Working With Oracle® Solaris 11.4 Directory and Naming Services: DNS and NIS

Part No: E61011
November 2020

Working With Oracle Solaris 11.4 Directory and Naming Services: DNS and NIS

Part No: E61011

License Restrictions Warranty/Consequential Damages Disclaimer

Warranty Disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

Hazardous Applications Notice

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

Third-Party Content, Products, and Services Disclaimer

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Pre-General Availability Draft Label and Publication Date

Pre-General Availability: 2020-01-15

Pre-General Availability Draft Documentation Notice

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

Oracle Confidential Label

ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties.

Revenue Recognition Notice

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Référence: E61011

Restrictions de licence/Avis d'exclusion de responsabilité en cas de dommage indirect et/ou consécutif

Ce logiciel et la documentation qui l'accompagne sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d'utilisation et de divulgation. Sauf stipulation expresse de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, accorder de licence, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d'interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Exonération de garantie

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu'elles soient exemptes d'erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Avis sur la limitation des droits

Si ce logiciel, ou la documentation qui l'accompagne, est livré sous licence au Gouvernement des Etats-Unis, ou à quiconque qui aurait souscrit la licence de ce logiciel pour le compte du Gouvernement des Etats-Unis, la notice suivante s'applique :

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

Avis sur les applications dangereuses

Ce logiciel ou matériel a été développé pour un usage général dans le cadre d'applications de gestion des informations. Ce logiciel ou matériel n'est pas conçu ni n'est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer un risque de dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d'applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l'utilisation de ce logiciel ou matériel pour des applications dangereuses.

Marques

Oracle et Java sont des marques déposées d'Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d'autres propriétaires qu'Oracle.

Intel et Intel Inside sont des marques ou des marques déposées d'Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées de SPARC International, Inc. AMD, Epyc, et le logo AMD sont des marques ou des marques déposées d'Advanced Micro Devices. UNIX est une marque déposée de The Open Group.

Avis d'exclusion de responsabilité concernant les services, produits et contenu tiers

Ce logiciel ou matériel et la documentation qui l'accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers, sauf mention contraire stipulée dans un contrat entre vous et Oracle. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'accès à des contenus, produits ou services tiers, ou à leur utilisation, sauf mention contraire stipulée dans un contrat entre vous et Oracle.

Date de publication et mention de la version préliminaire de Disponibilité Générale ("Pre-GA")

Version préliminaire de Disponibilité Générale ("Pre-GA") : 15.01.2020

Avis sur la version préliminaire de Disponibilité Générale ("Pre-GA") de la documentation

Si ce document est fourni dans la Version préliminaire de Disponibilité Générale ("Pre-GA") à caractère public ou privé :

Cette documentation est fournie dans la Version préliminaire de Disponibilité Générale ("Pre-GA") et uniquement à des fins de démonstration et d'usage à titre préliminaire de la version finale. Celle-ci n'est pas toujours spécifique du matériel informatique sur lequel vous utilisez ce logiciel. Oracle Corporation et ses affiliés déclinent expressément toute responsabilité ou garantie expresse quant au contenu de cette documentation. Oracle Corporation et ses affiliés ne sauraient en aucun cas être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'utilisation de cette documentation.

Mention sur les informations confidentielles Oracle

INFORMATIONS CONFIDENTIELLES ORACLE. Destinées uniquement à un usage autorisé. Ne pas distribuer à des tiers.

Avis sur la reconnaissance du revenu

Si ce document est fourni dans la Version préliminaire de Disponibilité Générale ("Pre-GA") à caractère privé :

Les informations contenues dans ce document sont fournies à titre informatif uniquement et doivent être prises en compte en votre qualité de membre du customer advisory board ou conformément à votre contrat d'essai de Version préliminaire de Disponibilité Générale ("Pre-GA") uniquement. Ce document ne constitue en aucun cas un engagement à fournir des composants, du code ou des fonctionnalités et ne doit pas être retenu comme base d'une quelconque décision d'achat. Le développement, la commercialisation et la mise à disposition des fonctions ou fonctionnalités décrites restent à la seule discrétion d'Oracle.

Accessibilité de la documentation

Pour plus d'informations sur l'engagement d'Oracle pour l'accessibilité de la documentation, visitez le site Web Oracle Accessibility Program, à l'adresse : `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

Contents

Using This Documentation

- **Overview** – Describes the DNS and NIS naming services, methods for planning their use, and steps to implement DNS and NIS.
- **Audience** – System administrators.
- **Required knowledge** – Familiarity with naming service concepts and terminologies that refer to DNS and NIS.

Product Documentation Library

Documentation and resources for this product and related products are available at `http://www.oracle.com/pls/topic/lookup?ctx=E37838-01`.

Feedback

Provide feedback about this documentation at `http://www.oracle.com/goto/docfeedback`.

♦ ♦ ♦ CHAPTER 1

About Naming and Directory Services

This chapter provides an overview of naming and directory services included in the Oracle Solaris 11 release. It also briefly describes DNS, NIS, and LDAP naming services.

This chapter contains the following topics:

- "What Is a Naming Service?" on page 13
- "Oracle Solaris Naming Services" on page 19
- "Comparing the Naming Services" on page 23

What Is a Naming Service?

A *naming service* performs lookups of stored information, such as:

- Access permissions
- Automount maps
- Group membership
- Host names and addresses
- Passwords
- User names

This information enables users to log in to their system, access resources, and be granted permissions. You can store name service information in files, maps, or various forms of database files. These information repositories can be local to the system or located in a central network-based repository or database.

Note - This documentation is about configuring the various naming services. Selection of the active naming service is done using profiles. For more information, see Chapter 5, "Administering External Network Modifiers in Oracle Solaris" in *Configuring and Managing Network Components in Oracle Solaris 11.4*.

Without a central naming service, each system would have to maintain its own copy of this information. If you centralize all data, administration becomes easier.

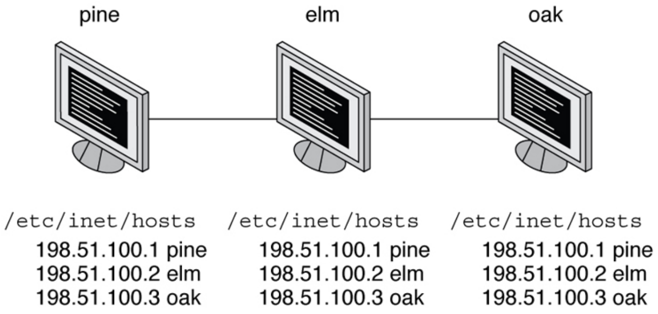
Naming services are fundamental to any computing network. Among other features, naming services provide functionality that performs the following actions:

- Associating (*binding*) names with objects
- Listing names
- Removing bindings
- Renaming information
- Resolving names to objects

A network information service enables systems to be identified by common names instead of numerical addresses. Communication is simpler because users do not have to remember and try to enter numerical addresses such as `192.0.2.0`.

For example, suppose you have a network of three systems that are named `pine`, `elm`, and `oak`. Before `pine` can send a message to either `elm` or `oak`, `pine` must know their numerical network addresses. For this reason, `pine` keeps a file, `/etc/inet/hosts`, that stores the network address of every system in the network, including itself. Likewise, in order for `elm` and `oak` to communicate with `pine` or with each other, the systems must keep similar files.

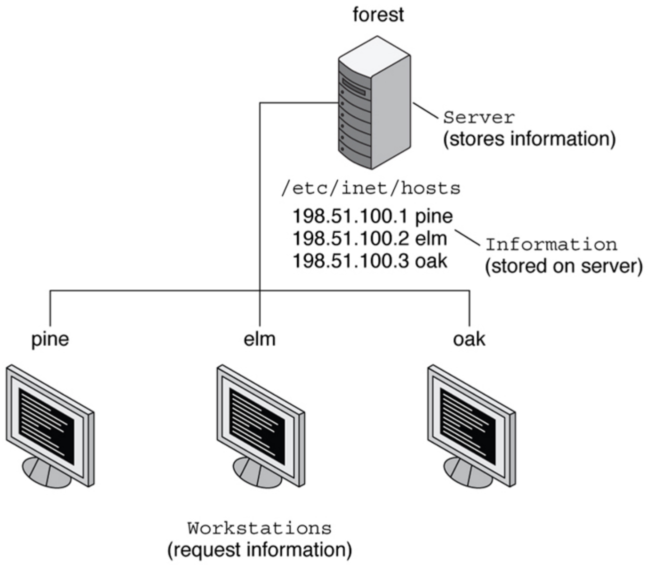
FIGURE 1 `/etc/inet/hosts` File Entries



In addition to storing addresses, systems store security information, mail data, and network services information. As networks offer more services, the stored list of information grows and each system might keep an entire set of files that are similar to `/etc/inet/hosts`.

A network information service stores network information on a server, which can be queried by any system. These systems are known as *clients* of the server. Whenever information about the network changes, instead of updating each client's local file, an administrator updates only the information stored by the network information service. Doing so reduces errors, inconsistencies between clients, and the size of the task. The following figure illustrates the client-server arrangement.

