービス名は、接頭辞 `service/` または `_service/` で識別されます。

Solaris オペレーティング環境では、サービスの名前空間は構造階層になっています。サービス名は、スラッシュ (/) で区切られた左から右の複合名です。サービスの名前空間を使用するアプリケーションは、この階層属性を利用し、そのアプリケーションのサブツリーを獲保します。たとえば、プリンタサービスはサービスの名前空間のサブツリー `printer` を獲保します。

FNS は、サービス名またはリファレンスタイプが選択される方法を指定します。これらは、サービスの名前空間を共有するサービス提供者によって決定されます。たとえば、カレンダサービスは、サービスのコンテキストにある名前 `_service/calendar` を使用してカレンダサービスをネーミングし、名前 `calendar` に割り当てられたものを決定します。

サービス名およびリファレンスの登録

米国 Sun Microsystems, Inc. は、`service` の名前空間の最初のレベルにある名前の登録を行います。新規名を登録するには、電子メールのリクエストを fns-register @sun.com まで送信するか、または書面で次の住所までご郵送ください。

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

名前の使用目的についての簡単な説明と、その名前に割り当てられるリファレンスのフォーマットの説明をお書き添えください。

プリンタの名前空間

プリンタの名前空間では、プリンタをネーミングするための名前空間が提供されます。プリンタの名前空間は、サービスの名前空間に関連 (従属) します。つまり、プリンタのサービスは、サービスの名前空間のサービスの一部です。プリンタ名は、接頭辞 `service/printer` または `_service/printer` で識別されます。たとえば、`service/printer/laser1` では、`laser1` という名前のプリンタが識別されます。

後続スラッシュの意味

後続の / は、次のネーミングシステムにあるオブジェクトをネーミングします。あるネーミングシステムから他のネーミングシステムに移動するときに必ず必要となります。

たとえば、名前 org/east.sales/service/printer では、org と east.sales の間のスラッシュは、上で説明した構成要素の区切り文字であり、組織名 (sales/) の最後の要素に続くスラッシュは、service 名前空間を組織単位名前空間から区切ります。したがって、org/west.sales/service/printer では、west.sales という組織単位の printer サービスがネーミングされます。

FNS 予約名

FNS は、表 25–25 に挙げたすべての原子名をそれ自身の使用のために名前空間識別子として予約します。たとえば、_orgunit、 org、 _site、site などが予約されています。この制限事項は、564 ページの「エンタープライズ名前空間の構造」にある名前空間の配列で定義されているように、名前空間識別子が現れるコンテキストに適用されます。それ以外の場合は、FNS は、他のコンテキストではこれらの原子名の使用を制限します。

たとえば、原子名 service は、user/fatima/service/calendar のように、ユーザー名に相対的な名前空間識別子として使用され、ユーザー fatima のサービスの名前空間のルートであることを意味します。FNS では、名前 user/ が割り当てられるコンテキストは、名前空間識別子にではなく、ユーザー名に指定されるため、user/service のようにユーザー名として名前 service を使用することが禁止されるわけではありません。この場合、service は明確にユーザー名として解釈されます。一方で、/user/mikhail を使用すると、ユーザー mikhail とファイル (またはサブディレクトリ) /user/mikhail の区別がつきにくくなるため、user という名前のディレクトリは作成しないでください。

複合名の例

この項では、FNS ポリシーに従う名前の例を示します。これらのポリシーについては、表 25–26 を参照してください。

組織名、サイト名、ユーザー名、ホスト名、ファイル名、サービス名 (calendar や printer など) の名前は、例として使用されています。これらの名前は、FNS ポリシーでは指定されません。

組織に関連する名前を作成する

組織の名前空間 (_orgunit、orgunit、または org) に関連付けることのできる名前空間は、user、host、service、fs、と site です。

たとえば :

- orgunit/doc.com/site/videoconf.bldg-5 では、組織 doc.com に関連するサイトの建物 5 にある会議室 videoconf がネーミングされる (orgunit に別名 org を使用して、org/doc.com/site/videoconf.bldg-5 を作成することもできる)

- orgunit/doc.com/user/mjones では、組織 doc.com のユーザー mjones がネーミングされる
- orgunit/doc.com/host/smptserver1 では、組織 doc.com に所属するマシン smptserver1 がネーミングされる
- orgunit/doc.com/fs/staff/agenda9604/ では、組織 doc.com に所属するファイル staff/agenda9604 がネーミングされる
- orgunit/doc.com/service/calendar では、組織 doc.com の calendar サービスがネーミングされます。ここでは、組織の会議スケジュールを管理する

ユーザーに関連する名前を作成する

user の名前空間に関連付けることができる名前空間は、service および fs です。

- user/helga/service/calendar では、helga という名前のユーザーの calendar サービスがネーミングされる
- user/helga/fs/tasklist では、ユーザー helga のホームディレクトリにあるファイル tasklist がネーミングされる

ホストに関連する名前を作成する

hosts の名前空間に関連付けることができる名前空間は、service および fs です。

- host/mailhop/service/mailbox では、マシン mailhop と関連する mailbox サービスがネーミングされる
- host/mailhop/fs/pub/saf/archives.96 では、マシン mailhop によってエクスポートされたファイルシステムのルートディレクトリにあるディレクトリ pub/saf/archives.96 がネーミングされる

サイトに関連する名前を作成する

sites の名前空間に関連付けることができる名前空間は、service および fs です。

- site/bldg-7.alameda/service/printer/speedy では、bldg-7.alameda サイトにあるプリンタ speedy がネーミングされる
- site/alameda/fs/usr/dist は、サイト alameda で使用可能なファイルディレクトリ usr/dist を指定する

サービスおよびファイルに関連する名前を作成する

ファイル (fs) またはサービス (service) の名前空間に関連付けることができる名前空間はありません。たとえば、/services/calendar/orgunit/doc.com などの名前は作成できません。つまり、ファイルまたはサービスの名前空間に関連付けた複合名は作成できません。もちろん、/user/esperanza/service/calendar のように、他の名前空間に関連付けてファイル名またはサービス名を作成することは可能です。

エンタープライズ名前空間の構造

FNS ポリシーでは、エンタープライズ名前空間の構造が定義されます。この構造の目的は、統一性のある名前を簡単に作成することです。このエンタープライズ名前空間構造には、2 つの主要なルールがあります。

- 狭い有効範囲のオブジェクトは、広い有効範囲のオブジェクトに関連してネーミングされる
- 名前空間識別子は、ある名前空間から次の名前空間への移行を表すために使用される

表 25–26 は、エンタープライズ名前空間を配列するための FNS ポリシーのまとめです。次の図は、これらの FNS ポリシーに準拠した名前空間の例です。

表 **25–26** フェデレートされたエンタープライズ名前空間用のポリシー

名前空間識別子	従属する名前空間	親コンテキスト	名前空間編成	構文

`orgunit`
`_orgunit`
`org`
	サイト、ユーザー、ホスト、ファイルシステム、サービス	エンタープライズのルート	階層	ドット区切り、右から左

`site` `_site`	サービス、ファイルシステム	エンタープライズのルート、組織単位	階層	ドット区切り、右から左
`user` `_user`	サービス、ファイルシステム	エンタープライズのルート、組織単位	フラット	Solaris ログイン名
`host` `_host`	サービス、ファイルシステム	エンタープライズのルート、組織単位	フラット	Solaris ホスト名
`service` `_service`	アプリケーション固有	エンタープライズのルート、組織単位、サイト、ユーザー、ホスト	階層	/ 区切り、左から右
`fs` `_fs`	なし	エンタープライズのルート、組織単位、サイト、ユーザー、ホスト	階層	/ 区切り、左から右
`printer`	なし	サービス	階層	/ 区切り、左から右

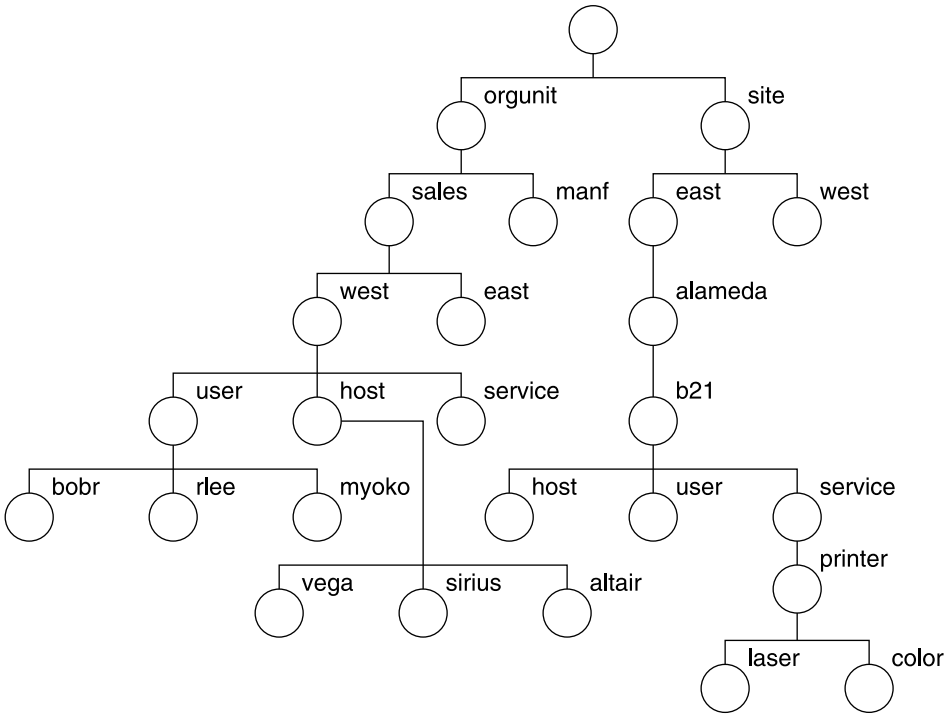


図 **25–1** エンタープライズ名前空間の例

エンタープライズの名前空間は、組織単位の周辺に構成されます。適切な名前空間識別子とオブジェクト名を使用して組織単位名を作成して、サイト、ホスト、ユーザー、ファイル、サービスの名前は、組織単位に関連付けてネーミングできます。

図 25–2 では、エンタープライズの sales 組織の西部事業部のユーザー `myoko` は、名前 `orgunit/west.sales/user/myoko` を使用してネーミングされます。

名前空間識別子 `user` を使用して、`orgunit` 名前空間から `user` 名前空間への移行を表していることに注意してください。同様に (適切な名前空間識別子を使用して)、ファイル名およびサービス名も、サイト、ユーザー、またはホストの名前に関連付けてネーミングできます。サイト名は、組織単位名に関連付けてネーミングできます。

簡単に均質な名前を作成することは、この構造を使用して達成されます。たとえば、エンタープライズ (`orgunit/west`) 内の組織単位名が分かれば、`user` の名前空間識別子とユーザーのログイン名で名前を作成して `orgunit/west/user/josepha` などの名前を作成し、エンタープライズに関連付けてユーザーをネーミングできます。

このユーザーのファイルシステムにあるファイルをネーミングするときには、`orgunit/west/user/josepha/fs/notes` などの名前を使用できます。

エンタープライズのルート

エンタープライズのルートコンテキストは、エンタープライズ名前空間のルートレベルにあるオブジェクトをネーミングするためのコンテキストです。エンタープライズのルートは、グローバルの名前空間で割り当てられます。

エンタープライズのルートをネーミングする方法には、次の 2 つがあります。

- `. . . /`*rootdomain*`.`
- `org//`

3 つのドットを使用してエンタープライズのルートを識別する

`. . . /`*rootdomain*`/` を使用して、エンタープライズのルートを識別できます。

- 最初の 3 つのドット (`. . .`) は、グローバルのコンテキストを示す原子名 (586 ページの「グローバル名前空間のポリシー」のグローバルのコンテキストについての説明を参照)
- *rootdomain*`/` は、エンタープライズのルートドメイン。たとえば、`doc.com/` など

  したがって、`. . . /doc.com/` では、ルートドメインが `doc.com` である会社のエンタープライズのルートが識別されます。この例では、エンタープライズのルートに関連するサイトをネーミングするためのコンテキストは、`. . . /doc.com/site/alameda` または `. . . /doc.com/site/alameda.bldg5` などの `. . . /doc.com/site/` となります。

注 – DNS のグローバルの建物を設定した場合は、`. . . /`*rootdomain* フォーマットだけを使用できます。

`org//` を使用してエンタープライズのルートを識別する

`org//` を使用して、エンタープライズのルートを識別できます。本質的に `org//` は、`. . . /`*domainname*`/` に対する別名または機能的に相当するものです。`org//` を使用するときは、二重のスラッシュによって、ルートのエンタープライズコンテキストとそれに関連する名前空間が識別されます。

たとえば、`org//site/alameda` では、エンタープライズのルートに関連する Alameda サイトがネーミングされます。

反対に、org/ または orgunit/ (一重のスラッシュ) では、エンタープライズのルートに必ずしも相対的にネーミングされていない組織のコンテキストを示します。たとえば、org/sales/site/alameda のようになります。

エンタープライズのルートの従属するコンテキスト

次のオブジェクトは、エンタープライズのルートに関連付けてネーミングできます。

- エンタープライズの組織単位
- エンタープライズの最上位の組織単位にあるサイト (XFN ポリシーへの拡張)
- エンタープライズの最上位の組織単位にいるユーザー
- エンタープライズの最上位の組織単位にあるホスト
- エンタープライズの最上位の組織単位に対するサービス
- エンタープライズの最上位の組織単位に対するファイルサービス

これらのオブジェクトは、ターゲットオブジェクトの名前でターゲットオブジェクトの名前空間の名前空間識別子を作成して、ネーミングします。

エンタープライズのルートと組織的なサブ単位

組織のサブ単位は、エンタープライズのルートに関連付けてネーミングできます。

組織ルート名の場合は、orgunit または _orgunit のどちらかの名前空間識別子を使用して、従属する組織単位コンテキストに名前を作成できます。

たとえば、.../doc.com がエンタープライズ名である場合、組織単位にネーミングするためのコンテキストのルートは、.../doc.com/orgunit/ であり、組織単位名は .../doc.com/orgunit/sales や .../doc.com/orgunit/west.sales のようになります。または、 org//orgunit/sales でも同じ結果を得ることができます。

次のオブジェクトは、組織単位名に関連付けてネーミングできます。

- その組織単位のサイト (XFN ポリシーからの拡張)
- その組織単位にあるホスト
- その組織単位にいるユーザー
- その組織単位に対するサービス
- その組織単位に対するファイルサービス

たとえば、名前 ...doc.com/orgunit/sales/service/calendar では、sales 組織単位のカレンダのサービスが識別されます。組織単位に関連付けてオブジェクトをネーミングする際の詳細については、558 ページの「組織単位の名前空間」および 562 ページの「組織に関連する名前を作成する」を参照してください。

エンタープライズのルートとサイト

サイトは、XFN ポリシーから拡張されています。

サイトは、次のものに関連付けてネーミングできます。

- エンタープライズのルート
- 組織単位

  エンタープライズのルートに関連付けてネーミングされたサイトは、最上位の組織単位に関連付けてネーミングされたサイトと同一になります。組織名の場合は、名前空間識別子の `site` または `_site` を使用して、サイトのコンテキストに名前を作成できます。たとえば、エンタープライズのルートが `../doc.com` である場合、エンタープライズのルートに関連付けてサイトにネーミングするためのコンテキストは、`../doc.com/site` となります。サイトは、`../doc.com/site/alameda` のような名前になります。

  次のオブジェクトは、サイト名に関連付けてネーミングできます。

- サイトのスケジュールまたはカレンダ、プリンタ、および FAX などのサイトでのサービス
- サイトで利用可能なファイルサービス

これらのオブジェクトは、ターゲットオブジェクトの名前空間とターゲットオブジェクト名の名前空間識別子でサイト名を作成して、ネーミングできます。たとえば、名前 `site/Clark.bldg-5/service/calendar` は、会議室 `Clark.bldg-5` のカレンダのサービスでネーミングし、サイト名 `site/Clark.bldg-5` とサービス名 `service/calendar` から作成されます。サイトに関連付けてオブジェクトにネーミングする際の詳細については、563 ページの「サイトに関連する名前を作成する」を参照してください。

エンタープライズのルートとユーザー

ユーザーは、次のものに関連付けてネーミングできます。

- 組織単位
- エンタープライズのルート

  エンタープライズのルートに関連付けてネーミングされたユーザーは、最上位の組織単位に関連付けてネーミングされたユーザーと同一になります。組織名の場合は、名前空間識別子の `user` または `_user` のどちらかを使用して、コンテキストに名前を作成できます。したがって、`orgunit/east.sales` で組織がネーミングされる場合は、`orgunit/east.sales/user/hirokani` で `east.sales` 組織単位にいるユーザー `hirokani` がネーミングされます。

  次のオブジェクトは、ユーザー名に関連付けてネーミングできます。

- ユーザーに関連するサービス
- ユーザーのファイル

これらのオブジェクトは、ターゲットオブジェクトの名前空間とターゲットオブジェクト名の名前空間識別子でユーザー名を作成して、ネーミングされます。たとえば、名前 user/sophia/service/calendar では、ユーザー sophia に対するカレンダがネーミングされます。ユーザーに関連付けてオブジェクトにネーミングする際の詳細については、560 ページの「ユーザーの名前空間」および 、568 ページの「エンタープライズのルートとユーザー」を参照してください。

エンタープライズのルートとホスト

ホストは、次のものに関連付けてネーミングできます。

- 組織単位
- エンタープライズのルート

  エンタープライズのルートに関連付けてネーミングされたホストは、最上位の組織単位に関連付けてネーミングされたホストと同一になります。組織名の場合は、名前空間識別子の host または _host のどちらかを追加して、hostname のコンテキストに名前を作成できます。したがって、orgunit/west.sales で組織がネーミングされる場合は、名前 org/west.sales/host/altair で west.sales 組織単位にあるマシン altair がネーミングされます。

  次のオブジェクトは、ホスト名に関連付けてネーミングできます。
- ホストに関連するサービス
- ホストによってエクスポートされたファイル

これらのオブジェクトは、ターゲットオブジェクトの名前空間とターゲットオブジェクト名の名前空間識別子でホスト名を作成して、ネーミングされます。たとえば、名前 host/sirius/fs/release では、マシン sirius によってエクスポートされたファイルディレクトリ release がネーミングされます。ホストに関連付けてオブジェクトにネーミングする際の詳細については、559 ページの「ホストの名前空間」および 563 ページの「ホストに関連する名前を作成する」を参照してください。

エンタープライズのルートとサービス

サービスは、次のものに関連付けてネーミングできます。

- 組織単位
- エンタープライズのルート
- ユーザー
- ホスト
- サイト

エンタープライズのルートに関連付けてネーミングされたサービスは、最上位の組織単位に関連付けてネーミングされたサービスと同一になります。

サービスのコンテキストは、名前空間識別子 service または _service を使用して、関連する組織、サイト、ユーザー、またはホストに関連付けてネーミングされます。たとえば、orgunit/corp.finance で組織単位がネーミングされる場合

は、orgunit/corp.finance/service/calendar で組織単位 corp.finance の calendar サービスがネーミングされます。ユーザーの名前空間および、ユーザーに関連付けてオブジェクトをネーミングする際の詳細については、561 ページの「サービスの名前空間」および、563 ページの「サービスおよびファイルに関連する名前を作成する」を参照してください。

FNS では、サービスの名前空間での割り当てのタイプが制限されるわけではありません。アプリケーションは、サービスのコンテキスト以外のタイプのコンテキストを作成し、サービスの名前空間で割り当てられます。

FNS は、サービスのコンテキストに「汎用」コンテキストを作成することをサポートします。汎用コンテキストは、アプリケーション決定のリファレンスタイプを持つこと以外は、サービスのコンテキストに類似しています。汎用コンテキストの他のすべての属性は、サービスのコンテキストと同一です。

たとえば、World Intrinsic Designs Corp (WIDC) という名前の会社は、サービスの名前空間で名前 extcomm を獲保し、製品の外部通信ラインに関連する割り当てを追加するための汎用コンテキストを参照します。extcomm に割り当てられるコンテキストは、リファレンスタイプ WIDC_comm を持つ汎用コンテキストです。このコンテキストとサービスのコンテキストの違いは、このコンテキストが異なるリファレンスタイプを持っていることだけです。

サービス名は、561 ページの「サービス名およびリファレンスの登録」で説明しているように、米国 Sun Microsystems, Inc. に登録する必要があります。

エンタープライズのルートとファイル

ファイルの名前空間は、次のものに関連付けてネーミングできます。

- エンタープライズのルート
- 組織単位
- ユーザー
- ホスト
- サイト

エンタープライズのルートに関連付けてネーミングされたファイルは、最上位の組織単位に関連付けてネーミングされたファイルと同一になります。ファイルのコンテキストは、名前空間識別子の fs または _fs を使用して、関連する組織、サイト、ユーザー、またはホストに関連付けてネーミングされます。たとえば、orgunit/accountspayable.finance で組織単位がネーミングされる場合は、名前 user/jsmith/fs/report96.doc でファイル report96.doc がネーミングされます。ユーザーのファイルサービスは、NIS+ の passwd テーブルで指定されているように、デフォルトではホームディレクトリになります。ユーザーの名前空間の詳細については、560 ページの「ファイルの名前空間」 を参照してください。

この他には、ファイルシステムに従属するコンテキストのタイプはありません。

エンタープライズのルートとプリンタ

プリンタのコンテキストは、XFN ポリシーの拡張です。

プリンタの名前空間は、サービスのコンテキストでネーミングできます。プリンタのコンテキストは、名前空間識別子の `printer` を使用して、次のものに関連したサービスのコンテキストでネーミングされます。

- 組織単位
- ユーザー
- ホスト
- サイト

たとえば、`org/east.sales` で組織単位がネーミングされる場合は、`org/eastsales/service/printer` では組織単位 `east.sales` のプリンタのサービスがネーミングされます。したがって `lp1` という名前の固有のプリンタは、次のように識別されます。 `org/east.sales/service/printer/lp1`.

この他には、プリンタに従属するコンテキストのタイプはありません。

エンタープライズ内のネーミングに対する初期コンテキストの割り当て

初期コンテキストは、クライアントのユーザー、ホスト、およびアプリケーションがエンタープライズ名前空間の任意のオブジェクトに (結果的に) ネーミングできる開始点です。

次の図のネーミングシステムは、図 25-1 のネーミングシステムと同じです。ただし、初期コンテキストの割り当てが影付きになっており、斜体で示されています。これらの初期コンテキストは、解決する名前を決定するユーザー、ホスト、またはアプリケーションの観点から示されています。

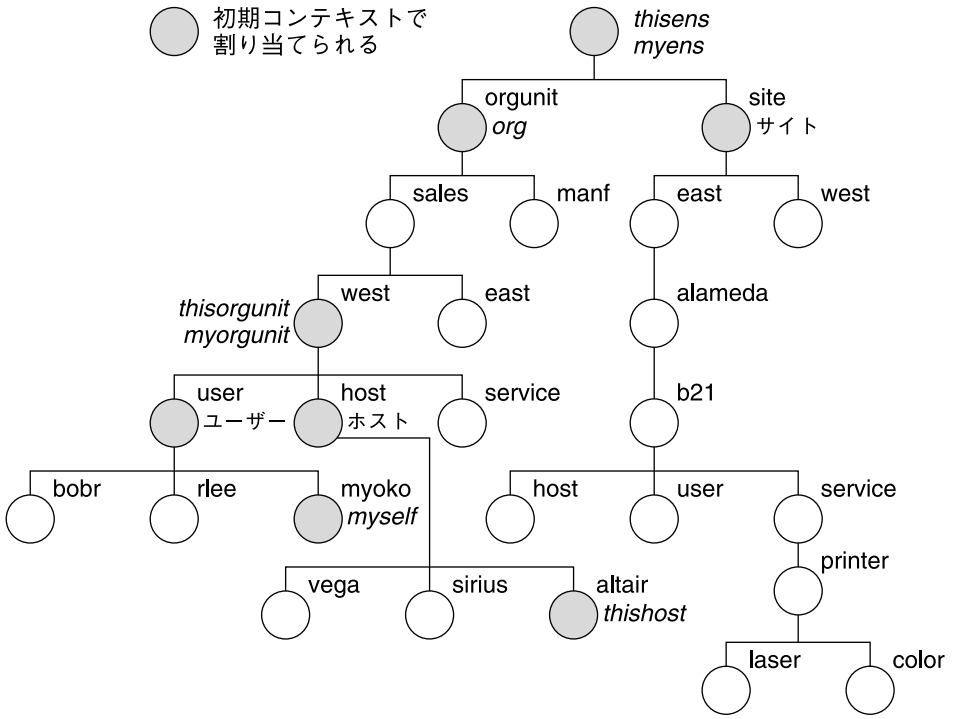


図 **25–2** 初期コンテキストのエンタープライズ割り当ての例

XFN では、「初期コンテキスト関数」の `fn_ctx_handle_from_initial()` が提供され、クライアントは初期コンテキストのオブジェクトを名前解決の開始点として得ることが可能になります。初期コンテキストには、名前空間識別子に対するフラットな名前空間があります。これらの初期コンテキスト識別子の割り当てについては表 25–27 に要約し、その後の節でさらに詳細に説明しています。これらの名前は、必ずしもすべての初期コンテキストに表示されるわけではありません。たとえば、スーパーユーザーがプログラムを起動したときには、ホストとクライアントに関連する割り当てのみが初期コンテキストに表示され、ユーザーに関連する割り当ては表示されません。

表 **25–27** エンタープライズ内のネーミングに対する初期コンテキストの割り当て

名前空間識別子	割り当て
`myself`、`_myself`、`thisuser`	名前解決しようとするユーザーのコンテキスト
`myens`、`_myens`	名前解決しようとするユーザーのエンタープライズの root

表 25–27 エンタープライズ内のネーミングに対する初期コンテキストの割り当て　(続き)

名前空間識別子	割り当て
myorgunit、_myorgunit	ユーザーの主要な組織単位のコンテキスト。たとえば、NIS+ 環境では、主組織単位はユーザーの NIS+ ホームドメインとなる
thishost、_thishost	名前解決しようとするホストのコンテキスト
thisens、_thisens	名前解決しようとするホストのエンタープライズのルート
thisorgunit、_thisorgunit	ホストの主要な組織単位のコンテキスト。たとえば、NIS+ 環境では、主組織単位はホストの NIS+ ホームドメインとなる
user、_user	ホストと同じ組織単位にいるユーザーがネーミングされるコンテキスト
host、_host	ホストと同じ組織単位にいるホストがネーミングされるコンテキスト
org、orgunit、_orgunit	ホストのエンタープライズにある組織単位の名前空間のルートコンテキスト。たとえば、NIS+ 環境では、これは NIS+ ルートドメインに対応する
site、_site	サイトの名前空間が構成された場合、最上位の組織単位にあるサイトの名前空間のルートコンテキスト

XFN 用語では、接頭辞として使用される下線は、「正規」名前空間識別子と呼ばれます。この下線がなく、org および thisuser が追加された名前は、Solaris 用にカスタマイズされた名前です。Solaris カスタマイズの名前空間識別子は、他の Solaris オペレーティング環境以外の環境で認識されるとは限りません。正規名前空間識別子は、他の環境でも必ず認識されるため移植性があります。

注 – 現在実装されている FNS では、初期コンテキストにある名前および割り当ての追加や修正はサポートされません。

初期コンテキストの割り当ては、基本的に次の 3 つに分けられます。

- ユーザーに関連する割り当て (573 ページの「ユーザーに関連する割り当て」を参照)
- ホストに関連する割り当て (575 ページの「ホストに関連する割り当て」を参照)
- 短縮形の割り当て (576 ページの「短縮形の割り当て」を参照)

ユーザーに関連する割り当て

FNS では、XFN 初期コンテキスト関数が呼び出されたときに、プロセスに関連付けられたユーザーが存在するものと仮定されます。この関連付けは、プロセスの実効ユーザー ID (*euid*) に基づいています。プロセスに対するユーザーの関連付けがプロセスの途中で変更されることはありますが、元のコンテキストハンドルは変更されません。

FNS では、名前解決を要求するユーザーに関連する初期コンテキストにある次の割り当てが定義されます。

myself

初期コンテキストにある名前空間識別子 `myself` (あるいは同義語の `_myself` または `thisuser` のどちらか) は、要求を行うユーザーのコンテキストで解決されます。ユーザー `jsmith` が所有するプロセスが名前解決を要求したときの例を示します。

- 名前 `myself` は、`jsmith` のユーザーのコンテキストとして初期コンテキストで解決される
- `myself/fs/.cshrc` は、`jsmith` のファイル `cshrc` にネーミングする

myorgunit

FNS では、各ユーザーがエンタープライズの組織単位に関連するものと仮定されます。1 人のユーザーを複数の組織単位に関連させることはできますが、組織の名前空間での位置または組織内でのユーザーの役割によって、主な組織単位が必ず 1 つ必要です。

- 「NIS+」。NIS+ 名前空間では、この組織単位はユーザーのホームドメイン (これは、サブドメインになることもある) に対応する
- 「NIS」。NIS 名前空間では、ユーザーのドメインに対応するエンタープライズレベルの組織単位は 1 つしかない
- 「ファイル」。ファイルベースの名前空間では、組織単位は `myorgunit` にマップする `org//` しかない

  名前空間識別子 `myorgunit` (または `_myorgunit`) は、要求を行うユーザーが所属する主組織単位のコンテキストとして初期コンテキストで解決されます。たとえば、要求を行うユーザーが `jsmith` で、`jsmith` のホームドメインが `east.sales` である場合は、`myorgunit` は `east.sales` の組織単位コンテキストとして初期コンテキストで解決され、名前 `myorgunit/service/calendar` は `east.sales` のカレンダのサービスで解決されます。

myens

FNS では、各ユーザーがエンタープライズに関連するものと仮定されます。これは、`myorgunit` を保持する名前空間に対応します。

名前空間識別子 `myens` (および `_myens`) は、要求を行うユーザーが所属するエンタープライズのルートとして初期コンテキストで解決されます。たとえば、`jsmith` が要求を行い、`jsmith` の NIS+ ホームドメインが `east.sales` である場合、それは `doc.com.` のルートドメイン名を含む NIS+ 階層にあります。名前 `myens/orgunit/` は、`doc.com` の最上部の組織単位で解決されます。

注 – myorgunit または myself/service などのユーザーに関連する複合名を使用するときは、これらの割り当てはプロセスの実効ユーザー ID に依存するため、ユーザー ID 設定プログラムに注意してください。ユーザー ID を root に設定して、呼び出し元に代わってシステムリソースにアクセスするプログラムでは、通常は fn_ctx_handle_from_initial() を呼び出す前に seteuid(getuid()) を呼び出してください。

ホストに関連する割り当て

XFN 初期コンテキスト関数が呼び出されたときには、特定のホストでプロセスが実行中です。FNS では、プロセス実行中のホストに関連する初期コンテキストにある次の割り当てが定義されます。

thishost

名前空間識別子 thishost (または _thishost) は、プロセス実行中のホストのホストコンテキストに割り当てられます。たとえば、プロセスがマシン cygnus で実行中である場合、thishost は cygnus のホストのコンテキストに割り当てられ、名前 thishost/service/calendar はマシン cygnus のカレンダのサービスを参照します。

thisorgunit

FNS では、ホストは組織単位に関連付けられていると仮定します。1 つのホストを複数の組織単位に関連付けることができますが、主となる組織単位が必要です。NIS+ 名前空間では、組織単位はホストのホームドメインに対応します。

名前空間識別子 thisorgunit (または _thisorgunit) は、プロセス実行中のホストの主要な組織単位に解決されます。たとえば、ホストがマシン cygnus であり、cygnus の NIS+ ホームドメインが west.sales である場合は、thisorgunit は west.sales として組織単位コンテキストで解決され、名前 thisorgunit/service/fax は組織単位 west.sales の FAX サービスを参照します。

thisens

FNS では、ホストがエンタープライズに関連するものと仮定されます。これは、thisorgunit を保持する名前空間に対応します。

名前空間識別子 thisens (または _thisens) は、プロセス実行中のホストのエンタープライズのルートで解決されます。たとえば、NIS+ で、ホストのホームドメインが sales.doc.com である場合、名前 thisens/site/ は doc.com. のサイトの名前空間のルートで解決されます。

短縮形の割り当て

FNS では、初期コンテキストにある次の短縮形割り当てを定義し、短い名前を使用して、共通に参照される特定の名前空間を参照できます。

user

名前空間識別子 `user` (または `_user`) は、プロセス実行中のホストの組織単位にある username コンテキストとして初期コンテキストに割り当てられます。これにより、同じ組織単位にいる他のユーザーをこのコンテキストからネーミングできます。

初期コンテキストから、名前 `user` および `thisorgunit/user` は同じコンテキストで解決されます。たとえば、プロセス実行中のホストがマシン `altair`であり、`altair` が `east.sales` 組織単位にある場合は、名前 `user/medici` で `east.sales` にいるユーザー `medici` がネーミングされます。

host

名前空間識別子 `host` (または `_host`) は、プロセス実行中のホストの組織単位にある `hostname` コンテキストとして初期コンテキストに割り当てられます。これにより、同じ組織単位にいる他のホストをこのコンテキストからネーミングできます。

初期コンテキストから、名前 `host` および `thisorgunit/host` は同じコンテキストで解決されます。たとえば、プロセス実行中のホストがマシン `sirius` であり、`sirius` が `east.sales` 組織単位にある場合は、名前 `host/sirius` で `east.sales` にあるマシン `sirius` がネーミングされます。

org

名前空間識別子 `org` (または `orgunit`、`_orgunit`) は、プロセス実行中のホストが所属するエンタープライズの組織のルートコンテキストとして初期コンテキストで割り当てられます。

初期コンテキストから、名前 `org` および `thisens/orgunit` は同じコンテキストで解決されます。たとえば、プロセス実行中のホストがマシン `aldebaran` であり、`aldebaran` がエンタープライズ `doc.com` にある場合は、名前 `org/east.sales` で `doc.com.` にある組織単位 `east.sales` がネーミングされます。

site

名前空間識別子 `site` (または `_site`) は、プロセス実行中のホストが所属するエンタープライズの最上位の組織単位のサイトネーミングシステムのルートに対する初期コンテキストに割り当てられます。

初期コンテキストから、名前 site および thisens/site は同じコンテキストで解決されます。たとえば、プロセス実行中のホストがマシン aldebaran であり、aldebaran がエンタープライズ doc.com にある場合は、名前 site/pine.bldg-5 で doc.com. の建物 5 にある会議室 pine がネーミングされます。

FNS およびエンタープライズのレベルのネーミング

FNS では、複数のネームサービスを 1 つの簡単なインタフェースを使用した基本的なネーミング操作でフェデレートすることができます。FNS は、NIS+、NIS、およびファイルという 3 つのエンタープライズレベルのネームサービスを操作できるように設計されています。また、581 ページの「FNS ポリシーの対象となるクライアントアプリケーション」 で説明するように、プリンタやカレンダサービスなどのアプリケーションも操作できます。

FNS ポリシーと NIS+ との関連

FNS では、NIS+ サーバー上のドメインレベルの org_dir NIS+ ディレクトリにある FNS テーブルに、エンタープライズのオブジェクトに対する割り当てが格納されます。FNS テーブルは NIS+ テーブルに類似しています。これらの FNS テーブルには、次のエンタープライズ名前空間に対する割り当てが格納されます。

- 577 ページの「NIS+ ドメインと FNS 組織単位」で説明する「組織」の名前空間
- 578 ページの「NIS+ ホストと FNS ホスト」で説明する「ホスト」の名前空間
- 579 ページの「NIS+ ユーザーと FNS ユーザー」で説明する「ユーザー」の名前空間
- 組織、ホスト、ユーザーに関連付けて地域サイトをネーミングするための「サイト」の名前空間
- 組織、ホスト、ユーザーに関連付けてプリンタやカレンダなどのサービスにネーミング可能な「サービス」の名前空間

NIS+ ドメインと FNS 組織単位

FNS では、優先 Solaris エンタープライズのネームサービスである NIS+ 内の組織、ユーザー、ホストのエンタープライズオブジェクトがネーミングされます。NIS+ ドメインは、ユーザーおよびマシンの論理コレクションとそれらについての情報で構成され、エンタープライズ内の階層組織構造のフォームを反映するように配列されます。

FNS は、NIS+ ドメインを FNS 組織にマッピングして、NIS+ で実行されます。組織単位名は、NIS+ ドメイン名に対応しており、完全指定された形式の NIS+ ドメイン名または NIS+ ルートに関連した NIS+ ドメイン名のどちらかを使用して識別されます。最上位の FNS 組織名前空間は、NIS+ ルートドメインにマップされ、初期コンテキストから名前 `org/` を使用してアクセスされます。

NIS+ では、ユーザーおよびホストに「ホームドメイン」の概念があります。ホストまたはユーザーのホームドメインは、それらの関連付けられた情報を保持する NIS+ ドメインです。ユーザーまたはホストのホームドメインは、直接 NIS+「主体名」を使用して決定できます。NIS+ 主体名は、原子ユーザー (ログイン) 名または原子ホスト名と、NIS+ ホームドメイン名からなります。たとえば、ホームドメイン `doc.com.` を持つユーザー `sekou` には NIS+ 主体名 `sekou.doc.com` が付けられ、マシン名 `vega` には NIS+ 主体名 `vega.doc.com` が付けられます。

ユーザーの NIS+ ホームドメインは、ユーザーの FNS 組織単位に対応します。同様に、ホストのホームドメインは、その FNS 組織単位に対応します。

組織名の後のドット

組織名の後のドットは、その名前が完全指定の NIS+ ドメイン名であることを示します。このドットがない場合は、その組織名は NIS+ ルートドメインに関連付けて解決される NIS+ ドメイン名です。

たとえば、NIS+ ルートドメインが `doc.com.` で、サブドメインが `sales.doc.com.` の場合、次の名前の組み合わせは同じ組織を参照します。

表 **25–28** NIS+ での相対および完全指定の組織名の例

相対名	完全指定名
org/	org/doc.com.
org/sales	org/sales.doc.com.