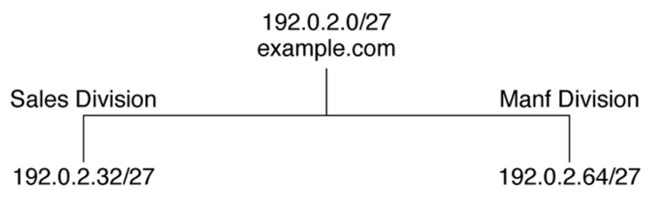
FIGURE 2 Client-Server Arrangement



This arrangement of a server providing centralized services to clients across a network is known as *client-server computing*.

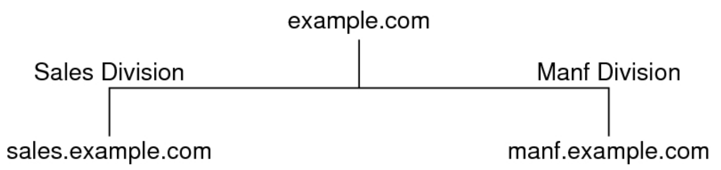
Although the main purpose of a network information service is to centralize information, the network information service can also simplify network names. For example, suppose your company has set up a network that is connected to the Internet with the network address `192.0.2.0` and the domain name `example.com`. Your company has two divisions, Sales and Manufacturing (Manf), so its network is divided into a main network and one subnet for each division as shown in the following figure. Each subnet has its own address.

FIGURE 3 Domain and Two Subnets With IP Addresses



You can identify each division by its network address, but descriptive names made possible by naming services are preferable.

FIGURE 4 Domain and Two Subnets With Descriptive Names

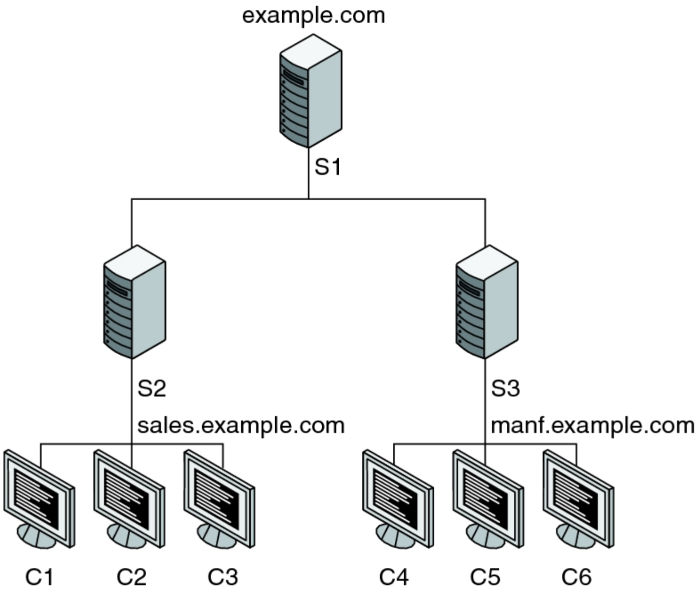


Instead of addressing mail or other network communications to `192.0.2.0/27`, you can address mail to `example.com`. Instead of addressing mail to `192.0.2.32/27` or `192.0.2.64/27`, you can address mail to `sales.example.com` or `manf.example.com`.

Names are also more flexible than physical addresses. Physical networks tend to remain stable but company organization tends to change.

For example, assume that the `example.com` network is supported by three servers, `S1`, `S2`, and `S3`, as shown in the following figure. Assume that two of those servers, `S2` and `S3`, support clients.

FIGURE 5 `example.com` Domain With Two Servers



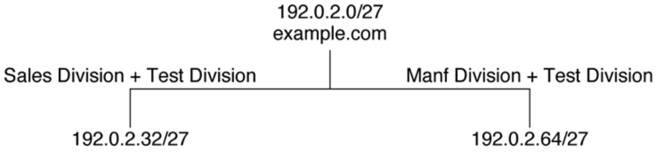
Clients `C1`, `C2`, and `C3` would obtain their network information from the server `S2`. Clients `C4`, `C5`, and `C6` would obtain information from the server `S3`. The following table summarizes the resulting network. The table is a generalized representation of the network but does not resemble an actual network information map.

TABLE 1 Representation of `example.com` Network

Network Address	Network Name	Server	Clients
`192.0.2.0/27`	`example.com`	`S1`	NA
`192.0.2.32/27`	`sales.example.com`	`S2`	`C1`, `C2`, `C3`
`192.0.2.64/27`	`manf.example.com`	`S3`	`C4`, `C5`, `C6`

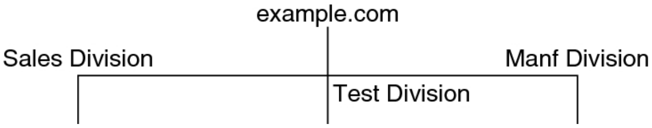
Suppose that you created a third division, Test, which borrowed some resources from the other two divisions but did not create a third subnet. The physical network would then no longer parallel the corporate structure, as shown in the following figure.

FIGURE 6 `example.com` Domain With Three Subnets



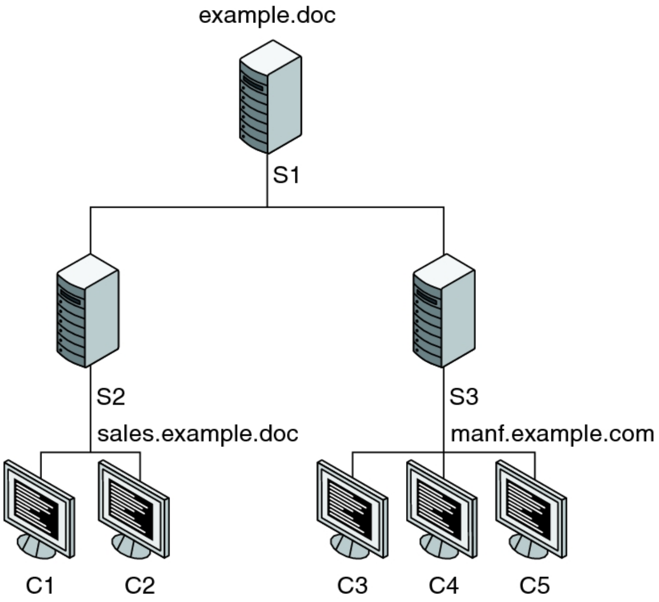
Network traffic for the Test division would not have its own subnet but would instead be split between `192.0.2.32/27` and `192.0.2.64/27`. However, with a network information service, the Test division traffic could have its own dedicated network.

FIGURE 7 `example.com` Domain With Dedicated Network for Each Division



Thus, when an organization changes, its network information service can change its mapping as shown in the following figure.

FIGURE 8 Modified `example.com` Domain



Clients `C1` and `C2` obtain their information from server `S2`. `C3`, `C4`, and `C5` obtain information from server `S3`.

You can accommodate subsequent changes in your organization by changes to the network information structure without reorganizing the network structure.

Oracle Solaris Naming Services

The Oracle Solaris platform provides the naming services described in the following sections:

- "DNS Naming Service" on page 20
- "`/etc` Files Naming Service" on page 21
- "NIS Naming Service" on page 21
- *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*

Most modern networks use two or more of these services in combination. The name service switch coordinates the naming service that is used for a particular lookup. For more information, see Chapter 2, “About the Name Service Switch”.

Naming Services and the Service Management Facility

In Oracle Solaris, all naming services are managed by the Service Management Facility (SMF). Configuration information is no longer stored in configuration files but in the SMF repository. Refer to the individual chapters in this book for more information about how SMF works with a specific naming service.

Legacy configuration files are retained in the Oracle Solaris 11 release only for purposes of compatibility with previous Oracle Solaris releases. The SMF service that is relevant to the specific naming service generates the contents of the legacy configuration files. You should no longer use these files for naming service configuration. Instead, you must use the general SMF commands such as `svcs`, `svcadm`, and `svccfg`.

When you upgrade from Oracle Solaris 10 to Oracle Solaris 11 and its update releases, the system's name service configuration is automatically migrated to SMF. However, if necessary, you can perform manual migration by using the `nscfg` command. For more information, see the `nscfg`(8) man page.

DNS Naming Service

The *Domain Name System* (DNS) is a hierarchical, distributed database implemented on a TCP/IP network. It is primarily used to look up IP addresses for Internet host names and host names for IP addresses. The data is distributed across the network and is located by using period-separated names that are read from right to left. DNS is also used to store other Internet-related host information, such as mail exchange routing information, location data, and available services. The hierarchical nature of the service enables the local administration of local domains while providing international coverage of other domains that are connected to the Internet, an intranet, or both.

DNS clients request information about a system from one or more name servers and wait for a response. DNS servers respond to requests from an information cache that was loaded from any of the following sources:

- A file or a third-party database on a DNS master server
- A file or a third-party database from a cooperating DNS slave server in the network
- Information stored from previous queries

If no response is found and the DNS server is not responsible for the domain in question, the service, if appropriately configured, will recursively request the host name from other DNS servers and cache that response.

Multicast DNS and Service Discovery

The `svc:network/dns/multicast` service manages two extensions to the DNS protocol. Multicast DNS (mDNS) implements DNS in a small network where no conventional DNS server has been installed. DNS Service Discovery (DNS-SD) extends Multicast DNS so that it also provide simple service discovery (network browsing). For more information, see "Multicast DNS" on page 35 and "Multicast DNS Service Discovery" on page 36.

The mDNS service uses the `.local` domain name, so do not use that name in DNS to avoid possible conflicts.

`/etc` Files Naming Service

The original host-based UNIX naming system was developed for stand-alone UNIX systems and then adapted for network use. Many legacy UNIX operating systems still manage all naming data by using only local files in `/etc`. However, managing hosts, users, and other naming data by using local files is not well suited for large complex networks. For a description of each file, refer to their associated man pages. For example, the `/etc/inet/hosts` file is described in the `hosts`(5) man page.

NIS Naming Service

The *Network Information Service* (NIS) was developed independently of DNS. DNS makes communication simpler by using host names instead of numerical IP addresses. NIS focuses on making network administration manageable by providing centralized control over a variety of network information. NIS stores information about the network, host names and addresses, users, and network services. This collection of network information is referred to as the *NIS namespace*.

NIS namespace information is stored in NIS maps. NIS maps replace UNIX `/etc` files as well as other configuration files. Because NIS maps store much more than names and addresses, the NIS namespace has a large set of maps. For more information, see "Working With NIS Maps" on page 93.

NIS uses a client-server arrangement, which is similar to DNS. Replicated NIS servers provide services to NIS clients. The principal servers are called *master* servers, and for reliability, the master servers have backup, or *slave* servers. Both master and slave servers use the NIS retrieval software and both store NIS maps. For more information about NIS architecture and NIS administration, see Chapter 6, "Setting Up and Configuring Network Information Service" and Chapter 7, "Administering the Network Information Service".

LDAP Naming Services

The Lightweight Directory Access Protocol (LDAP) is the secure network protocol that is used to access directory servers for distributed naming and other directory services. This standards-based protocol supports a hierarchical database structure. You can use the same protocol to provide naming services in both UNIX and multiplatform environments.

The Oracle Solaris OS supports LDAP in conjunction with the Oracle Directory Server Enterprise Edition (formerly Sun Java System Directory Server), as well as other LDAP directory servers.

For more information about LDAP and instructions to transition from NIS to LDAP, see *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.

For information about single sign-on as well as the setup and maintenance of Kerberos authentication services, see Chapter 1, "Kerberos on Oracle Solaris" in *Managing Kerberos in Oracle Solaris 11.4*.

Name Service Switch

The name service switch is a mechanism that enables clients to search through the DNS, LDAP, NIS, or local files data sources for naming information. The switch is managed through the `svc:/system/name-service/switch` service. For more information, see Chapter 2, "About the Name Service Switch".

Comparing the Naming Services

Naming Services	DNS	NIS	LDAP	Files
Namespace	Hierarchical	Flat	Hierarchical	Files
Data Storage	Files/resource records	Two-column maps	Directories (varied) Indexed database	Text-based files
Servers	Master/slave	Master/slave	Master/replica Multi master replica	None
Security	DNSSEC, varied	None (root or nothing)	Kerberos, TLS, SSL, varied	None
Transport	TCP/IP	RPC	TCP/IP	File I/O
Scale	Global	LAN	Global	Local host only
Data	Host	All	All	All

Note - Use DNS for host or network address lookups for LDAP and files-based naming.

IPv6 Extensions to Oracle Solaris Name Services

This section describes naming changes that were introduced by the implementation of IPv6. You can store IPv6 addresses in any of the Oracle Solaris naming services, such as NIS, LDAP, DNS, and files. You can also use NIS over IPv6 RPC transports to retrieve any NIS data.

DNS Extensions for IPv6

An IPv6-specific resource record, the AAAA resource record, is specified in RFC 1886 *DNS Extensions to Support IP Version 6*. The AAAA record maps a host name into a 128-bit IPv6 address. The pointer record (PTR) is still used with IPv6 to map IP addresses into host names. The 32 4-bit nibbles of the 128-bit address are reversed for an IPv6 address. Each nibble is converted to its corresponding hexadecimal ASCII value. Then, `ip6.arpa` is appended.

♦ ♦ ♦ CHAPTER 2

About the Name Service Switch

This chapter describes the name service switch. You use the name service switch to coordinate usage of different naming services.

This chapter covers the following topics:

- "Overview of the Name Service Switch" on page 25
- "Configuring the Name Service Switch" on page 30
- "Name Service Switch and Password Information" on page 33

Overview of the Name Service Switch

The name service switch is a configurable selection service that enables an administrator to specify which name information service or source to use for each type of network information. Each of the name information services is called a database. The name service switch is used by client applications that call any of the `getXbyY()` interfaces, such as the following:

- `gethostbyname()`
- `getpwuid()`
- `getpwnam()`
- `getaddrinfo()`

Each system has its own configuration in an SMF repository. Each property defined in the name service switch identifies a particular database, such as a host, password, or group. The value assigned to each property lists one or more sources from which to request the information. Sometimes, these values include guidance or options. The guidance might include how many retries to a service must be attempted, which timeout to apply, or what to do if the service fails.

The name service switch also controls DNS forwarding for clients as described in Chapter 3, "Managing DNS Server and Client Services". DNS forwarding grants Internet access to clients.

Databases and Sources for the Name Service Switch

You configure databases that are supported by the name service switch by using SMF services. To obtain a listing of these databases, use the `svccfg` command as shown in the following example.

```
# svccfg -s name-service/switch listprop config
config                      application
config/default              astring            files
config/password             astring            "files nis"
config/group                astring            "files nis"
config/host                 astring            "files nis"
config/network              astring            "nis [NOTFOUND=return] files"
config/protocol             astring            "nis [NOTFOUND=return] files"
config/rpc                  astring            "nis [NOTFOUND=return] files"
config/ether                astring            "nis [NOTFOUND=return] files"
config/netmask              astring            "files nis"
config/bootparam            astring            "nis [NOTFOUND=return] files"
config/publickey            astring            "nis [NOTFOUND=return] files"
config/netgroup             astring            nis
config/automount            astring            "files nis"
config/alias                astring            "files nis"
config/service              astring            "files nis"
config/printer              astring            "user nis"
config/auth_attr            astring            "files nis"
config/prof_attr            astring            "files nis"
config/project              astring            "files nis"
```

Note - The `timezone` table does not use the name service switch, so the table is not included in the property list for the switch.

From the SMF perspective, these databases are considered configurable properties of the service. Each database stores the following type of information:

- `alias` – Email addresses and aliases
- `auth_attr` – Authorization names and descriptions
- `automount` – Information about remote file systems that could be mounted locally
- `bootparam` – Boot information for diskless clients
- `ether` – Ethernet addresses and matching host names
- `group` – Information about groups that can be used to share access to files
- `host` – IP address and matching host names

- netgroup – Information for shared NFS file systems
- netmask – Network masks used to implement IP subnets
- network – Name and number for each network
- password – User account information
- prof_attr – Execution profile names, descriptions, and other attributes
- project – Project names, unique identifiers, and associated resource allocations
- protocol – Internet protocol names, protocol numbers, and aliases
- publickey – Public key information
- rpc – Names and numbers of RPC programs
- service – Name, port, and protocol for Internet services
- tnrhdb – Security attributes for a host using the Trusted Extensions feature of Oracle Solaris
- tnrhtp – Templates used by Trusted Extensions

In addition, the default property in the name service switch defines the source string for any database that is not otherwise defined. The value for this property is set to files to indicate that all the databases and their information are found locally in the /etc directory. You can set up a different configuration for the default property based on available sources. See “How to Change the Source for All Naming Databases” on page 32 for the procedure.

The default property enables you to configure a source that universally applies to the databases, instead of configuring each database's source.

The kind of sources that can be listed in the name service switch for the databases are as follows:

- ad – Identifies databases stored on an Active Directory server.
- pam_list – Replaces the obsoleted compat database. It can be used for password and group information to support old-style + or - syntax in the /etc/passwd, /etc/shadow, and /etc/group files.
- dns – Specifies that host information must be obtained from DNS.
- files – Specifies a file stored in the client's /etc directory, for example, /etc/passwd.
- ldap – Specifies that entries must be obtained from the LDAP directory.
- mdns – Specifies hosts information by using mDNS.
- nis – Specifies an NIS map, for example, the hosts map.

Note - The switch search criteria for the auto_home and auto_master tables and maps is combined into one category, which is called automount.

Source Formats for the Name Service Switch

You can use the following search criteria formats to select one or more information sources, and to specify the order in which the sources are used.