`manf.` という名前だけを含む NIS+ ドメインは存在しないため、名前 `org/manf.` (ドット付き) はありません。

NIS+ ホストと FNS ホスト

NIS+ 名前空間のホストは、ホストのホームドメインの `hosts.org_dir` テーブルにあります。FNS 組織にあるホストは、対応する NIS+ ドメインの `hosts.org_dir` テーブル内のホストに対応します。FNS では、`hosts` テーブル内の各ホストにコンテキストが提供されます。

NIS+ ユーザーと FNS ユーザー

NIS+ 名前空間にいるユーザーは、ユーザーのホームドメインの `passwd.org_dir` テーブルのリストに入っています。FNS 組織にいるユーザーは、対応する NIS+ ドメインの `passwd.org_dir` テーブル内のユーザーに対応します。FNS では、`passwd` テーブル中の各ホストにコンテキストが提供されます。

NIS+ セキュリティと FNS

FNS の `fncreate` コマンドを使用して、コマンドが実行されたホストのドメインに関連付けられた NIS+ 階層に FNS テーブルとディレクトリを作成します。`fncreate` を実行するには、そのドメインの NIS+ オブジェクトの読み取り、作成、変更、削除を許可する資格を得て、認証された NIS+ 主体になる必要があります。`fncreate` で作成した FNS テーブルの「所有者」になります。この許可を得る 1 つの方法は、ドメインの管理者特権を持つ NIS+ グループのメンバーになることです。

`fncreate` を実行する前に、`NIS_GROUP` 環境変数をドメインに対する NIS+ 管理グループ名に設定する必要があります。個別のユーザーが、ユーザーに関連する FNS データを変更可能かどうかを指定できます。

FNS ポリシーと NIS の関連

FNS では、NIS マスターサーバー (および存在する場合は NIS スレーブサーバー) `/var/yp/`*domainname* ディレクトリにある FNS マップのエンタープライズオブジェクトへの割り当てが格納されます。FNS マップは、構造と機能の面で FNS マップと類似しています。これらの NIS マップでは、次のエンタープライズ名前空間に対する割り当てが格納されます。

- エンタープライズ全体に関連するオブジェクトをネーミングするための名前空間を提供する「組織」。NIS がネームサービスの下にあるときは、NIS ドメインに対応する 1 つの組織単位コンテキストがある。この組織単位コンテキストは、NIS ドメイン名または、マシン NIS ドメイン名をデフォルトとする空白名によって FNS で識別される
- NIS ドメインの `hosts.byname` マップに対応する「ホスト」の名前空間。FNS では、`hosts.byname` マップにある各ホストにコンテキストが提供される
- `passwd.byname` マップに対応する「ユーザー」の名前空間。FNS では、ドメインの `passwd.byname` にある各ユーザーにコンテキストが提供される
- 組織、ホスト、ユーザーに関連付けて地域サイトをネーミングするための「サイト」の名前空間
- 組織、ホスト、ユーザーに関連付けてプリンタやカレンダなどのサービスにネーミングするための「サービス」の名前空間

FNS では、他のオブジェクトをこれら 5 つの名前空間に関連付けてネーミングするためのコンテキストが提供されます。

FNS の fncreate コマンドを使用して、NIS マスターサーバーの /var/yp/*domainname* ディレクトリに FNS マップを作成します。これは、NIS ネームサービスのマスターサーバーと同じマシンか、または FNS マスターサーバーとして機能する異なるマシンで行えます。スレーブサーバーが存在する場合は、NIS は、通常の処理の一部として FNS マップをそれらにプッシュします。fncreate を実行するには、FNS マップのホストとなるサーバーの特権ユーザーになる必要があります。各ユーザーは、FNS データを変更できません。

FNS ポリシーとファイルベースのネーミングの関連

FNS では、通常各マシンに NFS マウントされた /var/fn ディレクトリにあるファイル内のエンタープライズオブジェクトに対する割り当てが格納されます。これらの FNS ファイルでは、次のエンタープライズ名前空間に対する割り当てが格納されます。

- エンタープライズ全体に関連付けてオブジェクトをネーミングする名前空間を提供する「組織」。ローカルのファイルがネームサービスの下にある場合は、システム全体を表す 1 つの組織単位コンテキストがある。この組織単位コンテキストは、FNS では常に org// として識別される
- /etc/hosts ファイルに対応する「ホスト」の名前空間。FNS では、/etc/hosts ファイルにある各ホストにコンテキストが提供される
- /etc/passwd ファイルに対応する「ユーザー」の名前空間。FNS では、/etc/passwd ファイルにある各ユーザーにコンテキストが提供される
- 組織、ホスト、ユーザーに関連付けて地域サイトをネーミングするための「サイト」の名前空間
- 組織、ホスト、ユーザーに関連付けてプリンタやカレンダなどのサービスにネーミングするための「サービス」の名前空間

FNS では、他のオブジェクトをこれら 5 つの名前空間に関連付けてネーミングするためのコンテキストが提供されます。

FNS の fncreate コマンドによって、コマンドが実行されるマシンの /var/fn ディレクトリに FNS ファイルを作成します。fncreate を実行するには、そのマシンでのスーパーユーザー特権を持つ必要があります。UNIX ユーザー ID に基づいて、各ユーザーは、FNS コマンドを使用して自身のコンテキスト、割り当て、属性を変更することが許可されます。

FNS ポリシーの対象となるクライアントアプリケーション

FNS ポリシーの 1 つの目的は、ファイルシステム、およびカレンダマネージャ、印刷ツール、ファイルマネージャ、メールツールなどの DeskSet ツール、またこれらのツールをサポートする RPC、電子メール、印刷サブシステムなどのサービスなどの、共通して使用されるツールに一貫性を持たせることです。

注 – これらのツールの一部は、現在の Solaris オペレーティング環境 には実装されていません。ここでは、FNS の使用例を示すために挙げています。

- 「カレンダ」。「username@hostname」の形の名前を使用して誰かのカレンダにアクセスする代わりに、たいていの場合は単にサイト、組織、またはユーザー名を入力する。カレンダをネーミングするときには、複合名を使用することもできる。たとえば、FNS でフェデレートするときには、次の形式の名前がカレンダマネージャで使用可能となる
  - bernadette
  - user/bernadette
  - site/pine.bldg-5 (Pine 会議室用のカレンダ)
  - org/sales (sales 組織用のカレンダ)
- 「印刷」。特定のプリンタを名前でネーミングする代わりに、ユーザー、サイト、または組織に関連付けてプリンタをネーミングできる。たとえば:
  - ilych (ilych のデフォルトのプリンタ)
  - org/sales (組織のデフォルトのプリンタ)
  - site/pine.bldg-5 (Pine 会議室のプリンタ)
- 「ファイルアクセス」。ファイルシステムやファイルをネーミングするときに複合名を使用できる。オートマウンタは、FNS を使用して複合名の解決を可能にする。たとえば、/xfn/user/baruch/fs/.cshrc などのファイル名を使用して、ユーザー baruch の .cshrc ファイルを参照できる
- 「RPC」。サービスをホスト名、プログラム、およびバージョン番号でアドレス指定する代わりに、複合名を使用してサービスをネーミングできる。たとえば、user/hatori/service/rpc のようにユーザーまたは組織に関連付けて RPC サービスをネーミングできる
- 「メール」。同様に、複合名はメールの宛先をネーミングするために使用される。次のような名前を使用できる
  - angus
  - user/angus
  - org/mlist (組織のメールリスト)
  - site/pine.bldg-5 (会議室の調整係のメールボックス)
- 「他のデスクトップアプリケーション」。スプレッドシート、文書作成ツール、FAX ツールなどの他のデスクトップアプリケーションに複合名を引き渡すことができる。これらのアプリケーションには、それ自身の名前空間をサービスの名前空間に接続し、FNS フェデレーションの一部となるものもある

アプリケーションの例 - カレンダサービス

ここでは、カレンダサービスというアプリケーションを変更して、FNS ポリシーを使用する方法について説明します。この例では、ユーザーから FNS 複合名が提示され、承認される手順を示します。

DeskSet のカレンダサービスは、一般的なクライアントサーバーアプリケーションです。カレンダサーバーは、複数のマシンで動作し、ユーザーのカレンダを保持します。カレンダマネージャ (cm) は、デスクトップで動作し、適切なサーバーにコンタクトして必要なカレンダを入手します。

カレンダサービスは、次のような簡単なレジストリまたは検索のモデルを使用して FNS を利用します。

- 「割り当て」。起動時に、サーバーは、`user/jsmith/service/calendar` などの管理するそれぞれのカレンダの複合名に対するそれ自身の ONC+ RPC アドレス (ホスト、プログラム、バージョン) を含むリファレンスを割り当て、サーバーの管理するカレンダを登録する
- 「検索」`cm` を使用するときには、ユーザーは、単にユーザー名 (`hirokani` など) を入力するか、または以前に入力した名前のリストからユーザーを選択して、他のユーザーのカレンダを指定する。ユーザー名が `hirokani` の場合は、`cm` で複合名 `user/hirokani/service/calendar` が作成され、これを使用してカレンダを管理するサーバーと通信する必要のある RPC アドレスが検索される

前の例では、名前 `calendar` を使用して、カレンダの割り当てを示しています。RPC プログラムを RPC 管理者に登録するのと同じように、カレンダサービスの開発者は、名前 `calendar` を FNS 管理者に登録します。561 ページの「サービス名およびリファレンスの登録」 を参照してください。

注 – ここで使用した名前 `calendar` は 1 つの例です。FNS ポリシーによって、特定のサービス名が指定されるわけではありません。

カレンダサービスでは、カレンダをサイト、組織、およびホストに関連付け、一定の方法でそれらをネーミングして、FNS ポリシーをさらに利用できます。たとえば、カレンダを会議室 (サイト) と関連付け、ユーザーのカレンダのセットでその部屋の会議に利用可能な時間を見つけるのと同じように、サービスを使用して会議室のカレンダを多重表示できます。同様に、カレンダは、グループ会議の組織や、保守スケジュールを管理するホストに関連付けることができます。

カレンダマネージャ (cm) は、いくつかの簡単な手順に従って、ユーザーが指定する必要のあることを明確にします。

1. `cm` では、ユーザーからの複合名を承認して、カレンダが要求されたオブジェクト名を構成するためのツールが使用されます。

   オブジェクトは、ユーザー、サイト、ホスト、または組織の名前になります。たとえば、ユーザーが名前 `kuanda` を入力すると、カレンダマネージャで複合名 `user/kuanda` が生成されます。このツールは、DeskSet アプリケーションのグ

ループ間で共有できます。

2. cm は、XFN インタフェースを使用して、接尾辞 /service/calendar を含む名前を作成し、カレンダ名を入手します。
3. このカレンダ名は、プロセスの初期コンテキストに関連付けて解決されます。

   続いて、名前 user/kuanda/service/calendar が解決されます。同様に、ユーザーがサイト名 pine.bldg-5 を入力した場合は、cm で名前 site/pine.bldg-5/service/calendar が生成され解決されます。

   印刷やメールなどの他のサービスでも、類似の方法で FNS ポリシーを利用できます。

FNS ファイルシステム名前空間

FNS では、ファイル名はユーザー、ホスト、組織、サイトとの関係から設定できます。この場合実際には、オブジェクトの名前の後にエンタープライズの名前空間識別子 fs を指定し、その後にファイル名を指定します。たとえば、Sales 部門の、budget というファイルなら、org/sales/fs/budget/draft96という名前になります。

初期コンテキストは、ルートディレクトリの /xfn の下にあります。したがって、以下のように入力して参照できます。

```
% more /xfn/org/sales/fs/budget/draft96
```

アプリケーションからこのディレクトリへアクセスする方法は、他のディレクトリの場合とまったく同様です。アプリケーションを修正したり、XFN API を使用したりする必要はありません。

NFS ファイルサーバー

NFS は Sun Microsystems, Inc. の製品で、分散ファイルシステムの一種です。オブジェクトのファイルは、通常 1 つ以上のリモート NFS ファイルサーバーにあります。最も単純なのは、エクスポートされた NFS ファイルシステムに名前空間識別子 fs が対応するという図 25–3 のような場合です。

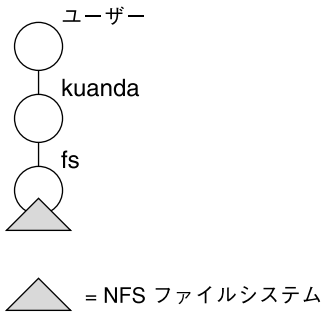


図 **25–3** NFS ファイルシステム - 単純な場合

しかしオブジェクトのファイルシステムの中には、複数の (重なり合う場合もある) リモートマウントで構成され、FNS によって管理される仮想ディレクトリ構造にまとめられているものもあります。

図 25–4 は、この機能を使用して 3 つの異なるファイルサーバーをまとめて組織のファイルシステムを 1 つ作成している例です。`project` ディレクトリと `lib` サブディレクトリは同じファイルサーバー上に常駐していますが、`src` サブディレクトリの常駐先は別です。しかしユーザー、およびアプリケーションの側からは 1 つのシームレスな名前空間に見えるので、複数のサーバーを使用していることを意識する必要はありません。

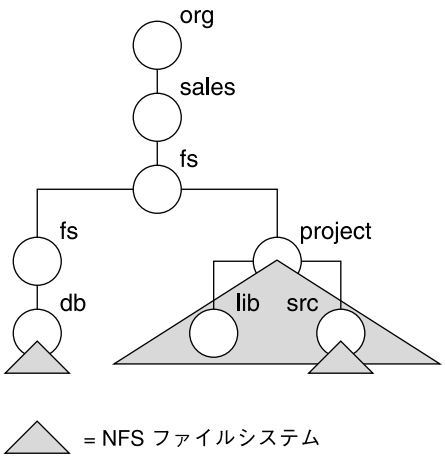


図 **25–4** NFS ファイルシステム - 複数サーバーで構成されている場合

オートマウンタ

効率を上げるため、FNS ディレクトリのマウントには必要に応じてオートマウンタが使用されます。`/etc/auto_master` 構成ファイルには、デフォルトでは以下のような行が含まれます。

```
/xfn -xfn
```

これは、「FNS 名前空間が、XFN の指定どおり `/xfn` の下にマウントされる」という意味です。

オートマウンタは、FNS によって指定されたディレクトリのマウントに使用されるため、FNS ディレクトリのサブディレクトリは、マウントされるまで表示できません。たとえば、sales 組織のファイルシステムが複数のファイルシステムで構成されているとします。以下の `ls` コマンドは最近アクセスして現在もマウントされている 2 つのファイルシステムだけを表示します。

```
% ls /xfn/org/sales/fs
customers products
```

完全なリストを表示するには、`fnlist` コマンドを使用します。

```
% fnlist org/sales/fs
Listing org/sales/fs:
products
goals
customers
incentives
```

FNS プリンタの名前空間

printer コンテキストは、XFN ポリシーには含まれていません。FNS に提供されているのは、プリンタ割り当てを保存するためです。

FNS では、FNS 名前空間にプリンタ割り当てを保存するための機能が提供されています。この機能によって、プリントサーバーは自分自身のサービスをユーザーに知らせることができ、ユーザーはクライアント側の管理がなくてもプリンタのブラウズおよび選択ができます。

プリンタ割り当ては、組織、ユーザー、ホスト、サイトごとに存在する printer コンテキストに格納されます。したがって、組織、ユーザー、ホスト、サイトは、それぞれ専用の printer コンテキストを所有できます。

printer コンテキストは、個々の複合名の service コンテキストの下に作成されます。例を以下に示します。

```
org/doc.com./service/printer
```

ホストのプリンタ名が deneb である場合、printer コンテキストは以下のように作成されます。

```
host/deneb/service/printer/laser
```

グローバル名前空間のポリシー

グローバルネームサービスには、固有の適用範囲があります。ここでは、グローバルネームサービスを使用するオブジェクトをネーミングするためのポリシーについて説明します。

ネーミングに関しては、エンタープライズは、グローバル名前空間にあるエンタープライズのルートを割り当てて、フェデレートされたグローバル名前空間にリンクします。これにより、エンタープライズ外部のアプリケーションおよびユーザーがそのエンタープライズ内部のオブジェクトをネーミングできます。たとえば、エンタープライズ内部のユーザーは、他のエンタープライズにいる同僚にファイルのグローバル名を渡すことができます。

DNS および X.500 のコンテキストは、エンタープライズをネーミングするためのグローバルレベルのネームサービスを提供します。FNS では、DNS と X.500 コンテキストの両方がサポートされます。FNS がない場合、DNS および X.500 では、限定された部分のエンタープライズ名前空間だけに外部からアクセスできます。FNS では、カレンダマネージャなどのサービスを含むエンタープライズ名前空間全体に外部からアクセスできます。

グローバルネーミングに対する初期コンテキスト割り当て

名前「. . .」(3 つのドット) は、各 FNS クライアントの初期コンテキストに表示されます。名前「. . .」は、グローバル名が解決されるコンテキストに割り当てられます。

表 **25–29** グローバルネーミングに対する初期コンテキスト割り当て

原子名	割り当て
. . .	DNS または X.500 の名前を解決するためのグローバルコンテキスト
/ . . .	3 つのドットの同義語

グローバル名は、完全指定のインターネットドメイン名か、または X.500 の識別名になります。

- インターネットドメイン名は、インターネット RFC 1035 で指定された構文に表示される。たとえば、`. . . /doc.com` (587 ページの「DNS のフェデレーティング」を参照) など
- X.500 名は、X/Open DCE ディレクトリで決定された構文に表示される。たとえば、`. . . /c=us/o=doc` など (588 ページの「X.500/LDAP のフェデレーティング」 を参照)

  名前「. . .」および「/. . .」は、初期コンテキストで解決されるときには等価になります。たとえば、名前 `/. . . /c=us/o=doc` および `. . . /c=us/o=doc` は、初期コンテキストで同じオブジェクトとして解決されます。

DNS のフェデレーティング

完全指定の DNS 名は、必ずグローバルコンテキストで使用できます。DNS 名がグローバル名前空間で見つかると、リゾルバライブラリを使用して解決されます。リゾルバライブラリは、DNS 名前解決メカニズムです。一般的 DNS 名は、インターネットのホストアドレスまたは DNS ドメインレコードで解決されます。グローバルコンテキストで DNS 名が検出されると、名前は DNS リゾルバに引き渡されて解決されます。結果は、XFN リファレンス構造に変換され、要求者に返されます。

DNS ドメインの内容は、表示できます。しかし、表示操作は、インターネットでの連結性とセキュリティなどの実用上の理由によって限定されることもあります。たとえば、DNS ドメインのグローバルルート表示は、一般的にルート DNS サーバーではサポートされません。しかし、ルートの下にあるたいていのエンティティは、表示操作をサポートします。

DNS ホストおよびドメインは、DNS リソース名と関連付けられたネームサービス (NS) のリソースレコードの有無によって確認されます。

- 「*DNS* ドメイン名」NS レコードがリソース名に存在する場合は、その名前はドメイン名であると考えられ、返されたリファレンスのタイプは `inet_domain` となる
- 「*DNS* ホスト名」NS レコードがリソース名に存在しない場合は、その名前はホスト名であると考えられ、返されたリファレンスのタイプは `inet_host` となる

DNS は、端末以外のネーミングシステムのように機能し、他のネーミングシステムをフェデレートするために使用されます。

たとえば、エンタープライズネーミングシステムは、FNS 名 `.../doc.com/` がそのエンタープライズの FNS 名前空間のルートを参照するような DNS にある `doc.com` に割り当てられます。

エンタープライズのネーミングシステムは、適切なテキスト (TXT) レコードをそのドメインの DNS マップに追加することにより、DNS ドメインに割り当てられます。そのドメインに対する FNS 名には後に続くスラッシュ (/) が含まれ、TXT リソースレコードは、エンタープライズのネーミングシステムへのリファレンスを構成するために使用されます。

DNS の一般的な情報については、`in.named` のマニュアルページか『*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*)*』を参照してください。

X.500/LDAP のフェデレーティング

X.500 は、グローバルのディレクトリサービスです。ここでは、情報が格納され、情報を表示して検索する機能と同様に、名前で情報を検索する機能が提供されます。

X.500 の情報は、ディレクトリ情報ベース (DIB) に格納されます。DIB にあるエントリは、ツリー構造で配列されます。各エントリは名前付きオブジェクトで、定義された属性のセットを含んでいます。各属性には、定義済みの属性タイプと 1 つまたは複数の値があります。

エントリは、ルートから名前付きエントリまでのパスでツリー内の各エントリから選択された属性の連結である「識別名」によって明確に識別されます。たとえば、図 25–5 に示す DIB を使用すると、`c=us/o=doc` は、合衆国にある `doc` 組織の識別名となります。X.500 ディレクトリのユーザーは、DIB にあるエントリと属性を問い合わせたり、変更したりすることができます。

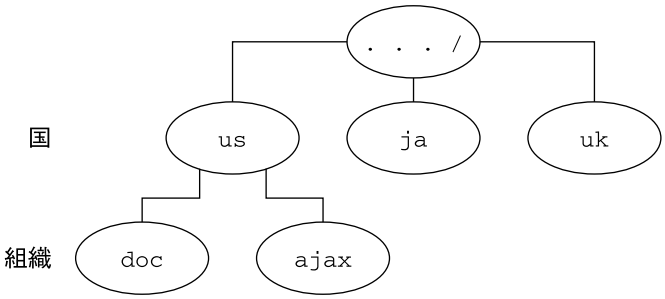


図 **25–5** X.500 ディレクトリ情報ベースの例

FNS は、名前空間をグローバル X.500 名前空間の下にシームレスに接続して表示するために必要なサポートを提供し、X.500 をフェデレートします。

たとえば、FNS は、X.500 の下の `doc` 組織にエンタープライズネーミングシステムをリンクします。初期コンテキストから始まって、`doc` 組織の `sales` 組織単位を識別する FNS 名は次のようになります。

```
.../c=us/o=doc/orgunit/sales
```

エンタープライズ内の名前は、単純にグローバルの X.500 名上に連結されます。FNS 名が初期コンテキストで名前「`...`」を使用して、グローバル名が後に続くことを示すことに注意してください。

FNS 名の名前解決は、次のように行われます。X.500 名がグローバル名前空間で見つかると、X.500 名前解決メカニズムを使用して解決されます。次の 3 つの結果が考えられます。

- フルネームが X.500 エントリに解決される。これは、エントリが X.500 に格納されていることを示す。要求された FNS 操作がそのエントリで実行される
- フルネームの接頭辞は、X.500 エントリに解決される。これは、名前の残りの部分が従属するネーミングシステムに所属することを示す

  従属するネーミングシステムへの次のネーミングシステムのポインタ (NNSP) が調べられ、XFN リファレンスが返されます。この後、名前解決は従属するネーミングシステムで続けられます。
- エラーが報告される

  X.500 エントリは、FNS 操作を使用して調べたり、変更したりできます (アクセス制御に従う)。しかし、現在は FNS を使用して X.500 名前空間のルートの下にある従属するエントリを表示できません。

FNS の問題と対策

この節では、問題と共に、考えられる原因、診断、対策を示します。

FNS エラーメッセージの一般的な情報については、515 ページの「FNS エラーメッセージ」 を参照してください。

初期コンテキストが取得できない

「症状」

「`Cannot obtain initial context`」というメッセージが表示されます。

「考えられる原因」

インストール時の問題により発生します。

「診断」

`/usr/lib/fn/fn_ctx_initial.so` というファイルを検索し、FNS が正しくインストールされているかどうかを確認します。

「対策」

`fn_ctx_initial.so` ライブラリをインストールします。

初期コンテキストが空になっている

「症状」

`fnlist` を実行して初期コンテキストの内容を確認すると何も表示されないという状態です。

「考えられる原因」

NIS+ の設定によって発生する問題です。`fn*` コマンドを実行するユーザーやマシンに関連した組織に、`ctx_dir` ディレクトリがないことが原因です。

「診断」

`nisls` コマンドを使用して `ctx_dir` ディレクトリの有無を確認します。

「対策」

診断の結果 `ctx_dir` ディレクトリがなければ、`fncreate -t org/`*`nis+_domain_name/`* を実行して作成します。

「No Permission」というメッセージが表示される (FNS)

「症状」

「no permission」というメッセージが表示されます。

「考えられる原因」

このメッセージは、「コマンドを実行しようとしたが、アクセス権がない」ということを意味します。

「診断」

適切な NIS+ コマンドを使用してアクセス権を確認します (543 ページの「FNS と NIS + の詳細情報」を参照)。NIS+ 主体名は、`nisdefaults` コマンドで知ることができます。

また、使用している名前が正しいかどうかも確認する必要があります。たとえば、ルート組織のコンテキスト名は、`org//` を使用して決定します。ルート組織を操作する権限があるかどうか確認してください。`myorgunit/` を指定する場合もあります。

「対策」

アクセス権が正しく設定されているにもかかわらずこのメッセージが表示される場合は、適切な資格が与えられていない可能性があります。

これには以下の原因が考えられます。

- `keylogin` が実行されていない (NIS+ 主体はデフォルトでは「未認証」になる)
- NIS+ 以外のソースに対して `keylogin` が実行された

`/etc/nsswitch.conf` ファイルに `publickey: nisplus` エントリがあるかどうか確認するこれはおそらく認証エラーとなる

`fnlist` で下位組織のリストが表示されない

「症状」

`fnlist` に組織名を指定して実行しても何も表示されません。

「考えられる原因」

NIS+ の設定によって発生する問題です。下位組織は NIS+ ドメインでなければなりません。NIS+ ドメインには、`org_dir` というサブディレクトリが必要です。

「診断」

nisls コマンドを使用して、どんなサブディレクトリが存在するかを調べます。nisls をサブディレクトリごとに実行すれば、どのサブディレクトリに org_dir があるかを確かめることができます。org_dir のあるサブディレクトリが下位組織になります。

「対策」

「考えられる原因」を参照してください。

ホストコンテキストまたはユーザーコンテキストが作成できない

「症状」

fncreate -t を、user (または username)、または host (または hostname) を指定して実行しても何も起こりません。

「考えられる原因」

NIS_GROUP 環境変数が設定されていません。ユーザーコンテキストあるいはホストコンテキストを作成した際、所有権が名前空間を設定した管理者ではなく、ホストまたはユーザーにあったようです。したがって、fncreate を実行するには、NIS_GROUP 変数を設定して、グループ内の管理者によってコンテキストの操作が行えるようにする必要があります。

「診断」

NIS_GROUP 環境変数をチェックします。

「対策」

NIS_GROUP 環境変数に、コンテキストの管理者のグループ名を設定します。

作成したコンテキストを削除できない

「症状」

ホストコンテキストまたはユーザーコンテキストに対して fndestroy を実行しても、コンテキストが削除されません。

「考えられる原因」

ホストコンテキストまたはユーザーコンテキストの所有権を持っていないことです。ユーザーコンテキストあるいはホストコンテキストを作成した際、所有権が名前空間を設定した管理者ではなく、ホストまたはユーザーにあったようです。

「診断」

NIS_GROUP 環境変数をチェックします。

「対策」

NIS_GROUP 環境変数に、コンテキストの管理者のグループ名を設定します。

fnunbind を実行すると「name in use」というメッセージが表示される

「症状」

割り当てを削除しようとすると、「name in use」というメッセージが表示されます。指定する名前によってコマンドが正しく実行される場合と、そうでない場合があります。

「考えられる原因」

コンテキストの名前の割り当てを解除できません。この制限は、名前のないコンテキスト (orphaned context) が残るような場合に適用されます。

「診断」

fnlist コマンドを実行して、fnbind、fncreate に指定しようとする名前がコンテキストのものかどうか確認します。

「対策」

名前がコンテキストのものであれば、fndestroy コマンドを使用してコンテキストを削除します。

fnbind/ fncreate -s を実行すると「name in use」というメッセージが表示される

「症状」

-s オプションをつけて fnbind および fncreate を実行すると、指定する名前によっては「name in use」というメッセージが表示されます。

「考えられる原因」

fnbind、および fncreate に -s オプションをつけて実行すると、既存の割り当ては上書きされます。ただしそれによって名前のないコンテキストができるような場合、「name in use」というエラーメッセージが表示され、この操作は失敗します。このエラーは、名前のないコンテキストを回避するために発生します。

「診断」

fnlist コマンドを実行して、fnbind、fncreate に指定しようとする名前がコンテキストのものかどうか確認します。

「対策」

fndestroy コマンドを実行してコンテキストを削除してから、fnbind または fncreate を (先にエラーになった場合と同じ名前を指定して) 実行します。

実体のない名前を指定して fndestroy/fnunbind を実行しても「Operation Failed」が返らない

「症状」

実体のない名前を指定して fndestroy、fnunbind を実行しても、操作が失敗したことがまったく知らされない。

「考えられる原因」

fndestroy、fnunbind のセマンティクスが、「端末名が割り当てられていなくても、操作が success を返す」というようになっているためだと考えられます。

「診断」

問題が発生した場合と同じ名前を指定して fnlookup コマンドを実行します。「name not found」というメッセージが表示されるはずです。

「対策」

「考えられる原因」を参照してください。

その他の一般的なエラーメッセージ

attribute no permission
: 「呼び出し側が属性に対して何らかの操作をしようとしたが、アクセス権がないため実行できなかった」ということを意味します。

attribute value required
: 「値を指定しないで属性の作成が試みられたが、ネーミングシステムがそれを許可しなかった」ということを意味します。

authentication failure
: 「要求を作成した主体が、関連するネームサービスによって認証されなかったため、操作が完了しなかった」ということを意味します。NIS+ サービスを使用している場合は、(コマンド nisdefaults を使用して)「ユーザーが正しい主体として

認識されているかどうか」を確認し、また「公開鍵のソースが、マシンに正しく指定されているか」を確認します (/etc/nsswitch.conf ファイルに publickey:nisplus というエントリがあるかどうかを確認する)。

bad reference
「FNS がリファレンスの内容を理解できなかった」ということを意味します。「リファレンスの内容が壊れている」、「FNS のものであるとされているリファレンスが、FNS で復号化できない」といった場合に発生します。

Cannot obtain Initial Context
インストールに問題があることを示します (590 ページの「初期コンテキストが取得できない」を参照)。

communication failure
「FNS がネームサービスとコミュニケーションできず、操作を完了できなかった」ということを意味します。

configuration error
構成上の問題により発生するエラーです。次に例を示します。

(1) 割り当てテーブルが削除されました (FNS の問題ではありません)。

(2) ホストは NIS+ ホストディレクトリオブジェクトの中に存在しますが、ホストに対応する FNS ホストコンテキストがありません。

context not empty
「割り当てを含んでいるコンテキストを削除しようとした」ということを意味します。

continue operation using status values
「ステータスオブジェクトの中に残っている名前、名前との対応づけのできたリファレンスを使用して動作を続行する」ということを意味します。

error
「要求処理中に、ここまでにとりあげたどのエラーにも該当しない状況が発生した」ということを意味します。関連のあるネームサービスの状態をチェックし、特殊な問題が発生していないことを確認します。

illegal name
指定された名前が正しくないことを示します。

incompatible code sets
「操作中、互換性のないコードセットの文字列が使用された」、あるいは「サポートされていないコードセットが提供されている」ということを意味します。

invalid enumeration handle
「提供されている列挙ハンドルが正しくない」ということを意味します。「処理中に更新が発生した」などの理由が考えられます。

`invalid syntax attributes`
「指定された構文属性が正しくないために、構文を完全に決定できない」ということを意味します。

`link error`
指定された名前を使用して XFN リンクを作成する際に発生するエラーです。

`malformed link`
「`fn_ctx_lookup_link()` の動作中に、不正なリンク参照が検出された」ということを意味します。指定された名前が、リンクされていないリファレンスに結びつけられたということです。

`name in use`
「指定された名前が、コンテキスト中ですでに使用されている」ということを意味します。

`name not found`
指定された名前が見つからないということを意味します。

`no permission`
アクセス制御の問題により、動作が失敗したことを意味します。591 ページの「「No Permission」というメッセージが表示される (FNS)」、および 451 ページの「アクセス権がない」を参照してください。

`no such attribute`
オブジェクトが、指定の属性を持たないことを意味します。

`no supported address`
「`/usr/lib/fn` ディレクトリに、(FNS の名前に結びつけられたリファレンス中にはいくつかのタイプのアドレスがあるが) どのタイプの共用ライブラリも存在しない」ということを意味します。共用ライブラリの名前は、`fn_ctx_` *address_type*`.so` という形になります。リンクは通常、「`fn_ctx_`*address_type*`.so` から `fn_ctx_`*address_type*`.so.1` へ」という形で行われます。

たとえば、リファレンスのアドレスのタイプが `onc_fn_nisplus` だとすると、共用ライブラリの存在するパスは `/usr/lib/fn/fn_ctx_onc_fn_nisplus.so` ということになります。

`not a context`
「リファレンスが、正しいコンテキストに対応していない」ということを意味します。

`partial result returned`
操作によって得られた結果が不完全であることを意味します。

`Success`
(1) 要求は成功しました。このメッセージは、NIS+ のエラーコード定数 `NIS_SUCCESS` によって生成されます。詳細については、`nis_tables(3N)` のマニュアルページを参照してください。

(2) 操作が成功しました。

`syntax not supported`
「サポートされていないタイプの構文が使用された」ということを意味します。

`too many attribute values`
「1 つの属性に、ネーミングシステムでサポートされている範囲を超える数の値を持たせようとした」ということを意味します。

`unavailable`
操作に必要なネームサービスが利用できないということを意味します。

`link loop limit reached`
「複数の名前を結びつける際、XFN リンクが原因で完了しないループが検出された」、または「一回の操作における XFN のリンクの数が、実装で定義された上限を超えた」ということを意味します。

パート V　ネームサービス間の移行

このパートでは、NIS から NIS+ への移行方法および NIS+ から LDAP への移行方法について説明します。

第 26 章

NIS から NIS+ への移行

この章では、NIS ネームサービスから NIS+ ネームサービスへの変換方法について説明します。ここでは、2 つのネームサービスの違いについて説明し、推奨する移行方法の概要を示します。

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。NIS+ から LDAP への移行支援ツールは、Solaris 9 オペレーティング環境で使用できます (パート V を参照)。 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 と NIS+ の相違

NIS と NIS+ の間には、移行に影響を与える相違点がいくつかあります。たとえば、NIS は、1 つのドメイン (またはいくつかの別々のドメイン) を持つ平坦な (階層型ではない) 名前空間を使用しますが、NIS+ は、DNS に似たドメイン階層を使用します。このため、NIS+ に変換する前に、NIS+ の名前空間を設計する必要があります。また、 NIS+ にはセキュリティを強化するための機能があります。これにより、名前空間内の情報だけでなく、名前空間の構造的な構成要素へのアクセスも制限されます。

このような相違点から、NIS+ が単に NIS をアップグレードしたものではなく、完全に新しい製品であることがわかります。NIS から NIS+ へ移行する際は、主にこれらの製品間の相違がポイントになります。

この章では、これらの相違点について、次に示す一般的な用語を使って説明します。NIS+ への移行を正しく行うには、これらの用語を理解することが重要です。

- ドメイン構造

- 相互運用性
- サーバーの構成
- 情報の管理
- セキュリティ

ドメインの構造

NIS+ は、NIS に置き換わるものとして設計されており、単に NIS をアップグレードしたものではありません。このことは、NIS+ のドメイン構造を調べると明らかです。NIS のドメインは平坦で、階層を持つことができません。これに対して、NIS+ のドメインは平坦な場合もありますが、階層構造のドメインを作成できます。この階層は、ルートドメインと、その下の任意の数のサブドメインから構成されます。

NIS のドメイン構造は、1980 年代に一般的であったクライアントとサーバー間のコンピューティングネットワークの管理の要件に対応したものでした。つまり、数百のクライアントと少数の多目的サーバーを持つ、クライアントとサーバー間のネットワークを対象としていました。

NIS+ は、世界中のサイトの専用サーバー 10〜100 台によってサポートされるクライアント 100〜10000 台をサポートするネットワークを対象としています。こうしたネットワークは、複数の「信頼性の低い」公衆ネットワークに接続されています。このようなネットワークの規模と構成を維持するには、新しい独立した管理方式が必要です。NIS+ のドメイン構造は、これらの要件を満たすように設計されています。NIS+ のドメイン構造は、次の図に示すように、DNS のドメイン構造に似ています。

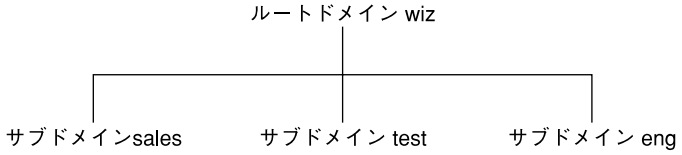


図 **26–1** NIS+ のドメイン

ドメインを階層構造にすると、規模の小さいものから非常に大きいものまで、広い範囲のネットワークに NIS+ を使用することができます。また、NIS+ のサービスを組織の成長に対応させることもできます。NIS+ ドメイン構造の詳細については、66 ページの「ドメイン」 を参照してください。

DNS、NIS、NIS+ の相互運用性

NIS+ が提供する相互運用性とは、NIS からの移行と、NIS サービスによって提供されていた DNS とを継続して併用できることを意味します。 NIS+ には、NIS からの移行に役立つ NIS 互換モードと情報転送ユーティリティがあります。NIS 互換モードを使