- **Single Source** – If an information type has only one source, such as `files`, a search routine that uses the switch searches for the information in that source *only*. If the routine finds the information, the routine returns a `success` status message. If the routine does not find the information, the routine stops searching and returns a different status message. What the routine does with the status message varies from routine to routine.
- **Multiple Sources** – If a database contains multiple sources for a given information type, the switch directs the search routine to search in the first listed source. If the routine finds the information, the routine returns a `success` status message. If the routine does not find the information in the first source, the routine tries the next source. The routine searches all sources until the routine has found the information, or until the routine is halted by a `return` specification. If all of the listed sources are searched without finding the information, the routine stops searching and returns a `non-success` status message.

By default in the Oracle Solaris 11 release, the first source is `files`. This configuration prevents system freezes if the next source listed is not available.

Status Messages for the Name Service Switch

If a routine finds the information, the routine returns a `success` status message. If the routine does not find the information, the routine returns one of three error status messages. Possible status messages are as follows:

- `SUCCESS` – Requested entry was found in the specified source.
- `UNAVAIL` – Source is either unresponsive or unavailable. In other words, none of the database sources could be found or accessed.
- `NOTFOUND` –Source responded with "No such entry." In other words, the database was accessed, but the needed information was not found.
- `TRYAGAIN` – Source is busy and might respond next time. In other words, the database was found but could not respond to the query.

Switch Action Options for the Name Service Switch

You can instruct the name service switch to respond to status messages with either of the following two actions:

- `return` – Stop looking for the information.
- `continue` – Try the next source.

In addition, for the `TRYAGAIN` status message, you can define the following actions:

- `forever` – Retry the current source indefinitely
- *n* – Retry the current source *n* more times

Default Search Criteria for the Name Service Switch

The combination of the name service switch status message and action options determine what the search routine does at each step. The combination of the status message and action options make up the *search criteria*.

The switch's default search criteria are the same for every source. Some of the search criteria are as follows:

- `SUCCESS=return` – Stop looking for the information. Proceed using the information that has been found.
- `UNAVAIL=continue` – Go to the next name service switch source and continue searching. If this source is the last or only source, return with a `NOTFOUND` status.
- `NOTFOUND=continue` – Go to the next name service switch source and continue searching. If this source is the last or only source, return with a `NOTFOUND` status.
- `TRYAGAIN=forever` – Searches the current name service switch source indefinitely.
- `TRYAGAIN=3` – Searches the current source three times. After exhausting three retries, the `TRYAGAIN` action transitions to `continue` and searches the next name service switch source.

You can change the default search criteria by explicitly specifying another criteria by using the *STATUS*=*action* syntax. For the procedure, see “How to Configure a Search Criterion for a Database” on page 31.

Note - Lookups in the name service switch are performed in the order in which items are listed. However, password updates are performed in reverse order unless otherwise specified by using the `passwd -r` *repository* command. For more information, see “Name Service Switch and Password Information” on page 33.

The client library routines contain compiled-in default entries that are used if no specific SMF property or `default` SMF property is defined in the name service switch, or if the property is syntactically incorrect. Typically, these compiled-in defaults are `files` only.

`keyserv` and `publickey` Entries in the Name Service Switch

The `keyserv` daemon reads the `publickey` properties in the name service switch only when `keyserv` is started. If you change the name service switch properties, `keyserv` does not register the changes until you restart the `keyserv` daemon by using `svcadm refresh svc:/network/rpc/keyserv:default`. You must run this command after the properties have been changed and the `name-service/switch` service has been refreshed so that the property changes are loaded into the SMF repository.

Configuring the Name Service Switch

When you configure the name service switch, you simultaneously perform the following actions:

- Indicate the source of the databases
- Specify a search sequence of the sources, if the database has multiple sources
- Define switch actions for corresponding search statuses, also known as switch criteria

The name service switch databases or properties are configured with default values. The procedures in this section explain how to configure certain properties differently.

▼ How to Change the Source for a Database

This procedure describes how to specify a different source for the `host` database. Assume that the original source configuration for the database are files and NIS, which means that for a host lookup, the local files are searched first, and then NIS. You reconfigure the name service switch to also use DNS in `host` lookups.

You can use this procedure as a template to configure the sources for other name service switch databases.

Before You Begin Make sure that the name service switch configuration reflects the actual setup of naming service on your system. For example, if you want DNS to be a source for `host` lookups, then DNS must be configured as well.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **(Optional) Display the current configuration of the `host` database.**

   ```
   # svccfg -s name-service/switch listprop config/host
   config                      application
   config/host                 astring             "files nis"
   ```

3. **Change the source definition for the `host` database.**

   ```
   # svccfg -s system/name-service/switch
   svc:/system/name-service/switch> setprop config/host = astring: "files dns nis"
   svc:/system/name-service/switch> quit
   ```

4. **Refresh the service for the name service switch.**

   ```
   # svcadm refresh name-service/switch
   ```

How to Configure a Search Criterion for a Database

The name service switch has default search criteria, as explained in "Default Search Criteria for the Name Service Switch" on page 29.

This procedure describes how to redefine the search mechanism for the `host` database when the information is not found at the first source. The search mechanism must stop instead of proceeding to search the next source.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **(Optional) Display the current configuration of the `host` database.**

```
# svccfg -s name-service/switch listprop config/host
config/host              astring            "files dns nis"
```

3. **Create a new search criterion for the `host` database lookups when information is not found at the first source.**

```
# svccfg -s system/name-service/switch
svc:/system/name-service/switch> setprop config/host = \
astring: "files [NOTFOUND=return] dns nis"
svc:/system/name-service/switch> quit
```

With this configuration, the search mechanism for the `network` database uses the default search criteria for the `SUCCESS` status and `UNAVAILABLE` status. However, if the information is not found, the search stops immediately.

4. **Refresh the service for the name service switch.**

```
# svcadm refresh name-service/switch
```

▼ How to Change the Source for All Naming Databases

This procedure describes how to define a common source for all the databases that are used by the name service for lookups. By default, the common source is `files`. You can use this procedure to add another source.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **(Optional) Display the current configuration of the `default` property.**

```
# svccfg -s name-service/switch listprop config/default
config/default              astring            files
```

3. **Add NIS as a default source.**

```
# svccfg -s system/name-service/switch
svc:/system/name-service/switch> setprop config/default = astring: "files nis"
svc:/system/name-service/switch> quit
```

4. **Refresh the service for the name service switch.**

   ```
   # svcadm refresh name-service/switch
   ```

▼ How to Use a Legacy nsswitch.conf File

Use this procedure if your existing naming switch configuration still uses the nsswitch.conf file. This procedure shows how to migrate your naming switch configurations from the file to SMF, which is the default method to configure naming switch in Oracle Solaris.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Copy the nsswitch.conf file to a new system.**

   Make sure to name the file /etc/nsswitch.conf.

3. **Load the information from the file into the SMF repository.**

   ```
   # nscfg import -f svc:/system/name-service/switch:default
   ```

4. **Refresh the service for the name service switch.**

   ```
   # svcadm refresh name-service/switch
   ```

Name Service Switch and Password Information

You can include and access password information in multiple repositories, such as files and nis. Use the config/password property in the name service switch to establish the lookup order for password information.

Note - Make files the first source in the name services switch for passwd information to help prevent a denial of service (DoS) attack on the system.

In an NIS environment, the config/password property in the name service switch must list the repositories in the following order:

```
config/password  astring             "files nis"
```

Listing files first allows the root user to log in under most circumstances, even when the system encounters some network or naming service issues.

Do not maintain multiple repositories *for the same user*. In most cases, the naming service looks up and returns the first definition only. Duplicate entries usually mask security problems.

For example, having the same user in both files and the network repository will (depending on the config/password name-service/switch configuration) use one login ID over the other. The first matched ID for a given system will become the ID used for the login session. If an ID is in both files and the network repository and the network repository has been disabled for security reasons, then any system where the ID resides and is accessed before the network ID is disabled might now be insecure and vulnerable to insecure and unwanted access.

♦ ♦ ♦ CHAPTER 3

Managing DNS Server and Client Services

This chapter provides information about the DNS server and client services. It contains the following topics:

- "Overview of DNS" on page 35
- "DNS and the Service Management Facility" on page 36
- "Administering DNS" on page 37
- "Administering Multicast DNS" on page 44
- "DNS Reference" on page 46

Overview of DNS

Like most networking protocols, DNS has two parts: a service providing answers and a client that queries the service. The BIND software and its associated daemon `named` provides the default DNS service in the Oracle Solaris operating system. BIND is maintained by the Internet Systems Consortium (ISC). The DNS client consists of a collection of utilities and libraries.

Multicast DNS

Multicast DNS (mDNS) provides a naming service system that is easy to set up and enables you to easily maintain systems on a local link. All participating network devices on the same local link perform standard DNS functions, using mDNS rather than unicast, and do not need a unicast DNS server. For administrators, the primary advantage of mDNS is that you do not need to maintain a unicast DNS server on the local network. For example, you do not need to update and maintain host names in files to resolve hostname-to-IP address requests for systems on the local link that are using mDNS.

Multicast DNS Service Discovery

Network services include printing, file transfer, music sharing, servers for photo, document, and other file sharing, and services provided by other local devices. DNS service discovery support in Oracle Solaris includes an open-source framework and tools to enable applications to advertise and discover network services.

For users, network service discovery makes computing easier by enabling them to browse for services on the network rather than needing to find the service manually. Existing standards and work by other companies and groups ensure that cross-platform support is available.

Related Materials About DNS

For information about DNS and BIND administration, see the following sources:

- *BIND 9 Administrator Reference Manual* on the ISC web site at http://www.isc.org
- BIND 9 Migration Notes documentation in the `/usr/share/doc/bind/migration.txt` file
- Listings of BIND features, known bugs and defects, and links to additional material on the ISC web site at http://www.isc.org
- *DNS and Bind (5th Edition)*, by Paul Albitz and Cricket Liu, (O'Reilly, 2006)

DNS and the Service Management Facility

The DNS server daemon, `named`, is managed by using the SMF. For an overview of SMF, see Chapter 1, "Introduction to the Service Management Facility" in *Managing System Services in Oracle Solaris 11.4*. Also refer to the `svcadm`(8), `svcs`(1), and `svccfg`(8) man pages for more details.

Note the following important information needed to use the SMF service to administer the DNS service:

- To perform administrative actions on this service, such as enabling, disabling, or restarting, use the `svcadm` command.
- To provide some protection for the service configuration, you can temporarily disable a service by using the `-t` option. If the service is disabled with the `-t` option, the original settings are restored for the service after a reboot. If the service is disabled without `-t`, the service remains disabled after a reboot.

- The Fault Managed Resource Identifiers (FMRIs) for the DNS service are `svc:/network/dns/server:`*instance*, and `svc:/network/dns/client:`*instance*.
- You can use the `svcs` command to display the status of the DNS server and client.
  - The following example shows the output of the `svcs` command.

    ```
    # svcs \*dns\*
    STATE          STIME    FMRI
    disabled       Nov_16   svc:/network/dns/multicast:default
    online         Nov_16   svc:/network/dns/server:default
    online         Nov_16   svc:/network/dns/client:default
    ```
  - The following example shows the output of the `svcs -l` command.

    ```
    # svcs -l dns/server
    fmri         svc:/network/dns/server:default
    name         BIND DNS server
    enabled      true
    state        online
    next_state   none
    state_time   Tue Jul 26 19:26:12 2011
    logfile      /var/svc/log/network-dns-server:default.log
    restarter    svc:/system/svc/restarter:default
    contract_id  83
    manifest     /lib/svc/manifest/network/dns/server.xml
    dependency   require_all/none svc:/system/filesystem/local (online)
    dependency   require_any/error svc:/network/loopback (online)
    dependency   optional_all/error svc:/network/physical (online)
    ```
- To start the DNS service with different options, change the properties of the `svc:/network/dns/server` service by using the `svccfg` command. For an example, see "How to Configure DNS Server Options" on page 39.

Because the DNS server daemon, `named`, is managed by SMF, the server is automatically restarted when an unexpected event occurs that causes `named` to exit abnormally. Additionally, you can use the `svcadm` command to restart the service. The BIND-specific management that is available by using `rndc` command can be used simultaneously with SMF.

Administering DNS

This section covers the following DNS administration tasks:

- "How to Install the DNS Package" on page 38

- "How to Configure a DNS Server" on page 38
- "How to Create an `rndc.conf` File" on page 39
- "How to Configure DNS Server Options" on page 39
- "How to Run the DNS Service as an Alternative User" on page 40
- "How to Enable a DNS Client" on page 41
- "Troubleshooting DNS Server Startup Issues" on page 43
- "How to Verify DNS Configuration Changes" on page 42

▼ How to Install the DNS Package

The DNS package is automatically installed with the Oracle Solaris 11 release. If the package was not included when the system was installed, use this procedure to install the package.

- **Install the DNS package.**

```
$ pkg install pkg:/service/network/dns/bind
```

▼ How to Configure a DNS Server

Note - Do not configure `named` to specify a root directory. A more secure option is to create an Oracle Solaris zone and configure `named` to run within that zone.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Create and verify a DNS configuration file.**

   Before the `named` daemon starts, a valid configuration file must exist. This file is called `/etc/named.conf` by default. The configuration of `named` is simple. An empty file provides sufficient information to configure a caching-only server, assuming that DNS root servers are accessible.

   ```
   # touch /etc/named.conf
   # named-checkconf -z /etc/named.conf
   ```

3. **(Optional) Create an `rndc` configuration file.**

You use this file to configure remote control access of the DNS server. See "How to Create an rndc.conf File" on page 39.

4. **(Optional) Change configuration information for the dns/server service.**

   See "How to Configure DNS Server Options" on page 39.

5. **Start the DNS service.**

   ```
   # svcadm enable dns/server
   ```

How to Create an rndc.conf File

You use the /etc/rndc.conf file to configure remote control access of the DNS server daemon, named, by using the rndc command. This procedure creates a default file. For more information, see the rndc.conf(5) man page.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Create the rndc configuration file.**

   ```
   # rndc-confgen -a
   wrote key file "/etc/rndc.key
   ```

3. **(Optional) Restart the DNS service.**

   If you are creating the rndc.conf file as part of the DNS server configuration, you can skip restarting the DNS service until the DNS server configuration is completed.

   ```
   # svcadm restart dns/server:default
   ```

How to Configure DNS Server Options

You can use the svccfg command to set the configuration options for the DNS server. For more information about configuration options, see the named(8) man page. This procedure describes how to set the options/ip_interfaces option to select the IPv4 transport protocol for named traffic.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Change the configuration information for the `dns/server` service.**

   ```
   # svccfg -s dns/server:default
   svc:/network/dns/server:default> setprop options/ip_interfaces = "IPv4"
   svc:/network/dns/server:default> quit
   ```

   Tip - You can change the configuration information with a single command.

   ```
   # svccfg -s dns/server:default setprop options/ip_interfaces=IPv4
   ```

3. **Update the SMF repository and enable the DNS service.**

   ```
   # svcadm refresh network/dns/server:default
   # svcadm enable network/dns/server:default
   ```

4. **(Optional) Verify the change.**

   ```
   # svccfg -s dns/server:default listprop options/ip_interfaces
   options/ip_interfaces  astring  IPv4
   ```

▼ How to Run the DNS Service as an Alternative User

This procedure describes how to assign the relevant authorizations to a user to manage the `named` daemon.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Provide the alternative user with the appropriate authorization.**

   ```
   # useradd -c "Trusted DNS administrator user" -s /usr/bin/pfbash \
    -A solaris.smf.manage.bind user
   ```

3. **Set service properties for the user.**

   ```
   # svccfg -s dns/server:default
   svc:/network/dns/server:default> setprop start/user = user
   svc:/network/dns/server:default> setprop start/group = user
   svc:/network/dns/server:default> exit
   ```

4. **Create a directory for a new process ID file.**

   Because only `root` has write access to create the default process ID file, `/var/run/named/named.pid`, you must configure the `named` daemon to use an alternative file.

   ```
   # mkdir /var/named/tmp
   # chown user /var/named/tmp
   ```

5. **Change the configuration to use the new directory by adding the following lines to the `named.conf` file:**

   ```
   # head /etc/named.conf
   options {
   directory "/var/named";
   pid-file "/var/named/tmp/named.pid";
   };
   ```

6. **Update the SMF repository and restart the DNS service.**

   ```
   # svcadm refresh svc:/network/dns/server:default
   # svcadm restart svc:/network/dns/server:default
   ```

How to Enable a DNS Client

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Configure the DNS domain.**

   a. **List the domains that need to be searched and the IP addresses for the DNS name servers.**

   ```
   # svccfg -s dns/client
   svc:/network/dns/client> setprop config/search = \
   astring: ("example.com" "sales.example.com")
   ```

```
svc:/network/dns/client> setprop config/nameserver = \
net_address: (192.0.2.10 192.0.2.11)
svc:/network/dns/client> quit
```

b. **Update the SMF repository.**

```
# svccfg -s dns/client
svc:/network/dns/client> select network/dns/client:default
svc:/network/dns/client:default> refresh
svc:/network/dns/client:default> quit
```

3. **Update the name service switch information in the SMF repository to use DNS.**

```
# svccfg -s system/name-service/switch
svc:/system/name-service/switch> setprop config/host = astring: "files dns"
svc:/system/name-service/switch> select system/name-service/switch:default
svc:/system/name-service/switch:default> refresh
svc:/system/name-service/switch:default> quit
```

4. **Start the services needed to run the DNS client.**

```
# svcadm enable network/dns/client
# svcadm enable system/name-service/switch
```

▼ How to Verify DNS Configuration Changes

When modifying the DNS configuration, you can verify the syntax of the `/etc/named.conf` file with the `named-checkzone` command.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Change the configuration file, as needed.**

   In this example, the default directory is changed.

```
# echo 'options {directory "/var/named";};' > /etc/named.conf
```

3. **Verify the file contents.**

```
# named-checkconf
/etc/named.conf:1: change directory to '/var/named' failed: file not found
```

```
/etc/named.conf:1: parsing failed
```

In this example, the check failed because the `/var/named` directory has not yet been created.