用すれば、Solaris オペレーティング環境で動作している NIS+ サーバーは、NIS+ クライアントからの要求に応答しながら、NIS クライアントからの要求にも応答できます。また、管理者は、情報転送ユーティリティを使って NIS のマップと NIS+ のテーブルを同期させることができます。

NIS 互換モードの設定に必要な手順は、標準 NIS+ サーバーで使用する手順と若干異なります。また、NIS 互換モードは、NIS+ 名前空間内のテーブルとセキュリティ上の関連を持っています。

NIS+ サーバーが NIS 互換モードで実行されている場合、NIS のクライアントコンピュータは、 NIS+ のクライアントコンピュータとは異なる方法で NIS+ 名前空間にアクセスします。次にこの違いを示します。

- NIS のクライアントマシンは、NIS+ のテーブルパスまたはリンクをたどることも、他のドメインのデータを読み取ることもできません。
- NIS+ サーバー上で `rpc.nisd` に `-Y -B` オプションを付けて実行している場合、NIS+ サーバー内で解決できない NIS のクライアントマシンからのホスト要求を DNS に転送することができます。しかし、NIS+ クライアントからのこのような要求は転送されません。NIS+ クライアントマシンの DNS 要求の転送は、`/etc/resolv.conf` ファイルと `/etc/nsswitch.conf` ファイルの構成によって制御されます。詳細については、第 1 章を参照してください。
- 許可を持つ NIS+ の管理者は、`passwd` コマンドを使用して、パスワードの有効期限やロックの設定など、パスワードに関連するすべての管理業務を実行できます。NIS+ クライアントのユーザーは、`passwd` コマンドを使用して、自分自身のパスワードを変更できます。
- ローカルサブネット上のすべてのサーバーが応答しなくなった場合でも、NIS+ クライアントマシンは、そのドメインの複製サーバーのどれかと通信できれば、そのネームサービスの呼び出しに応答を得ることができます。NIS クライアントマシンは、サーバー名が設定されていないと、そのサブネットの外部にあるネットワーク上の情報にアクセスすることができません。サーバー名は、`ypset` によって、または Solaris NIS クライアントの場合には `ypset` サブネットの外部にあるネットワーク上の情報にアクセスすることによって設定されます。
- NIS クライアントマシンは、受信中のデータが承認された NIS サーバーから送信されたものかどうかについては確認できません。これに対して、承認された NIS+ クライアントでは、承認された NIS+ サーバーからデータが送信されていることを確認できます。
- NIS 環境では、サーバーが応答しなくなったとき、NIS の `yp_match()` 呼び出しは、サーバーが応答して要求に応じるまで、呼び出しを試行し続けます。NIS+ の API (アプリケーションプログラムインタフェース) では、このような事態が発生すると、アプリケーションに対してエラーメッセージを返します。

Solaris 2.3 では、NIS 互換モードで DNS 転送をサポートします。Solaris 2.2 では、DNS 転送を可能にする「パッチ (patch `#101022-06`)」が提供されています。DNS 転送を可能にするパッチは、Solaris 2.0 と 2.1 では利用できません 。

NIS+ ドメインは、インターネットに直接接続できません。ただし、NIS+ クライアントマシンは、ネームサービススイッチ経由でインターネットに接続できます。クライアントのスイッチ構成ファイル (`/etc/nsswitch.conf`) を設定して、NIS+ テーブル以外に、DNS ゾーンファイルまたは NIS マップの情報をクライアントから検索できます。

サーバーの構成

NIS+ のクライアント サーバー構成は、各ドメインが複数のサーバーによってサポートされているという点で、NIS と DNS の構成に似ています。メインサーバーは「マスターサーバー」と呼ばれ、バックアップサーバーは「複製サーバー」と呼ばれます。マスターサーバーと複製サーバーの両方で 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維持しています。

ただし、NIS+ では、NIS の場合とはまったく異なる方法でデータベースが更新されます。NIS が開発された時点では、NIS が格納する情報のほとんどが静的なものと想定されていました。したがって、NIS の変更は手作業で処理し、そのマップ内の情報が変更されるたびにマップを作成しなおし、すべてを伝達させる必要があります。

これに対して NIS+ では、複製サーバーに対して変更分だけの更新ができます。マスタサーバー上のマスタデータベースを変更する必要はありますが、変更内容は複製サーバーにも自動的に伝達されます。「make」マップを再度作成したり、情報が伝達されるまで何時間も待つ必要はありません。伝達は、3 〜 4 分で終了します。

情報の管理

NIS+ は、マップやゾーンファイルではなく、「テーブル」に情報を格納します。NIS+ には、図 26–2 に示すように、17 種類の定義済みテーブル (システムテーブル) があります。

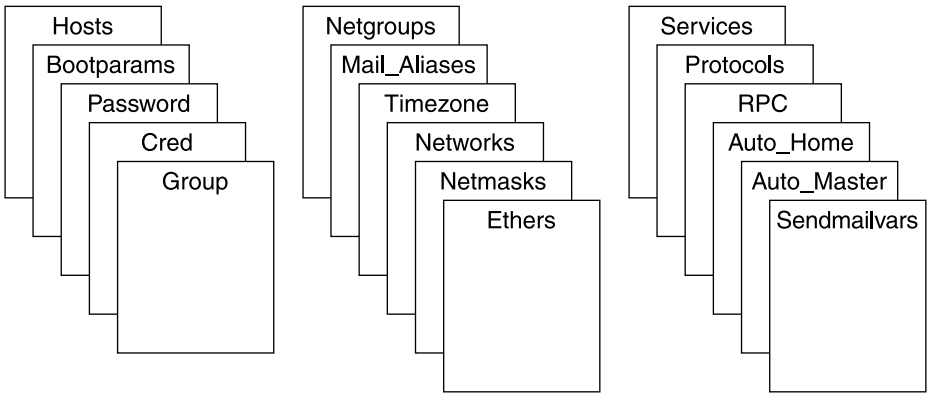


図 **26–2** NIS+ の標準テーブル

NIS+ テーブルは、ASCII ファイルでなく、NIS+ リレーショナルデータベース内のテーブルです。NIS+ テーブルの内容は、NIS+ のコマンドを使用しなければ表示したり編集したりできません。

NIS+ テーブルには、NIS で使用したマップに比べて大きく改善された機能が 2 つあります。その 1 つは、NIS+ テーブルを、最初の列 (「キー」とも呼ぶ) だけでなく、任意の検索可能な列によって検索できるという機能です。特定の列が検索可能であるかどうかを知るには、テーブルに対して `niscat -o` コマンドを実行してください。コマンドは、そのテーブルの列と属性のリストを返します。この検索機能により、NIS によって使用される `hosts.byname` マップと `hosts.byaddr` マップのような重複するマップを持つ必要性がなくなります。もう 1 つは、NIS+ テーブル内の情報に対して、テーブルレベル、エントリ (行) レベル、列レベルという 3 つのレベルでアクセスを制御できることです。

NIS マップがサーバーのディレクトリ `/var/yp/`*domainname* に置かれるのに対して、NIS+ ディレクトリは `/var/nis/data` に置かれます。NIS+ テーブルはデータベースの中に格納されます。テーブルの情報は、データベースへの要求が出されると、メモリーに読み込まれます。データを要求された順序でメモリーに保存すると、ディスクへのアクセスを最小限に抑えられるため、要求への応答時間を短縮することができます。

セキュリティ

NIS+ のセキュリティ機能は、名前空間の情報と名前空間そのものの構造を不正なアクセスから保護します。NIS+ のセキュリティ機能は、「認証」と「承認」という 2 つの手段によって行われます。 認証とは、NIS+ サーバーが、特定の要求を送信した NIS+ の「主体」(クライアントユーザーまたはクライアントマシン) を識別する処理を指します。承認とは、サーバーが、その主体 (クライアントマシンまたはクライアントユーザー) に許可されたアクセス権を識別する処理を指します。

つまり、最高レベルの NIS+ セキュリティ機能がサイトに導入されている場合、名前空間内の情報にアクセスするには認証された NIS+ クライアントでなければならず、その情報にアクセスするための適切なアクセス権を持っていなければなりません。さらに、名前空間へのアクセス要求は、NIS+ のクライアントライブラリルーチンか NIS+ の管理コマンドによって行われた場合にだけ有効になります。また、NIS+ のテーブルと構造を直接編集することはできません。

推奨する移行手順

次に、NIS から NIS+ への移行において推奨する移行手順の概略を示します。

1. 基本的な移行の方針について確認します。
2. NIS+ について理解します。
3. 最終的な NIS+ 名前空間を設計します。
4. セキュリティの方式を選択します。
5. NIS 互換モードの使用方法を決定します。
6. 移行の準備を完了します。
7. 移行を実行します。

この章の以下の部分では、上記の手順の各段階について詳しく説明します。

移行の方針

移行を開始するにあたって、次に示す基本的な方針を確認してください。

移行をすぐに実行するのではなく、別の方法を考慮する

NIS+ へのアップグレードは、NIS+ を Solaris オペレーティング環境に完全に移行した段階で行ってもかまいません。このようにすれば、リソースを移行作業だけに利用することができます。NIS を Solaris オペレーティング環境上で稼動しながら、NIS+ への移行を準備することができます。

処理を簡略化する

いくつかの手順により、移行を簡略にすることができます。これらの手順を実行すると、NIS+ の効果は減少しますが、サーバーの台数は少なくて済み、管理に要する時間も減ります。移行が完了すれば、NIS+ の設定を変更して、サイトの要求に完全に適合させることができます。次にいくつかの推奨事項を示します。

- ドメイン名を変更しない

- 階層を使用しないで、平坦な NIS+ 名前空間を使用する
- NIS 互換機能を使用する
- デフォルトのテーブルとディレクトリ構造を使用する
- Solaris 2.5 以降のリリースを使用する場合は、クライアントの資格を設定しない

1 種類のソフトウェアリリースを使用する

移行に使用する Solaris オペレーティング環境と NIS+ のバージョンを決定します。各バージョン間には若干の違いがあるため、複数のバージョンを同時に使用すると、移行の処理が不必要に複雑になります。Solaris 製品の 1 バージョンだけを選択して、それに対応するバージョンの NIS+ を使用してください。

現在のリリースは、設定スクリプトなどのほとんどの機能を備えています。通常の操作に必要な Solaris 2.6 のパッチを用意し、すべてのサーバーとクライアントに同じパッチがインストールされていることを確認してください。

クライアントユーザーへの影響を最小限に抑える

クライアントユーザーに関して、考慮するべき点が 2 つあります。1 つは、ユーザーがサービスの変更に気が付かないようにするということです。もう 1 つは、移行作業そのものがユーザーに与える混乱を最小限に抑えるということです。2 番目のポイントについては、ユーザーに移行作業を要求するのではなく、必ず各ドメインの管理者がそのクライアントマシンの NIS+ への移行作業を行うようにすれば解決できます。これにより、正しい手順が実行され、その手順がクライアントマシン全体でも一貫して実行されます。したがって、問題があっても、管理者がただちに処理することができます。

禁止事項

- 現在 NIS によって提供されているネームサービス、または NIS の動作を変更しないこと
- DNS の構造を変更しないこと
- IP ネットワークの形態を変更しないこと
- NIS を使用するアプリケーションを NIS+ にアップグレードしないこと。NIS+ API への移行はあとで行う
- 実装段階では、NIS+ に機能は追加しないこと。 機能の追加はあとで行う

NIS+ について理解する

NIS+ に関する理解を深めておいてください。特に、この章の前半で要約した概念 (このマニュアルの後半で詳しく説明します) については、よく理解しておく必要があります。

NIS+ を理解するための最もよい方法の 1 つは、プロトタイプの名前空間を作成することです。製品を実際に経験するということに優る方法はありません。システム管理者には、業務に支障をきたさないテスト環境での練習が必要です。

注 – プロトタイプのドメインを、実際の NIS+ 名前空間としては使用しないでください。プロトタイプですべてを学んだら、そのプロトタイプは削除して、名前空間の構成上の問題が起こらないようにします。計画をすべて終えたら、新しく実際の名前空間を作成してください。

テストドメインを作成するときは、小規模の管理しやすいドメインを作成してください。NIS+ 設定スクリプトを使用して、単純なテストドメインとサブドメインを計画および作成する方法 (NIS 互換モードも使用可能) については、95 ページの「NIS+ 名前空間サンプルの作成」 を参照してください。

注 – NIS+ 名前空間を設定するときは、説明に従って NIS+ スクリプトを利用することをお勧めします。この手順では、最初に NIS+ スクリプトを使用して、基本的な NIS+ 名前空間を設定します。続いて NIS+ コマンドセットを使用して、各自のニーズに合わせて名前空間をカスタマイズします。

最終的な NIS+ 名前空間を設計する

609 ページの「NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする」の指針に従って、最終的な NIS+ 名前を設計します。名前空間の設計中は、NIS からの移行によって生じる制限を気にする必要はありません。これらの制約は、最終的な NIS+ の目的を明確にしてから変更することができます。

セキュリティの方式を選択する

NIS+ のセキュリティは、ユーザーと管理者にとって非常に有益ですが、ユーザーにも管理者にも、より詳しい知識と設定作業が必要になります。 また、計画上の決定をいくつか行う必要もあります。631 ページの「NIS+ セキュリティの影響について理解する」では、NIS+ セキュリティの持つ意味と、NIS+ 名前空間でセキュリティを使用する場合に必要な決定事項について説明します。

NIS 互換モードの使用方法を決定する

移行の間は、NIS と NIS+ の名前空間を並行して使用することは事実上避けられません。2 つの名前空間を同時に使用するには、さらに資源を追加する必要があるため、各サイトが二重のサービスを使用する時間、または名前空間内の二重サービスの適用範囲を減らす (たとえば、可能なかぎり多くのドメインを NIS+ に変換するなど) ように努めてください。

650 ページの「NIS+ への移行の準備 - 他システムに対する NIS+ の影響を調べる」では、NIS 互換モードに関連する移行の問題を説明し、 NIS から、NIS 互換を経由して、NIS+ へ完全に移行する方法を示します。

移行を実行する

657 ページの「NIS+ の実装の概要」では、推奨される一連の手順を示して、それまでに計画した移行を実際に実行します。

NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする

名前空間を設計するときは、NIS からの移行によって生じる制限を気にしないでください。NIS+ ドメインは、後で最終的な NIS+ 構成がどのようになるかがわかってから変更することができます。

ドメイン構造など、各サイトで使用する情報管理のモデルを選択します。各サイトでの情報の作成、格納、使用、管理について明確な方針がないと、この節で示す設計の決定を行うことが困難になります。たとえば、作業に必要以上の経費がかかる設計をしてしまう可能性があります。また、要求に合わない名前空間を設計してしまうおそれもあります。一度設定した名前空間の設計の変更には時間と手間がかかります。

名前空間の構造を設計する

NIS+ 名前空間の設計は、いろいろな作業の内で最も重要なものの 1 つです。これは、 NIS+ を一度設定した後にドメイン構造を変更するのは、時間のかかる複雑な作業になるからです。この作業が複雑になるのは、情報、セキュリティ、管理の各方針が名前空間のドメイン構造に組み込まれているからです。ドメインを編成しなおす場合は、情報の再編成、セキュリティの再設定、管理方針の再設計が必要になります。

NIS+ 名前空間の構造を設計するときは、この章の以下の節で説明する要素を考慮してください。

ドメインの階層

NIS+ ドメイン階層には、名前空間をより管理しやすい複数の構成要素に分割できる、という利点があります。各構成要素には独自のセキュリティ、情報管理、管理方針を持たせることができます。クライアントの数が 500 を超える場合、あるユーザーグループに異なるセキュリティに方針を設定したい場合、あるいは地理的に分散したサイトがある場合は、階層を使用することをお勧めします。

ドメイン階層の必要がなければ、階層を使用しません。こうすることにより、NIS+ への移行が簡略化されます。すべてのユーザーが同じ NIS ドメイン内にいる場合、これらのユーザーは、完全指定名を使用しなくてもお互いを直接認識することができます。しかし、NIS+ 階層を作成すると、ユーザーは別々のドメインに置かれます。つまり、完全指定名か完全指定パスを使用しないかぎり、あるドメインにいるユーザーは別のドメインにいるユーザーを直接認識することができません。

たとえば、`sales.com.` および `factory.com.` というサブドメインが、 `.com.` ドメインから作成されているとします。このとき、`sales.com.` ドメインのユーザー `juan` が `factory.com.` ドメインのユーザー `myoko` にメールを送信するには、送信先のユーザー名を `myoko` ではなく `myoko@hostname.factory.com.` または `myoko@hostname.factory` と指定する必要があります。送信先のユーザーが同じドメインに存在する場合は、`myoko` だけでかまいません。リモートログインでもドメイン間の完全指定名が必要です。

テーブル間のパスを使用すると、あるドメインのテーブルと別のドメインのテーブルとの間に接続を設定することができますが、ドメイン階層を使用するメリットはなくなります。また、NIS+ サービスの信頼性も低くなります。これは、クライアントが、各自のホームドメインの利用状況だけでなく、各自のテーブルにパス指定される他のドメインの利用状況にも依存するようになるためです。テーブル間のパスを使用すると、要求への応答時間も長くなります。

ドメインの階層 – Solaris 2.6 以前のリリース

Solaris 2.6 以前のリリースでは、各サブドメインの NIS+ サーバーは、そのドメインではなく、親ドメインに含まれます。ただし、ルートドメインは除きます。サーバーとサブドメインの関係がこのような関係になっていると、サーバーがネームサービスデータをサブドメインから取得できることを想定しているアプリケーションの場合に、問題が発生します。たとえば、サブドメインの NIS+ サーバーが NFS サーバーを兼ねている場合、サーバーはネットグループ情報をサブドメインからは取得しません。代わりに、サブドメインの上位ドメインからネットグループ情報を取得します。この点に注意してください。階層によって問題が発生する可能性がある場合の別の例としては、遠隔ログインするユーザーが自分のマシンからでは実行できないコマンドを実行する場合に、この NIS+ サーバーも使用する場合です。ルートドメインが 1 つしかない場合には、NIS+ ルートサーバーは自分がサーバーであるドメイン内にいるので、このような問題は発生しません。

ドメインの階層 – Solaris 9

Solaris オペレーティング環境では、ドメインの NIS+ サーバーは、自分がサーバーであるドメイン内に存在できます。したがって、サーバーはクライアントが使用している名前をドメイン名に設定することができます。残りのドメインの階層との機密保護通信を実行できるサーバーの機能には影響を与えません。

ドメインの階層を設計する

ドメイン階層を使い慣れていない場合は、まず 第 2 章 を参照してください。 このマニュアルでは、NIS+ のドメイン構造、情報の格納、セキュリティについて説明しています。

ドメイン階層の各構成要素を理解したら、最終的な階層を示す図を作成します。この図は、設定手順を進めるうえで非常に参考になります。少なくとも、次の問題について考慮する必要があります。

- 組織的、または地理的な構造を用いた階層
- 上位ドメインへの接続
- ルートドメインでのクライアントサポート
- ドメインの大きさとドメインの数の比較
- レベルの数
- セキュリティレベル
- 複製サーバーとその数
- 情報の管理

ドメインは 1 つのオブジェクトではなく、オブジェクトの集合に対する参照であることを忘れないでください。したがって、ドメインをサポートするサーバーは、実際にはドメインと関連しないでドメインのディレクトリと関連しています。ドメインは、次の図に示すように、*domain*、`ctx_dir.` *domain*、`org_dir.`*domain*、および`groups_dir.`*domain* という 4 つのディレクトリで構成されます。

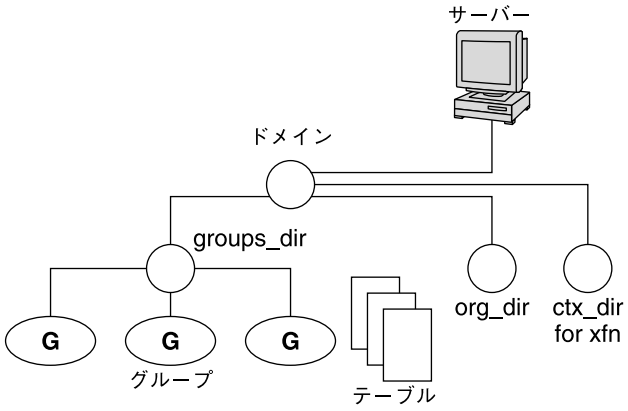


図 **26–3** サーバーとドメインの関係

組織的または地理的な構造による階層

NIS+ の主な利点の 1 つに、名前空間をより小さく、より管理しやすい部分に分割できるという機能があります。たとえば、次の図 (仮想企業 Doc Inc. の階層) のような組織の階層を作成できます。

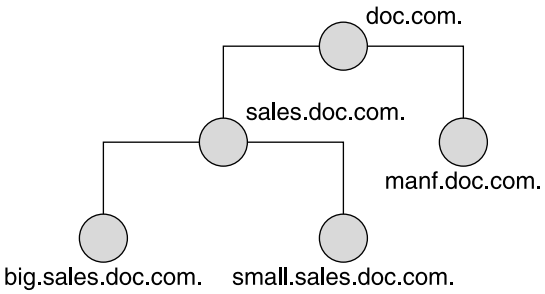


図 **26–4** 論理的な組織構造による NIS+ 階層の例

次の図に示すように、組織ではなく建物によって階層を構成することもできます。

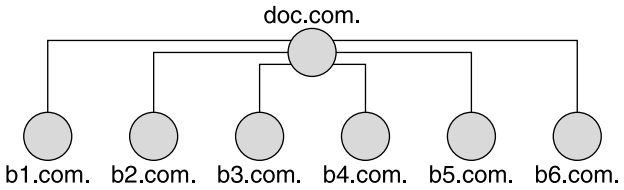


図 **26–5** 物理的な位置による NIS+ 階層の例

どの構成を選択するかは、主に名前空間の管理方法とクライアントによる名前空間の使用方法によって決まります。たとえば、`factory.com.` ドメインに属するクライアントが Doc Inc. の建物全体に分散している場合は、名前空間を建物によって構成しないでください。 クライアントは他のドメインに常にアクセスしなければならないため、他のドメインへの資格をクライアントに与えなければならず、ルートマスタサーバーとの通信量が増加します。この場合は、組織ごとにクライアントを構成してください。これに対して、建物に基づくドメインは、組織に変更があっても影響を受け付けません。

ネットワークの物理的配置による制限を受けないようにしてください。NIS+ 名前空間は、NIS クライアントをサポートしなければならない場合を除いて、物理的ネットワークに一致する必要はありません。名前空間で必要なドメインの数は、選択した階層の種類によって決まります。

今後の拡張計画を検討します。現在の NIS+ ルートドメインが、将来別の NIS+ ドメインの下に配置されるかどうかを検討してください。現在の設定を変更するには、膨大な作業が必要になります。名前空間での今後のドメインの必要性を見積って、混乱なくそれらのドメインを収容できる構造を設計してください。

上位ドメインへの接続

NIS+ 名前空間をインターネットや DNS のドメインなどの上位ドメインに接続するかどうかを検討します。現在 DNS 階層のもとで NIS を使用している場合は、NIS+ 名前空間に、 NIS ドメインだけを置き換えるか、サイト全体の DNS/NIS 構造を置き換えるかを決めます。

ルートドメインでのクライアントサポート

図 26–4 と図 26–5 に示した Doc Inc. の 2 つのドメイン階層を例にして説明します。まず、すべてのクライアントを、ルートドメインの下のドメインに配置するかどうかを調べます。また、一部のクライアントをルートドメインに配置するかどうかを調べます。ルートドメインの目的がそのサブドメインのルートとして動作することだけかどうか、あるいはルートドメインがそれ自身のクライアントグループをサポートするかどうかを調べます。 すべてのクライアントをドメインの最下層に配置し、管理に使用するクライアントだけを中間ドメインに配置することができます。たとえば、図 26–4

でこの計画を実行すると、すべてのクライアントが
big.sales.com.、small.sales.com.、factory.com. の各ドメインに配置され、管理に使用されるクライアントだけが .com. と sales.com. ドメインに配置されます。

また、汎用部門のクライアントを上位レベルのドメインに置くこともできます。たとえば図 26–5 では、.com. ドメインは建物によって構成されていて、設備部門のクライアントを .com. ドメインに置くことができます。しかし、ルートドメインは単純で比較的ゆとりのある状態に維持しておく必要があるため、この配置はお勧めできません。

ドメインの大きさとドメインの数と比較

現在 NIS+ の実装は、1 つのドメインあたり最大 1000 の NIS+ クライアント、1 つのドメインあたり最大 10 の複製サーバーを設定するように最適化されています。このようなドメインには、通常 10000 のテーブルエントリがあります。この制約は、現在のサーバー発見プロトコルに起因しています。NIS+ クライアントが 1000 を超える場合は、名前空間を異なる複数のドメインに分割して、階層を作成してください。

しかし、階層を作成すると、状況が複雑になって対処しにくくなるおそれがあります。 1 つのドメインを大きくした方が、複数の小さいドメインを作成するよりも管理が容易なため、階層ではなくより大きなドメインを作成したいと考えるかもしれません。数の少ない大きいドメインでは、各自が作成するスクリプトを使って、作業をより容易に自動化できるため、それらのドメインのサービスを担当する熟練の管理者が少なくて済み、管理に要する手間と費用を削減することができます。しかし、ドメインを小さくすると性能が向上し、各自のテーブルをより簡単にカスタマイズすることができます。また、小さいドメインでは、管理をより柔軟に行うこともできます。

レベルの数

NIS+ は、複数レベルのドメインを処理するように設計されています。NIS+ は、任意の数のレベルに対応することができますが、レベルの数が多すぎる階層は、管理が困難です。たとえば、オブジェクトの名前は、長くてややこしいものになる場合があります。したがって、1 つのドメインに対するサブドメインの数は 20 までとし、NIS+ 階層のレベルの数は 5 までに制限するようお勧めします。

セキュリティレベル

名前空間は、通常、セキュリティレベル 2 で管理します。ただし、ドメインごとに異なるセキュリティレベルを使用する場合は、ここでそのレベルを指定する必要があります。631 ページの「NIS+ セキュリティの影響について理解する」では、セキュリティレベルの詳細を説明しています。

複数の時間帯にまたがるドメイン

地理的に分散した組織では、ドメイン階層を機能のグループによって構成すると、1つのドメインが複数の時間帯にまたがることがあります。ドメインが複数の時間帯にまたがることが「決して」ないようにしてください。複数の時間帯にまたがるドメインを構成する必要があるときは、複製サーバーの時刻は、マスタサーバーの時刻に合わせられることに注意してください。これにより、データベースの更新は、万国標準時 (グリニッジ標準時) を使って正しく行われます。時刻が重要な他のサービスに複製サーバーが使用されると、このことが問題の原因となるおそれがあります。複数の時間帯にまたがるドメインを動作させるには、 NIS+ をインストールするときに、複製サーバーの `/etc/TIMEZONE` ファイルを、マスタサーバーの時間帯に合わせてローカルに設定する必要があります。複製サーバーがいったん動作を始めると、時刻が重要なプログラムの中には、万国標準時かローカル時刻のどちらを使用するかによって、正しく作動するものとしないものがでてきます。

情報の管理

NIS+ 名前空間の情報の管理は、中央の制約の範囲内でローカルに行うことをお勧めします。情報は、できるかぎりそのホームドメインで管理するべきですが、広域の名前空間レベルで設定された指針または方針に従ってください。これにより、ドメインの独立性を強化する一方で、ドメイン間の整合性を維持することができます。

ドメイン名

名前の長さと複雑さについて検討します。まず、内容がわかりやすい名前を選択します。たとえば、 Sales は BW23A よりも内容をわかりやすく表しています。次に、短い名前を選択します。管理業務をより簡単にするためには、あまり長い名前を付けないでください (長い名前の例: `administration_services.corporate_headquarters.doc.com.`。

ドメイン名は、左から右に形成され、ローカルドメインから始まって、ルートドメインで終わります。ルートドメインには、常に少なくとも 2 つのラベルがなければならず、ドットで終了しなければなりません。2 番目のラベルは、「`com.`」などの Internet のドメイン名にすることができます。

サイト内とインターネット全体の電子メールドメインを対象に、このドメイン固有の名前の意味について検討する必要があります。

移行の方式によっては、NIS 上のドメイン名を希望の構造に変更してから、 NIS+ ドメインに移行することもできます。

電子メール環境

NIS が平坦なドメイン空間を持つのに対して、NIS+ はドメイン階層を持つことができるため、NIS+ への移行はメール環境にも影響があります。NIS では、必要なメールホストは 1 つだけです。NIS+ でドメイン階層を使用すると、名前空間の各ドメインごとに 1 つのメールホストが必要になります。これは、各ドメインの名前が一意ではなくなるためです。

したがって、ルートドメインにないクライアントの電子メールアドレスが変更される場合があります。一般的に、クライアントの電子メールアドレスは、ドメイン名が変更されるか、または階層に新しいレベルが追加されると変更されます。

以前の Solaris のリリースでは、これらの変更に非常に手間がかかりました。このリリースでは、`sendmail` の拡張機能がいくつか追加され、作業が簡単になっています。さらに、NIS+ には、`sendmailvars` テーブルが追加されています。`sendmail` プログラムは、まず `sendmailvars` テーブル (図 26–7 を参照) を見てから、ローカルな `sendmail.cf` ファイルを調べます。

注 – メールサーバーが、そのサポート対象となるクライアントの NIS+ ドメイン内にあることを確認してください。また、パフォーマンス上の理由から、 メールサーバーに対して他のドメイン内のテーブルへのパスを指定しないでください。

DNS での新しいメールアドレスの影響について検討してください。DNS の MX レコードを修正しなければならない場合があります。

サーバーの必要条件を決める

各 NIS+ ドメインは、一組の NIS+ サーバーによってサポートされています。各組は、1 つのマスタサーバーと 1 つ以上の複製サーバーを持っています。これらのサーバーは、ドメインのディレクトリ、グループ、テーブルを格納して、ユーザー、管理者、アプリケーションからのアクセス要求に応答します。各ドメインをサポートしているのは、一組のサーバーだけです。一組のサーバーで、複数のドメインをサポートすることができますが、お勧めできません。

NIS+ サービスでは、マスターサーバーを少なくとも 1 つ各 NIS+ ドメインに割り当てる必要があります。各ドメインが必要とする複製サーバーの数は、通信量の負荷、ネットワークの構成、NIS クライアントの有無などによって決まります。サーバーメモリーの量、ディスク記憶容量、プロセッサの速度は、クライアントの数とサーバー上に置かれる通信量の負荷によって決まります。

Solaris オペレーティング環境で稼動する任意のマシンを NIS+ サーバーにすることができます。ただし、ハードディスク要件を満たしていなければなりません。NIS+ のサーバー用、クライアント用のソフトウェアは、どちらも Solaris 製品に含まれています。つまり、Solaris オペレーティング環境がインストールされていれば、任意のマシンをサーバーまたはクライアント、あるいはその両方にすることができます。

NIS+ 名前空間をサポートするために必要なサーバーを決定するとき、次の節で説明する要因を考慮する必要があります。

サポートするドメインの数

まず、階層内のドメインごとに、マスターサーバーを 1 つずつ割り当てます。次の図は、可能な割り当ての例です。

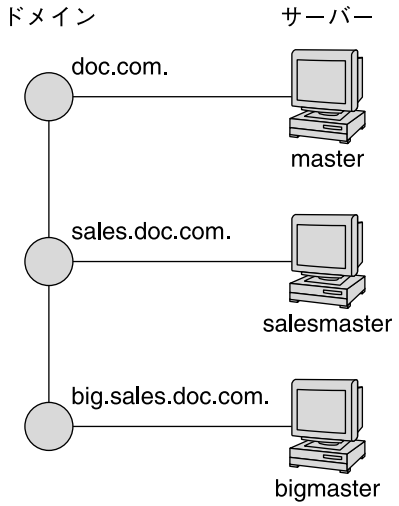


図 **26–6** サーバーをドメインに割り当てる

1 つ以上の複製を各ドメインに割り当てます。複製を使うと、マスターサーバーが一時的に使用不可能な場合でも、要求に応答することができます。使用する複製の数については、618 ページの「複製サーバーの数」を参照してください。

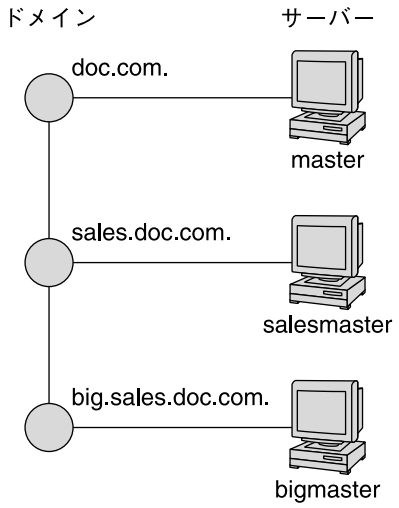


図 **26–7** ドメインへの複製サーバーの追加

複製サーバーの数

ドメインに最適なサーバーの数 (マスターと複製) は、数多くの要因によって決まります。

- NIS+ はローカルサブネット上の同時通信に依存しないため、NIS+ マスターサーバーが必要とする複製の数は NIS サーバーよりも少なくなります。
- すべてのドメインには少なくとも 1 つの複製サーバーがなければなりません。その理由は、マスターサーバーが一時的に使用不可能になったときに NIS+ サービスが破壊されないようにするためです。
- ドメインには 10 以上の複製があってはなりません。その理由は、情報更新が多くの複製に伝わるときにネットワークの通信量とサーバーの負荷が増加するためです。
- クライアントの種類。クライアントマシンが古くて遅いと、新しくて高速のマシンより必要な複製の数が少なくなります。
- 設計するドメイン階層が広域ネットワーク (WAN) リンクをまたがる場合、WAN リンクの両側に複製を置くと安全に実行できるようになります。この場合、リンクの 1 つの側にマスターサーバーと 1 つ以上の複製を設置して、他の側にも 1 つ以上の複製を設置するようにします。こうすると、WAN リンクが一時的に使用不可能になった場合でも、リンクの片側にいるクライアントは NIS+ サービスを継続して使用することができます (しかし、サーバーを WAN の両側に置くと、物理的配置によってではなくグループ機能別に構成されている名前空間の構造が変化します。その原因は、複製は地理的に異なったドメインで、物理的に常駐するためです)。

多くのサイトが分散されている組織では、各サイトは独自のサブドメインを必要とする場合もあります。サブドメインマスターは、さらにレベルの高いドメインに配置されます。その結果、ポイントツーポイントのリンク間では非常に多くの通信量が発生します。地域的な複製を作成すると、要求への応答を早くすることができ、さらにリンク両端でのポイントツーポイントの通信量を少なくすることもできます。この構成の場合、ルックアップはサブドメインで処理できます。

- ドメイン内のサブネット数。できれば、1 つの複製を各サブネット上に置きます (ただし、ドメイン全体で 10 以上の複製を使わないでください)。Solaris 1.x NIS クライアントがない場合または、NIS クライアントをサポートするために NIS+ サーバーを NIS 互換モードで使う場合、これらの場合以外には、すべてのサブネットに複製を配置する必要はありません。NIS クライアントは、同じサブネット上にないサーバーにアクセスしません。唯一の例外は Solaris オペレーティング環境の NIS クライアントであり、`ypinit`(1M) を使って NIS サーバーのリストを指定することができます。この場合、ネットマスク数は正しく設定しなければなりません。
- ユーザーと管理者がルックアップを実行する方法。`niscat` *table* | `grep` *name* コマンドは、`nismatch` *name table* コマンドが使用するものよりもはるかに多くのサーバー資源を使用します。
- サーバーの種類。新しくて高速なサーバーは、古くて遅いマシンが実行するサービスよりも高速で、より効率的なサービスを行うことができます。したがって、サーバーが強力になるほど、必要とするサーバーが少なくなります。
- クライアントの数。ドメイン内のクライアントの数が多くなるほど、必要とする複製サーバーの数も多くなります。ドメイン内のクライアントの数は 1000 以下になるようにしてみてください。 NIS+ クライアントは 、 NIS クライアントよりもサーバー上での負荷が大きくなります。非常に多くのクライアントがほんのわずかのサーバーからサービスを提供されると、ネットワークのパフォーマンスに影響を与えることになります。

  次の表は、一連のサーバーが応答時間を遅らせることなく処理できるビジークライアントのピーク数を示しています。この結果を作成したベンチマークテストでは、クライアントは、 NIS+ サービスを集中的に利用するように設計されています。各クライアントは、通常のドメインが経験する平均的な負荷ではなく、ピーク負荷をシミュレートするため、多くの NIS+ コールを行いました。したがって、次の表に示した数字は応答時間を長くしないでピーク負荷 (平均負荷ではなく) に適合するように設定された構成を示しています。

表 26–1 サーバーの構成と NIS+ クライアントの数

サーバーと複製の構成	ビジークライアントのピーク数
Master: SS5-110	120
Master: SS5-110 Replica: SS10-40	220
Master: SS5-110 Replica: SS10-40 Replica: SS20-50	580

表 **26–1** サーバーの構成と NIS+ クライアントの数　(続き)

サーバーと複製の構成	ビジークライアントのピーク数
Master: Ultra-167	420
Master: Ultra-167 Replica: SS10-40	840

表の数字は、クライアントが NIS+ サービスを広範囲に使用した場合、約 100 から 400 のクライアントごとに余分な複製を追加する必要があることを示しています。複製が SS5 の場合、 100 のクライアントごとに新しい複製を 1 つ追加する必要があり、複製が Ultra の場合 400 のクライアントごとに新しい複製を追加する必要があります。この数字は、必要性に応じて調整します。