4. **Correct any errors reported.**

   ```
   # mkdir /var/named
   ```

5. **Repeat steps 3 and 4 until no errors are reported.**

6. **(Optional) Reflect the change in the running service by using one of the following methods:**

   - **Use the `rndc` command to update the configuration using the `reload` or `reconfig` subcommand, depending on the changes made.**
   - **Restart the `named` service.**

     ```
     # svcadm restart svc:/network/dns/server:default
     ```

Troubleshooting DNS Server Startup Issues

Try the following checks if you have problems with DNS startup. You must first become an administrator.

- Check the DNS service status.

  ```
  # svcs -x dns/server:default
  svc:/network/dns/server:default (BIND DNS server)
  State: online since Tue Oct 18 19:35:00 2011
  See: named(8)
  See: /var/svc/log/network-dns-server:default.log
  Impact: None.
  ```

- Check the DNS service log file.

  ```
  # tail /var/svc/log/network-dns-server:default.log
  ```

- Check system log messages.

  ```
  # grep named /var/adm/messages
  ```

- Start the `named` daemon manually. Running `named` in the foreground forces all logging to standard error so that it is easier to identify problems.

```
   # named -g
```

After the issue has been fixed, clear the maintenance required state.

```
# svcadm clear dns/server:default
# svcs dns/server:default
STATE          STIME    FMRI
online         17:59:08 svc:/network/dns/server:default
```

Administering Multicast DNS

You use the mDNS service to advertise the availability of services provided on the system. For mDNS and DNS service discovery to function, the mDNS service must be enabled on all systems that are to participate in mDNS. The following sections explain how to enable multicast DNS (mDNS) and DNS service discovery.

▼ How to Enable mDNS and DNS Service Discovery

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Check whether the mDNS package has been installed.**

   ```
   # pkg info pkg:/service/network/dns/mdns
   ```

3. **If the mDNS package has not been installed, install it.**

   ```
   $ pkg install pkg:/service/network/dns/mdns
   ```

4. **Update name service switch information by changing the `config/host` property of the `name-service/switch` service to include `mdns` as a source.**

   For example:

   ```
   # /usr/sbin/svccfg -s svc:/system/name-service/switch
   svc:/system/name-service/switch> setprop config/host = astring: "files dns mdns"
   svc:/system/name-service/switch> select system/name-service/switch:default
   svc:/system/name-service/switch:default> refresh
   ```

```
svc:/system/name-service/switch> quit
```

5. **Enable the mDNS service.**

   ```
   # svcadm enable svc:/network/dns/multicast:default
   ```

   Enabling mDNS by using the `svcadm enable` command ensures that your changes persist through upgrades and reboots. For more information, see the `svcadm`(8) man page.

6. **(Optional) Check the mDNS error log, `/var/svc/log/network-dns-multicast:default.log`, for errors or messages.**

Advertising Resources for DNS

You can use the `dns-sd` command interactively as a network diagnosis tool to browse and discover services, similarly to the `ping` or `traceroute` command.

Because the `dns-sd` command-line arguments and its output format can change over time, invoking it from a shell script is unpredictable and risky. Additionally, the asynchronous nature of DNS-SD does not easily lend itself to script-oriented programming.

For more information, see the `dns-sd`(8) man page. To incorporate the DNS service in applications, see the `libdns_sd`(3LIB) man page.

The following examples show how to advertise services by using DNS service discovery.

EXAMPLE 1 Advertising a Printing Service

This example advertises the existence of the LPR printing service on port `515` on a system called `My Test` so that it will be available to DNS-SD compatible printing clients.

```
# dns-sd -R "My Test" _printer._tcp. . 515 pdl=application/postscript
```

For this registration to be useful, the LPR service must be available on port `515`.

EXAMPLE 2 Advertising a Web Page

This example advertises a web page being served by an HTTP server on port `80` on the `My Test` system. The web page will appear on the Bonjour list in Safari and other DNS-SD compatible web clients.

```
# dns-sd -R "My Test" _http._tcp . 80 path=/path-to-page.html
```

DNS Reference

This section describes the files, daemons, and commands that are associated with the DNS service. In addition, it describes some of the flags that were used when the ISC version of BIND was built.

DNS Files

The following files are associated with the DNS service:

- `/etc/named.conf` – Provides configuration information for the `named` daemon. For more information, see the `named.conf`(5) man page.
- `/etc/rndc.conf` – Provides configuration information for the `rndc` command. For more information, see the `rndc.conf`(5) man page.

DNS Commands and Daemons

The following commands and daemons are associated with the DNS service:

- `/usr/sbin/dig` – Requests DNS responses from a DNS server. Often used to troubleshoot. For more information, see the `dig`(1) man page.
- `/usr/bin/dns-sd` – Finds or lists resources used by the mDNS service. For more information, see the `dns-sd`(8) man page.
- `/usr/sbin/dnssec-dsfromkey` – Generates a Delegation Signer resource record (DS RR) from a key file. For more information, see the `dnssec-dsfromkey`(8) man page.
- `/usr/sbin/dnssec-keyfromlabel` – Retrieves selected keys from cryptographic device and builds a key file. For more information, see the `dnssec-keygen`(8) man page.
- `/usr/sbin/dnssec-keygen` – Creates keys and key files for secure DNS and for transaction signatures (TSIG). For more information, see the `dnssec-keygen`(8) man page.
- `/usr/sbin/dnssec-signzone` – Signs a DNS zone. For more information, see the `dnssec-signzone`(8) man page.
- `/usr/sbin/host` – Performs simple DNS lookups, often converting host names to IP addresses or IP addresses to host names. For more information, see the `host`(1) man page.
- `/usr/sbin/named` – DNS server daemon, which responds to information requests from clients. For more information, see the `named`(8) man page.
- `/usr/sbin/named-checkconf`– Checks the syntax of the `named.conf` file. For more information, see the `named-checkconf`(8) man page.

- /usr/sbin/named-checkzone– Checks the syntax and integrity of a DNS zone file. For more information, see the named-checkzone(8) man page.
- /usr/sbin/named-compilezone – Converts a DNS zone file. For more information, see the named-compilezone(8) man page.
- /usr/sbin/nscfg – Legacy name service configuration utility that imports or exports name service configuration between legacy name service configuration files and the SMF repository. For more information, see the nscfg(8) man page.
- /usr/sbin/nslookup – Deprecated: Queries the DNS server. Instead, use the dig command.
- /usr/sbin/nsupdate – Submits DNS update requests to a DNS server. For more information, see the nsupdate(1) man page.
- /usr/sbin/rndc – Provides remote control of the DNS server daemon. For more information, see the rndc(8) man page.
- /usr/sbin/rndc-confgen – Generates configuration files for the rndc command. For more information, see the rndc-confgen(8) man page.

Compilation Flags Used When BIND Was Built

To view the flags that were used to compile BIND, use the named -v command. The following list shows some of the compilation flags that were used when building the ISC version of BIND for the Oracle Solaris release:

- disable-openssl-version-check – Disables the OpenSSL version check because OpenSSL is provided by a separate dynamic library
- enable-devpoll – Uses the /dev/poll driver for fast poll on many file descriptors
- enable-fixed-rrset – Enables fixed resource record set ordering, which is needed for backward compatibility
- enable-threads – Enables multithreading
- with-openssl – Builds BIND with cryptographic and Secure Sockets Layer (SSL) support, which is required for DNSSEC
- with-pkcs11 – Enables the use of OpenSSL cryptographic hardware support

♦ ♦ ♦ CHAPTER 4

Setting Up Oracle Solaris Active Directory Clients

The nss_ad naming service module provides a back end for the passwd, shadow, and group files. The nss_ad module uses Active Directory (AD) and its native schema as the naming service to resolve user and group names and IDs from across an AD forest.

This chapter covers the following topics:

- "Overview of the nss_ad Naming Service Module" on page 49
- "Password Updates and AD" on page 52
- "How nss_ad Module Retrieves Data From AD" on page 52

Overview of the nss_ad Naming Service Module

The Oracle Solaris client must be joined to an AD domain before any of the AD interoperability functionality, including nss_ad, can be used. Use the kclient utility to join the client to AD. During the join operation, kclient configures Kerberos v5 on the client. Thereafter, you can use nss_ad to resolve naming service requests by specifying ad as a source in the nsswitch.conf file for the supported databases. The nss_ad module uses host credentials to look up naming service information in AD.

The nss_ad module uses DNS server records to auto-discover AD directory servers, such as domain controllers and global catalog servers. Therefore, DNS must be properly configured on the Oracle Solaris client. The nss_ad module also uses the LDAP v3 protocol to access naming information from AD servers. The AD server schema requires no modification because nss_ad works with the native AD schema.