各種のマシンを使わないでドメインあたりの複製の数を十分なものにする 1 つの方法は、マルチホームのサーバーを作成することです。マルチホームサーバーとは、複数のイーサネットまたはネットワークインタフェースを持っているマシンをいいます。マルチホームサーバーは、1 つのドメイン内にある複数のサブネットにサービスを提供することができます (マスターあるいは複製サーバーに複数のドメインを設定することもできますが、これはお勧めできません)。

サーバーの速度

サーバーの速度が早いほど、 NIS+ のパフォーマンスは向上します (しかし、その場合 NIS+ サーバーは SMP マルチスレッドハードウェアを有効に利用することはできません)。NIS+ サーバーは、平均的なクライアントと同等かそれ以上に強力にする必要があります。 新しいクライアントのサーバーとして古いマシンを使うことはお勧めできません。

サーバーの速度以外に、その他の多くの要因が NIS+ のパフォーマンスに影響を与えます。ユーザーおよびホストの数と種類、実行しているアプリケーションの種類、ネットワークトポロジ、負荷の密度、その他の要因すべてが NIS+ のパフォーマンスに影響します。したがって、 2 つの異なるネットワークにおいて、同じサーバーハードウェアからまったく同じパフォーマンスを期待することはできません。

下の表のベンチマークは、比較のために掲載しています。各ネットワーク上のパフォーマンスは、この数字とは違うこともあります。下に示したベンチマークの数字は、10000 エントリという標準的なテーブルサイズのテストネットワークに基づいています。

表 **26–2** ハードウェア速度と NIS+ 動作の比較

対象マシン	秒当たりの整合動作数	秒当たりの追加動作数
SS5-110	400	6
SS20-50	440	6

表 **26–2** ハードウェア速度と NIS+ 動作の比較　*(続き)*

対象マシン	秒当たりの整合動作数	秒当たりの追加動作数
PPro-200	760	13
Ultra-167	800	11
Ultra-200	1270	8

サーバーメモリーの容量

サーバーの絶対最低メモリー必要量は 32M バイトですが、中から大規模ドメインのサーバーは少なくとも 64M バイトを装備した方が良いでしょう。

理想的には、 NIS+ サーバーは、 有効な NIS+ テーブルすべての検索可能列のエントリすべてを RAM 内に一度に保存できるほど十分なメモリーが必要です。要するに、最適なサーバーメモリーは、すべてのNIS+ テーブルが必要とするの合計メモリー必要量になります。

分かりやすくするため、下の表は検索可能列が 5 つある `netgroup` テーブルのメモリー必要量を示し、表 26–4 は `passwd` 、`host` および `cred` テーブルのおおよそのメモリー必要量を示しています。

表 **26–3** `netgroups` テーブルに必要なサーバーメモリー

エントリの数	サーバーメモリー使用量 **(M** バイト**)**
6000	4.2
60000	39.1
120000	78.1
180000	117.9
240000	156.7
300000	199.2

表 **26–4** `passwd` テーブルに必要なおおよそのメモリー

エントリの数	サーバーメモリー使用量 **(M** バイト**)**
6000	3.7
60000	31.7
120000	63.2
180000	94.9

表 **26–4** passwd テーブルに必要なおおよそのメモリー　(続き)

エントリの数	サーバーメモリー使用量 (**M** バイト)
240000	125.8
300000	159.0
1000000	526.2

他のテーブルには、検索可能な各列に対してエントリ当たりの平均バイト数に予測エントリ数を掛けると、メモリーサイズを予測することができます。たとえば、エントリが 10,000 で検索可能列が 2 のテーブルがあるとします。最初の列でのエントリ当たりの平均バイト数は 9 で、 2 番目の列でのエントリ当たり平均バイト数は 37 です。したがって、計算結果は、 (10,000 × 9) + (10,000 × 37) = 460,000 になります。

注 – cred テーブルのエントリ数を予測するときは、ユーザーのローカル資格証明書に 1 つ、DES 資格証明書に 1 つずつ、すべてのユーザーが 2 つのエントリを持つことを忘れないでください。各マシンが使用するエントリは 1 つだけです。

標準的な NIS+ テーブルのそれぞれにある検索可能列の数については、623 ページの「NIS+ 標準テーブル」を参照してください。

サーバーのディスク容量

必要なディスク容量は、次の 4 つの要因によって決まります。

- Solaris オペレーティング環境で使用されるディスク容量
- /var/nis (および /var/yp 互換モードで使用する場合) のディスク容量
- メモリーの容量
- NIS+ サーバー処理に必要なスワップ空間

Solaris オペレーティング環境に必要なディスク容量は、インストール方法によっては、220M バイトを超えることがあります。正確な数字については、Solaris のインストールガイドを参照してください。また、サーバーが使用する他のソフトウェアの使用するディスク容量も計算に入れる必要があります。NIS+ ソフトウェア自体は、Solaris オペレーティング環境配布の一部であるため、余分なディスク容量を使用しません。

NIS+ のディレクトリ、グループ、テーブル、クライアント情報は、/var/nis に格納されています。この /var/nis ディレクトリは、1 つのクライアントごとにおよそ 5K バイトのディスク容量を使用します。たとえば、名前空間に 1000 のクライアントがあると、/var/nis には、およそ 5M バイトのディスク容量が必要になります。ただし、同じく /var/nis に格納されるトランザクションのログが大量になる場合があるため、クライアントごとにディスク容量を追加する必要があるかもしれません。この場合は 10〜15M バイトの容量を追加するようお勧めします。つまり、1000 のクラ

イアントがあるときは、15〜20M バイトを /var/nis に割り当ててください。トランザクションのログに対して定期的にチェックポイントを実行する場合、この数字を減らすことができます。/var/nis には独立したパーティションを設けることをお勧めします。パーティションが独立していることにより、オペレーティングシステムのアップグレードを行う際、その作業が容易になります。

NIS+ を NIS と並行して使用するときは、/var/yp に対して、/var/nis に割り当てている量と同じ容量を割り当てて、NIS から転送する NIS マップを格納してください。

さらに、サーバーの通常のスワップ空間の所要量に加えて、rpc.nisd のサイズの 2 倍のスワップ空間も必要になります。システム上で rpc.nisd が使用しているメモリーの量を確認するには、 nisstat コマンドを実行します。詳細は、 rpc.nisd マニュアルページを参照してください。この空間のほとんどは、コールバック操作中や、 nisping-C によってディレクトリに対しチェックポイントを実行するか、複製サーバーが作成されるときに使用されます。これは、このような手続き中には、NIS+ サーバープロセス全体がフォークされるためです。使用するスワップ空間が、64M バイト未満になることはありません。

テーブルの構成を決める

NIS+ テーブルには、単純なテキストファイルやマップにはない、いくつかの機能があります。これらのテーブルは、列エントリ構造を持ち、検索パスを受け付けます。また、これらのテーブルをリンクして、いくつかの異なる方法で構成することもできます。さらに、独自のカスタム NIS+ テーブルを作成することもできます。各自のドメイン用にテーブル構成を選択するときは、以下の節の内容を検討してください。

NIS+ テーブルと NIS マップとの違い

NIS+ テーブルは、様々な点で NIS マップと異なりますが、次の 2 つの相違点は、名前空間を設計する場合に念頭においておく必要があります。

- NIS+ が使用する標準テーブルの数は NIS よりも少ない
- NIS+ テーブルは、SunOS 4.x での NIS マップとは異なる方法で、/etc 内のファイルと相互運用される

NIS+ 標準テーブル

17 の標準 NIS+ テーブルを参照して、各サイトの必要に応じたものかどうかを確認してください。これらのテーブルは、609 ページの「名前空間の構造を設計する」 に示してあります。表 26–6 は、 NIS マップと NIS+ テーブルの対応を示しています。

関連するテーブルの同期化については心配する必要はありません。NIS+ テーブルには、基本的に NIS マップと同じ情報が格納されます。ただし、NIS+ テーブルでは、類似の情報が 1 つのテーブルに統合されます (たとえば、NIS+ の hosts テーブルには、NIS マップの `hosts.byaddr` と `hosts.byname` と同じ情報が格納されます)。NIS+ テーブルでは、NIS マップで使用されていた「キー - 値」ペアの代わりに、列と行が使用されます。表 26–6 を参照してください。「キー - 値」テーブルは、2 つの列で構成されます。1 列目がキー、2 列目が値です。したがって、ホスト情報などの情報を変更するときは、その情報を、`hosts` テーブルなど 1 か所で変更するだけですみます。関連するマップ全体の情報の整合性の維持について注意する必要はなくなりました。

オートマウンタテーブルの新しい名前は次のとおりです。

- `auto_home` (旧名 : `auto.home`)
- `auto_master` (旧名 : `auto.master`)

NIS+ では、ドットを使ってディレクトリを区切るため、ドットは下線に変更されました。テーブル名にドットを使用すると、NIS+ は名前の変換を誤ります。同じ理由で、マシン名にドットを使用することはできません。ドットを含むマシン名は、かならず他の名前に変更してください。たとえば、マシン名に `sales.alpha` を使用できません。 `sales_alpha`、`salesalpha` などのドットを含まない任意の名前に変更してください。

NIS から NIS+ への移行を行うには、NIS 自動マウンタマップのドットを下線に変更する必要があります。また、クライアントのオートマウンタ構成ファイルでも、同じ処理が必要です。次の表を参照してください。

表 **26–5** NIS+ テーブル

NIS+ テーブル	テーブル内の情報
`hosts`	ドメイン内にあるすべてのマシンのネットワークアドレスとホスト名
`bootparams`	ドメイン内にあるすべてのディスクレスクライアントのルート、スワップ、ダンプの各パーティションの位置
`passwd`	ドメイン内のすべてのユーザーに関するパスワード情報
`cred`	ドメインに属する主体の資格
`group`	ドメイン内のすべての UNIX ® グループのグループパスワード、グループ ID、メンバー
`netgroup`	ドメイン内のマシンやユーザーが所属できるネットグループ
`mail_aliases`	ドメイン内のユーザーの mail 別名に関する情報
`timezone`	ドメインの時間帯
`networks`	ドメイン内のネットワークとその標準的な名前

表 **26–5** NIS+ テーブル　(続き)

NIS+ テーブル	テーブル内の情報
netmasks	ドメイン内のネットワークとそれに関連するネットマスク
ethers	ドメイン内にあるすべてのマシンの Ethernet アドレス
services	ドメインで使用される IP サービスの名前とそのポート番号
protocols	ドメインで使用される IP プロトコルのリスト
rpc	ドメインで使用できる RPC サービスの RPC プログラム番号
auto_home	ドメイン内のすべてのユーザーホームディレクトリの位置
auto_master	オートマウンタマップ情報
sendmailvars	mail ドメインを格納

表 **26–6** NIS マップと NIS+ テーブルの対応表

NIS マップ	**NIS+** テーブル	注
auto.home	auto_home	
auto.master	auto_master	
bootparams	bootparams	
ethers.byaddr	ethers	
ethers.byname	ethers	
group.bygid	group	NIS+ グループとは異なる
group.byname	group	NIS+ グループとは異なる
hosts.byaddr	hosts	
hosts.byname	hosts	
mail.aliases	mail_aliases	
mail.byaddr	mail_aliases	
netgroup	netgroup	
netgroup.byhost	netgroup	
netgroup.byuser	netgroup	
netid.byname	cred	
netmasks.byaddr	netmasks	
networks.byaddr	networks	
networks.byname	networks	

表 **26–6** NIS マップと NIS+ テーブルの対応表　(続き)

NIS マップ	**NIS+** テーブル	注
passwd.byname	passwd	
passwd.byuid	passwd	
protocols.byname	protocols	
protocols.bynumber	protocols	
publickey.byname	cred	
rpc.bynumber	rpc	
services.byname	services	
ypservers		必要なし

NIS+ には、NIS テーブルと対応しない sendmailvars という新しいテーブルが 1 つあります。 この sendmailvars テーブルには、sendmail で使用される mail ドメインが格納されます。

NIS+ テーブルは、NIS とは異なる方法で /etc 内のファイルと相互運用される

NIS および 他のネットワーク情報サービスが SunOS 4.x 環境の /etc 内のファイルとの間で行う相互運用は、+/- 構文を使用して /etc 内のファイルによって管理されていました。NIS+、NIS、DNS、および他のネットワーク情報サービスと、Solaris オペレーティング環境の /etc ファイルとを相互運用する方法は、ネームサービススイッチによって決まります。ネームサービススイッチは、/etc/nsswitch.conf という名前の構成ファイルとして、各 Solaris オペレーティング環境クライアントに配置されます。すべての Solaris オペレーティング環境クライアントにある構成ファイルの nsswitch.conf は、そのクライアントの情報源を指定します。 これは、 /etc 内のファイル、DNS ゾーンファイル (ホストだけ)、NIS マップ、または NIS+ テーブルなどです。NIS+ クライアントの nsswitch.conf 構成ファイルを簡単に示すと、次の例のようになります。

例 **26–1** 簡略化されたネームサービススイッチファイルの例

```
passwd: files

group: compat

group_compat: nisplus

hosts: nisplus dns [NOTFOUND=return] files

services: nisplus [NOTFOUND=return] files

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

例 **26–1** 簡略化されたネームサービススイッチファイルの例　(続き)

```
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
```

つまり、ほとんどのタイプの情報で、情報源はまず NIS+ テーブルであり、次に `/etc` 内のファイルということになります。`passwd` および `group` エントリの場合、ネットワーク情報のソースは、ネットワークファイルか、または `/etc` 内のファイルおよび `/etc` ファイルの +/- エントリによって表された NIS+ テーブル のいずれかとなります。

3 種類のスイッチ構成ファイルから選択するか、または独自のスイッチ構成ファイルを作成することができます。詳細は第 1 章を参照してください。

カスタム NIS+ テーブルの使用

どの標準以外の NIS マップを使用するか、またその使用目的を決定してください。NIS+ に変換できるか、あるいは NIS+ 標準マップと置き換えられるかを検討します。

アプリケーションの中には、NIS マップに依存するものがあります。これらのアプリケーションが、NIS+ でも同様に機能するか、また混合環境で正しく機能できるかを検討します。

NIS+ でカスタムテーブルを作成するには、`nistbladm` を使用します。テーブル名にはドットを使用できないことを忘れないでください。

NIS+ を使用して、独自の NIS マップをサポートできるようにしたい場合は、2 つの列を使用するキー値テーブルを作成する必要があります。最初の列はキー、2 番目の列は値を示します。このテーブルを作成して、NIS+ サーバーを NIS 互換モードで実行すると、NIS クライアントは機能の変更に気がつきません。

テーブル間の接続

NIS+ テーブルには、そのホームドメインの資源とサービスに関する情報だけが含まれています。したがってクライアントは、別のドメインに格納された情報を検索する際には、そのドメインの名前を指定しなければなりません。この「転送」を自動化するには、ローカルテーブルをリモートテーブルに接続してください。NIS+ テーブルは、次の 2 つの方法で接続することができます。

- パスを使用する方法
- リンクを使用する方法

NIS+ 名前空間で NIS クライアントを使用するときは、パスとリンクを使用してはいけません。NIS クライアントは、パスまたはリンクでは、正しい情報を検索することができません。

パス

他のドメインのクライアントが、特定の NIS+ テーブル内の情報を頻繁に要求する場合は、そのローカル NIS+ テーブルから他のドメインのテーブルへのパスを設定することを検討してみてください。次の図を参照してください。

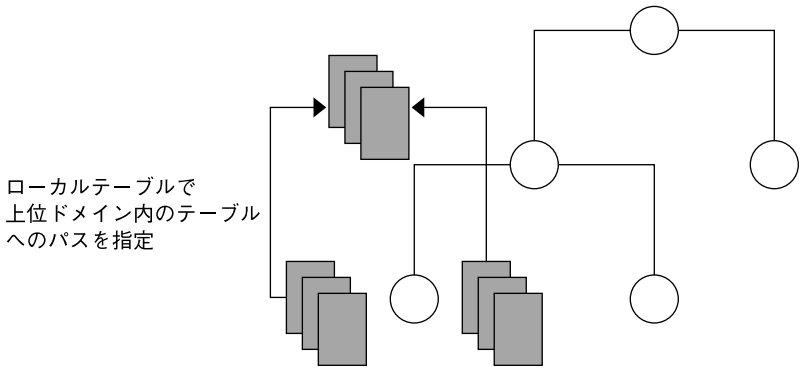


図 **26–8** 上位ドメイン内のテーブルへのパスを設定する

このようなパスがもたらす主な利点は 2 つあります。まず、下位ドメインのクライアントが、別のテーブルを明示的に検索しなくてもすみます。さらに、上位ドメインの管理者があるテーブルで変更を行い、その変更を他のドメインのクライアントに見えるようにすることができます。ただし、このようなパスを設定すると、パフォーマンスが低下します。特に検索がうまくいかないと、パフォーマンスに影響が出ます。これは、NIS+ サービスで、1 つのテーブルではなく 2 つのテーブルを検索しなければならないためです。パスを使用すると、テーブル検索も、他のドメインの利用状況に依存することになります。この依存によって、ドメインの実質の利用度が低下する可能性があります。このような理由から、パスは、他に問題を解決する手段がない場合に限って使用するようにしてください。

mailhost は、別名として使用されることが多いため、特定の mailhost に関する情報を検索する必要がある時は、検索パスに完全指定名 (たとえば、`mailhost.sales.com.` など) を使用する必要があることに注意してください。 そうしないと、 NIS+ は、検索したすべてのドメインで見つかった mailhost をすべて返します。

パスをローカルテーブルに設定するには、`nistbladm` コマンドに `-p` オプションを付けて使用します。テーブルのパスを変更するには、テーブルオブジェクトへの変更アクセス権がなければなりません。テーブルの検索パスを調べるには、`niscat -o` コマンドを使用してください (テーブルへの読み取りアクセス権が必要です)。

リンク

テーブル間にリンクを設定すると、パスと同様の効果が生じますが、リンクでは 1 つのテーブル、つまりリモートテーブルの検索だけが行われる点が異なります。検索パスでは、 NIS+ はまずローカルテーブルを検索し、うまくいかなかった場合にのみリモートテーブルを検索します。リンクでは、検索は、リモートテーブルに対して直接行われます。実際には、リモートテーブルがローカルテーブルと置き換わります。リンクを設定すると、下位ドメインが、独自のテーブルを管理しなくても、上位ドメインの情報を使用することができます。

リンクを作成するには、`nisln` コマンドを使用してください。また、テーブルオブジェクトに対する変更権が必要です。

パスを使用するか、またはドメイン内の NIS+ テーブルをリンクするかを決定するのは、容易ではありません。この決定を行う際の基本的な方針をいくつか、次に示しておきます。

- すべてのドメインに、すべての標準テーブルへのアクセス権がなければなりません。
- 内容の更新が多く、またアクセス頻度が高いデータは、階層の下位に位置していなければなりません。このようなデータは、最も使用頻度の高い場所の近くに置くようにしてください。
- いくつかのドメインが使用するデータは、階層内の上位に位置していなければなりません。ただし、それらのドメインを独立した状態にしておく必要がある場合は除きます。
- データの格納場所が階層の下位であればあるほど、自律的な管理が容易になります。
- NIS+ クライアントだけが、パスおよびリンクで接続されたテーブルを見ることができます。これらのテーブルは、次の図に示すように、NIS クライアントからは参照できません。

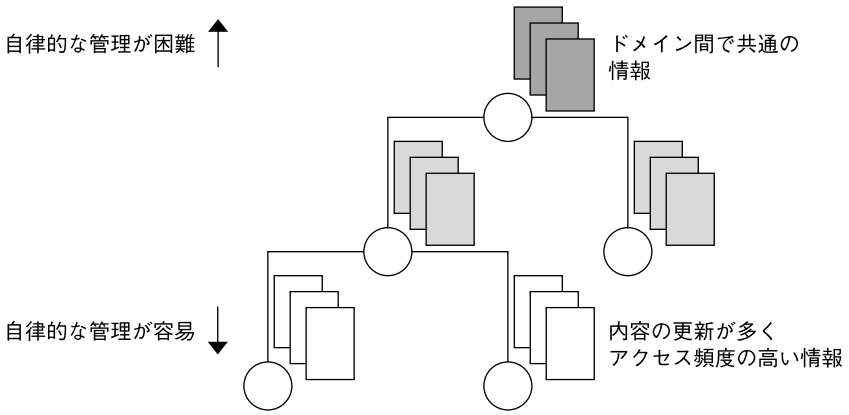


図 **26–9** NIS+ 階層での情報の配布

ユーザー名とホスト名の重複の解決

NIS+ は、要求が実行された場合に、その主体が人間なのかマシンなのかを区別できません。したがって、すべてのユーザー名は、同一の名前空間におけるマシン名と違うものでなければなりません。すなわち、一定の名前空間においては、ユーザーがマシン名と同じユーザー名を持つことができず、またユーザー ID と同じマシン名を付けることもできません。

たとえば、NIS 環境ではローカルマシンの名前が `irina` の場合も、ユーザーは `irina` というログイン名を使用できます。このユーザーのネットワークアドレスは、`irina@irina` となります。これは、NIS+ 環境では成立しません。サイトを NIS + に変換する場合、ユーザーがログイン名を変更するか、あるいはユーザーのマシン名を変更する必要があります。同一のユーザー名とマシン名が存在する場合、その名前を持つマシンが同じ名前のユーザーに属していない場合でも問題となります。次に示す例は、NIS+ では無効となる重複した名前の例です。

- 同一名前空間における `jane@jane`
- 同一名前空間における `patna@peshawar` と `rani@patna`

この問題の一番の解決方法は、`/etc` 内のファイルすべてと NIS マップ をチェックしてから、 NIS+ テーブルを生成することです。重複している名前を見つけた場合は、ログイン名ではなくマシン名を変更し、後でマシンの元の名前の別名を作成してください。

NIS+ セキュリティの影響について理解する

NIS+ には、NIS にはなかったセキュリティが備わっているため、さらに多くの管理作業が必要になります。chkey、keylogin、または keylogout の手順の実行に慣れていないユーザーにも、より多くの作業が要求されることがあります。さらに、NIS+ によって提供される保護は、完璧に安全というものではありません。十分な計算能力と知識があれば、Diffie-Hellman 公開鍵暗号システムを破ることができます。

192 ビットを超える Diffie-Hellman 鍵を使用すると、NIS+ セキュリティが大幅に向上します。ただし、鍵が長くなるにつれて認証に必要な時間が長くなるため、パフォーマンスが低下する可能性があります。

注 – nisauthconf を使用して、この種類の Diffie-Hellman 鍵を設定します。長い鍵の使用については、nisauthconf(1M) を参照してください。

また、キーサーバープロセスによって格納された秘密鍵は、資格を持つ、root 以外のユーザーがログアウトしても、そのユーザーが keylogout によってログアウトしないかぎり、自動的に削除されません。セキュリティは、ユーザーが keylogout によってログアウトしたとしても、完全ではありません。これは、セッションキーが、その期限が切れるか、または再初期化されるまで、有効なためです (詳細については keylogout(1) のマニュアルページを参照してください)。ルートキーは、keylogin -r によって作成されて、/etc/.rootkey に格納されますが、これは .rootkey ファイルが明示的に削除されるまで残ります。スーパーユーザーは keylogout を使用することができません。しかしそれでも、NIS+ は、NIS よりもはるかに安全です。

NIS+ セキュリティがユーザーに与える影響

NIS+ セキュリティを使用すると、NIS+ から取得する情報の信頼性が向上し、情報への不正なアクセスを防げるため、ユーザーにとって有益です。ただし、NIS+ セキュリティを使用するには、ユーザーは、セキュリティに関する若干の知識を習得して、2 ～ 3 の管理手順を実行しなければなりません。

NIS+ ではネットワークログインが必要ですが、ユーザーがさらにキーログインを実行する必要はありません。これは、クライアントが正しく設定されていれば、login コマンドにより、そのクライアントのネットワーク鍵が自動的に取得されるためです。クライアントは、そのログインパスワードと SecureRPC パスワードが同じであれば、正しく設定されます。ユーザー root の秘密鍵は、通常、/etc/.rootkey ファ

イルで入手することができます (潜在的なセキュリティの問題として前述しました)。NIS+ ユーザーのパスワードと資格が、passwd コマンドによって変更されると、そのユーザーの資格情報も自動的に変更されます。

- NIS+ マシンのローカルの root パスワードを変更するには、passwd コマンドを実行します。
- root の資格を変更するには、chkey コマンドを実行します。

ただし、ユーザーがそのネットワークパスワードだけでなく、ローカルの /etc/passwd ファイルのパスワードも管理できて、これらのパスワードがネットワークパスワードと異なる場合、ユーザーは login を実行するたびに keylogin を実行しなければなりません。

NIS+ セキュリティがシステム管理者に与える影響

Solaris オペレーティング環境は、認証のために DES 暗号機構を備えているため、セキュリティ保護操作を必要とするシステム管理者は、別に暗号キットを購入する必要がありません。ただし、システム管理者は、ユーザーに、passwd コマンドと passwd -r コマンドを使用する方法と、これらのコマンドをいつ使用するかを指示する必要があります。

また、セキュリティを強化した NIS+ 名前空間の設定は、通常の名前空間の設定よりも複雑です。この複雑さは、名前空間の設定に必要なステップが多いことだけではなく、すべての NIS+ 主体に対するユーザーの資格とマシンの資格を作成して管理しなければならないということに原因があります。管理者は、passwd テーブルとhosts テーブルから不要なアカウント情報を削除するのと同様に、不要な資格を削除する必要があります。また、管理者は、サーバーの公開鍵が変更された場合、nisupdkeys を使用して、名前空間全体の鍵も変更しなければなりません。さらに管理者は、他のドメインからこのドメインへのリモートログインを望んだり、NIS+ への認証されたアクセスを望むユーザーに対して、LOCAL 資格を追加しなければなりません。

NIS+ セキュリティが移行の計画に与える影響

NIS+ セキュリティの利点と管理上の要件をよく理解したら、NIS+ セキュリティを、移行中または移行後のどちらで実装するかを決める必要があります。NIS 互換モードで、ドメイン内のサーバーの一部またはすべてを操作している場合でも、完全な NIS+ セキュリティを使用するようお勧めします (ドメイン内のすべてのサーバーが同じ NIS 互換モードを使用しているのが、望ましい状態です)。ただし、これには管理の手間が非常にかかります。簡単な方法としては、NIS 互換セキュリティによって NIS+ サーバーと名前空間を設定し、NIS+ クライアントの資格は作成しないことです。ただし、管理者とサーバーには、やはり資格が必要です。NIS+ クライアントは、NIS クライアントとともに、未認証カテゴリに割り当てられます。これにより、学習と設定の作業は軽減されますが、次のような欠点があります。

- ユーザーは、NIS+ テーブルを更新できなくなります。ただし、ログインパスワードの変更は可能です (ただし Solaris 2.5 以降のリリースの場合だけ)。

- ユーザーは、ネームサービス情報が、認証された NIS+ サーバーのものかどうかを確認できなくなります。

資格を選択する

NIS+ には、LOCAL と DES という 2 つの種類の資格があります。

注 – このマニュアルでは、「DES 資格」という用語は、拡張 640 ビット Diffie-Hellman 鍵と、オリジナルの 192 ビット Diffie-Hellman (デフォルト) 鍵の長さについて使用します。`cred` テーブルでは、拡張鍵は `DES` キーワードではなく、`DH640-0` のような着信先を使用します。長い鍵の使用については、`nisauthconf`(1M) を参照してください。

どの NIS+ 主体も、これらの資格のうち少なくとも 1 つを必要とします。名前空間がセキュリティレベル 2 (デフォルト) で管理されているときは、すべての NIS+ 主体 (クライアント) は、そのホームドメインに、DES 資格がなければなりません。また、すべてのユーザー (マシンではなく) は、そのホームドメインと、ログインアクセスが必要な他のすべてのドメインに、LOCAL 資格がなければなりません。

名前空間の資格の必要を調べるには、次のことを検討してください。

- 主体の種類
- 資格の種類

NIS+ の主体になれるのは、クライアントマシン上のユーザーかスーパーユーザーです。図 26–10 を参照してください。

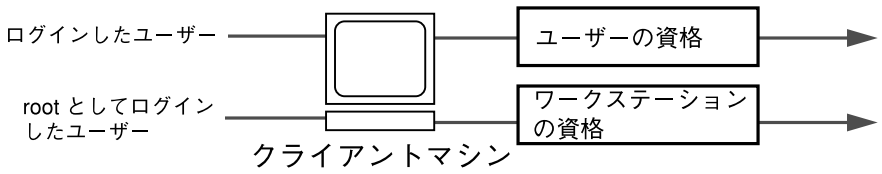


図 **26–10** NIS+ の主体について

作成する資格を決定したら、その資格を必要とする主体の種類を確認します。たとえば、NIS+ クライアントを `nisclient` によって設定する場合、マシンとユーザーの両方に資格を作成することになります。ユーザーの資格を作成しないと、そのユーザーには、未認証クラスに許可されたアクセス権しか与えられません。意図してそのように名前空間を設定している場合は、この設定は十分正しく機能します。しかし、未認証クラスにアクセス権を何も与えていないと、ユーザーはその名前空間を使用することができません。

セキュリティレベルを選択する

NIS+ はセキュリティレベル 2 で実行されるように設計されており、このレベルがデフォルトです。セキュリティレベルの 0 および 1 は、テストとデバッグの目的にのみ用意されています。実際にユーザーが存在しているネットワーク (operational network) は、レベル 2 以外で運用しないでください。詳細については、第 11 章を参照してください。

パスワード有効期限の基準、原則、および規則を確立する

パスワードの有効期限は、ユーザーに対し定期的にパスワードを変更させる仕組みです。パスワードの有効期限については、次の設定が可能です。

- 次にパスワードを変更するまでに使用可能な日数 (最大値) を指定
- パスワードが変更 (設定) されてから、次の変更が可能になるまでの日数 (最小値) を指定
- パスワードが有効期限に達する前の一定の日数、警告メッセージを表示するよう指定
- ログインとログインの間隔の最大値 (日数) を指定する。指定した日数の間、このアカウントにユーザーがログインしなかった場合、ユーザーのパスワードはロックされます。

さまざまな最大日数や期間に到達していても、それまでにすでにログインしているユーザーは、上記の機能には影響されないことを覚えておいてください。これらのユーザーは、通常どおりに作業を継続できます。パスワード有効期限に関する制限と動作は、ユーザーがログインした場合、または次の操作のうちの 1 つを実行した場合のみ実行されます。

- `login`
- `rlogin`
- `telnet`
- `ftp`

パスワード有効期限のパラメータは、ユーザー単位を基本として適用されます。また、ユーザーごとに異なるパスワード有効期限を設定できます。さらに、個別に有効期限を設定したユーザー以外のすべてのユーザーに適用される、一般的なデフォルトのパスワード有効期限パラメータを設定できます。

NIS+ の名前空間を計画する場合、実装したいパスワード有効期限の機能と指定したいデフォルト値を決定してください。パスワードの有効期限の詳細については、第 16 章 を参照してください。

NIS+ グループの計画

NIS+ は、NIS にはない新しい種類のグループをネームサービス管理に導入します。NIS+ グループは、NIS+ アクセス権を一度にいくつかの NIS+ 主体に与える手段としてだけ使用します。またこれは、NIS+ の承認にだけ使用します。

NIS+ グループは、アクセス権が基準を置いている、4 つの承認クラスのうちの 1 つです。4 つの承認クラスは次のとおりです。

- 「所有者」。すべての NIS+ オブジェクトには、単一ユーザーの所有者が 1 人ずつ割り当てられます。所有者は通常、オブジェクトの作成者に割り当てられます。ただし、所有権は他のユーザーに譲渡できます。
- 「グループ」。ユーザーの集合に所定の NIS+ のアクセス権を与える目的で付けられたグループ名の元に集められたユーザーの集合です。
- 「その他」。「認証された」ユーザー。つまり、有効な DES 資格を持つユーザー。オブジェクトの所有者およびオブジェクトグループのメンバーは、資格が有効な間は、その他クラスのメンバーでもあります。
- 「未認証」。有効な DES 資格を持たないユーザー。他のクラスのメンバーの資格が、無効である、失われている、破損している、または見つからないのいずれかの場合、そのユーザーは未認証クラスに分類されます。

NIS+ スクリプトにより、アクセス権を与える目的で作成されたデフォルトのグループ名は、*admin* グループです。また、別の名前を付けて別のグループを作成することも、別のグループに別の NIS+ オブジェクトを割り当てることもできます。

あるオブジェクトグループのメンバーユーザーには、そのオブジェクトに特定の変更を行うアクセス権のような特権があります。たとえば、admin グループに何人かの見習い管理者を追加し、`passwd` テーブルと `hosts` テーブルだけを変更できるが、他のテーブルは変更できないようにすることができます。admin グループを使用すると、管理作業を、階層全体のスーパーユーザーだけに行わせるのではなく、多数のユーザーに分散させることができます。NIS+ admin グループには、NIS 互換モードでドメインを管理している場合でも、そのメンバーのために作成された資格がなければなりません。これは、認証を受けたユーザーだけが、NIS+ テーブルを変更するアクセス権を持つためです。

必要な資格の種類を確認したら、名前空間に必要なアクセス権を選択する必要があります。この作業を容易にするには、まずいくつの管理用のグループが必要かを決めます。複数のグループに異なる権利を割り当てたい場合は、独立したグループを使用すると便利です。通常、グループはドメイン別に作成します。各ドメインには、admin グループが 1 つだけなければなりません。

NIS+ グループとディレクトリへのアクセス権の計画

主体をグループに配置したあと、名前空間のオブジェクトによって、他のカテゴリの主体 (未認証、所有者、グループ、その他) だけでなく、これらのグループに許可されるアクセス権の種類を決めます。これらの割り当てを事前に決めておくと、一貫性のあるセキュリティの方針を確立するうえで役立ちます。

表 26–7 に示すように、NIS+ では名前空間の各オブジェクトに対してデフォルトのアクセス権を提供します。

表 **26–7** NIS+ オブジェクトへのデフォルトのアクセス権

オブジェクト	未認証	所有者	グループ	その他
ルートディレクトリオブジェクト	`r---`	`rmcd`	`rmcd`	`r---`
ルート以外のディレクトリオブジェクト	`r---`	`rmcd`	`rmcd`	`r---`
groups_dir ディレクトリオブジェクト	`r---`	`rmcd`	`rmcd`	`r---`
org_dir ディレクトリオブジェクト	`r---`	`rmcd`	`rmcd`	`r---`
NIS+ グループ	`----`	`rmcd`	`r---`	`r---`
NIS+ テーブル	*varies*	*varies*	*varies*	*varies*

デフォルトのアクセス権を使用するか、または独自のアクセス権を割り当てることができます。独自のアクセス権を割り当てるときは、名前空間内のオブジェクトがどのようにアクセスされるかをよく考える必要があります。未認証クラスは、NIS+ クライアントからのすべての要求から構成されており、その要求は認証されていてもいなくても構わない、ということに注意してください。その他のクラスは、NIS+ クライアントからの認証を受けているすべての要求から構成されます。したがって、認証されていない要求に対し、名前空間へのアクセス権を与えたくなければ、未認証クラスへのアクセス権を割り当てず、その他のクラスだけを割り当てます。一方で、いくつかのクライアントが、たとえばアプリケーションを通して、認証されていない読み取り要求を出すことが予測されるときは、未認証クラスに読み取り権を割り当てる必要があります。NIS クライアントを NIS 互換モードでサポートしたい場合は、未認証クラスに読み取り権を割り当てなければなりません。

また、各種の名前空間オブジェクトが、最初に指定した NIS+ グループに割り当てる権利についても検討してください。名前空間の管理方法によって、利用できるアクセス権の一部またはすべてを、このグループに割り当てることができます。マスターサーバー上のユーザー root を、admin グループの所有者にすることをお勧めします。admin グループには、ルートドメインのオブジェクトに対する作成権と削除権が

必要です。1 人の管理者にだけ、ルートドメインを作成、変更させたい場合は、その管理者だけを admin グループに所属するようにしてください。グループには、いつでもメンバーを追加することができます。設定を行う管理者が何人かいる場合は、その管理者をすべてグループに追加して、そのグループにすべての権利を割り当ててください。その方が、所有者を切り替えたり元に戻したりするよりも簡単です。

オブジェクトの所有者にはすべての権利が与えられていなければなりません。ただし、グループにすべての権利が与えられていれば、このことはさほど重要ではありません。すべての権利を所有者にだけ与えると、名前空間のセキュリティ保護はより安全なものになります。しかし、管理グループにすべての権利を与えた方が管理は容易です。

NIS+ テーブルのアクセス権の計画

NIS+ テーブル以外の NIS+ オブジェクトは、主に構造的なものです。一方、NIS+ テーブルは、オブジェクトの種類が異なり、情報を伝えるものです。 NIS+ テーブルへのアクセスは、すべての NIS+ 主体と、これらの主体に代わって実行されるアプリケーションで必要とされます。このため、NIS+ へのアクセス要件は若干異なります。

表 26–8 は、NIS+ テーブルに割り当てられるデフォルトのアクセス権を示しています。列が、テーブルの権利以外の権利を持つ場合は、それらの権利も示しています。テーブルとエントリのレベルの権利は、`nischmod` コマンドによって変更することができます。また、列レベルの権利は、`nistbladm -u` コマンドによって変更することができます。639 ページの「暗号化されているパスワードフィールドの保護」 には、テーブル権利を変更して異なる要求に応える方法の一例を示してあります。

表 26–8 NIS+ テーブルと列のデフォルトのアクセス権

テーブル / 列	未認証	所有者	グループ	その他
hosts テーブル	`r---`	`rmcd`	`rmcd`	`r---`
bootparams テーブル	`r---`	`rmcd`	`rmcd`	`r---`
passwd テーブル	`----`	`rmcd`	`rmcd`	`r---`
名前 (name) 列	`r---`	`----`	`----`	`----`
パスワード (passwd) 列	`----`	`-m--`	`----`	`----`
ユーザー ID (uid) 列	`r---`	`----`	`----`	`----`
グループ ID (gid) 列	`r---`	`----`	`----`	`----`
GCOS (gcos) 列	`r---`	`-m--`	`----`	`----`

表 **26–8** NIS+ テーブルと列のデフォルトのアクセス権　*(続き)*

テーブル / 列	未認証	所有者	グループ	その他
ホームディレクトリ (home) 列	r---	----	----	----
ログインシェル (shell) 列	r---	----	----	----
シャドー (shadow) 列	----	----	----	----
group テーブル	----	rmcd	rmcd	r---
名前 (name) 列	r---	----	----	----
パスワード (passwd) 列	----	-m--	----	----
グループ ID (gid) 列	r---	----	----	----
メンバー (members) 列	r---	-m--	----	----
cred テーブル	r---	rmcd	rmcd	r---
cname 列	----	----	----	----
auth_type 列	----	----	----	----
auth_name 列	----	----	----	----
public_data 列	----	-m--	----	----
private_data 列	----	-m--	----	----
networks テーブル	r---	rmcd	rmcd	r---
netmasks テーブル	r---	rmcd	rmcd	r---
ethers テーブル	r---	rmcd	rmcd	r---
services テーブル	r---	rmcd	rmcd	r---
protocols テーブル	r---	rmcd	rmcd	r---
rpc テーブル	r---	rmcd	rmcd	r---
auto_home テーブル	r---	rmcd	rmcd	r---
auto_master テーブル		rmcd	rmcd	r---

注 – NIS 互換ドメインは、テーブルオブジェクトレベルの `passwd` テーブルに、未認証クラスの読み取り権を与えます。

暗号化されているパスワードフィールドの保護

表 26–8 を見るとわかるように、passwd テーブルを除くすべてのテーブルで、未認証クラスに読み取り権が与えられています。NIS+ テーブルは、未認証クラスの読み取りアクセス権を与えます。これは、NIS+ テーブルにアクセスする必要がある多くのアプリケーションが、認証されていないクライアントとして実行されるためです。ただし、passwd テーブルに対して同じことを行うと、暗号化されているパスワードの列が、認証されていないクライアントに公開されてしまいます。

表 26–8 に示す構成は、NIS 互換ドメインに対するデフォルトのアクセス権です。NIS 互換ドメインは、passwd 列に、未認証クラスの読み取りアクセス権を与えなければなりません。これは、NIS クライアントが認証されておらず、未認証クラスの読み取りアクセス権を与えないと、その passwd 列にアクセスできないためです。したがって、NIS 互換ドメインでは、パスワードが暗号化されていても、復号化されやすい状態にあります。パスワードを、所有者以外には読めないようにしておくと、より安全になります。

標準の NIS+ ドメイン (NIS 互換ではない) には、さらに別のレベルのセキュリティがあります。 nissetup によって提供されるデフォルトの構成では、列ごとに制御する方法で、passwd 列を未認証ユーザーから保護しますが、passwd テーブルの残りの部分に対するアクセス権は与えられます。テーブルレベルでは、未認証の主体に読み取りアクセス権はありません。列レベルでは、passwd 列を除くすべての列への読み取りアクセス権があります。

エントリ所有者が、パスワード列に対するアクセス権を取得する方法について説明します。エントリ所有者は、各自のエントリに対する読み取り権と変更権の両方を持ちます。エントリ所有者は、その他のクラスのメンバーになることによって、読み取り権を取得します (テーブルレベルでは、その他のクラスに読み取り権があることに注意してください)。また、エントリ所有者は、列レベルでの明示的な割り当てによって、変更権を取得します。

テーブルの所有者とエントリの所有者は、同じ NIS+ 主体であることはほとんどなく、また同じである必要もないことに注意してください。したがって、所有者にテーブルレベルの読み取り権があっても、特定のエントリの所有者に読み取り権があるということではありません。

前に述べたように、これは、Solaris 2.3 以降のリリース (Solaris 9のデフォルト設定です。テーブル、エントリ、および列レベルのセキュリティの詳細については、第 16 章を参照してください。

NIS 互換モードの使用方法の概要

NIS と平行して NIS+ を実行するかどうか、実行する場合にはその方法、および停止する時を決定するのは、おそらくユーザーが直面する最も難しい移行問題の 1 つでしょう。 NIS+ には、 NIS といっしょに使用できる機能がいくつかありますが、中でも NIS 互換モードという機能があります。

NIS 互換モードを使用する計画がある場合、 NIS 互換モードで利用できる基本的な利点を考えておく必要があります。 NIS クライアントにはまったく変更を行う必要はありません。基本的な欠点は、完全な NIS+ セキュリティと階層を利用できないことと、クライアントのドメイン名を変更する必要があることです。

図 26–11 は、NIS だけの名前空間から、NIS と NIS+ の両方の要求に応じる名前空間に変換する方法を示しています。

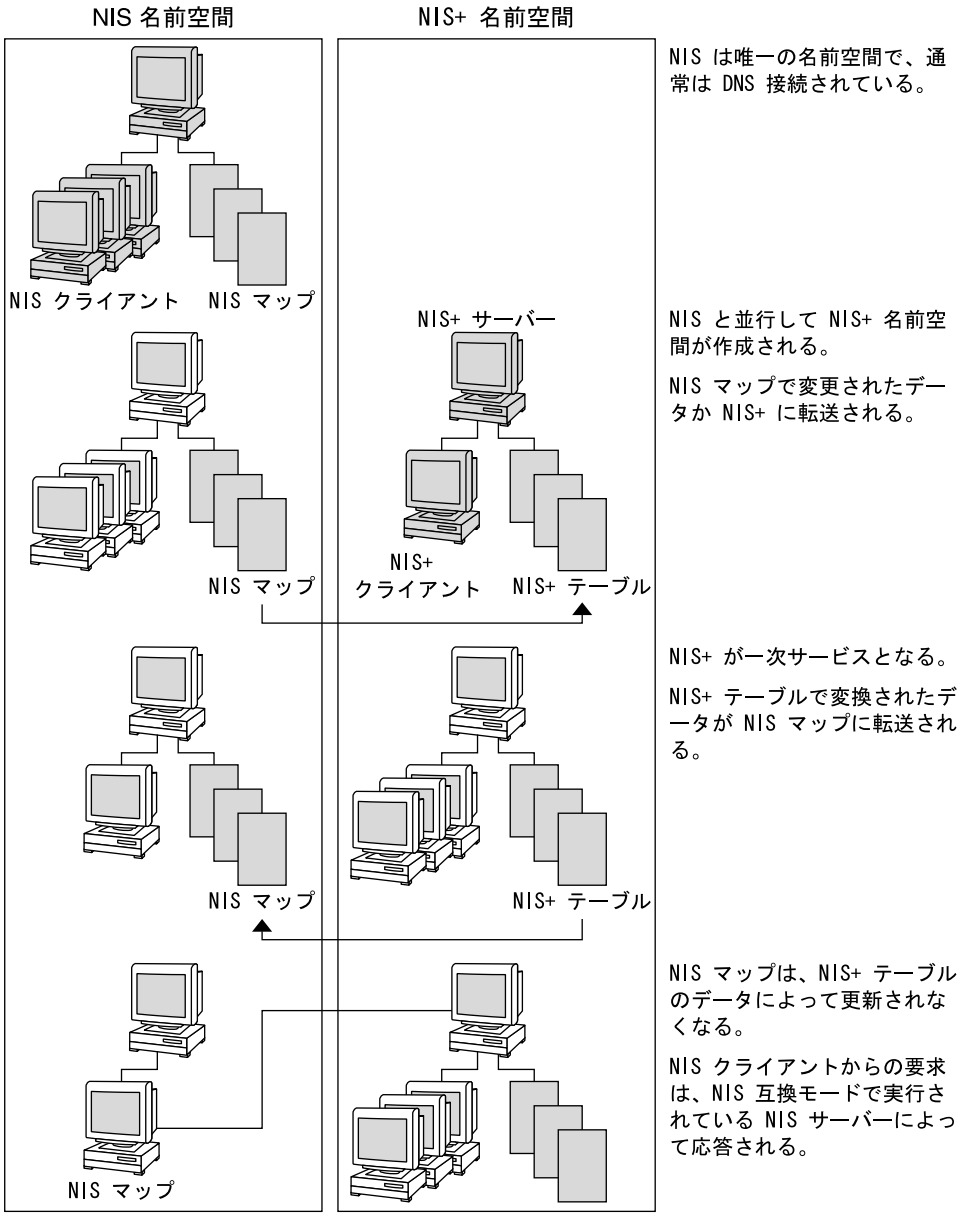


図 **26–11** NIS　互換モードへの移行

NIS 互換になるドメインを選ぶ

NIS クライアントのリストを作成して、それらを最終的な NIS+ ドメインにグループ化してください。NIS 互換モードで管理される NIS+ ドメインの名前が、その NIS クライアントの元の NIS ドメインと異なる場合は、NIS クライアントのドメイン名を、NIS 互換の NIS+ サーバーによってサポートされる NIS+ ドメインの名前に変更しなければなりません。

まず、NIS は間違いなく一次サービスです。情報共有の複雑さに慣れれば、NIS+ を一次サービスに移行する計画を立てることもできます。NIS+ ユーザーは、主な NIS ドメインと新しい NIS+ ドメインを切り替える機能を必要とする場合があります。`nisclient` スクリプトを使用すると、バックアップファイルが作成されるとき、この切り替えを行うことができます。

NIS 互換サーバーの構成を決める

NIS+ サーバーの要件を考慮した上で、各自の NIS サーバーについて判断を行います。最終的に、それらのサーバーを NIS+ サービスに使用する場合は、それらを NIS+ で推奨されるものに変更します。どの NIS サーバーを使用してどの NIS+ ドメインを、またどの機能 (マスターか複製か) でサポートするかを明らかにします。NIS+ サーバーは、それらがサポートするドメインよりも上位のドメインに属することを忘れないでください (ルートドメインサーバーは例外です)。NIS+ サーバーは、そのサービス対象となるドメインに属さないため、ドメインに依存する情報を必要とする他のサービスに、そのマシンを使用することはできません。

可能であれば、NIS+ サーバーマシンは、NIS+ にだけ使用するようにしてください。この構成では、DNS ネームサービス、ブートサーバー、ホームディレクトリ、NFS サーバーなどの他のネットワークサービスを、NIS+ ではないサーバーマシンに転送しなければならない可能性があります。

サイトの多くで、NIS サーバーは、NFS サーバー、計算サーバー、`rlogin` サーバー、メールホストサーバーなど、複数の役割を果たします。NIS サーバーは、そのクライアントと同じ情報を使用してその名前を解決するため、他のサービスも提供することができます。610 ページの「ドメインの階層」で説明したように、ルートドメインを除くすべての NIS+ サーバーが、それがサービスを提供するドメインよりも上位のドメインに存在します。したがって、NIS+ サーバー上ではネームサービスを利用しなければならないサービスを実行しないようにするか、あるいは `nsswitch.conf` のファイルのような他の手段を使用して、これと同じ情報を取得してください。この問題は、階層がない場合には起こりません。この場合、NIS+ ルートサーバーは、そのサービス対象のドメイン内に存在します。NIS+ サーバーの資源の要件は、NIS サーバーの要件よりも大きいため、NIS+ とともに他のサービスを実行しないようにしてください。

Solaris 以外のマシンがネットワーク上にある場合は、NIS 互換モードで NIS+ サーバーを引続き使用することも、このようなマシンをすべて独自のドメインに移動させることもできます。Solaris 以外のマシンをすべて、1 つのサブネットに移動すると、NIS 互換クライアントの場合と同様に、同じサブネットに NIS+ サーバーがなければならないという制約をなくすことができます。これにより、ドメインに必要な複製サーバーの数が減ります。

サービス間で情報を転送する方法を決める

情報を同期させるには、一方の名前空間がもう一方の名前空間に従属する関係になるようにしてください。まず、NIS 名前空間を「主」とします。この場合、NIS マップを変更すると、次にその変更内容を「従」である NIS+ テーブルに反映します。したがって、NIS 名前空間がマスタデータベースになります。

NIS 互換モードの NIS+ サーバーは、標準の NIS マップをサポートします。これらのマップの完全なリストは、`ypfiles`(4) のマニュアルページの注意事項の節にあります。ただし、マップのサポートにはいくつかの制約があります。NIS+ サーバーは、ネットグループマップに対する `ypmatch` 要求には応じますが、逆マップに対する要求には応じません。また、`ypcat` などのネットグループマップの表示要求をサポートしません。`passwd.adjunct` マップもサポートしません。

最終的には NIS+ 名前空間が「主」になります。この場合は、NIS+ テーブルで変更を行って、それを NIS マップにコピーします。

NIS+ コマンドの `nisaddent` と NIS+ スクリプトの `nispopulate` を使用すると、表 26–9 に示すように、NIS マップと NIS+ テーブルの間で、情報を転送することができます。

表 **26–9** `passwd` テーブルでの情報変更のためのコマンド

NIS+ コマンド	説明
`/usr/lib/nis/nisaddent -y`	`ypxfr` を実行して、NIS サーバーからローカルディスクへマップを転送した後で、NIS マップから NIS+ テーブルへ情報を転送する。標準以外の NIS マップは、情報がキーと値のペアになっていれば、NIS+ テーブルに転送できる。複数列のマップは転送されない

表 **26–9** passwd テーブルでの情報変更のためのコマンド　*(続き)*

NIS+ コマンド	説明
/usr/lib/nis/nisaddent -d	NIS+ テーブルからファイルへ情報をコピーする。この情報は、標準 NIS ユーティリティを使って、さらに NIS マップに転送することができる
/usr/lib/nis/nispopulate -Y	NIS マップから NIS+ テーブルへ情報を転送する

Solaris 2.5 より前のリリースの NIS+ のバージョンでは、ユーザーのパスワード情報が /etc 内のファイル、NIS マップ、NIS+ テーブルのどこに格納されているかによって、別々のパスワードコマンド (passwd、yppasswd、nispasswd) を使用して、パスワード関連の処理を行う必要がありました。Solaris 2.5 からは、パスワード関連の処理をすべて passwd または passwd -r nisplus のコマンドで自動的に行うとともに、ユーザーの nsswitch.conf ファイルの passwd エントリによって管理することになりました。

NIS+ または NIS 互換のネットワーク上に、passwd コマンドとパスワードの有効期限を正しく設定するには、各マシン上の nsswitch.conf ファイルの passwd エントリが正しくなければなりません。このエントリによって、passwd コマンドがパスワード情報を取得する場所と更新する場所が決まります。

次の 5 つのパスワードエントリ構成のみが使用できます。

例 **26–2** nsswitch.conf ファイルにおける使用可能な passwd エントリ

```
passwd:files

passwd: files nis

passwd: files nisplus

passwd: compat

passwd_compat: nisplus
```

注意 – 使用しているネットワークのすべてのマシン上に存在する nsswitch.conf ファイルは、必ず上記の passwd 構成のうちの 1 つを使用していなければなりません。別の方法で、passwd エントリを構成した場合、ユーザーがログインできない可能性があります。

NIS 互換モードで作成されたドメインのアクセス権は、NIS+ の場合と少し異なります。テーブルレベルのアクセス権は、その他クラスに読み取りアクセス権を割り当てる必要があります。列レベルのアクセス権は、未認証クラスに読み取りアクセス権を割り当てる必要があります。

DNS 転送を実装する方法を決める

NIS サーバーは、Solaris 1.x の NIS クライアントからの DNS 要求を転送することができます。 NIS 互換モードで実行される NIS+ サーバーにも、DNS 転送機能がありますが、Solaris 2.3 のリリースからしか使用できません (この機能は、パッチ `#101022-06` を使用すれば、Solaris 2.2 でも使用できます)。このため、Solaris 2 または Solaris 9 オペレーティング環境で動作している NIS クライアントは、`/etc/nsswitch.conf` と `/etc/resolv.conf` ファイルをローカルに設定しておく必要があります。

NIS 互換モードで実行される Solaris 2.0 または 2.1 のサーバーによってサポートされている Solaris 1.X NIS クライアントは、DNS 転送を利用することができません。これらのサーバーは、Solaris 2.3 にアップグレードする必要があります。

DNS ドメインの区分を再度作成するときは、新しい DNS ゾーンファイルを定義しなければなりません。ただし、クライアントによっては、`/etc/resolv.conf` ファイルの変更が必要な場合があります。クライアントが DNS クライアントでもある場合は、そのネームサービススイッチ構成ファイルを設定して、NIS+ テーブルだけでなく、DNS ゾーンファイルまたは NIS マップのホスト情報を検索することができます。

NIS+ クライアントの DNS 転送

NIS+ クライアントは、NIS クライアントのような暗黙の DNS 転送機能を持ちません。そのかわり、ネームサービススイッチを使用することができます。NIS+ クライアントに DNS 機能を与えるには、その `hosts` エントリを次のように変更してください。

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

Solaris 2 または Solaris 9 オペレーティング環境の NIS クライアントの DNS 転送

NIS クライアントが、 *NIS* 互換の NIS+ サーバーの DNS 転送機能を使用している場合、そのクライアントの `nsswitch.conf` ファイルは、`hosts` エントリに次の構文があってはいけません。

```
hosts: nis dns files
```

DNS 転送では、ホスト要求が DNS に自動的に転送されるため、この構文があると、NIS+ サーバーとネームサービススイッチの両方が、不必要な要求を DNS サーバーに転送してしまいます。この結果、パフォーマンスが低下します。

Solaris 1、Solaris 2、Solaris 9 における NIS コマンドと NIS+ コマンドの対応

この節で示す表によって、Solaris 1 オペレーティング環境の NIS コマンド、Solaris 2 または Solaris 9 オペレーティング環境の NISコマンド、およびそれに対応する NIS+ コマンドがわかります。

- 表 26–10 は、Solaris 2 および Solaris 9 でサポートされている NIS コマンドを示しています。
- 表 26–11 と表 26–12 は、Solaris 2 および Solaris 9 の NIS クライアントとサーバーのコマンドに対応する NIS+ コマンドを示しています。
- 表 26–13 には、NIS アプリケーションプログラマ用のインタフェース関数と、それに対応する NIS+ API が示されています。詳細については、該当するマニュアルページを参照してください。

Solaris 2 および Solaris 9 でサポートされている NIS コマンド

Solaris 2 および Solaris 9 では、一部の NIS コマンドだけがサポートされています。NIS サーバーコマンドは、Solaris 2 および Solaris 9 では提供されません。NIS クライアントコマンドだけがこのリリースに含まれています。これらの NIS コマンドが実行されるかどうかも、Solaris 2 または Solaris 9 の NIS クライアントが、NIS サーバーまたは NIS 互換モードの NIS+ サーバーにサービスを要求するかどうかによって決まります。NIS クライアントは、NIS 互換モードで実行されている NIS+ サーバーに更新を依頼することができません。たとえば、このようなクライアントは、`chkey`、`newkey` の各コマンドを実行することができません。表 26–10 は、Solaris 2 および Solaris 9 でサポートされている NIS コマンドの一覧です。

表 **26–10** Solaris 2 および Solaris 9 オペレーティング環境でサポートされる NIS コマンド

コマンドの種類	**Solaris 2** および **9** でサポートされる **NIS** コマンド	**Solaris 2** および **9** でサポートされない **NIS** コマンド
ユーティリティ	`ypinit ypxfr ypcat ypmatch yppasswd ypset ypwhich`	`yppush yppoll ypchsh ypchfn ypmake`
デーモン	`ypbind`	`ypserv ypxfrd rpc.ypupdated rpc.yppasswdd`

表 **26–10** Solaris 2 および Solaris 9 オペレーティング環境でサポートされる NIS コマンド　*(続き)*

コマンドの種類	**Solaris 2** および **9** でサポートされる **NIS** コマンド	**Solaris 2** および **9** でサポートされない **NIS** コマンド

NIS API	`yp_get_default_domain()`
	`yp_bind()`
	`yp_unbind()`
	`yp_match()`
	`yp_first yp_next()`
	`yp_all()`
	`yp_master()`
	`yperr_string()`
	`ypprot_err()`
		`yp_order() yp_update()`

クライアントコマンドとサーバーコマンドの対応

この節で示す 2 つの表には、NIS コマンドと、それに相当する NIS+ コマンドを示してあります。これらのコマンドは、2 つのカテゴリに分けられます。表 26–11 では、ネームサービスクライアントからネームサービスサーバーへのコマンドを示しています。 表 26–12 では、ネームサービスサーバーからネームサービスサーバーへのコマンドを示しています。

対応するクライアントコマンド

表 26–11 では、ネームクライアントからネームサーバーへのコマンドを示しています。これらのコマンドを、ネームサービスのクライアントマシン上で入力してネームサービスサーバーの情報を要求します。表の 1 列目のコマンドは、Solaris 1 の NIS サーバーに接続されている、 Solaris 1、Solaris 2、または Solaris 9 の NIS クライアントで実行されます。表の 2 列目のコマンドは、 NIS 互換モードで実行されている Solaris 2 または Solaris 9 の NIS+ サーバーに接続されている、 Solaris 1、Solaris 2、または Solaris 9 の NIS クライアント上で実行されます。 表の 3 列目のコマンドは、Solaris 2 または Solaris 9 の NIS+ サーバーに接続されている、Solaris 2 または Solaris 9 の NIS+ クライアント上でのみ実行されます。1 つの行に示されたコマンドは、ほぼ同じ機能を持ちます。「なし」は、対応するコマンドがないことを示しています。

表 **26–11** NIS クライアントコマンドと対応する NIS+ コマンド

SunOS 4.x NIS サーバー	**NIS** 互換モードの **NIS+** サーバー	**NIS+** サーバー
`ypwhich -m`	`ypwhich -m`	`niscat -o org_dir`
`ypcat`	`ypcat`	`niscat`
`ypwhich`	`ypwhich`	なし
`ypmatch`	`ypmatch`	`nismatch/nisgrep`

表 **26–11** NIS クライアントコマンドと対応する NIS+ コマンド　(続き)

SunOS 4.x NIS サーバー	**NIS** 互換モードの **NIS+** サーバー	**NIS+** サーバー
`yppasswd`	`passwd`	`passwd`
`ypbind`	`ypbind`	なし
`yppoll`	なし	なし
`ypset`	`ypset`	なし
なし	`ypinit -c`	`nisclient -c`

以下の点に注意してください。

- Solaris 2.5 では、`passwd` コマンドは NIS または NIS+ の状態に関係なく使用できます。以前 `nispasswd` および `yppasswd` コマンドによって実行していた機能は、このリリースでは `passwd` コマンドに含まれました。
- `ypinit -c` コマンドは、Solaris 2 または Solaris 9 の NIS クライアント上でのみ使用できます。
- `ypcat` コマンドは、netgroup テーブルに対する照会ではサポートされません。NIS クライアントの要求は、応答が返される前に時間切れになります。これは、このテーブルの形式が、`netgroup` NIS マップの形式と大幅に異なるためです。

対応するサーバーコマンド

表 26–12 では、ネームサーバーからネームサーバーへのコマンドを示しています。NIS サーバーコマンドは、Solaris 2 または Solaris 9 に含まれていないため、NIS + サーバーにも NIS 互換モードの NIS+ サーバーにも使用できません。また、NIS サーバーは、NIS+ サーバーに変更を依頼することができません。この逆もできません。このテーブルの 3 番目の列には、最初の列の NIS サーバーコマンドに対応する NIS+ サーバーコマンドが示されています。NIS 互換モードはクライアントコマンドだけを参照するため、NIS 互換モードのサーバーには同じ機能を提供するコマンドはありません。

表 **26–12** NIS サーバーコマンドと対応する NIS+ コマンド

SunOS 4.x NIS サーバー	**NIS** 互換モードの **NIS+** サーバー	**NIS+** サーバー
`ypxfr`	なし	なし
`makedbm`	なし	`nisaddent`
`ypinit -m ypinit -s`	なし	`nisserver`
`ypserv`	`rpc.nisd -Y`	`rpc.nisd`

表 **26–12** NIS サーバーコマンドと対応する NIS+ コマンド　*(続き)*

SunOS 4.x NIS サーバー	**NIS** 互換モードの **NIS+** サーバー	**NIS+** サーバー
`ypserv -d`	`rpc.nisd -Y -B`	DNS 転送は不要、`/etc/nsswitch.conf` を使用すること
`ypxfrd`	なし	なし
`rpc.ypupdated`	なし	なし
`rpc.yppasswd`	`rpc.nispasswdd`	`rpc.nispasswdd`
`yppush`	なし	`nisping`
`ypmake`	なし	`nissetup`、`nisaddent`
`ypxfr`	なし	なし

NIS と NIS+ の API 関数の対応

サイトを完全に NIS+ に移行するには、ネームサービスを変更するだけでなく、すべてのアプリケーションを NIS+ に移植する必要があります。NIS の呼び出しを行う、サイトの内部で作成されたアプリケーションはすべて、NIS+ の呼び出しを実行するように変更しなければなりません。そうしないと、常に NIS 互換モードで NIS+ サーバーを実行しなければならず、このモードの欠点をすべて抱えることになります。サイトの外部で作成されたアプリケーションでは、必要な変更が行われるまで、NIS 互換モードで名前空間を管理しなければならないことがあります。

表 26–13 では、NIS API 機能と、それに対応する NIS+ API 機能を示しています。

表 **26–13** NIS の API の関数と NIS+ の API の関数の対応

NIS API の関数	**NIS API** の関数
`yp_get_default_domain()`	`nis_local_directory()`
`ypbind()`	なし
`ypunbind()`	なし
`ypmatch()`	`nis_list()`
`yp_first()`	`nis_first_entry()`
`yp_next()`	`nis_next_entry()`
`yp_all()`	`nis_list()`
`yp_master()`	`nis_lookup()`
`yperr_string()`	`nis_perror() nis_sperrno()`

表 **26–13** NIS の API の関数と NIS+ の API の関数の対応　(続き)

NIS API の関数	**NIS API** の関数
`ypprot_err()`	`nis_perror()` `nis_sperrno()`
`yp_order()`	なし
`yp_update()`	`nis_add_entry()`、`nis_remove_entry()`、`nis_modify_entry()`

NIS 互換モードのプロトコルサポート

表 26–14 は、NIS 互換モードで動作する NIS+ サーバー によってサポートされる NIS プロトコルについて示しています。

表 **26–14** NIS+ サーバーによる NIS プロトコルのサポート

NIS プロトコル	互換性についての説明
NIS クライアント V2 プロトコル	サポートしている
NIS サーバー間プロトコル (NIS server-to-server プロトコル)	サポートしていない
NIS クライアント更新プロトコル	`yppasswd`プロトコルをサポートしている
NIS クライアント V1 プロトコル	`YPPROC_NULL`、`YPPROC_DOMAIN`、`YPPROC_DOMAIN_NONACK` を除き、サポートしていない

NIS+ への移行の準備 - 他システムに対する NIS+ の影響を調べる

システム管理者を教育するだけでなく、サイトで NIS+ を紹介し、テストを行い、NIS+ に習熟できるような教育を行なってください。また、NIS+ への移行によって影響を受ける他のシステムやアプリケーションの NIS への依存関係を調べてください。

たとえば、アプリケーションの中には、NIS マップの一部に依存するものがあります。これらのアプリケーションが標準 NIS+ テーブルまたはカスタム NIS+ テーブルのどちらで機能するか、また、それらのアクセスの必要性によりセキュリティ計画全体にどのような影響が及ぶかを調べてください。

さらに、サイトで使用される標準以外の NIS マップはどれか、また、それらのマップを NIS+ テーブルに変換するかあるいは標準以外の NIS+ テーブルを作成して、情報を格納できるかを調べてください。アクセス権をチェックする必要もあります

各サイトで、NIS に依存する、ローカルで作成したアプリケーションを使用するかどうかも調べます。また、 `yp_match()` 関数の呼び出しが埋め込まれているかどうかなど、直接 NIS を呼び出すコマンドやアプリケーションがあるかどうかも調べる必要があります (詳細については、649 ページの「NIS と NIS+ の API 関数の対応」を参照してください)。

名前空間でユーザー名とホスト名が重複していないかチェックしてください (詳細については、630 ページの「ユーザー名とホスト名の重複の解決」を参照してください)。

NIS+ への移行がネットワークのインストール手順に与える影響も調べます。もし影響があれば、必要な変更を分析してください。 NIS+ のサイト管理作業に対する影響を調べると、発生する可能性がある障害を事前に明らかにすることができます。

システム管理者の教育

607 ページの「NIS+ について理解する」 で説明した紹介と教育プログラムのもう 1 つの目的として、各サイトの管理者に NIS+ の概念を理解してもらい、作業手順に慣れる機会を与えるということがあります。教室での学習だけでは不十分です。システム管理者には、業務に支障をきたさない環境で作業する機会が必要です。この教育には、次の内容が含まれます。

- NIS+ の正しい概念と管理を教えるコース
- 基本的な NIS+ の障害追跡情報とその実習
- 各サイトの実装方針と計画に関する情報

ユーザーへの事前の連絡

NIS+ へのクライアントの移行を実際に開始するかなり前から、ユーザーに事前に移行についての連絡を行なってください。実装の計画を伝え、詳細を入手する方法を提供してください。606 ページの「推奨する移行手順」 で説明したように、移行の主な目的の 1 つは、クライアントに対する移行の影響を最小限に抑えるということですが、ユーザーが新しい変更に関心を持つ場合があります。このような場合には、電子メールで通知を送って、情報セミナーの開催を知らせ、ユーザーが質問を送信できる電子メールの別名または個人名を指定してください。

必要な変換ツールとプロセスを明らかにする

移行のツールを作成または入手して、実装に利用してください。各サイトですでに自動化ツールを使用して、個々のシステムやネットワークサービスを管理している場合は、それらを移植して、移行に使用するバージョンの Solaris ソフトウェアや NIS+ で動作するようにしてください (607 ページの「1 種類のソフトウェアリリースを使用する」を参照)。次に、スクリプトを作成する際の推奨事項を示します。

- NIS+ へユーザーを移行させるためのスクリプト。`nisclient` シェルスクリプトに追加を行う
- ユーザーの NIS+ 環境を確認するチェックスクリプト
- バックアップスクリプトと復元スクリプト
- 日常的な NIS+ 管理の `crontab` エントリ
- 障害通知手順

このようなスクリプトを作成すると、ドメイン全体で一貫した移行を行い、移行の効率をあげて、問題を減らすことができます。また、名前空間全体のすべてのクライアントで使用できるような `nsswitch.conf` といった一連の標準構成ファイルとオプションを用意する必要もあります。

移行に使用される管理用のグループを明らかにする

移行の際に調べた管理資源に対応する名前空間の設計 (634 ページの「パスワード有効期限の基準、原則、および規則を確立する」 を参照) の一部として、NIS+ グループが作成されたことを確認してください。NIS+ 名前空間の日常的な操作に使用されるもの以外の NIS+ グループを、移行に使用することもできます。リモートの管理者の援助が至急必要な場合には、グループにこれらの管理者を追加してください。

グループのメンバーに正しい資格があり、名前空間オブジェクトがグループに正しいアクセス権を承認していることを確認してください。また、その適切なグループが、適切な名前空間オブジェクトのグループ所有者として識別されることを確認してください。

次の表は、NIS+ グループとグループアクセス権を操作するコマンドの一覧です。

表 **26–15** グループ用の NIS+ コマンド

コマンド	説明
`nisgrpadm`	グループを作成または削除し、メンバーを追加、変更、一覧表示、削除する
`niscat -o`	NIS+ グループのオブジェクト属性値を表示する
`nissetup`	ドメインのグループが格納されているディレクトリの基本構造を作成する
`nisls`	ディレクトリの内容を一覧表示する
`NIS_GROUP`	シェルで設定されている `nisdefaults` の値を上書きする環境変数
`nischmod`	オブジェクトのアクセス権を変更する
`nischown`	NIS+ オブジェクトの所有者を変更する
`nischgrp`	NIS+ オブジェクトのグループ所有者を変更する
`nistbladm -u`	NIS+ テーブルの列へのアクセス権を変更する
`nisdefaults`	現在の NIS+ のデフォルトを表示、または変更する

ドメインの所有者を決める

ドメイン階層の機能を十分利用するには、ドメインの所有権を、それらのドメインをサポートしている組織に分散させてください。これにより、ルートドメインの管理者が、ローカルレベルの基本的な作業を行わなくてもすむようになります。 誰が何を所有しているのかがわかれば、管理用のグループを作成するための指針を用意して、オブジェクトへのアクセス権を設定することができます。

NIS+ ドメインの所有権と DNS ドメインの所有権をどのように調整するかを検討してください。次のいくつかの指針を示します。

- DNS ドメイン構造の管理には、サイトの最上位レベルの管理用のグループの作業を含みます。
- これと同じ管理用のグループが、最上位のレベルの NIS+ ドメインを所有します。
- 下位レベルの DNS ドメインと NIS+ ドメインの管理責任は、上位レベル管理グループによって、個々のサイトに委任されます。NIS+ ドメインが、DNS ドメインと同じ設計で作成される場合 (たとえば地理的に構成されるなど)、この委任は説明しやすいものになります。

資源の利用度を調べる

実装にどの管理資源が必要かを調べます。これらの資源は、通常の NIS+ の操作に必要な資源をはるかに超えます。移行を行うときに、NIS+ と NIS の互換性が必要な期間が長期に及ぶ場合は、さらに資源が必要になります。

名前空間の設計を実装する作業だけでなく、多くのクライアントを移行して、特殊な要求や問題を処理する作業についても検討してください。このとき、NIS+ の習得にかなり時間がかかることを考慮に入れておいてください。NIS+ でサポート作業を行う場合、NIS で作業していたときとくらべて、しばらくの間、管理者の作業効率がやや低下する場合があります。このため、通常の教育だけでなく、実習経験をともなう上級コースの研修を受けることも検討してください。

さらに、移行が完了した後でも、管理者は、NIS+ をサポートするための毎日の作業に慣れるまでに若干の時間を要します。

ハードウェア資源についても検討してください。NIS サーバーは、経路指定、印刷、ファイル管理などの他のネットワークサービスをサポートするために使用されることがよくあります。NIS+ サーバーにかかる可能性のある負荷を考えて、専用の NIS+ サーバーを使用する必要があります。このように負荷を分散すると、障害追跡と性能監視が簡単になるため、移行が簡略化されます。もちろん、システムを追加するとコストがかかります。必要なサーバーの数と、その構成方法については、609 ページの「NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする」に説明があります。

これらのサーバーは、NIS サーバーの他に必要であることを忘れないでください。NIS サーバーは、移行が完了すると、必要なくなるか、あるいは他の用途で使用される場合がありますが、 NIS+ サーバーは引き続き使用されます。

ログイン名とホスト名の衝突を解決する

NIS+ 認証では、マシンとユーザーが、1 つのドメイン内で同じ名前を使用することはできません。たとえば、joe@joe は使用できません。NIS+ は、ホストの資格とログイン名の資格を区別しないため、1 つの名前に 1 種類の資格を使用するだけですみます。名前空間に重複した名前があり、何らかの理由でその重複したホスト名を維持しなければならないときは、次のように変更してください。つまり、ユーザーログイン名をそのままにして、重複したホスト名を別名に指定します。ホストに新しい名前を作成して、古い名前を新しい名前の別名として使用します。不正な名前の組み合わせの例については、630 ページの「ユーザー名とホスト名の重複の解決」を参照してください。

名前の衝突を解決してから実装を開始する必要がありますが、通常の NIS+ の操作中に新しいマシンとユーザーの名前を常時チェックする計画をたてる必要もあります。これには、nisclient スクリプトを使用してクライアントの資格を作成すると、名前の比較が行われます。

すべての情報源となるファイルを調べる

/etc 内のファイルと NIS マップをすべて調べて、空のフィールドやこわれているデータがないかを確認してから、NIS+ を構成してください。NIS+ テーブルの生成スクリプトとコマンドは、データファイルに空のフィールドや余分な文字があると、正常に実行されない場合があります。空フィールドを埋めるか、またはデータを修正してから、作業を開始してください。NIS+ スクリプトをそのまま実行してデータを破壊する危険をおかすよりも、問題のあるユーザーやマシン名を /etc 内のファイルや NIS マップから削除してから、 NIS+ スクリプトを実行して、NIS+ のインストール後にそれらを戻すことをお勧めします。

ホスト名から「.」を削除する

NIS+ では、マシン名とドメインの区切りや親ドメインとサブドメインとの区切りにドット (ピリオド) を使用するため、ドットを含むマシン名 (ホスト名) を付けることはできません。NIS+ へ移行するにあたって、ホスト名からドットを必ず削除する必要があります。ドットの代わりにハイフン (-) を使用できます。たとえば、sales.alpha というマシン名を付けることはできません。この名前は、sales-alpha に置き換えることができます (使用可能なホスト名の詳細については、 hosts のマニュアルページを参照してください)。

NIS マップ名から「.」を削除する

609 ページの「NIS+ 名前空間の設計 - 管理モデルの目的を明らかにする」で説明したように、NIS+ のオートマウンタテーブルでは、その名前とファイル内容の「.」(ドット) が下線で置き換えられています。この変更は、移行中に使用する NIS マップの名前に対しても行う必要があります。この変更を行わないと、NIS+ は、名前の「.」を、オブジェクト名のドメインレベルを区別するピリオドと混同します。

注 – オートマウンタだけでなく、すべての NIS マップの「.」を必ず下線に変換してください。ただし、標準以外の NIS マップの名前をドットから下線に変更した場合、標準以外のマップを使用するアプリケーションを、NIS+ 構文を認識するように変更しないと、そのアプリケーションに障害が生じるおそれがあります。

既存の NIS 名前空間を文書化する

現在の構成を文書化しておくと、移行を開始する開始点を明確にすることができます。次の項目のリストを作成してください。

- 現在のすべての NIS ドメインとネットワークの名前と位置
- 現在のすべての NIS サーバー (マスタとスレーブの両方) のホスト名と位置
- 現在のすべての NIS サーバーの構成次のものが含まれます。
  - ホスト名
  - CPU の種類
  - メモリーサイズ
  - 使用可能なディスク容量
  - root アクセス権を持つ管理者の名前
- 標準以外の NIS マップ

NIS クライアントのリストと、最終的な NIS+ ドメインを対応させてください。これらは、Solaris オペレーティング環境にアップグレードする必要があります。

NIS サーバーの移行計画を作成する

NIS サーバーについてよく考慮します。移行が完了した後でも NIS サーバーを他の用途に使用することができますが、「両方」のサービスにサーバーを必要とする段階があることを忘れないでください。したがって、既存の NIS サーバーを使って、すべての NIS+ サーバーの必要を満たす計画を立てることはできません。

NIS サーバーの詳しい移行計画を作成して、NIS+ に使用される NIS サーバーと、その移行時期を明らかにしておくと役立ちます。NIS から NIS+ への移行の初期段階では、NIS サーバーを NIS+ サーバーとして使用しないでください。657 ページの「NIS + の実装の概要」で説明するように、名前空間全体に対する操作を調べてからクライアントを NIS+ に移行すると、より安定した実装を行うことができます。

NIS サーバーを NIS+ ドメインに割り当てて、各サーバーの役割 (マスタまたは複製) を明らかにしてください。NIS+ サービスへの移行を計画しているサーバーを指定したら、それらを NIS+ の要件に合わせてアップグレードしてください (621 ページの「サーバーメモリーの容量」を参照)。

NIS+ の実装の概要

以上の節で説明した作業を実行すれば、手のかかる仕事はほとんど終わったことになります。ここでしなければならないことは、設計した名前空間の設定と、クライアントの追加だけです。この章では、これらの処理を行う方法について説明します。これらの手順を実行するにあたっては、移行前の準備作業すべてが完了していて、各サイトのユーザーが移行の計画を了解していることを確認してください。

NIS+ ドメインを DNS ドメインと並行して使用する場合は、各 DNS ドメインに 1 つの NIS+ サブドメインを設定します。最初の DNS ドメインに完全な NIS+ 名前空間を設定して、すべてが正しく動作していることを確認したら、別の NIS+ 名前空間を並行して設定することができます。

第 1 段階 - NIS+ 名前空間を設定する

ドメインを NIS 互換モードで管理している場合でも、完全な DES 認証を備えた名前空間を設定します。NIS+ スクリプトを使用します。NIS+ の構造と概念の詳細については、第 4 章を参照してください。そして、次の手順を行います。

1. ルートドメインを設定します。

   NIS 互換モードでルートドメインを管理する場合は、`nisserver` を使用します(セットアップスクリプトを使用しない場合は、`rpc.nisd` および `nissetup` に `-y` フラグを付けて使用してください)。

2. ルートドメインテーブルを生成します。

   NIS マップまたはテキストファイルから、 `nispopulate` を使用して、情報を転送することができます。もちろん、 `nistbladm` や `nisaddent` を使用して、同時に複数のエントリを作成することもできます。

3. ルートドメインのクライアントを設定します。

   ルートドメインに 2、3 のクライアントを設定して、その操作を正しくテストできるようにします。完全な DES 認証を使用してください。これらのクライアントマシンの中には、後でルートの複製サーバーに変換されたり、ルートドメインをサポートする管理者のマシンとして機能するものがあります。NIS+ サーバーは、個人用のマシン

にはなりません。

4. サイト固有の **NIS+** テーブルを作成、または変換します。

   新しい NIS+ ルートドメインにサイト固有のカスタム NIS+ テーブルが必要な場合は、 `nisaddent` を使用してそれらのテーブルを作成し、 `nistbladm` を使用して、それらのテーブルに NIS データを転送します。

5. ルートドメインのグループに管理者を追加します。

   管理者には、LOCAL 資格と DES 資格が必要です (`nisaddcred` を使用します)。管理者のマシンは、ルートドメインのクライアントでなければなりません。また、管理者の root 識別情報は、DES 資格を持つ NIS+ クライアントでなければなりません。

6. 必要に応じて、`sendmailvars` テーブルを変更します。

   新しいドメイン構造の結果、電子メール環境が変更された場合は、新しいエントリを使って、ルートドメインの `sendmailvars` テーブルを生成します。

7. ルートドメインの複製サーバーを設定します。

   まず、クライアントをサーバーに変換します (NIS 互換のために `rpc.nisd` に `-Y` オプションを使用し、DNS 転送が必要な場合は `-B` も使用します)。次に、`nisserver -R` を使って、それらのサーバーをルートドメインに関連付けます。

   NIS 互換の場合は、 `rpc.nisd` に `-Y` オプションを使用して実行します。そして、`/etc/init.d/rpc` ファイルを編集して、`EMULYP` 行からコメント記号 (#) を削除します。DNS 転送の場合は、`rpc.nisd` に `-B` オプションを使用します。

8. ルートドメインの動作をテストします。

   一連のインストール固有のテストルーチンを作成して、NIS+ に切り替えた後に、クライアントの機能を確認します。これにより、移行処理の効率が向上して、問題が減ります。このドメインをおよそ 1 週間操作してから、他のユーザーを NIS+ に移行してください。

9. 名前空間の残りを設定します。

   これ以上クライアントを NIS+ に移行しないで処理を進め、ルートドメインの下にある他のドメインをすべて設定します。これには、マスターサーバーと複製サーバーの設定も含まれます。新しい各ドメインを、ルートドメインの場合と同様に完全にテストして、構成とスクリプトが正しく機能することを確認します。

10. 名前空間の動作をテストします。

    保守、バックアップ、復元などのすべての操作手順をテストします。名前空間内のすべてのドメイン間での情報共有処理をテストします。NIS+ 全体の操作環境を検査し終わるまでは、第 2 段階に進まないでください。

11. **NIS+** ドメインのセキュリティ構成をカスタマイズします。

    これは、すべてが正しく機能していれば必要ありません。ただし、不正なアクセスから保護したい情報がある場合は、NIS+ テーブルのデフォルトアクセス権を変更して、 NIS クライアントであっても、それらの情報にアクセスできないようにすることができます。また、NIS+ グループのメンバーの関係と NIS+ の構成オブジェクトのアクセス権を再構成して、管理責任を割り当てることもできます。

第 2 段階 - NIS+ 名前空間を他の名前空間に接続する

1. ルートドメインを **DNS** 名前空間に接続します **(**任意**)**。

   NIS+ クライアントは、ネームサービススイッチを使って、インターネットに接続することができます。マシンが DNS クライアントでもある場合、そのネームサービススイッチ構成ファイルを設定して、NIS+ テーブルや NIS マップだけでなく、DNS ゾーンファイル内の情報を検索することもできます。

   各クライアントの `/etc/nsswitch.conf` ファイルと `/etc/resolv.conf` ファイルを正しく構成してください。`/etc/nsswitch.conf` ファイルは、クライアントのネー ムサービススイッチ構成ファイルです。`/etc/resolv.conf` では、クライアントの DNS サーバーの IP アドレスを一覧にします。

2. **NIS+** と **DNS** の共同操作をテストします。

   情報への要求を複数の名前空間の間で、問題なく渡せることを確認します。

3. **NIS+** を **NIS** と並行して操作している場合は、情報の転送をテストします。

   `nispopulate` スクリプトを使用して、NIS から NIS+ へ情報を転送します。NIS+ から NIS へデータを転送するには、 `nisaddent -d` を実行後、 `ypmake`を実行します (詳細についてはマニュアルページを参照してください)。 スクリプトを使用して、この処理を自動化します。テーブル、特に `hosts` テーブルと `passwd` テーブルを同期させる方針を設定します。NIS 環境と NIS+ 環境の間で整合性を維持するために使用するツールをテストします。NIS+ テーブルを実際の情報源にする時期を決めます。

4. **DNS** と **NIS** の両方との **NIS+** の操作をテストします。

   3 つの名前空間をすべて一緒にテストして、追加した接続に問題がないことを確認します。

第 3 段階 - NIS+ 名前空間を十分に稼働させる

1. クライアントを **NIS+** に移行します。

   一度に 1 つのワークグループのクライアントを移行して、1 つのサブネット内のワークグループをすべて移行してから、他のサブネットでの移行を行います。このように、1 つのサブネット内のクライアントをすべて移行したら、そのサブネット上の NIS サービスを削除することができます。各クライアントを移行したら、検査スクリプトを実行して、移行が正しく行われたことを確認します。この検査スクリプトは、ユーザーに対して、問題とその問題の報告に役立つサポート形態を通知します。実際

に必要は手順は、サイトによって異なります。

`nisclient` スクリプトを使用して、NIS クライアントを NIS+ クライアントに移行します。クライアントの DNS 構成を変更する必要がある場合は、独自のスクリプトを作成して、その処理を自動化しなければなりません。

`/usr/local` のような共有の、マウントされるディレクトリがサイトにあれば、時間を節約することができます。このディレクトリにスクリプトを置いて、移行時に、このスクリプトをスーパーユーザーとして実行するよう依頼する電子メールをクライアントに送ることができます。

2. クライアントの移行中、移行の状況を監視します。

   計画と突き合わせて進行状況を追跡し、計画段階では予測できなかった重大な問題がないかを監視します。状況を通知して、関係するグループがそれを追跡できるようにします。

3. **NIS** サーバーを削除します。

   サブネット上のすべてのクライアントが NIS+ に移行されたら、NIS サーバーを削除します。特定のサブネットに、NIS サービスを必要とするクライアントがある場合は、NIS+ サーバーの NIS 互換モード機能を使用します。NIS サーバーをそのままにしないでください。

4. **NIS+** のパフォーマンスを評価します。

   実装が完了したら、NIS+ が正しく機能しているかどうかをテストします。

5. **NIS+** 環境を最適化します。

   性能の評価結果に基づいて、必要であれば NIS+ の環境を変更します。このような改善には、選択した複製サーバーを負荷の高いドメインに追加するような簡単なものや、ドメインのグループの NIS+ 情報の記憶領域を再編成したりすることが含まれます。

6. 新しいドメインを整理します。

   移行の際に、処理の簡便化のため、古いドメインの名前を変更しなかった場合は、それらをここで新しい NIS+ の命名方式に合わせて変更します。たとえば、物理的な位置を表す名前を持つドメインをいくつか残し、その一方で組織的な階層へ移行した場合は、物理的な位置を表す名前を組織を表す名前に変更します。

第 4 段階 - NIS 互換ドメインを移行する

1. 最後の **NIS** クライアントを **NIS+** に移行します。

   NIS 互換の NIS+ ドメインが、できるだけ早く不要になるようにします。最後の NIS クライアントを NIS+ に移行すると、NIS+ のセキュリティ機能を利用できるようになります。ネットワーク上で Solaris 以外のコンピュータを実行している場合は、NIS 互換の NIS+ ドメインを削除することはできません。

2. セキュリティ構成を調整します。

   NIS クライアントがなくなったら、NIS+ サーバーを標準モードで再起動して、NIS+ テーブルに対して `nischmod` を実行し、アクセス権レベルを変更して、NIS 互換によって生じたセキュリティの不備を削除することができます。NIS+ 主体に以前に資格を作成しなかった場合は、ここで資格を作成してください。認証されていない主体のアクセスは制限してください。

3. その他の評価とプログラムの改善を行います。

   操作手順を評価して、改善できるものがないかを調べます。特に、問題回復に使用される手順を調べてください。新しい NIS+ リリースと機能強化の可能性について検討します。また、新しい NIS+ テーブルを必要とする可能性がある Solaris の構成要素の開発を調査します。さらに、NIS+ 管理作業をより効率的に実行できるようにする自動化ツールを探します。最後に、社内の開発担当者と協力して、NIS+ API を利用できるように援助してください。

   これで NIS+ への移行は完了です。

第 27 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 *: DNS*、*NIS*、*LDAP* 編*)*』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、690 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。690 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 (698 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。

移行の準備

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、681 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、682 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、671 ページの「`nisplusLDAPobjectDN`」 を参照してください。

- `nisplusLDAPbaseDomain`

  NIS+ オブジェクト仕様 (669 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 から 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 (数万から数十万のエントリ) 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 684 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。このコンテナには、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、672 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。複数値 cn ((cn) として示す) が、複数の name 列の値 ((name) として示す) に対応づけられます。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。675 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 (675 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。675 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。675 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。676 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。674 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、683 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、『*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*)*』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、『*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*)*』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 (667 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。

  次の例に示すように、`nisplusObject` 属性と `nisplusObjectContainer` オブジェクトクラスを作成します。LDIF データは、`ldapadd(1)` に適用することができます。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: ( 1.3.6.1.4.1.42.2.27.5.42.42.1.0 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、675 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")), \
```