Note - The nss_ad module does not support logins of Windows users onto an Oracle Solaris system. Until such logins are supported, Windows users must continue to log in by using traditional back ends such as nis and ldap.

You must enable the idmap and svc:/system/name-service/cache services to use nss_ad. The nss_ad module uses the idmap service to map between Windows security identifiers (SIDs), UNIX user identifiers (UIDs), and group identifiers (GIDs).

Make sure to qualify all AD user and group names with domain names such as user@domain or group@domain. For example, getpwnam(abc) will fail, but getpwnam(abc@domain) will succeed provided that abc is a valid Windows user in the domain named domain.

The following additional rules also pertain to the nss_ad module:

- Like AD, nss_ad performs case-insensitive matching of user and group names.
- Only use the nss_ad module in UTF-8 locales or in domains where users and groups have only ASCII characters in their names.
- Well-known SIDs are a set of SIDs that identify generic users or generic groups in the Windows world. These SIDs are not domain specific and their values remain constant across all Windows operating systems. The names of well-known SIDs are qualified with the string BUILTIN, for example, Remote Desktop Users@BUILTIN.
- The nss_ad module does not support enumeration. Therefore, the getpwent() and getgrent() interfaces and commands that use enumeration such as getent passwd and getent group cannot retrieve information from AD.
- The nss_ad module currently supports only the passwd and group files. nss_ad does not support other naming service databases that follow the passwd entry, such as audit_user and user_attr. If the ad back end is processed, based on the configuration, it returns NOT FOUND for these databases.

▼ How to Configure the nss_ad Module

The nss_ad module requires that the Oracle Solaris client use DNS for host resolution.

1. **Configure the DNS service.**
   See "How to Enable a DNS Client" on page 41 for instructions.

   Note - The AD domain name must be specified either by means of the domain directive or as the first item in the list specified by the search directive.

   If both directives are specified, then whichever is last takes precedence in order for the idmap auto-discovery feature to work properly.

   ---

2. **Use the dig command to verify that the AD server can be resolved by using its name and IP address.**

```
# dig -x 192.0.2.22 +short
myserver.ad.example
# dig myserver.ad.example +short
192.0.2.22
```

3. **Add dns to the list of naming services for hosts.**

```
# svccfg -s svc:/system/name-service/switch
svc:/system/name-service/switch> setprop config/host = astring: "files dns"
svc:/system/name-service/switch> select system/name-service/switch:default
svc:/system/name-service/switch:default> refresh
svc:/system/name-service/switch:default> quit
```

Note - To include additional naming services such as nis or ldap for host resolution, add them after dns.

4. **Verify whether the DNS service is enabled and online.**

   For example:

```
# svcs svc:/network/dns/client
STATE  STIME  FMRI
online Oct_14 svc:/network/dns/client:default
```

5. **Use the kclient utility to join the system to the AD domain.**

   For example:

```
# /usr/sbin/kclient -T ms_ad
```

6. **Add ad to the list of naming services for password and group.**

```
# svccfg -s svc:/system/name-service/switch
svc:/system/name-service/switch> setprop config/password = astring: "files nis ad"
svc:/system/name-service/switch> setprop config/group = astring: "files nis ad"
svc:/system/name-service/switch> select system/name-service/switch:default
svc:/system/name-service/switch:default> refresh
svc:/system/name-service/switch:default> quit
```

7. **Enable the idmap service.**

```
# svcadm enable idmap
```

8. **Update the SMF repository for the name service switch service.**

```
# svcadm refresh name-service/switch
```

Note - The nscd module automatically restarts whenever name service switch is refreshed.

9. **Verify whether you can access user and group information from AD.**

   For example:

```
# getent passwd 'test_user@example'
test_user@example:x:2154266625:2154266626:test_user::
# getent passwd 2154266625
test_user@example:x:2154266625:2154266626:test_user::
```

Password Updates and AD

The passwd(5) man page contains a list of valid formats for the config/passwd property in the name service switch. Adding ad to these configurations is supported. However, changing AD user passwords through the passwd command is not supported. If found in the passwd entry during a password update, ad is skipped. Use the kpasswd command to update AD user passwords.

You can add the ad search order to existing valid password and group entries in name service switch. For example:

```
# svccfg -s svc:/system/name-service/switch
svc:/system/name-service/switch> setprop config/password = astring: "files nis ad"
svc:/system/name-service/switch> setprop config/group = astring: "files nis ad"
svc:/system/name-service/switch> select system/name-service/switch:default
svc:/system/name-service/switch:default> refresh
svc:/system/name-service/switch:default> quit
```

How nss_ad Module Retrieves Data From AD

This section describes how the nss_ad module resolves naming service requests for the passwd, shadow, and group files by retrieving corresponding data from AD.

nss_ad Retrieving passwd Information

Use the following syntax for the passwd entry:

username:*password*:*uid*:*gid*:*gecos*:*home-directory*:*login-shell*

For more information, see the passwd(5) man page.

The nss_ad module retrieves passwd information from AD as follows:

- *username* – Uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, johnd@example.com.
- *password* – Uses the value of x because the user password is not available in the AD object.
- *uid* – Uses the Windows user's SID from the objectSID AD attribute, which is mapped to the UID by using the idmap service.
- *gid* – Uses the Windows user's primary group SID, which is mapped to the GID by using the idmap service. The group SID is obtained by appending the value of the primaryGroupID AD attribute to the domain SID. Because for users in AD the primaryGroupID attribute is an optional attribute, it might not exist. If the attribute does not exist, nss_ad uses the idmap diagonal mapping facility to map the user SID from the objectSID attribute.
- *gecos* – Value of the CN AD attribute.
- *home-directory* – Uses the value of the homeDirectory AD attribute if a value exists. Otherwise, the field is left empty.
- *login-shell* – The field is left empty because the native AD schema has no login shell attribute.

nss_ad Retrieving shadow Information

Use the following syntax for the shadow entry:

username:*password*:*lastchg*:*min*:*max*:*warn*:*inactive*:*expire*:*flag*

For more information, see the shadow(5) man page.

The nss_ad module retrieves shadow information from AD as follows:

- *username* – Uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, johnd@example.com.
- *password* – Uses the value of *NP* because the user password is not available in the AD object.

The rest of the shadow fields are left empty because shadow fields are irrelevant with AD and Kerberos v5.

nss_ad Retrieving group Information

Use the following syntax for the group entry:

groupname:*password*:*gid*:*user-list*

For more information, see the group(5) for man page.

The nss_ad module retrieves information from AD as follows:

- *groupname* – Uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, admins@example.com.
- *password* – Field is left empty because the Windows groups do not have passwords.
- *gid* – Uses the Windows group's SID from the objectSID AD attribute, which is mapped to the GID by using the idmap service.
- *user-list* – Field is left empty.

♦ ♦ ♦ CHAPTER 5

About the Network Information Service

This chapter provides an overview of the Network Information Service (NIS), which is a distributed naming service. NIS is a mechanism for identifying and locating network objects and resources. It provides a uniform storage and retrieval method for network-wide information in a transport-protocol and media-independent fashion.

This chapter covers the following topics:

- "NIS Hierarchical Design" on page 55
- "NIS System Types" on page 57
- "NIS Elements" on page 58
- "NIS Binding" on page 64

NIS Hierarchical Design

By running NIS, the system administrator can distribute administrative databases, called *maps*, among a variety of servers (*master* and *slaves*). The administrator can update those databases from a centralized location in an automatic and reliable fashion to ensure that all clients share the same naming service information in a consistent manner throughout the network.

NIS was developed independently of DNS and has a slightly different focus. Whereas DNS focuses on making communication simpler by using host names instead of numerical IP addresses, NIS focuses on making network administration manageable by providing centralized control over a variety of network information. NIS stores information not only about host names and addresses, but also about users, the network itself, and network services. This collection of network *information* is referred to as the NIS *namespace*.

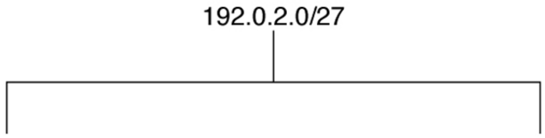
Note - Screen messages or NIS map names might use "host name" or "machine name".

NIS uses a client-server arrangement. NIS servers provide services to NIS clients. The principal server is called a *master* server, and for reliability, it can have several backup servers or *slave* servers. Both master and slave servers use the NIS information retrieval software, and both store NIS maps.

NIS uses domains to arrange the systems, users, and networks in its namespace. However, it does not use a domain hierarchy. An NIS namespace is flat.

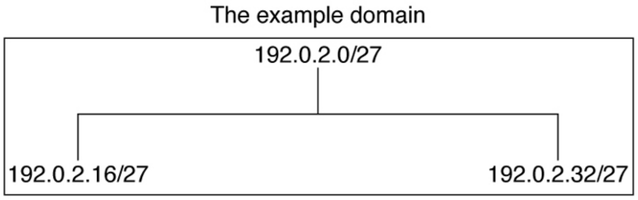
The following figure shows a domain that has unidentified hierarchical structure.