    次のように変更します。

```
(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きに、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、674 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、674 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、684 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

エラーメッセージ

この付録では、Solaris オペレーティング環境 の DNS、NIS、および NIS+ ネームサービスで使用される一般的なエラーメッセージを、アルファベット順に説明します。メッセージごとに説明を行い、該当するものがある場合は、解決策や、このマニュアルの他の部分の参照箇所を示します。

エラーメッセージについて

ここで説明したエラーメッセージの一部は、マニュアルページでさらに詳しく説明します。

エラーメッセージの内容

エラーメッセージは、ポップアップウィンドウ、シェルツールのコマンド行、ユーザーコンソールウィンドウや、各種のログファイルに表示または記録されます。`/etc/syslog.conf` ファイルに指定されているエラーの重大度の基準を上げたり下げたりすることもできます。

ほとんどの場合、入力したコマンドか、コマンドが送られたコンテナオブジェクト (ファイル、マップ、テーブル、またはディレクトリ) によって、エラーメッセージが生成されます。しかし時には、コマンドに応答したサーバーによって、エラーメッセージが生成されることもあります (メッセージは通常、`syslog` に記録される)。たとえば、「`permission denied`」メッセージは、ほとんどの場合ユーザーかマシンが原因となって発生しますが、コマンドやマシンから要求された機能を実行するのに必要なアクセス権を、サーバー上のソフトウェアが持っていない場合に発生することもあります。

同様に、一部のコマンドは、非常に多くの種類のオブジェクトに対して検索や照会を行います。オブジェクトによっては、原因が明確であるとは限りません。オブジェクトのアクセス権 (読み取り専用の状態、利用できないなどの状態) が原因で、エラーメッセージが返されることがあります。このような場合は、どのオブジェクトが原因となって問題が起こったのか、メッセージからわからない場合もあります。

通常の操作では、ネーミングなどのソフトウェアとサーバーは、関数ルーチン呼び出しを行います。時として、これらの呼び出しが障害を起こし、エラーメッセージが生成されることがあります。また、ユーザーが入力したコマンドをクライアントやサーバーが処理する前に、それ以外のコマンドによる呼び出しが障害を起こし、エラーメッセージが生成されることもあります。そのようなエラーメッセージは、あたかも今入力したコマンドに対する応答のように見えるかもしれませんが、実際はそれまでの操作に対する応答です。

注 – 名前空間で作業を行なっているときに、遠隔手続き呼び出し (RPC) によってエラーメッセージが生成される可能性もあります。これらの RPC エラーメッセージは、このマニュアルでは説明していません。ご使用のシステムのマニュアルを参照してください。

状況によって異なる意味

NIS+ のソフトウェアのどの部分がエラーメッセージを生成したかにより、同じエラーメッセージがやや異なる意味を持つことがあります。たとえば、`nisls` コマンドが「`Not Found`」というメッセージを生成した場合、指定された名前の NIS+ のオブジェクトが存在しないことを意味します。しかし、`nismatch` コマンドが同じメッセージを表示した場合、検索基準に一致するテーブル項目が見つからなかったことを意味します。

エラーメッセージのアルファベット順ソート規則

この付録のエラーメッセージは、次の規則に従ってアルファベット順に示します。

- 大文字と小文字を区別しません。つまり、「A」で始まるメッセージと「a」で始まるメッセージは同じ順になります。
- アルファベット以外の記号は無視します。つまり、`_svcauth_des` で始まるメッセージは、文字「S」で始まる他のメッセージと同じ順になります。
- 「NIS+」で始まる (または「NIS+」を含む) エラーメッセージは、アルファベット順で「NIS」で始まる (または「NIS+」を含む) メッセージの後になります。
- エラーメッセージの前に、エラーメッセージを生成したホスト、アプリケーション、プログラム、ルーチンの日付または名前が付き、その後にコロンが付く場合があります。この場合には、コマンドの最初の名前はメッセージをアルファベット順にするために使われます。

- メッセージの中には、ユーザー ID、プロセス番号、ドメイン名などの変数が含まれています。この章では、これら変数は斜体の文字で示されます。変数は任意のものにできるので、この章に示すメッセージの分類には含まれていません。たとえば、実際には `sales:is not a table` (`sales` は変数) と表示されるメッセージは、この付録では「*name*: is not a table」と表現しており、「`is not a table` 」として文字「I」で始まるメッセージに入っています。
- 「`** ERROR:`*domainname* `does not exist`」のように、アスタリスクで始まるエラーメッセージは、NIS+ のインストールスクリプトや設定スクリプトによって生成されたものです。これらのメッセージは、アスタリスクを無視し、最初の文字でソートされます。

エラーメッセージ内の番号

- 多くのメッセージには、IP アドレスが含まれます。IP アドレスは *n.n.n.n.* で示されます。
- エラーメッセージにプロセス ID 番号、項目の数などのような番号が含まれる場合があります。エラーメッセージ内の番号は *nnnn* で示されます。

NIS+、FNS に共通するエラーメッセージ

`abort_transaction: Failed to action NIS+` *objectname*
: `abort_transaction` ルーチンが、サーバーのクラッシュや他の回復不可能なエラーにより、不完全なトランザクションから抜けることに失敗しました。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`abort_transaction: Internal database error abort_transaction: Internal error, log entry corrupt NIS+` *objectname*
: これら 2 つのメッセージは、名前空間のデータベースやログが壊れていることを意味します。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`add_cleanup: Cant allocate more rags.`
: このメッセージは、システムで使用できるメモリーが不足していることを示しています。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`add_pingitem: Couldn't add` *directoryname* `to pinglist (no memory)`
メモリー不足の問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`add_update: Attempt add transaction from read only child. add _update Warning: attempt add transaction from read only child`
読み取り専用の子プロセス `rpc.nisd` が、ログに項目を追加しようとしました。ログの中にこのメッセージが記録されることがあっても、重大な問題ではありません。しかし、頻繁に記録される場合は、ご購入先にご連絡ください。

`Attempting to free a free rag!`
このメッセージは、`rpc.nisd` にソフトウェア上の問題が起こったことを意味します。`rpc.nisd` は、異常終了したはずです。`ps -ef | grep rpc.nisd` を実行して、`rpc.nisd` が現在も動作しているかどうか確認します。まだ動作している場合は、`rpc.nisd` を終了させ、前回と同じオプションを指定して再起動します。動作していない場合は、前回と同じオプションで再起動します。`/var/nis` をチェックして、`core` ファイルのダンプが行われていないか確認します。`core` ファイルが存在する場合は、削除してください。

注 – `-YB` オプションを指定して `rpc.nisd` を起動した場合は、`rpc.nisd_reply` デーモンも終了しなければなりません。

`Attempt to remove a non-empty table`
`nistbladm` が、まだエントリを含んでいる NIS+ テーブルを削除しようとしました。または、`nisrmdir` が、ファイルあるいはサブディレクトリを含むディレクトリを削除しようとしました。

- テーブルを削除する場合には、`niscat` を使ってテーブルの内容をチェックし、`nistbladm` を使って既存の内容を削除します。
- ディレクトリを削除する場合には、`nisls -l -R` を使って既存のファイルまたはサブディレクトリをチェックして、それらを最初に削除します。
- `nisrmdir -s` を使って、ドメインから複製を分離するときに、複製がダウンしているか、またはマスターとの通信不能の状態にある場合に、このエラーメッセージが表示されます。このような場合は、マスターサーバー上で `nisrmdir -f -s` *replicaname* コマンドを実行すれば、強制的に切り離すことができます。しかし、`nisrmdir -f -s` *replicaname* を使って通信不能な複製を分離する場合には、複製がオンライン状態に戻ったらすぐに、`nisrmdir -f -s` *replicaname* を再実行して、複製の `/var/nis` ファイルシステムを消去する必要があります。`nisrmdir -f -s` *replicaname* を再実行しないと、複製がサービスを再開した時に複製上に残された古い情報によって問題が発生します。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOMEMORY` によって生成されます。詳細については、`nis_tables(3N)` のマニュアルページを参照してください。

`authdes_marshal: DES encryption failure`
認証データの DES 暗号化に失敗しました。考えられる原因は次のとおりです。

- ライブラリ関数か引数が壊れている
- DES 暗号化チップを使用している場合、そのチップに問題があります。

ご購入先にご連絡ください。

`authdes_refresh: keyserv is unable to encrypt session key`
keyserv プロセスは、公開鍵を使用してセッション鍵の暗号化を行うことができませんでした。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`authdes_refresh: unable to encrypt conversation key`
keyserv プロセスは、公開鍵を使用してセッション鍵の暗号化を行うことができませんでした。このメッセージが表示された場合、通常はユーザーの側で何らかの操作を行う必要があります。考えられる原因は次のとおりです。

- keyserv プロセスが終了しているか、応答しません。ps -ef を使用して、keyserv プロセスが keyserv のホストで動作しているかどうかチェックします。動作していない場合は、keyserv を起動し、keylogin を実行します。
- クライアントが keylogin を実行していません。クライアント側で keylogin を実行し、問題が解決できるかどうか調べます。
- クライアントのホストに、資格が割り当てられていません。クライアントのホームドメインの cred テーブルに対して nismatch を使用して、クライアントのホストに適切な資格が割り当てられているかどうか調べます。割り当てられていない場合は、DES の資格を作成します。
- DES 暗号化に失敗しました。エラーメッセージ「authdes_marshal:DES encryption failure」を参照してください。

セキュリティ鍵の問題の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`authdes_refresh: unable to synchronize clock`
このエラーメッセージは、クライアントとサーバーそれぞれのクロックが同期していないことを示しています。通常は、この問題は自動的に訂正されます。しかし、このメッセージの後に、タイムスタンプに関係したエラーが発生した場合は、手作業でクロックを同期させなければなりません。この問題が再度発生した場合は、rpcbind が正しく機能しているかどうかチェックします。

`authdes_refresh: unable to synch up w/server`
クライアントとサーバーそれぞれのクロックを同期させることに失敗しました。サーバー上の rpcbind プロセスが応答しなかったことが原因となった可能性もあります。サーバー上で ps -ef を実行し、rpcbind が動作しているかどうか調べ

ます。動作していない場合は、rpcbind を再度起動します。このメッセージの後に、タイムスタンプに関係したエラーが発生した場合は、rdate *servername* を使用して、クライアントのクロックを手作業でサーバーのクロックに同期させなければなりません。

`authdes_seccreate: keyserv is unable to generate session key`
このエラーメッセージは、keyserv がこのセッション用のランダムな DES キーを生成できなかったことを示しています。このメッセージが表示された場合、ユーザーの側で何らかの操作を行う必要があります。

- keyserv が正しく動作しているかどうかチェックします。動作していない場合は、automountd、rpc.nisd、sendmail などのように、secure RPC を使用したり NIS+ のコールを行う、長時間にわたって動作するプロセスとともに、keyserv を再度起動します。次に、keylogin を実行します。
- keyserv が起動されていて正しく動作している場合は、このエラーをログに記録したプロセスをもう一度起動します。

`authdes_seccreate: no public key found for` *servername*
クライアント側で、サーバー名 *servername* の DES の資格を取得できませんでした。このメッセージが表示された場合、ユーザーの側で何らかの操作を行う必要があります。

- *servername* に DES の資格が割り当てられているかどうかチェックします。割り当てられていない場合は、DES の資格を作成します。
- スイッチ構成ファイルをチェックして、指定されているネームサービスを調べ、そのサービスが応答するかどうか確認します。応答しない場合は、ネームサービスを再度起動します。

`authdes_seccreate: out of memory`
メモリー不足の問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`authdes_seccreate: unable to gen conversation key`
keyserv プロセスは、ランダムな DES キーを生成できませんでした。最も可能性の高い原因は、keyserv プロセスが停止しているか、応答しないことです。ps -ef を使用して、keyserv プロセスが keyserv のホストで動作しているかどうかチェックします。動作していない場合は、keyserv を起動し、次に keylogin を実行します。

keyserv を起動し直しても問題が解決しない場合は、secure RPC を使用したり NIS+ をコールするプロセス (たとえば、automountd、rpc.nisd、sendmail) が動作していない可能性があります。これらのプロセスが動作しているかどうかチェックし、動作していない場合は再度起動します。

セキュリティ鍵の問題の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`authdes_validate: DES decryption failure`
エラーメッセージ「authdes_marshal: DES encryption failure」で DES 暗号化の失敗について参照してください。

`authdes_validate: verifier mismatch`
クライアントがサーバーに送ったタイムスタンプが、サーバーから受け取ったタイムスタンプに一致していません (Secure RPC セッションの中で回復できません) 。考えられる原因は次のとおりです。

- クライアントかサーバーのキャッシュに格納されているセッション鍵またはタイムスタンプのデータが壊れています。
- サーバーがキャッシュから、まだ有効なセッションのセッション鍵を削除しました。
- ネットワークデータが壊れています。

コマンドを再度実行してください。

`CacheBind:xdr_directory_obj failed`
最も可能性のある原因は次のとおりです。

- `xdr_directory_obj` ルーチンに渡されたパラメータが不適切です。直前に入力したコマンドの構文が正確かどうかチェックします。
- システムへのメモリーの割り当てに失敗しました。メモリーの問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。
- コマンドの構文が正確で、システムもメモリー不足を起こしていないと思われる場合は、ご購入先にご連絡ください。

`Cache expired`
オブジェクトのキャッシュから返されたエントリが、有効期限を過ぎています。つまり、持続時間の値が 0 になり、エントリが変更された可能性があることを示しています。フラグ `-NO_CACHE` が検索用関数に渡された場合は、その検索関数は動作を繰り返し行い、有効期限を過ぎていないオブジェクトのコピーの取得を試みます。

このメッセージは、NIS+ のエラーコード定数 `NIS_CACHEEXPIRED` によって生成されます。詳細については、`nis_tables` と `nis_names` のマニュアルページを参照してください。

`Callback: - select failed` *`message nnnn`*
内部システムコールがエラーを起こしました。ほとんどの場合、この問題は自動的に修正されます。自動的に修正されない場合は、`rpc.nisd` が異常終了していないかどうか確認します。異常終了している場合は、再度起動します。この問題が頻繁に発生する場合は、ご購入先にご連絡ください。

`CALLBACK_SVC: bad argument`
内部システムコールがエラーを起こしました。ほとんどの場合、この問題は自動的に修正されます。自動的に修正されない場合は、`rpc.nisd` が異常終了していないかどうか確認します。異常終了している場合は、再度起動します。この問題が頻繁に発生する場合は、ご購入先にご連絡ください。

`Cannot grow transaction log error` *string*
システムは、ログファイルへの追加を行うことができません。理由は *string* が示すとおりです。このメッセージが表示されたときは、多くの場合、ディスク領域が不足しています。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Cannot truncate transaction log file`
ログにチェックポイントを実行しようとしましたが、`rpc.nisd` デーモンは、チェックポイントが設定されたエントリをログから削除した後で、ログファイルの圧縮を試みました。このルーチンが失敗する要因については、 `ftruncate` のマニュアルページを参照してください『*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「NIS の障害追跡」も参照してください。

`Cannot write one character to transaction log, error`*message*
`rpc.nisd` デーモンは、現在のトランザクションを使用して、トランザクションログに更新部分を追加しようとしましたが失敗しました。原因は、関数が返した *message* の中に示されています。書き込みルーチンのマニュアルページの中に詳細が記されていることもあります。

`Can't compile regular expression` *variable*
`keypat` 内の正規表現が不適切だったために、`nisgrep` コマンドによってエラーメッセージが返されました。

`Can't get any map parameter information`
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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Can't find name service for passwd`
「`nsswitch.conf` ファイルがない」、「`nsswitch.conf` ファイルに `passwd` のエントリがない」、「`passwd` のエントリが意味をなさない」、「`passwd` のエントリの形式が正しくない」のいずれかを意味するエラーメッセージです。

`Can't find` *name* `'s secret key`
考えられる原因は次のとおりです。

- 正しくないパスワードを入力しました。
- `cred` テーブルの中に、*name* のエントリがない
- NIS+ が鍵を復号できませんでした (エントリが壊れている可能性があります)。

- nsswitch.conf ファイルが、cred テーブルの中に記録されている NIS+ パスワードとは異なるパスワードを使用して、/etc/passwd ファイルの中に格納されているローカルパスワードの照会を行なっています。

これらの問題を診断および解決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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`checkpoint_log: Called from read only child ignored.`
このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`checkpoint_log: Unable to checkpoint, log unstable.`
安定した状態にないログに対して、チェックポイントを実行しようとしました。つまり、ログは再同期、更新、チェックポイント、どれかの状態にありました。ログが安定した状態になるまで待って、nisping コマンドを再度実行します。

`check_updaters: Starting resync.`
システムの状態メッセージです。なんらかの操作を行う必要はありません。

`Child process requested to checkpoint!`
このメッセージは、システムが修正することのできる、ソフトウェアの小さな問題が発生したことを意味しています。これらのメッセージが頻繁に表示される場合は、/etc/syslog.conf ファイル内のしきい値レベルを変更します。詳細については、syslog.conf のマニュアルページを参照してください。

`Column not found:` *columnname*
指定した列名は、指定したテーブルの中に存在しません。

`Could not find` *string* `'s secret key`
考えられる原因は次のとおりです。

- 正しくないパスワードを入力しました。
- cred テーブルの中に、name のエントリがありません。
- NIS+ が鍵を復号できませんでした (エントリが壊れている可能性があります)。
- nsswitch.conf ファイルで、公開鍵の指定方法を間違えています。cred テーブルの中に登録されている NIS+ パスワードとは異なるパスワードを使用して、/etc/publickey ファイルの中に格納されているローカル公開鍵の照会を行なっています。

これらの問題を診断および解決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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Could not generate netname`
`keylogin`を実行する際、Secure RPC のソフトウェアが、ユーザーの UID に対応する Secure RPC のネット名を生成できませんでした。考えられる原因は次のとおりです。

- マシンのホームドメインにある NIS+ の `cred` テーブルで、LOCAL の資格が割り当てられていません。
- `/etc/passwd` ファイル内にあるローカルエントリの UID が、NIS+ の `passwd` テーブル内にある UID と違っています。

string`: could not get secret key for '`*string*
考えられる原因は次のとおりです。

- 正しくないパスワードを入力しました。
- `cred` テーブルの中に、`name` のエントリがありません。
- NIS+ が鍵を復号できませんでした (エントリが壊れている可能性があります)。
- `nsswitch.conf` ファイルで、公開鍵の指定方法を間違えています。`cred` テーブルの中に登録されている NIS+ パスワードとは異なるパスワードを使用して、`/etc/publickey` ファイルの中に格納されているローカル公開鍵の照会を行なっています。

これらの問題を診断および解決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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Couldn't fork a process!`
サーバーは、コールバックの要求を満たす子プロセスをフォークできませんでした。おそらく、システムが最大のプロセス数に達したためです。不要なプロセスをいくつか終了するか、システムが処理できるプロセス数を増やしてください。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Couldn't parse access rights for column` *string*
このメッセージは、通常 `nistbladm -u` コマンドで、演算子として `+` (プラス記号)、`-` (マイナス記号)、`=` (等号) 以外の記号を入力したときに返されます。他に考えられる原因として、列の権利をコンマで区切っていないことや、アクセス権の種類として `r`、`d`、`c`、`m` のどれでもない文字を指定したことが挙げられます。エントリのタイプによるエラーの場合は、構文をチェックしてください。入力が正しくてもエラーが発生する場合は、テーブルが壊れていることも考えられます。

`Database for table does not exist`
テーブルの参照に失敗しました。原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOMEMORY` によって生成されます。詳細については、`nis_tables` と `nis_names` のマニュアルページを参照してください。

`_db_add: child process attempting to add/modify _db_addib: non-parent process attempting an add`
これらのメッセージは、読み取り専用プロセスや、親プロセス以外のプロセスが、データベース内のオブジェクトを追加または変更しようとしたことを示します。ほとんどの場合、なんらかの操作を行う必要はありません。この問題が頻繁に発生する場合は、ご購入先にご連絡ください。

`db_checkpoint: Unable to checkpoint` *string*
このメッセージは、なんらかの理由で、NIS+ がディレクトリにチェックポイントを実行できなかったことを示します。ほとんどの場合、ディスクが一杯になっています。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`_db_remib: non-parent process attempting an remove _db_remove: non-parent process attempting a remove`
これらのメッセージは、読み取り専用プロセスや、親プロセス以外のプロセスが、テーブルエントリを削除しようとしたことを示します。ほとんどの場合、なんらかの操作を行う必要はありません。この問題が頻繁に発生する場合は、ご購入先にご連絡ください。

`Do you want to see more information on this command?`
このメッセージは、スクリプトのコマンド行の中に、なんらかの構文エラーかスペルミスがあることを示します。

`Entry/Table type mismatch`
テーブルでエントリを追加または変更しようとして、他のタイプのテーブルからエントリを引き渡した場合に、このメッセージが表示されます。たとえば、列数が等しくないことが考えられます。更新に使っているテーブルのタイプが正確に一致しているかどうかチェックします。

このメッセージは、NIS+ のエラーコード定数 `NIS_TYPEMISMATCH` によって生成されます。詳細については、`nis_tables` のマニュアルページを参照してください。

`**ERROR: chkey failed again. Please contact your network administrator to verify your network password.`
このメッセージは、ネットワークのパスワードの入力ミスを示します。

- マシンを初めて初期設定するときは、ネットワーク管理者に連絡を取り、ネットワークのパスワードを確認します。
- このマシンが、同じドメインの中で NIS+ のクライアントとして初期設定されたことがある場合は、Secure RPC のパスワードプロンプトで、root のログインパスワードを入力してみます。
- このマシンが現在 NIS+ のクライアントであり、他のドメインのクライアントに変更したい場合は、`/etc/.rootkey` ファイルを削除し、ネットワーク管理者から受け取ったパスワード (または `nispopulate` スクリプトから生成したパスワード) を使用して、`nisclient` スクリプトを再度実行します。

`Error: Could not create a valid NIS+ coldstart file`
このメッセージは、NIS+ の初期設定ルーチン `nisinit` によって生成されます。この後に、「`lookup:..`」で始まるメッセージが表示されます。2 番目のメッセージは、NIS+ の有効なコールドスタートファイルが作成できなかった理由を示します。

`**ERROR: could not restore file` *filename*
このメッセージは、NIS+ が、*filename*`.no_nisplus` を *filename* に変更できなかったことを示します。

システムコンソールを使用して、システムエラーメッセージの有無をチェックします。

- システムエラーメッセージがある場合は、そのエラーメッセージの示す問題を解決し、`nisclient -i` を再度実行します。
- システムエラーメッセージがない場合は、そのファイルを手作業で名前を変更し、`nisclient -i` を再度実行します。

`**ERROR: Couldn't get the server NIS+_server's address.`
このスクリプトは、指定されたドメインのサーバーの IP アドレスを取り出すことができませんでした。`/etc/hosts` ファイルまたは `/etc/inet/ipnodes` ファイル内のサーバー *NIS+_server* に、手作業で IP アドレスを追加し、`nisclient -i` を再度実行します。

`**ERROR: directory` *directory-path* `does not exist`
このメッセージは、正しくないディレクトリパス名を入力したことを示します。正しいディレクトリパス名を入力します。

`**ERROR:` *domainname* `does not exist.`
このメッセージは、存在しないドメインを複製しようとしたことを示します。

- *domainname* のスペルが正しくない場合は、正しいドメイン名を指定して、このスクリプトをもう一度実行します。
- *domainname* のドメインが存在しない場合は、そのドメインを作成します。次に、複製を行うことができます。

`**ERROR:` *parent-domain* `does not exist`
このメッセージは、コマンド行で指定されたドメインの親ドメインが存在しないことを示します。このメッセージは、ルートマスター以外のサーバーの設定を行なっているときにだけ表示されるはずです。

- ドメイン名のスペルが正しくない場合は、正しいドメイン名を指定して、もう一度このスクリプトを実行します。
- このドメインの親ドメインが存在しない場合は、最初に親ドメインを作成し、次にこのドメインを作成します。

**ERROR: Don't know about the domain " *domainname*". Please check your *domainname*

このメッセージは、認識できないドメイン名が入力されたことを示します。正しいドメイン名を指定して、もう一度このスクリプトを実行します。

**ERROR: failed dumping *tablename* table

このスクリプトは、指定されたテーブルのダンプに失敗したため、cred テーブルを生成できませんでした。

- niscat *tablename* .org_dir の実行に失敗した場合は、すべてのサーバーが動作しているかどうか確認し、もう一度このスクリプトを実行して、tablename テーブルを生成します。
- niscat *tablename* .org_dir が実行できる場合は、NIS+ のサーバーが一時的にビジーだったために、エラーが発生した可能性があります。もう一度このスクリプトを実行して、tablename テーブルを生成します。

**ERROR: host *hostname* is not a valid NIS+ principal in domain *domainname* .This host name must be defined in the credential table in domain *domainname*. Use nisclient -c to create the host credential

あるマシンを NIS+ のサーバーにする前に、そのマシンを NIS+ のクライアントとし、適切な資格を割り当てる必要があります。マシンを NIS+ のルート複製サーバーに変換するには、最初にそのマシンをルートドメインの NIS+ クライアントにする必要があります。ドメインに新しいクライアントを追加するための指示に従い、nisserver -R を再度実行します。

マシンを NIS+ のルート以外のマスターサーバーや複製サーバーに変換するには、そのマシンを、サービスを提供するドメインの親ドメインの NIS+ クライアントにしておかなければなりません。ドメインに新しいクライアントを追加するための指示に従い、nisserver -M か nisserver -R を再度実行します。

ルートマスターサーバーを設定するときは、この問題は発生しないはずです。

Error in accessing NIS+ cold start file is NIS+ installed?

このメッセージは、NIS+ がマシンにインストールされていない場合や、何らかの理由で /var/nis/NIS_COLD_START ファイルが見つからないかアクセスできない場合に発生します。/var/nis/NIS_COLD_START ファイルが存在しているかどうかチェックします。ファイルが存在している場合は、パスが正しく設定されているかどうか、また NIS_COLD_START に正しいアクセス権が割り当てられているかどうか確認します。次に、古いコールドスタートファイル名の変更か削除を行い、nisclient スクリプトをもう一度実行して、NIS+ をマシンにインストールします。

このメッセージは、NIS+ エラーコード定数 NIS_COLDSTART_ERR を送信したキャッシュマネージャにより生成されます。ファイルにアクセスできない原因の詳細については、write および open のマニュアルページを参照してください。

`Error in RPC subsystem`
これは致命的なエラーで、RPC サブシステムになんらかの障害が発生したことを示します。通常は、クライアント側かサーバー側に、RPC 要求が失敗した理由を示す `syslog` メッセージがあります。

このメッセージは、NIS+ のエラーコード定数 `NIS_RPCERROR` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`**ERROR: it failed to add the credential for root`
NIS+ のコマンド `nisaddcred` は、ルートマスターサーバーの設定を行なっているときに、`root` の資格を作成できませんでした。システムコンソールを使用して、システムエラーメッセージの有無をチェックします。

- システムエラーメッセージが発生している場合は、そのエラーメッセージの示す問題を解決し、`nisserver` を再度実行します。
- システムエラーメッセージが発生していない場合は、`rpc.nisd` プロセスが動作しているかどうかチェックします。動作していない場合は、`rpc.nisd` をもう一度起動し、`nisserver` を再度実行します。

`**ERROR: it failed to create the tables`
NIS+ のコマンド `nissetup` は、ディレクトリやテーブルを作成できませんでした。システムコンソールを使用して、システムエラーメッセージの有無をチェックします。

- システムエラーメッセージが発生している場合は、そのエラーメッセージの示す問題を解決し、`nisserver` を再度実行します。
- システムエラーメッセージが発生していない場合は、`rpc.nisd` プロセスが動作しているかどうかチェックします。動作していない場合は、`rpc.nisd` をもう一度起動し、`nisserver` を再度実行します。

`**ERROR: it failed to initialize the root server.`
NIS+ のコマンド `nisinit -r` は、ルートマスターサーバーの初期設定に失敗しました。システムコンソールを使用して、システムエラーメッセージの有無をチェックします。システムエラーメッセージが発生している場合は、そのエラーメッセージの示す問題を解決し、`nisserver` を再度実行します。

`**ERROR: it failed to make the` *domainname* `directory`
`nisserver` を実行して、ルート以外のマスターを作成しているときに、NIS+ のコマンド `nismkdir` は、*domainname* という新しいディレクトリの作成に失敗しました。親ドメインに、新しいドメインを作成するためのアクセス権が割り当てられていません。

- そのドメインの所有者ではなく、親ドメインのグループメンバーでもない場合は、所有者か親ドメインのグループメンバーとして、そのスクリプトをもう一度実行します。
- `rpc.nisd` が、作成中のドメインのサーバーとなるマシンで動作していない場合は、`rpc.nisd` をもう一度起動します。

**ERROR: it failed to promote new master for the *domainname* directory

nisserver スクリプトを使ってルート以外のマスターを作成しているときに、NIS+ のコマンド nismkdir は、ディレクトリ *domainname* の新しいマスターの変換に失敗しました。

- このドメインの親ドメインで変更権を割り当てられていない場合は、親ドメインの所有者かグループメンバーとして、そのスクリプトをもう一度実行します。
- rpc.nisd が、変換先のドメインのサーバーで動作していない場合は、これらのサーバー上で rpc.nisd をもう一度起動し、nisserver を再度実行します。

**ERROR: it failed to replicate the *directory-name* directory

NIS+ のコマンド nismkdir は、ディレクトリ *directory-name* の新しい複製サーバーを作成できませんでした。

- rpc.nisd が、複製サーバーを作成したいドメインのマスターサーバー上で動作していない場合は、マスターサーバーで rpc.nisd をもう一度起動し、nisserver を再度実行します。
- rpc.nisd が、新しい複製サーバー上で動作していない場合は、新しい複製サーバーで rpc.nisd をもう一度起動し、nisserver を再度実行します。

**ERROR: invalid group name. It must be a group in the *root-domain* domain.

このメッセージは、ルートマスターサーバーを構成しているときに、不適切なグループ名が使用されたことを示します。*root-domain* の適切なグループ名を指定して、nisserver -r をもう一度実行します。

**ERROR: invalid name "*client-name*" It is neither an host nor an user name

このメッセージは、不適切な *client-name* が入力されたことを示します。

- *client-name* のスペルが正しくない場合は、正しい *client-name* を指定して、nisclient -c をもう一度実行します。
- *client-name* のスペルは正しくても、適切なテーブルの中に存在していない場合は、テーブルの中に *client-name* を記述して、nisclient -c をもう一度実行します。たとえば、ユーザーのクライアントは passwd テーブルの中に存在し、ホストのクライアントは hosts テーブルの中に存在します。

**ERROR: *hostname* is a master server for this domain. You cannot demote a master server to replica. If you really want to demote this master, you should promote a replica server to master using nisserver with the M option.

同じドメインの中で、マスターサーバーを直接複製サーバーに変換することはできません。しかし、複製サーバーのホスト名を指定して nisserver -M を実行することにより、複製サーバーを新しいマスターサーバーに変換できます。同時に、元のマスターは自動的に複製サーバーに変換されます。

**ERROR: missing hostnames or usernames
このメッセージは、コマンド行でクライアント名が入力されなかったことを示します。クライアント名を指定して、nisclient -c をもう一度実行します。

**ERROR: NIS+ group name must end with a "."
このメッセージは、ピリオドで終わる完全指定のグループ名が指定されなかったことを示します。完全指定のグループ名を使用して、このスクリプトをもう一度実行します。

**ERROR: NIS+ server is not running on *remote-host*. You must do the following before becoming a NIS+ server: 1. become an NIS+ client of the parent domain or any domain above the domain which you plan to serve. (nisclient) 2. start the NIS+ server. (rpc.nisd)
このメッセージは、NIS+ に変換しようとしているリモートマシン上で、rpc.nisd が動作していないことを示します。nisclient スクリプトを使用して、そのマシンを、サービスを提供したいドメインの親ドメイン、またはそれより上位のドメインの NIS+ クライアントにします。次に、*remote-host* 上で、rpc.nisd を起動します。

**ERROR: nisinit failed
nisinit は、NIS_COLD_START ファイルを作成できませんでした。

次のことをチェックします。

- -H オプションで指定した NIS+ サーバーが動作しているかどうか (ping を使用して確認する)
- 正しいドメイン名を入力したかどうか
- rpc.nisd がサーバー上で動作しているかどうか
- 未認証クラスに、そのドメインでの読み取り権が割り当てられているかどうか

**ERROR: NIS map transfer failed. *tablename* table will not be loaded
NIS+ は、このテーブルの NIS マップを、NIS+ データベースに転送できませんでした。

- NIS サーバーのホストが動作している場合は、もう一度このスクリプトを実行します。一時的な障害が原因で、エラーが発生した可能性があります。
- すべてのテーブルでこのエラーが発生した場合は、他の NIS サーバーを使用して、このスクリプトをもう一度実行します。

**ERROR: no permission to create directory *domainname*
親ドメインに、新しいドメインを作成するために必要な作成権が割り当てられていません。そのドメインの所有者ではなく、親ドメインのグループメンバーでもない場合は、所有者か親ドメインのグループメンバーとして、そのスクリプトをもう一度実行します。

**ERROR: no permission to replicate directory *domainname*
このメッセージは、ドメインを複製するためのアクセス権が割り当てられていないことを示します。このドメインの所有者かグループメンバーとして、スクリプトをもう一度実行します。

error receiving zone transfer
DNSのエラーメッセージです。通常これは、主サーバーのDNS ファイルの 1 つの構文エラーを示します。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

**ERROR: table *tablename* .org_dir.*domainname* does not exist." *tablename* table will not be loaded."
スクリプトは、NIS+ のテーブル tablename を見つけることができませんでした。

- *tablename* のスペルが正しくない場合は、正しいテーブル名を指定して、スクリプトをもう一度実行します。
- *tablename* が NIS+ の標準テーブルの 1 つであり、かつ *tablename* テーブルが存在しない場合は、nissetup を使用して、そのテーブルを作成します。*tablename* が NIS+ の標準テーブルではなく、また存在しない場合は、nistbladm を使用して、独自のテーブル *tablename* を作成します。次に、このスクリプトを再度実行して、このテーブルを生成します。
- *tablename* テーブルが存在する場合は、NIS+ のサーバーが一時的にビジーであったためにエラーが発生した可能性があります。このスクリプトをもう一度実行して、*tabelname* テーブルを生成します。

**ERROR: this name "*clientname*" is in both the passwd and hosts tables.You cannot have an username same as the host name.
client-name が、passwd テーブルと hosts テーブルの両方に記述されています。両方のテーブルに、共通の名前を記述することは認められていません。passwd と hosts どちらかのテーブルから、手作業でこのエントリを削除します。次に、nisclient -c を再度実行します。

**ERROR: You cannot use the -u option as a root user.
このメッセージは、スーパーユーザーが nisclient -u を実行しようとしたことを示します。-u オプションは、普通のユーザーだけを初期設定するためのものです。NIS+ クライアントとは異なり、スーパーユーザーは初期設定を行う必要がありません。

**ERROR: You have specified the Z option after having selected the *X* option. Please select only one of these options [*list*] . Do you want to see more information on this command?
いま実行しているスクリプトでは、リスト表示されているオプションのうち、どれか 1 つしか使用できません。

- y と入力すると、詳細な情報が表示されます。
- n と入力すると、スクリプトは停止して、元のプロンプトに戻ります。

スクリプトを終了した後で、オプションを 1 つだけ指定して、もう一度実行します。

**ERROR: you must specify a fully qualified groupname.
このメッセージは、ピリオドで終わる完全指定のグループ名が指定されなかったことを示します。完全指定のグループ名を使用して、このスクリプトをもう一度実行します。

**ERROR: you must specify both the NIS domainname (-y) and the NIS server host name (-h).
このメッセージは、NIS のドメイン名と NIS サーバーのホスト名の一方または両方を指定しなかったことを示します。プロンプトかコマンド行で、NIS のドメイン名と NIS サーバーのホスト名を入力します。

**ERROR: you must specify one of these options: -c, -i, -u, -r.
このメッセージは、これらのオプション -c、-i、-u、-r のどれかがコマンド行に指定されていなかったことを示します。正しいオプションを指定して、スクリプトをもう一度実行します。

**ERROR: you must specify one of these options: -r, -M or -R"
このメッセージは、-r、-M、-R のどのオプションも指定されなかったことを示します。正しいオプションを指定して、スクリプトをもう一度実行します。

**ERROR: you must specify one of these options: -C, -F, or -Y
このメッセージは、-Y、-F どちらのオプションも入力しなかったことを示します。正しいオプションを指定して、スクリプトをもう一度実行します。

**ERROR: You must be root to use -i option.
このメッセージは、普通のユーザーが nisclient -i を実行しようとしたことを示します。nisclient -i を実行できるアクセス権を割り当てられているのは、スーパーユーザーだけです。

Error while talking to callback proc
サーバーがクライアントからコールバックされているときに、サーバー上で RPC エラーが発生しました。この時点でトランザクションは異常終了し、まだ送信されていないデータは破棄されます。サーバー上の syslog で詳細をチェックします。

このメッセージは、NIS+ のエラーコード定数 NIS_CBERROR によって生成されます。詳細については、nis_tables(3N) のマニュアルページを参照してください。

First/Next chain broken
このメッセージは、コールバックルーチンが結果を返しているときに、クライアントとサーバーの接続が解除されたことを示します。プロセスの実行中にサーバーが終了すると、このようなメッセージが発生することがあります。

このメッセージは、NIS+ のエラーコード定数 NIS_CHAINBROKEN によって生成されます。

Generic system error
要求をしているときに、何らかの一般的なシステムエラーが発生しました。システムの syslog の記録をチェックし、サーバーから返されたエラーメッセージを探します。

このメッセージは通常、サーバーがクラッシュしたか、データベースが壊れたことを示します。このメッセージは、サーバーまたは複製サーバーの名前を正しく指定していない場合にも生成される可能性があります。たとえば、サーバーや複製サーバーが自らサービスを提供するドメインより上位ドメインに属しているが、サービスを提供しているドメインに属しているように指定した場合に生成されます。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

このメッセージは、NIS+ のエラーコード定数 NIS_SYSTEMERROR によって生成されます。詳細については、nis_tables と nis_names のマニュアルページを参照してください。

Illegal object type for operation
これらの問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

このメッセージは、NIS+ のエラーコード定数 DB_BADOBJECT によって生成されます。

insufficient permission to update credentials
このメッセージは、動作に必要なアクセス権が割り当てられていない状態で、nisaddcred コマンドを実行したときに発生します。テーブル、列、エントリどれかのレベルでアクセス権が不足している可能性があります。niscat -o cred.org_dir を使用して、cred テーブルでどのようなアクセス権が割り当てられているを確認します。アクセス権を追加する場合は、ユーザー自身またはシステム管理者がオブジェクトのアクセス権の要件を変更するか、必要なアクセス権が割り当てられているグループにユーザーを追加します。

アクセス権の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

Invalid Object for operation

- *Name context* - 関数に渡された名前は、NIS+ の有効な名前ではありません。
- *Table context* - 指定されたオブジェクトは、このテーブルにとって有効な NIS+ エントリのオブジェクトではありません。列数が一致しない場合や、テーブルの中で関連する列とデータ型 (たとえば、バイナリかテキストか) が異なる場合に、このメッセージが発生する可能性があります。

このメッセージは、NIS+ のエラーコード定数 NIS_NOMEMORY によって生成されます。詳細については、nis_tables(3N) と nis_names(3N) のマニュアルページを参照してください。

invalid usecs *Routine_name* : invalid usecs
このメッセージは、struct time stamp タイプの変数フィールド tv_usecs の値が、1 秒未満マイクロ秒単位の値より大きいときに発生します。通常は、何らかのタイプのソフトウェアエラーが原因です。

tablename is not a table
tablename という名前を持つオブジェクトは、テーブルオブジェクトではありません。たとえば、nisgrep や nismatch のコマンド行で、テーブルオブジェクト以外のオブジェクトを指定すると、このメッセージが発生します。

Link Points to illegal name
渡された名前を LINK タイプのオブジェクトとして解決しましたが、そのオブジェクトの内容が不適切な名前を指しています。

テーブルエントリをリンクさせることができません。このエラーメッセージはエントリレベルでリンクを作成しようとすると発生します。

このメッセージは、NIS+ のエラーコード定数 NIS_NOMEMORY によって生成されます。詳細については、nis_tables と nis_names のマニュアルページを参照してください。

Load limit of *number* reached!
サーバー上で子プロセスの数が既に上限に達しているときに、子プロセスを作成しようとしました。このメッセージは、メッセージをログに記録する基準のレベルが LOG_WARNING を含む場合に、サーバーのシステムログに記録されます。

login and keylogin passwords differ
nispasswd を使用してパスワードを変更し、システムはパスワードを変更しましたが、cred テーブル内の資格のエントリを新しいパスワードに更新できず、さらに、passwd テーブルの中で元のパスワードに復元もできない場合に、このメッセージが発生します。このメッセージの後に、次の指示が表示されます。

```
Use NEW password for login and OLD password for
keylogin. Use "chkey -p" to reencrypt the credentials with
the new login password. You must keylogin explicitly after
your next login.
```

この指示の後に、なぜ元のパスワードに復元することができなかったのか、理由を示す状態メッセージが表示されます。これらのメッセージが表示された場合は、その指示に従ってください。

`Login incorrect`
このメッセージの原因として最も多いのは、パスワードのタイプミスです。もう一度入力し直してください。また覚えているパスワードが正しいかどうかも確認してください。特に、大文字と小文字、アルファベットの o と数字の 0、アルファベットの l と数字の 1 を間違えないよう注意してください。

このメッセージが生成されるその他の原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`log_resync: Cannot truncate transaction log file`
ログにチェックポイントを実行しようとしましたが、`rpc.nisd` デーモンは、チェックポイントが設定されたエントリをログから削除した後で、ログファイルの圧縮を試みました。このルーチンが失敗する要因については、 `ftruncate` のマニュアルページを参照してください。『*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「NIS の障害追跡」も参照してください。

`Malformed Name or illegal name`
関数に渡された名前は、NIS+ の有効な名前ではありません。

考えられる理由の一つは、誰かがドメイン名を変更したことです。既存のドメイン名は変更しないでください。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

このメッセージは、NIS+ のエラーコード定数 `NIS_BADNAME` によって生成されます。詳細については、`nis_tables(3N)` のマニュアルページを参照してください。

`_map_addr: RPC timed out`
プロセスやアプリケーションが、必要なデータの取得や NIS+ からのホスト名の解決を行うデフォルト時間内に、NIS+ と通信できませんでした。ほとんどの場合、しばらく待機すれば解決します。パフォーマンスの遅れに関する問題の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Master server busy full dump rescheduled`
このメッセージは、マスターサーバーがビジーであったため、複製サーバーがマスターサーバーから得られたフルダンプを使って自らを更新することができなかったことを示します。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

String `Missing or malformed attribute`
属性の名前が、テーブル内の指定された列に一致していません。または、属性に値が割り当てられていません。

コマンドの構文エラーが原因となっている可能性があります。*string* から、何が間違っているのか推測できます。一般的な原因は、スペルミス、等号 (=) を不適切な場所に置いたこと、列名やテーブル名の間違いなどです。

このメッセージは、NIS+ のエラーコード定数 NIS_BADATTRIBUTE によって生成されます。詳細については、nis_tables のマニュアルページを参照してください。

Modification failed
コマンドを実行しているときに、誰かがグループ名を変更したため、nisgrpadm がこのメッセージを返しました。誰かがグループの操作を行なっているかどうかチェックします。次に、このコマンドを再度実行します。

このメッセージは、NIS+ のエラーコード定数 NIS_IBMODERROR によって生成されます。

Modify operation failed
試みた変更が、何らかの理由で失敗しました。

このメッセージは、NIS+ のエラーコード定数 NIS_MODFAIL によって生成されます。詳細については、nis_tables(3N) と nis_names(3N) のマニュアルページを参照してください。

Name not served by this server
指定された名前へのサービスを提供していないサーバーに、要求を行いました。通常、このメッセージは発生しないはずです。しかし、内蔵のサーバー検索メカニズムを使用しない場合は、独自のメカニズムが壊れた結果、発生する可能性もあります。

他に考えられる原因は次のとおりです。

- コールドスタートファイルが壊れています。/var/nis/NIS_COLD_START を削除して、再起動します。
- ローカルキャッシュの期限切れのような、キャッシュの問題。nis_cachemgr と /var/nis/NIS_SHARED_DIRCACHE のプロセスを終了し、次に再起動します。ルートディレクトリ以外の場所で問題が発生した場合は、単純にドメインのキャッシュマネージャのプロセスを終了し、コマンドをもう一度実行します。
- 誰かが複製サーバーから、このディレクトリを削除しました。

このメッセージは、NIS+ のエラーコード定数 NIS_NOT_ME によって生成されます。詳細については、nis_tables と nis_names のマニュアルページを参照してください。

Named object is not searchable
テーブル名を NIS+ のオブジェクトとして解決しましたが、オブジェクトが検索できませんでした。

このメッセージは、NIS+ のエラーコード定数 NIS_NOTSEARCHABLEliteral> によって生成されます。詳細については、nis_tables のマニュアルページを参照してください。

Name/entry isn't unique
複数のエントリを返す特定の検索基準に基づく動作が要求されました。たとえば、nistbladm -r を使用して、passwd テーブルからユーザーを削除しようとしました。しかし、このテーブルには、そのユーザーに対応するエントリが 2 つ存在したため、次のメッセージが表示されます。

```
mymachine# nistbladm -r [name=arnold],passwd.org_dir
Can't remove entry: Name/entry isn't unique
```

-r ではなく -R オプションを使用すると、コマンドを複数のエントリに適用できます。たとえば、arnold に対応するすべてのエントリを削除するには、次のように入力します。

```
mymachine#  nistbladm -R [name=arnold],passwd.org_dir
```

NIS+ error
「NIS+ サーバーがエラーを返したが、passwd コマンドがエラーの種類を特定できなかった」という場合に使用されるエラーメッセージです。

NIS+ operation failed
この一般的なエラーメッセージは、めったに発生しないはずです。通常はシステムが修正することのできる、ソフトウェアの小さな問題が発生したことを意味しています。これらのメッセージが頻繁に表示される場合や、システムにより修正されないと思われる場合は、ご購入先にご連絡ください。

このメッセージは、NIS+ のエラーコード定数 NIS_FAIL によって生成されます。

string: NIS+ server busy try again later
原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

NIS+ server busy try again later
サーバーは現在ビジーです。後でコマンドをもう 1 度実行します。

原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

NIS+ server for *string* not responding still trying
原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

NIS+ server not responding
原因については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`NIS+ server needs to be checkpointed. Use nisping -C`*domainname*

注意 – すぐにチェックポイントを設定してください。

サーバーのシステムログが `LOG_CRIT` のレベルにある場合は、このメッセージが生成されます。これは、ログが大きくなりすぎたことを示します。`nisping -C` *domainname* を使用してチェックポイントを実行し、ログを切り捨てます。

ログサイズの詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`NIS+ servers unreachable`

このソフトエラーは、指定したテーブルオブジェクトが置かれているディレクトリにサービスを提供するサーバーに到達できなかったことを示します。ネットワークの障害やサーバーのクラッシュが起こった場合に、このメッセージが発生する可能性があります。もう一度実行するだけで、成功する可能性もあります。`nis_tables(3N)` と `nis_names(3N)` のマニュアルページの `-HARD_LOOKUP` フラグの説明を参照してください。

このメッセージは、NIS+ のエラーコード定数 `NIS_NaMEUNREACHABLE` によって生成されます。

`NIS+ service is unavailable or not installed`

NIS+ のサービスが利用できないか、インストールされていません。このメッセージは、NIS+ のエラーコード定数 `NIS_UNAVAIL` によって生成されます。

`NIS+: write ColdStart File: xdr_directory_obj failed`

最も可能性のある原因は次のとおりです。

- パラメータが正しくありません。直前に入力したコマンドの構文が正確かどうかチェックします。
- システムへのメモリーの割り当てに失敗しました。メモリーの問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。
- コマンドの構文が正確で、システムもメモリー不足を起こしていないと思われる場合は、ご購入先にご連絡ください。

`nis_checkpoint_svc: readonly child instructed to checkpoint ignored.`

このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`nis_dumplog_svc: readonly child called to dump log, ignore`
このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`nis_dump_svc: load limit reached.`
システムで許可されている子プロセスの最大数に達しました。

`nis_dump_svc: one replica is already resyncing.`
マスターとの同期を実行できる複製は、1 度に 1 つだけです。後でコマンドをもう 1 度実行します。

上記の 3 つのエラーメッセージについては、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`nis_dump_svc: Unable to fork a process`
フォークを行うシステムコールが障害を起こしました。原因については、`fork(2)` のマニュアルページを参照してください。

`nis_mkdir_svc: read-only child called to mkdir, ignored`
このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`nis_ping_svc: read-only child was ping ignored`
このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`nis_rmdir_svc: readonly child called to rmdir, ignored`
このメッセージは、読み取り専用プロセスが、親プロセスに限定されている操作を試みて、その操作が異常終了したことを示す状態メッセージです。なんらかの操作を行う必要はありません。

`nisaddcred: no password entry for uid` *userid* `nisaddcred: unable to create credential`
これら 2 つのメッセージが発生するのは、`nispopulate` スクリプトを実行しているときです。NIS+ のコマンド `nisaddcred` は、リモートドメイン上で、ユーザー ID `userid` に LOCAL の資格を割り当てることに失敗しました 。このメッセージは、リモートドメインで `passwd` テーブルを生成しているときにだけ発生します。

問題を解決するには、ローカルの `passwd` テーブルに、次のテーブルパスを追加します。

```
# nistbladm -u -p passwd.org_dir.remote-domain passwd.org_dir
```

remote-domain は、nispopulate を実行したときに -d オプションで指定したドメインと同じものでなければなりません。このスクリプトをもう一度実行して、passwd テーブルを生成します。

No file space on server
サーバーにファイル領域がありません。

このメッセージは、NIS+ のエラーコード定数 NIS_NOFILESPACE によって生成されます。

No match
最も可能性の高い原因は、インデックスつきの名前を指定するために角括弧を使用し、そのためにシェルが解釈できなくなったことです。たとえば、インデックスつきの名前の両側を角括弧で囲む場合は、全体をさらに引用符で囲まない限り、シェルは角括弧を解釈することができません。そのため、次のメッセージが表示されます。

```
# nistbladm -m shell=/bin/csh [name=miyoko],passwd.org_dir No match
```

正しい構文は次のとおりです。

```
# nistbladm -m shell=/bin/csh `[name=miyoko],passwd.org_dir`
```

No memory
システムのメモリーが不足しているため、指定された操作を行うことができません。メモリーの問題の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

Non NIS+ namespace encountered
名前を完全に解決できませんでした。通常は、関数に渡された名前を解決した結果、NIS+ のネームツリーの中にない名前空間が得られたことを示します。つまり、この名前は登録されていないディレクトリの中にあります。このメッセージが発生した場合、エラーとともに、DIRECTORY タイプの NIS+ オブジェクトが返されます。

このメッセージは、NIS+ のエラーコード定数 NIS_FOREIGNNS によって生成されます。詳細については、nis_tables(3N) と nis_names(3N) のマニュアルページを参照してください。

No password entry for uid *userid* No password entry found for uid *userid*
これらのメッセージは両方とも、ユーザーの資格の作成や追加を試みたときに、passwd テーブルの中にそのユーザーが見つからなかったことを示します。資格の作成や追加を行う前に、passwd テーブルの中にユーザーを追加しておかなければなりません。

- 最も可能性のある原因は、コマンド行でユーザーの *userid* のスペルミスをしたことです。コマンド行の構文が正確か、またミススペルがないかチェックします。

- 正しいドメインにいるか、またはコマンド行で正しいドメインを指定しているかチェックします。
- コマンド行が正しい場合は、passwd テーブルをチェックして、入力した *userid* のユーザーが存在するかどうか確認します。次のように nismatch を使用します。

```
mymachine# nismatch uid=userid passwd.org_dir.
```

passwd テーブルの中にユーザーが含まれていない場合は、資格を作成する前に、nistbladm か nisaddent を使用して、passwd テーブルにユーザーを追加します。

No shadow password information
「制御に必要な情報が欠けているため、パスワードの有効期間の設定が行えなかった」ということを意味します。

Not found *String* Not found
Names context - 名前空間の中に、指定された名前が存在しません。

Table context - 検索基準に一致するエントリが、テーブルの中に存在しません。検索基準が null (すべてのエントリを返す) の場合は、このテーブルが空であり、安全に削除できることを意味します。

-FOLLOW_PATH フラグが設定されている場合にこのエラーが発生するのは、検索基準に一致するエントリが、パス内のどのテーブルにも含まれていないことを意味します。

このメッセージは、NIS+ のエラーコード定数 NIS_NOTFOUND によって生成されます。詳細については、nis_tables(3N) と nis_names(3N) のマニュアルページを参照してください。

この種類の問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

Not Found no such name
このエラーメッセージは、テーブルオブジェクトが置かれているはずの、指定されたディレクトリが存在しないことを示します。サーバーは、自らサービスを提供するテーブルの親サーバーであるはずですが、このテーブルがどのディレクトリに置かれているか認識していません。

このメッセージは、NIS+ のエラーコード定数 NIS_NOSUCHNAME によって生成されます。詳細については、nis_tables と nis_names のマニュアルページを参照してください。

この種類の問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Not master server for this domain`
このメッセージは、複製サーバー上のデータベースを直接更新しようとしたことを意味します。

また、ネームサービスを提供するサーバーに変更の要求が行われたものの、そのサーバーがマスターサーバーでなかったことを意味することもあります。ディレクトリオブジェクトが変更され、その結果新しいマスターサーバーが指定された場合に、これが発生します。`/var/nis/NIS_SHARED_DIRCACHE` ファイル内に、ディレクトリオブジェクトのキャッシュ済みコピーを保持しているクライアントは、`ps` を実行して `nis_cachemgr` のプロセス ID を確認し、`nis_cachemgr` のプロセスを終了させ、`/var/nis/NIS_SHARED_DICACHE` ファイルを削除して、`nis_cachemgr` を再度起動します。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOTMASTER` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`Not owner`
オブジェクトの所有者だけに認められている操作を試みましたが、あなたは所有者ではありません。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOTOWNER` によって生成されます。

`Object with same name exists`
既に存在する名前を追加しようとしました。この名前を追加するには、最初に既存の名前を削除して新しい名前を追加するか、既存のオブジェクトの名前を変更します。

このメッセージは、NIS+ のエラーコード定数 `NIS_NAMEEXISTS` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`parse error:` *string* (`key` *variable*)
`nisaddent` コマンドが `/etc` ディレクトリ内のデータベースファイルを使用しているときに、ファイルのエントリにエラーが見つかると、このメッセージが表示されます。最初の *string* は問題について説明し、`key` の右側の *variable* は、障害のあるエントリを示します。`/etc/passwd` ファイルで問題が起こった場合は、`/usr/sbin/pwck` を使用して、このファイルをチェックします。

`Partial Success`
「要求の受信は正しく行われたが、対応するエントリがなかった」という点を除き、`NIS_NOTFOUND` とほぼ同じです。

このエラーが発生すると、サーバーは、エントリではなくテーブルオブジェクトのコピーを返します。パスの処理や、その他のローカルポリシーの実装をクライアントが行えるようにするためです。

このエラーメッセージは、NIS+ のエラーコード定数 NIS_PARTIAL によって生成されます。詳細については、nis_tables(3N) のマニュアルページを参照してください。

Passed object is not the same object on server
「オブジェクトを名前空間から削除しようとしたが、削除の対象となるオブジェクトとは別のオブジェクトが要求の中で渡されたために処理が異常終了した」ということを意味します。

このエラーメッセージは、NIS+ のエラーコード定数 NIS_NOTSAMEOBJ によって生成されます。詳細については、nis_tables(3N) と nis_names(3N) のマニュアルページを参照してください。

Password does not decrypt secret key for *name*
考えられる原因は次のとおりです。

- 正しくないパスワードを入力しました。
- cred テーブルの中に、*name* のエントリがありません。
- NIS+ が鍵を復号できませんでした (エントリが壊れている可能性があります)。
- Secure RPC パスワードとログインパスワードが一致していません。
- nsswitch.conf ファイルが、cred テーブルの中に記録されている NIS+ パスワードとは異なるパスワードを使用して、/etc/passwd ファイルの中に格納されているローカルパスワードの照会を行なっています。実際に暗号化されているパスワードは、ローカルの /etc/shadow ファイルに格納されています。

これらの問題を診断および解決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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

Password has not aged enough
「パスワードが新しすぎるため、変更ができない」ということを意味します。パスワードは、作成されてから一定の日数 (*N* 日) が経過するまでは変更ができません。

Permission denied
ある操作を試みましたが、必要なアクセス権が割り当てられていなかったため、このエラーが発生しました。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

このエラーメッセージは、ログイン、パスワードの問題、あるいは NIS+ のセキュリティの問題に関係があります。発生原因として最も一般的なのは、「ユーザーのパスワードが管理者によってロックされている」あるいは「ユーザーのアカウントが終了している」ということです。

`Permissions on the password database may be too restrictive`
「NIS+ テーブル中のパスワードフィールドの内容を参照する (あるいは使用する) 権限を、ユーザーが持っていない」ということを意味します。NIS+ のアクセス権については、第 15 章 を参照してください。

`Please notify your System Administrator`
`passwd` コマンドでパスワード情報を更新しようとして表示された場合は、「何らかの理由 (数多く考えられる) で成功しなかった」ということを意味します。成功しない理由としては、「サービスが利用できない」、「必要なサーバーが停止している」、「アクセス権がないといった類の問題がある」などがあげられます。セキュリティの問題については、『*Solaris* のシステム管理 (ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編)』の「NIS の障害追跡」を参照してください。

`Please check your /etc/nsswitch.conf file`
「`nsswitch.conf` ファイルで、パスワード更新についてサポートされていない設定が行われている」ということを意味します。どのような設定がサポートされているかは、300 ページの「`nsswitch.conf` ファイルの必要条件」を参照してください。

`Probable success`
Name context - 要求は成功しました。しかし、返されたオブジェクトはオブジェクトキャッシュからのものであり、サーバーから直接返されたものではありません。オブジェクトキャッシュからオブジェクトを取り出したくない場合は、検索用関数をコールするときに、`-NO_CACHE` フラグを指定しなければなりません。

Table context - 要求は成功しました。しかし、検索パスの中でテーブルを見つけることができなかったので、テーブルがアクセス可能だった場合と比べると、得られた結果は違っている可能性があります。

このメッセージは、NIS+ のエラーコード定数 `NIS_S_SUCCESS` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`Probably not found`
指定されたエントリは、このテーブルの中に存在しません。しかし、パス内のすべてのテーブルが検索されたわけではないので、それらのテーブルの中に、このエントリが含まれている可能性もあります。

このメッセージは、NIS+ のエラーコード定数 `NIS_S_NOTFOUND` によって生成されます。詳細については、`nis_tables(3N)` のマニュアルページを参照してください。

`Query illegal for named table`
要求された構造体をクライアントのライブラリに渡すときに、問題が検出されました。

このメッセージは、NIS+ のエラーコード定数 NIS_BADREQUEST によって生成されます。詳細については、nis_tables(3N) のマニュアルページを参照してください。

Reason: can't communicate with ypbind
『*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「NIS の障害追跡」を参照してください。

replica_update: Child process attempting update, aborted
このメッセージは、読み取り専用プロセスが更新を試みて、それが失敗したことを示す状態メッセージです。

replica_update: error result was string
このメッセージは、複製サーバーへのダンプを行う際に問題が起こったことを示します (*string* は理由を示します)。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

replica_update: error result was Master server busy, full dump
rescheduled replica_update: master server busy rescheduling the
resync. replica_update: master server is busy will try later.
replica_update: nis dump result Master server busy, full dump
rescheduled
これらのメッセージはどれも、サーバーがビジーなので、ダンプが後で行われることを示します。

replica_update: nis dump result nis_perror *errorstring*
このメッセージは、複製サーバーへのダンプを行う際に問題が起こったことを示します (*error string* は理由を示します)。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

replica_update: *nnnn* updates *nnnn* errors
更新に成功したことを示す状態メッセージです。

replica_update: WARNING: last_update (*directoryname*) returned 0!
NIS+ のプロセスは、このディレクトリに対応するトランザクションログが最後に更新されたタイムスタンプを見つけることができませんでした。この結果、システムは、問題の起こったディレクトリについて完全な同期を実行しなければなりません。

Results Sent to callback proc
これは状態メッセージです。なんらかの操作を行う必要はありません。

このメッセージは、NIS+ のエラーコード定数 NIS_CBRESULTS によって生成されます。詳細については、nis_tables(3N) のマニュアルページを参照してください。

`root_replica_update: update failed` *string*`: could not fetch object from master.`

このメッセージは、複製サーバーへのダンプを行う際に問題が起こったことを示します。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`RPC failure: "RPC failure on yp operation`

このメッセージは、NIS クライアントの `nsswitch.conf` ファイルが、`nis` ではなく `files` に設定され、サーバーが `/etc/hosts` ファイルまたは `/etc/inet/ipnodes` ファイルに含まれていないときに、`ypcat` によって戻されます。

`Security exception on local system. UNABLE TO MAKE REQUEST.`

ユーザーのログイン ID がマシン名と重複したときに、このメッセージが表示されることがあります。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

date: hostname: `sendmail`（*nmnn*）`: gethostbyaddr failed`

この問題の一般的な原因の 1 つに、0 を前に付けた状態での NIS+、NIS、ファイル、DNS データセットの IP アドレスへの入力があります。たとえば、`151.029.066.001` のような IP アドレスは入力しないでください。このアドレスの正確な入力方法は、`151.29.66.1` です。

`Server busy, try again`

サーバーがビジーのため、要求をすぐに処理できません。

- 追加、削除、変更の操作を行なったときにこのメッセージが返されるのは、あるディレクトリのサービスを提供しているマスターサーバーが使用できないか、データベースのチェックポイントが実行中であることを意味します。
- サーバーが内部の状態を更新しているときに、このメッセージが返されることもあります。
- `nis_list` を実行した場合、クライアントがコールバックを指定し、サーバーがリソース不足でコールバックを行えないことも考えられます。

サーバーが使用可能になるのを待って、コマンドをもう一度実行します。

このメッセージは、NIS+ のエラーコード定数 `NIS_TRYAGAIN` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`Server out of memory`

ほとんどの場合、このメッセージは致命的な結果が発生したことを示します。サーバーがヒープ領域を使い果たしたことを意味します。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOMEMORY` によって生成されます。詳細については、`nis_tables(3N)` と `nis_names(3N)` のマニュアルページを参照してください。

`Sorry`
ログイン、またはパスワードの変更が拒否されたときに表示されるメッセージです。拒否の理由が表示されないのは、「表示された理由を読むことで、権利のない人がシステムに不正にアクセスできるようになる可能性がある」というセキュリティ上の理由からです。

`Sorry: less than` *nn* `days since the last change`
「パスワードが新しすぎるため、変更ができない」ということを意味します。パスワードは、作成されてから一定の日数 (*N* 日) が経過するまでは変更ができません。詳細については、297 ページの「パスワードの変更方法」を参照してください。

`_svcauth_des: bad nickname`
クライアントから受け取ったニックネームが不適切か、壊れています。原因は、おそらくネットワークの混雑です。このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは情報を示すだけです。レベルが高い場合は、このコマンドをもう一度実行しなければならない可能性があります。

`_svcauth_des: corrupted window from` *principalname*
送られたウィンドウが、ベリファイアで送られたものと一致しません。

このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。高いセキュリティレベルでは、しばらくたってからコマンドを再実行するか、あるいは次に説明するような修正作業を行う必要があります。

考えられる原因は次のとおりです。

- サーバーの鍵の組が変更されています。クライアントが、サーバーの古い公開鍵を使って、その一方でサーバーが `keyserv` でキャッシュされた新しい非公開鍵を持っています。`keylogin` をクライアントとサーバーの両方で実行してください。
- クライアントの鍵の組が変更されて、クライアントがクライアントシステムで `keylogin` を実行していません。このためシステムはまだ、クライアントの古い非公開鍵をサーバーに送付しており、そのサーバーはクライアントの新しい公開鍵を使用しています。このため、2 つの鍵は一致しません。`keylogin` を再度クライアントとサーバーの両方で実行してください。
- ネットワークでデータが壊れました。コマンドをもう一度実行してください。これで機能しない場合には、`snoop` コマンドを使って、ネットワークの問題を調査、修正してください。その後で、再度クライアントとサーバーの両方で `keylogin` を実行してください。

`_svcauth_des: decryption failure`
特定の認証データの DES 復号化に失敗しました。考えられる原因は次のとおりです。

- ライブラリ関数か引数が壊れています。

- DES 暗号化チップを使用している場合、そのチップに問題があります。

このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。レベルが高い場合は、ご購入先にご連絡ください。問題が、DES 暗号化チップに関連するように思われる場合は、ご購入先に連絡してください。

`_svcauth_des: corrupted window from` *principalname*
送られたウィンドウが、ベリファイアで送られたものと一致しません。

このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。高いセキュリティレベルでは、しばらくたってからコマンドを再実行するか、もしくは次に説明するような修正作業を行う必要があります。

考えられる原因は次のとおりです。

- サーバーの鍵の組が変更されています。クライアントが、サーバーの古い公開鍵を使って、その一方でサーバーが `keyserv` でキャッシュされた新しい非公開鍵を持っています。`keylogin` をクライアントとサーバーの両方で実行してください。
- クライアントの鍵の組が変更されて、クライアントがクライアントシステムで `keylogin` を実行していません。このためシステムはまだ、クライアントの古い非公開鍵をサーバーに送付しており、そのサーバーはクライアントの新しい公開鍵を使用しています。このため、2 つの鍵は一致しません。`keylogin` を再度クライアントとサーバーの両方で実行してください。
- ネットワークでデータが壊れました。コマンドをもう一度実行してください。これで機能しない場合には、`snoop` コマンドを使って、ネットワークの問題を調査、修正してください。その後で、再度クライアントとサーバーの両方で `keylogin` を実行してください。

`_svcauth_des: decryption failure for` *principalname*
特定の認証データの DES 復号化に失敗しました。考えられる原因は次のとおりです。

- ライブラリ関数か引数が壊れています。
- DES 暗号化チップを使用している場合、そのチップに問題があります。

このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。レベルが高い場合は、ご購入先にご連絡ください。問題が、DES 暗号化チップに関連するように思われる場合は、ご購入先に連絡してください。

`_svcauth_des: invalid timestamp received from` *principalname*
クライアントから受け取ったタイムスタンプが壊れています。または、サーバーが間違った鍵を使って、復号しようとしています。考えられる原因は次のとおりです。

- ネットワークの混雑。コマンドをもう一度実行します。

- サーバーのキャッシュから、このクライアント用のエントリが削除されました。ネットワークの負荷をチェックします。

_svcauth_des: key_decryptsessionkey failed for *principalname*
keyserv プロセスは、特定の公開鍵を使用してセッションキーを復号化することに失敗しました。考えられる原因は次のとおりです。

- keyserv プロセスが終了しているか、応答しません。ps -ef を使用して、ホスト上で keyserv プロセスが動作しているかどうか確認します。動作していない場合は、keyserv を再度起動し、keylogin を実行します。
- サーバー主体のキーログインが行われていません。サーバー主体の keylogin を実行します。
- サーバー主体 (ホスト) に資格が割り当てられていません。クライアントのホームドメインにある cred テーブルを対象にして、nismatch *hostname*.*domainname*. cred.org_dir を実行します。必要に応じて、新しい資格を作成します。
- keyserv は再度起動されているようですが、長時間にわたって動作するアプリケーション rpc.nisd、sendmail、automountd なども再度起動する必要があります。
- DES 暗号化に失敗しました。ご購入先にご連絡ください。

_svcauth_des: no public key for *principalname*
サーバーがクライアントの公開鍵を取得できません。考えられる原因は次のとおりです。

- 主体に公開鍵がありません。主体のホームドメインにある cred テーブルを対象に、niscat を実行します。テーブルの中で、この主体に DES の資格が割り当てられていない場合は、nisaddcred を使用して資格を作成し、その主体で keylogin を実行します。
- nsswitch.conf ファイルで指定されたネームサービスが応答しません。

_svcauth_des: replayed credential from *principalname*
サーバーは要求を受け取り、キャッシュの中でそのクライアント名に一致するエントリを見つけましたが、受け取った要求の会話鍵のタイムスタンプが、キャッシュの中に格納されているタイムスタンプより古いものでした。

このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。レベルが高い場合は、次の説明に従って問題を解決するための操作を行なってください。

考えられる原因は次のとおりです。

- クライアントとサーバーそれぞれのクロックが同期していません。rdate を使用して、クライアントのクロックをサーバーのクロックに同期させます。

- サーバーがランダムな順序で要求を受け取っています。マルチスレッドアプリケーションを使用している場合に、これが発生します。アプリケーションが tcp をサポートしている場合、/etc/netconfig (または NETPATH 環境変数) を tcp に設定します。

_svcauth_des: timestamp is earlier than the one previously seen from *principalname*

クライアントからいくつかのコールを受け取りましたが、後から受け取ったコールが、最初のコールより古いタイムスタンプを示しています。このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。レベルが高い場合は、次の説明に従って問題を解決するための操作を行なってください。

考えられる原因は次のとおりです。

- クライアントとサーバーそれぞれのクロックが同期していません。rdate を使用して、クライアントのクロックをサーバーのクロックに同期させます。
- サーバーのキャッシュから、このクライアント用のエントリが削除されました。サーバーはキャッシュの中に、現在のクライアントの情報を維持しています。キャッシュのサイズは、64 個のクライアントに相当します。

_svcauth_des: timestamp expired for *principalname*

ウィンドウを受信する際、デフォルトでは 35 秒以内の間隔が規定されていますが、クライアントから受け取ったタイムスタンプが、この範囲に収まっていません。このメッセージの重大度は、有効にしているセキュリティのレベルに依存します。セキュリティレベルが低い場合は、このメッセージは主に情報を示すために表示されます。レベルが高い場合は、次の説明に従って問題を解決するための操作を行なってください。

考えられる原因は次のとおりです。

- サーバーやネットワークが低速なので、35 秒のウィンドウでは間隔が不足しています。
- クライアントとサーバーそれぞれのクロックがかなりずれているため、ウィンドウの間隔に収まりません。rdate を使用して、クライアントのクロックをサーバーのクロックに同期させます。
- サーバーのキャッシュから、このクライアント用のエントリが削除されました。操作をもう一度実行します。

Too Many Attributes

サーバーに渡された検索基準が、検索可能なテーブルの列より多くの属性を持っています。

このメッセージは、NIS+ のエラーコード定数 NIS_TOOMANYATTRS によって生成されます。詳細については、nis_tables(3N) のマニュアルページを参照してください。

`Too many failures - try later`

`Too many tries; try again later`
「ログイン、またはパスワードの変更のやり直しの回数が多すぎる、あるいは時間がかかりすぎる」ということを意味します。詳細については、296 ページの「`Login incorrect` メッセージ」または、298 ページの「パスワード変更の失敗」を参照してください。

`Unable to authenticate NIS+ client`
サーバーがクライアントのコールバック要求を実行したときに、RPC の `clnt_call()` から `RPC_AUTHERR` という状態を受け取ると、このメッセージが生成されます。通常これは、古い認証情報が原因となっています。システムが、更新されていないキャッシュから取り出したデータを使用した場合や、認証情報が最近変更され、その変更結果がサーバーに伝わっていない場合に、このような古い認証情報が発生します。ほとんどの場合、この問題は短時間のうちに自動的に解決されます。

この問題が自動的に解決されない場合は、次のいずれかの問題があると考えられます。

- `/var/nis/NIS_SHARED_DIRCACHE` ファイルが壊れています。キャッシュマネージャのプロセスを終了させ、このファイルを削除し、キャッシュマネージャを再度起動します。
- `/var/nis/NIS_COLD_START` ファイルが壊れています。このファイルを削除し、`nisinit` を実行して、このファイルを再度作成します。
- `/etc/.rootkey` ファイルが壊れています。`keylogin -r` を実行します。

このメッセージは、NIS+ のエラーコード定数 `NIS_CLNTAUTH` によって生成されます。

`Unable to authenticate NIS+ server`
ほとんどの場合、システムが容易に修正することのできる、ソフトウェアの小さなエラーが発生したことを意味しています。サーバーが `RPC` の `clnt_call` から `RPC_AUTHERR` という状態を受け取ると、このメッセージが生成されます。

この問題が自動的に解決されない場合は、`/var/nis/NIS_COLD_START` ファイル、`/var/nis/NIS_SHARED_DIRCACHE` ファイル、`/etc/.rootkey` ファイルのどれかが壊れていることが考えられます。

このメッセージは、NIS+ のエラーコード定数 `NIS_SRVAUTH` によって生成されます。

`Unable to bind to master server for name '`*`string`*`'`
これらの問題については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。このメッセージは、`/etc/defaultdomain` ファイルの中にあるサーバーのドメイン名の右端にピリオドを追加したときに発生することがあります。

`Unable to create callback.`
サーバーは、ユーザーのマシンのコールバックサービスと通信できませんでした。この結果、データを返すことができませんでした。

詳細については、`nis_tables(3N)` のマニュアルページを参照してください。

`Unable to create process on server`
NIS+ のサービスルーチンが、サポートしていない手続き番号を指定する要求を受け取ると、このエラーが生成されます。

このメッセージは、NIS+ のエラーコード定数 `NIS_NOPROC` によって生成されます。

string`: Unable to decrypt secret key for` *string*`.`
考えられる原因は次のとおりです。

- 正しくないパスワードを入力しました。
- `cred` テーブルの中に、*name* のエントリがありません。
- NIS+ が鍵を復号化できませんでした。おそらく、エントリが壊れている可能性があります。
- `nsswitch.conf` ファイルが、`cred` テーブルの中に記録されている NIS+ パスワードとは異なるパスワードを使用して、`/etc/passwd` ファイルの中に格納されているローカルパスワードの照会を行なっています。

これらの問題を診断および解決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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`Unknown error`
NIS+ のエラー処理ルーチンが、未登録のタイプのエラーを受け取ったときに、このメッセージが表示されます。

`Unknown object`
返されたオブジェクトのタイプが、未登録のタイプです。

このメッセージは、NIS+ のエラーコード定数 `NIS_UNKNOWNOBJ` によって生成されます。詳細については、`nis_names` のマニュアルページを参照してください。

`update_directory:` *nnnn* `objects still running`
複製を更新するときに、サーバーがディレクトリを更新している間、表示されるメッセージです。なんらかの操作を行う必要はありません。

`User` *principalname* `needs Secure RPC credentials to login but has none.`
キーログインが成功しなかったことを示します。この問題は一般に、`/etc/shadow` とリモートの NIS+ パスワードテーブルとで記憶しているパスワードが異なっているために発生します。

警告 : couldn't reencrypt secret key for *principalname*
この問題の原因として最も可能性が高いのは「Secure RPC パスワードがログインパスワードと異なっていて (またはローカルの /etc/shadow とリモートの NIS+ パスワードテーブルとで記憶しているパスワードが異なっていて)、まだ明示的に keylogin を実行していない」ということです。この種類の問題の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

WARNING: db::checkpoint: could not dump database: No such file or directory
このメッセージは、チェックポイント実行の際に、システムがデータベースファイルをオープンできなかったことを示しています。考えられる原因は次のとおりです。

- データベースファイルが削除されています。
- サーバーがファイル記述子を使い果たしました。
- ディスクに問題があります。
- ユーザーまたはホストに、必要なアクセス権が割り当てられていません。

WARNING: db_dictionary::add_table: could not initialize database from scheme
データベーステーブルを初期設定できませんでした。考えられる原因は次のとおりです。

- システムリソースに問題があります。『 *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「NIS の障害追跡」を参照してください。
- コマンド構文での新しいテーブルの指定が正しくありませんでした。
- データベースが壊れています。

WARNING: db_query::db_query:bad index
ほとんどの場合、このメッセージは、インデックス付きの名前の指定が正しくないことを示しています。指定されたテーブルの中に、インデックス付きの名前が存在するかどうか確認します。コマンドのスペルと構文にエラーがないかチェックします。

**WARNING: domain *domainname* already exists
このメッセージは、作成しようとしたドメインがすでに存在することを意味しています。

- ルート以外のマスターサーバーを新しく変換しようとしている場合や、直前に起こった nisserver の問題を回復しようとしている場合は、このスクリプトを続けて実行します。
- *domainname* のスペルが正しくない場合は、正しいドメイン名を指定して、このスクリプトをもう一度実行します。

**WARNING: failed to add new member *NIS+_principle* into the *groupname* group.You will need to add this member manually: 1. /usr/sbin/nisgrpadm -a groupname NIS+_principal

NIS+ のコマンド nisgrpadm が、NIS+ のグループ *groupname* に新しいメンバーを追加することに失敗しました。次のように入力して、NIS+ のこの主体を追加します。

```
# /usr/sbin/nisgrpadm -a groupname NIS+_principal
```

**WARNING: failed to populate *tablename* table

nisaddent コマンドは、NIS+ の tablename テーブルをロードできませんでした。通常、この警告メッセージの前に、詳細なエラーメッセージが表示されます。

**WARNING: hostname specified will not be used. It will use the local hostname instead.

このメッセージは、-H オプションを指定して、リモートホスト名を入力したことを示します。nisserver -r スクリプトを使って、ルートマスターサーバーと同様の方法でリモートマシンを構成できません。

- ローカルマシンを NIS+ のルートマスターサーバーに変換したい場合は、別に操作を行う必要はありません。nisserver -r スクリプトは、入力されたホスト名を無視します。
- リモートホスト (ローカルマシンではなく) を NIS+ のルートマスターサーバーに変換したい場合は、このスクリプトを終了します。リモートホストを対象にして、nisserver -r を再度実行します。

**WARNING: *hostname* is already a server for this domain. If you choose to continue with the script, it will try to replicate the groups_dir and org_dir directories for this domain.

このメッセージは、複製サーバーを作成しようとしたドメインの中で、すでに *hostname* が複製サーバーとなっていることを警告します。

- nisserver によって以前に発生した問題を解決するためにこのスクリプトを実行している場合は、スクリプトを続けて実行します。
- *hostname* を間違って入力した場合は、正しいホスト名を指定して、このスクリプトをもう一度実行します。

**WARNING: *alias-hostname* is an alias name for host *canonical_hostname* You cannot create credential for host alias.

このメッセージは、nisclient -c の名前リストに、ホストの別名を入力したことを示します。ホストの別名に資格を作成すべきではないので、標準的なホスト名にその資格を作成するかどうか、スクリプトが尋ねています。

**WARNING: file *directory-path/tablename* does not exist!*tablename* table will not be loaded.

このスクリプトは、*tablename* の入力ファイルを見つけることができませんでした。

- *directory-path*/*tablename* が間違って入力されている場合は、正しいテーブル名を指定して、このスクリプトをもう一度実行します。
- *directory-path*/*tablename* が存在しない場合は、適切なデータを使ってファイルを作成し更新します。次に、このスクリプトを再度実行して、このテーブルを生成します。

`**WARNING: NIS auto.master map conversion failed. auto.master table will not be loaded.`

`auto.master` テーブル内のすべてのピリオドを下線に変換しているときに、`auto.master` マップの変換に失敗しました。別の NIS サーバーを使って、このスクリプトをもう一度実行します。

`**WARNING: NIS netgroup map conversion failed. netgroup table will not be loaded.`

`netgroup` マップの中で NIS のドメイン名を NIS+ のドメイン名に変換しているときに、`netgroup` マップの変換に失敗しました。別の NIS サーバーを使って、このスクリプトをもう一度実行します。

`**WARNING: nisupdkeys failed on directory` *domainname* `This script will not be able to continue. Please remove the` *domainname* `directory using 'nisrmdir'.`

NIS+ のコマンド `nisupdkeys` は、リスト指定されたディレクトリオブジェクト内の鍵を更新することに失敗しました。新しいドメインにサービスを提供することになっている新しいマスターサーバー上で `rpc.nisd` が動作していない場合は、`rpc.nisd` を再度起動します。次に、`nisrmdir` を使用して、*domainname* ディレクトリを削除します。最後に、`nisserver` を再度実行します。

`WARNING: nisupdkeys failed on directory` *directory-name* `You will need to run nisupdkeys manually: 1. /usr/lib/nis/nisupdkeys` *directory-name*

NIS+ のコマンド `nisupdkeys` は、リスト指定されたディレクトリオブジェクト内の鍵を更新することに失敗しました。次のように入力して、ディレクトリオブジェクト内の鍵を手作業で更新します。

```
# /usr/lib/nis/nisupdkeys directory-name
```

`**WARNING: once this script is executed, you will not be able to restore the existing NIS+ server environment. However, you can restore your NIS+ client environment using "nisclient -r" with the proper domainname and server information. Use "nisclient -r" to restore your NIS+ client environment.`

これらのメッセージは、このスクリプトを以前に実行して NIS+ サーバーをすでに設定している場合に表示されます。このスクリプトの実行を続行すると、必要に応じて NIS+ 関連のファイルが削除および再作成されることを示しています。

- NIS+ のファイルを削除してもよい場合は、このスクリプトを続けて実行します。

- NIS+ のファイルを保存するには、`Do you want to continue?` プロンプトで `n` と入力し、このスクリプトを終了します。次に、NIS+ のファイルを別のディレクトリに保存し、このスクリプトをもう一度実行します。

`**WARNING: this script removes directories and files related to NIS+ under /var/nis directory with the exception of the NIS_COLD _START and NIS_SHARED_DIRCACHE files which will be renamed to <file>.no_nisplus. If you want to save these files, you should abort from this script now to save these files first.`

すぐ上の「WARNING:once this script is executed,...」のメッセージを参照してください。

`**WARNING: you must specify the NIS domainname.`

このメッセージは、プロンプトで NIS のドメイン名を入力しなかったことを示しています。プロンプトで、NIS サーバーのドメイン名を入力します。

`**WARNING: you must specify the NIS server` *hostname*`. Please try again.`

このメッセージは、プロンプトで NIS のサーバーのホスト名を入力しなかったことを示しています。プロンプトで、NIS サーバーのホスト名を入力します。

`Window verifier mismatch`

これは、`_svcauth_des` コードによって生成されたデバッグ用メッセージです。鍵がキャッシュから削除されたため、ベリファイアが無効になっている可能性があります。このメッセージが生成されると、`_svcauth_des` は、`AUTH_BADCRED` の状態を返します。

`You (`*string*`) do not have Secure RPC credentials in NIS+ domain '`*string*`'`

コマンドを実行するために必要とされる資格が割り当てられていないサーバーで `nispasswd` を実行すると、このメッセージが生成されることがあります。セキュリティレベル 0 で動作しているサーバーは、資格の作成や維持を行わないことに注意してください。

資格、所有権、およびアクセス権の詳細については、『*Solaris* のシステム管理 *(*ネーミングとディレクトリサービス *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

`You may not change this password`

パスワードの変更が管理者によって禁止されていることを意味します。

`You may not use nisplus repository`

「コマンド行で `-r nisplus` が使用されたが、NIS+ パスワードテーブルに適切なエントリがない」ということを意味します。パスワードテーブルに必要なエントリがあるかどうか確認してください。また `nsswitch.conf` ファイルに `nisplus` というエントリを追加してみてください。

`Your password has been expired for too long`

`Your password is expired`
上記の 2 つのメッセージは、パスワードの有効期間に関するものです。パスワードの有効期間を過ぎているため、すぐに変更する必要があることを示しています。詳細については、296 ページの「`will expire` メッセージ」 を参照してください。

`Your password will expire in` *nn* `days`

`Your password will expire within 24 hours`
上記の 2 つのメッセージは、パスワードの有効期間に関するものです。パスワードがまもなく期限切れになるため、すぐに変更する必要があることを示しています。詳細については、296 ページの「`will expire` メッセージ」を参照してください。

`Your specified repository is not defined in the nsswitch file!`
「`-r` オプションでリポジトリのパスワード情報を更新しようとしたが、該当するパスワードのリポジトリが `nsswitch.conf` ファイルのパスワードエントリにない」という意味の警告メッセージです。使用したコマンド自体は機能し、`-r` フラグで指定したリポジトリへの変更はすべて行われます。しかし変更の対象となっているリポジトリが `nsswitch.conf` ファイルで指定されているものではないため、該当するリポジトリを `nsswitch.conf` ファイルで指定するまでこの変更の影響はまったく表れません。

たとえば、このスイッチファイルの passwd エントリに、 `files nis` と指定されているとします。このファイルに対して

```
passwd -r nisplus
```

を使用してパスワードの有効期限を設定しようとしても、無視されます。スイッチファイルの設定は、`files nis` のままです。

`verify_table_exists: cannot create table for` *string* `nis_perror` *message*
テーブルを対象にして操作を行う場合、NIS+ は最初にテーブルが存在するかどうかを確認します。テーブルが存在しない場合は、NIS+ はそのテーブルを作成しようとします。作成できない場合は、このエラーメッセージを返します。*string* は、検索や作成ができなかったテーブルを示します。`nis_perror` *message* の部分は、問題の原因に関係する情報を示します。メッセージのこの部分を、独立したメッセージと考えて、この付録にある説明を参照できます。このタイプの問題の考えられる原因は次のとおりです。

- サーバーはディレクトリの複製として追加されるだけであり、ディレクトリオブジェクトを持つことはできません。`nisping -C` により、チェックポイントを実行します。
- ディスク容量を使い果たしました。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 *: DNS*、*NIS*、*LDAP* 編*)*』の「NIS の障害追跡」を参照してください。

- データベースが壊れています。
- その他のタイプのソフトウェアエラー。ご購入先にご連絡ください。

`ypcat: can't bind to NIS server for domain` *domainname*`. Reason: can't communicate with ypbind.`
『*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「NIS の障害追跡」を参照してください。

`yppoll: can't get any map parameter.`
『*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「NIS の障害追跡」を参照してください。

用語集

BNF FNS (XFN) の用語。Backus-Naur の頭文字

data encryption standard (DES) (データ暗号化規格**)** 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。

DES 「data encryption standard (DES) (データ暗号化規格)」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、1 つまたは複数のサブドメインから構成されるのが普通。

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・ネーミング・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。

MIS 経営情報システム

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+ オブジェクト　NIS+ のドメイン、ディレクトリ、テーブル、グループのこと。「ドメイン」、「ディレクトリ」、「グループ」、「テーブル」の項を参照。

NIS+ 環境　管理上使用する用語。「該当する `nsswitch.conf` ファイルで、`nisplus` が情報源として指定されている」という状況を指す。また「コマンドが、NIS+ 名前空間のオブジェクトに操作をするようなオプションを付けて実行されている」という状況 (例：`passwd -r nisplus`) を指すこともある。

NIS+ 主体　「主体」の項を参照。

NIS+ トランザクションログ　名前空間内のオブジェクトに関する NIS+ テーブル宛のデータ更新が収められたファイル。名前空間内の変更は、複製サーバーに伝達されるまでは、トランザクションログに格納される。トランザクションログは、マスターサーバーの複製サーバーがすべて更新された後で、消去される。

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、たとえば、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。

NNSP　「次のネーミングシステム参照」の項を参照。

ping　NIS+ データに加えられた変更の内容が、NIS+マスターサーバーから当該ドメインの複製サーバーに送られるプロセス。

RPC　「遠隔手続き呼び出し (RPC)」の項を参照。

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。

TCP　「Transport Control Protocol (TCP)」の項を参照。

TCP/IP　Transport Control Protocol/Internet Protocol の頭文字。このプロトコル体系は、最初はインターネット用に開発された。「インターネット」プロトコル体系とも呼ばれる。Solaris ネットワークは、デフォルトでは TCP/IP 上で動作する。

Transport Control Protocol (TCP)　インターネットプロトコル体系での主要な転送プロトコルであり、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。配送には IP を使用する。「TCP/IP」の項を参照。

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。

XFN リンク	複合名でアドレスを示す特殊な形式のリファレンス。他の種類のリファレンスと同様に、XFNリンクはコンテキストの原子名と結合される。
アクセス権	NIS+ 主体 (principal) のクラスに割り当てられるアクセス権であり、NIS+ オブジェクト上でこれらのクラスが実行できる動作を決定する。読み取り、変更、作成、または 削除の 4 種類がある。
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。
暗号化鍵	「データ暗号化鍵」の項を参照。
暗黙的なネームシステムポインタ	FNS (XFN) の用語。他のネームシステムのコンテキストを指す、名前のないリファレンスのこと。
インターネット	一連のルーターによって相互接続された世界規模のネットワークの集まり。これらのネットワークは 1 つの大きな仮想ネットワークとして機能できる。
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、要求がリモートシステムに送信され、指定された手順が実行される。そして、その結果が呼び出し側に返される。
エンタープライズのルート	エンタープライズのルートコンテキストのこと。エンタープライズの名前空間のルートにあるオブジェクトの名前を管理する。
エンタープライズレベルのネームサービス	エンタープライズレベルネットワーク内のマシン (ホスト) 名、ユーザー名、ファイル名を管理するサービスのこと。FNS でも、組織ユニット、地理的なサイト、アプリケーションサービスのネーミングが可能です。
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) にできます。または、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 にできます。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。

親コンテキスト　あるコンテキストとその兄弟的なコンテキストが存在するコンテキスト。

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　非公開鍵を格納する Solaris オペレーティング環境のプロセス。

キー **(**列**)**　NIS+ テーブルエントリのデータは、そのテーブルのキーとは無関係に、どの列からでもアクセスできる。

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) NIS+ では、NIS+ サーバーに対して NIS+ サービスを要求する主体 (マシンまたはユーザー) がクライアント。

(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、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がクライアント。

(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、「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」がクライアント。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または別のマシン上で動作できる。

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。これらの例としては、「ドメイン名システム (DNS)」のネームサーバー / ネームリゾルバパラダイム、および「NFS」やディスクレスホストなどのファイルサーバー / ファイルクライアント関係がある。「クライアント」の項も参照。

グループ　(1) 共通の名前で参照されるユーザーの集り。

(2) NIS+ では、NIS+ オブジェクトに共通のアクセス権を与えられたユーザーの集まり。NIS+ グループの情報は、NIS+ グループテーブルに格納されている。

(3) UNIX では、ユーザーのファイルへのアクセスを決定する。デフォルトユーザーグループと標準ユーザーグループの 2 つがある。

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。

グローバルコンテキスト	FNS (XFN) で、オブジェクトのグローバル名を管理するコンテキスト (現在、XFNで指定されているグローバルコンテキストは、DNS と X.500 のみ)。
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をも識別することができる。
原子名	名前表記規則によって決められた最小かつ不可分の名前要素のこと。FNS (XFN) の用語。
広域ネットワーク **(WAN)**	地理的に離れた複数の LAN (Local Area Network) 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、非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。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、固有の公開鍵と非公開鍵が 1 対ある。
合成名 **(compound name)**	ネームシステムの名前表記規則に従って複数の原子名を並べて構成した名前。
コールドスタートファイル	クライアントが初期設定されたときにクライアントに与えられる NIS+ ファイル。クライアントがホームドメイン内のマスターサーバーと通信を開始できるだけの情報が収められている。
子ドメイン	「ドメイン」の項を参照。
コンテキスト	異なる複数の原子名との割り当てを状態として持つオブジェクトのこと。どのコンテキストもそれに対応する名前表記規則を持つ。各コンテキストでは、ルックアップ (名前解決) 操作でリファレンスを返す。その他、名前の結合、切り離し、結合された名前を一覧表示するなどの操作も実行できる場合もある。
サーバー	(1) NIS+ 、NIS 、DNS 、FNS (XFN) 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、サーバーとは計算資源 (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リモートアクセスが可能であり、これらの資源を使用できる。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。

サービスコンテキスト　FNS (XFN) では、サービスを提供するオブジェクトの名前を管理するコンテキストのこと。

サイトコンテキスト　物理的なサイトに関連するオブジェクトの名前を管理するコンテキスト。

サブコンテキスト　他のコンテキストに属するコンテキスト。

サブネット　ルーティングを簡単にするため、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　NIS+ 主体 (principal) についての認証情報。クライアントソフトウェアは、各要求と一緒にこの情報を NIS+ サーバーに送信する。この情報によって、ユーザーまたはマシンの確認検査が行われる。

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。

主体　NIS+ 情報のユーザーのうち、名前空間に資格が格納されているものを指す。また NIS+ サーバーに対してなんらかのリクエストを行うユーザー、マシンを指す。クライアントユーザー、クライアントマシンの 2 種類がある。

- 「ルート主体」マシンの root ユーザー (ユーザー ID = 0) のこと。DES資格だけ必要。
- 「ユーザー主体」root を除くすべてのユーザー (ユーザー ID > 0)。ローカル資格、および DES 資格が必要。

初期コンテキスト　FNS (XFN) では、すべての XFN 名が何らかのコンテキストに関連付けられる形で解釈され、どの XFN 操作もコンテキストオブジェクト上で行われる。XFN インタフェースには、複合名解決の出発点となる初期コンテキストオブジェクトをクライアント側で得るための関数が用意されている。

初期コンテキスト関数　FNS 関数の 1 つ、`fn_ctx_handle_from_initial()` のこと。これを実行すると、名前解決の出発点となる初期コンテキストをクライアント側で得ることができる。

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ある。

(2) スレーブサーバーは、NIS+ では「複製サーバー」と呼ばれる。

接続　FNS (XFN) の用語。ある名前空間にあって、別のネーミングシステムのコンテキストに結合されている名前。

属性　FNS (XFN) では、名前の付いている個々のオブジェクトに対して、ゼロないし複数の属性が割り当てられる。個々の属性は一意の識別子、構文、ゼロないし複数の個別の値を持つ。

組織単位
: エンタープライズは本部、研究所、部、課などで構成される。エンタープライズのサブユニットが組織単位である。

組織単位のコンテキスト
: エンタープライズ内の組織単位に関連するオブジェクト名を管理するコンテキスト。

チェックポイントの実行
: NIS+ データセットの変更 (サーバーのメモリに格納され、トランザクションログに記録される) をディスク上の NIS+ テーブルに書き込む一連のプロセス。つまり、NIS+ データセットの変更を反映して NIS+ テーブルを更新すること。

次のネーミングシステム参照 **(NNSP)**
: 下位のネーミングシステムで生成された複合名を解決するコンテキスト。

強い分離
: XFN コンテキストが XFN コンポーネントセパレータをネーミングシステムの境界として扱うこと。

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。

ディレクトリ
: (1) NIS+ においては、NIS+ オブジェクトのコンテナのこと。NIS+ テーブル、グループ、サブディレクトリなどを指す。

  (2) UNIX においては、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。

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。

テーブルの生成 **(populate)**
: ファイルまたは NIS マップから NIS+ テーブルにデータを入れること。

ドット表記
: 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。

ドメイン名サービス (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。すなわち 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ポリシーとメカニズムを提供する。

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、そして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。

(3) NIS+ 名前空間 - NIS+ ソフトウェアによって使用される階層型ネットワーク情報の集まり。

(4) NIS 名前空間 - NIS ソフトウェアによって使用される非階層型ネットワーク情報の集まり。

(5) DNS 名前空間 - DNS ソフトウェアを使用するネットワークワークステーションの集まり。

名前空間識別子 — FNS (XFN) の用語。名前空間のルートを表す特殊な名前のこと。

認証 — ある NIS+ サーバーが、NIS+ 名前空間に対するアクセス要求の送信者を識別できるかどうかの判定。認証された要求は、所有者、グループ、およびその他という承認カテゴリに分類される。送信者が識別できない未認証要求は、未承認カテゴリに入れられる。

ネーミング規則 — FNS (XFN) では、すべての名前が構文ルールによって生成される。「ネーミング規則」は、これら構文ルールの総称。

ネーミングシステム — 接続された同種の複数のコンテキスト (同じネーミング規則を持ち、同じ意味に対して同じ操作を行う) の総称。たとえば、UNIX では、

	ファイルシステム内のディレクトリ群 (ディレクトリ上での名前管理操作を含む) がネーミングシステムを構成する。
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	NIS+ 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`)。
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。
汎用コンテキスト	FNS (XFN) の用語。アプリケーションで使われる名前を結合させるコンテキスト。
非公開鍵	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、固有の公開鍵と非公開鍵が 1 対ある。
フェデレーテッド・ネーミング・サービス	フェデレーテッド・ネーミング・システムが提供するサービスのこと。
フェデレーテッド・ネーミング・システム	複合名を標準のインタフェースで解決できるようにするため、複数の自律的なネーミングシステムを集めてネーミングシステム同士を連係させたもの。連合内の各ネーミングシステムは、名前解決以外の操作については、自律的に選択できる。
フェデレートされた名前空間	FNS (XFN) の用語。連合を作っているネーミング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。
複合名 **(composite name)**	複数のネーミングシステムが使われている名前のこと。ゼロまたは複数の要素が順番に並んだ構成になっている。各構成要素は、単一のネーミングシステムの名前空間からとられた名前です。「複合名の解決」といえば、複数のネーミングシステムが使われている名前を解決することを指す。
複製サーバー	NIS+ サーバーのうち、ドメインのマスター NIS+ サーバーデータベースの複製を持つもの。複製サーバーは、NIS+ サーバーソフトウェアを実行し、NIS+ テーブルのコピーを管理する。複製サーバーによって、NIS+ サービスの可用性が向上する。NIS+ ドメインには個々に 1 つ以上の複製サーバーが必要である (「複製サーバー」は、NIS 名前空間では「スレーブサーバー」と呼ばれる)。
ホストコンテキスト	あるコンピュータに関連するオブジェクトの名前を管理するコンテキスト。

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レシーバとして機能するワークステーション。
ユーザーコンテキスト	ユーザーに関連するオブジェクトの名前を管理するコンテキスト。
優先サーバー	クライアントマシンから見て、優先 NIS+ サーバーとは、そのクライアントが名前空間に関する情報を得る際に、他のサーバーより先に照会を試みるサーバーのことをいう。あるクライアントまたはドメインの優先サーバーリストにあるサーバーは、そのクライアントまたはドメインの優先サーバーであると考えられる。
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。
優先順位の番号	クライアントマシンが名前空間に関する情報を取得する際に、どの NIS+ サーバーから順に照会するかを示す番号。マシンは、ある優先順位番号を持つすべてのサーバーに要求を送ったあと、その次に高いランク番号を持つサーバーに要求を送る。たとえば、ランク番号 0 の NIS+ サーバーに要求を送った後、ランク番号 1 のサーバーに要求を送る。
弱い分離	XFN コンテキストが XFN コンポーネントセパレータをネーミングシステムの境界としては扱わないこと。
リファレンス	ある名前を指し示すもの。具体的には、オブジェクト通信の終端を示すアドレスなど。
ルートコンテキスト	名前空間のルートにあってオブジェクト名を管理するコンテキスト。
ルートドメイン	「ドメイン」の項を参照。
ルート複製サーバー	NIS+ サーバーのうち、ルートドメインのマスター NIS+ サーバーデータベースの複製を持つものを指す。
ルートマスターサーバー	NIS+ ルートドメインのマスターサーバーのこと。
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、交換の目的で接続したもの。
割り当て **(**バインド**)**	原子名とオブジェクトリファレンス (オブジェクト) を関連付けること。簡素化のため、本書では「オブジェクトリファレンス」とそれが表す「オブジェクト」を同じ意味で使うことがある。

索引

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

あ

い

え

お

け

こ

さ

し

す

そ

と