FIGURE 9 Domain With Unidentified Hierarchical Structure



The following figure shows how to arrange a physical network into an NIS domain.

FIGURE 10 Domain With NIS Namespace



An NIS domain cannot be connected directly to the Internet using just NIS. However, organizations that want to use NIS and also be connected to the Internet can combine NIS with DNS. You can use NIS to manage all local information and use DNS for Internet host lookup. NIS also provides a forwarding service that forwards host lookups to DNS if the information cannot be found in an NIS map.

You can set up the name service switch in a Oracle Solaris system so that hosts lookup requests can be directed in the following ways:

- To access only DNS

- To access DNS, but if a host is not found in DNS, then access NIS
- To access NIS, but if a host is not found by NIS, then access DNS

For maximum interoperability, DNS is the recommended service for host lookups. For more information, see Chapter 2, “About the Name Service Switch”.

NIS System Types

The three types of NIS systems are as follows:

- Master server
- Slave servers
- Clients of NIS servers

Any system can be an NIS client, but only systems with disks should be NIS servers, either master or slave. Servers are also clients, typically of themselves.

NIS Servers

NIS servers come in two varieties, master and slave. The system designated as master server contains the set of maps that the system administrator creates and updates as necessary. Each NIS domain must have one, and only one, master server, which can propagate NIS updates with the least performance degradation.

You can designate additional NIS servers in the domain as slave servers. A slave server has a complete copy of the NIS maps in the master server. Whenever maps in the master server are updated, the updates are propagated among the slave servers. Slave servers can handle any overflow of requests from the master server, minimizing "server unavailable" errors.

Normally, the system administrator designates one master server for all NIS maps. However, because each individual NIS map has the host name of the master server encoded within it, you could designate different servers to act as master and slave servers for different maps. To minimize confusion, designate a single server as the master for all the maps you create within a single domain. The examples in this chapter assume that one server is the master for all maps in the domain.

NIS Clients

NIS clients run processes that request data from maps on the servers. Clients do not make a distinction between master and slave servers, since all NIS servers should have the same information.

Note - Oracle Solaris does not support a configuration in which an NIS client and a native LDAP client coexist on the same client system.

NIS Elements

The NIS naming service is composed of the following elements:

- "NIS Domain" on page 58
- "NIS Daemons" on page 59
- "NIS Commands" on page 59
- "NIS Maps" on page 60

NIS Domain

An NIS *domain* is a collection of systems that share a common set of NIS maps. Each domain has a domain name, and each system that shares the common set of maps belongs to that domain.

NIS domains and DNS domains are not necessarily the same. In some environments, NIS domains are defined based on enterprise-wide network subnet administrative layouts. DNS names and domains are defined by Internet DNS naming standards and hierarchies. The two naming domain naming systems might be or might not be configured to match up identically. The domain name for the two services are controlled separately and might be configured differently.

Any system can belong to a given domain, as long as there is a server for that domain's maps in the same network or subnet. NIS domain lookups use remote procedure calls (RPCs). Therefore, NIS requires that all the clients and all the servers that provide direct services to those clients must exist on the same accessible subnet. It is not uncommon to have each administrative subnet managed as a separate NIS domain (distinct from an enterprise-wide DNS domain) but using common databases managed from a common master server. You can use the `svc:/network/nis/domain` SMF service to manage the NIS domain name and all the shared NIS configuration information.

NIS Daemons

The NIS service is managed by SMF. Administrative actions on this service, such as enabling, disabling, or restarting, can be performed by using the `svcadm` command. For an overview of SMF, refer to Chapter 1, "Introduction to the Service Management Facility" in *Managing System Services in Oracle Solaris 11.4*. Also refer to the `svcadm`(8) and `svcs`(1) man pages for more details.

The following daemons provide the NIS service:

- `nscd` – A client service that provides a cache for most name service requests, which is managed by the `svc:/system/name-service/cache` service
- `rpc.yppasswdd` – The NIS password update daemon managed by the `svc:/network/nis/passwd` service

> **Note -** The `rpc.yppasswdd` daemon considers all shells that begin with an `r` to be restricted. For example, if you are in `/bin/rksh`, you are not allowed to change from that shell to another shell. If you have a shell that begins with `r` but is not intended to be restricted as such, refer to Chapter 8, "Troubleshooting Network Information System" for the workaround.

- `rpc.ypupdated` – A daemon that modifies other maps such as `publickey` and is managed by the `svc:/network/nis/update` service
- `ypbind` – The binding process managed by the `svc:/network/nis/client` service
- `ypserv` – The server process managed by the `svc:/network/nis/server` service
- `ypxfrd` – A high-speed map transfer daemon managed by the `svc:/network/nis/xfr` service

NIS Commands

The following commands support the NIS service:

- `make` – Updates NIS maps by reading the `/var/yp/Makefile` file when the command is run in the `/var/yp` directory. You can use `make` to update all maps based on the input files or to update individual maps. For information about the functionality of `make` for NIS, see the `ypmake`(8) man page.

- `makedbm` – Takes an input file and converts it into `dbm.dir` and `dbm.pag` files. NIS uses valid `dbm` files as maps. You can also use `makedbm -u` to disassemble a map so that you can see the key-value pairs that comprise it.
- `ypcat` – Displays the contents of an NIS map.
- `ypinit` – Automatically creates maps for an NIS server from the input files. It is also used to construct the initial `/var/yp/binding/`*domain*`/ypservers` file on the clients. Use `ypinit` to set up the master NIS server and the slave NIS servers for the first time.
- `ypmatch` – Prints the value for one or more specified keys in an NIS map. You cannot specify which version of the NIS server map you are seeing.
- `yppoll` – Shows which version of an NIS map is running on a server that you specify. It also lists the master server for the map.
- `yppush` – Copies a new version of an NIS map from the NIS master server to its slaves. You run the `yppush` command on the master NIS server.
- `ypset` – Instructs a `ypbind` process to bind to a named NIS server. This command is not for casual use, and its use is discouraged because of security implications. See the `ypset`(8) and `ypbind`(8) man pages for information about the `ypset` and `ypsetme` options to the `ypbind` process.
- `ypwhich` – Shows which NIS server a client is using at the moment for NIS services. If invoked with the `-m` *mapname* option, this command shows which NIS server is master of each map. If only `-m` is used, the command displays the names of all the available maps and their respective master servers.
- `ypxfr` – Pulls an NIS map from a remote server to the local `/var/yp/`*domain* directory by using NIS itself as the transport medium. You can run `ypxfr` interactively or periodically from a `crontab` file. It is also called by `ypserv` to initiate a transfer.

NIS Maps

The information in NIS maps is stored in ndbm format. For more information about the format of the map file, see the `ypfiles`(5) and `ndbm`(3C) man pages.

NIS maps extend access to UNIX `/etc` data and other configuration files, such as `passwd`, `shadow`, and `group`, so that the same data can be shared between a network of systems. Sharing these files simplifies administrative updates and management of the data files. You can deploy NIS with minimal effort. However, larger enterprises, especially those with security requirements, should consider using LDAP naming services instead. On a network running NIS, the NIS master server for each NIS domain maintains a set of NIS maps for other systems in the domain to query. NIS slave servers also maintain duplicates of the master server's maps. NIS client systems can obtain namespace information from either master or slave servers.

NIS maps are essentially two-column tables. One column is the *key* and the other column is information related to the key. NIS finds information for a client by searching through the keys. Some information is stored in several maps because each map uses a different key. For example, the names and addresses of systems are stored in two maps: `hosts.byname` and `hosts.byaddr`. When a server has a host's name and needs to find its address, it looks in the `hosts.byname` map. When a server has the address and needs to find the name, it looks in the `hosts.byaddr` map.

An NIS `Makefile` is stored in the `/var/yp` directory of a system that is designated as an NIS server at installation time. Running `make` in that directory causes `makedbm` to create or modify the default NIS maps from the input files.

Note - Always create maps on the master server, as maps created on a slave will not automatically be pushed to the master server.

Default NIS Maps

A default set of NIS maps are provided in the Oracle Solaris system. You might want to use all these maps or only some of them. NIS can also use whatever maps you create or add when you install other software products.

Default maps for an NIS domain are located in each server's `/var/yp/`*domain–name* directory. For example, the maps that belong to the domain `test.com` are located in each server's `/var/yp/test.com` directory.

The following table describes the default NIS maps and lists the appropriate source file name for each map.

TABLE 2 NIS Map Descriptions

NIS Map Name	Source File	Description
`audit_user`	`audit_user`	Contains user auditing preselection data.
`auth_attr`	`auth_attr`	Contains authorization names and descriptions.
`bootparams`	`bootparams`	Contains path names of files that clients need during boot: root, swap, possibly others.
`ethers.byaddr`	`ethers`	Contains host names and Ethernet addresses. The Ethernet address is the key in the map.
`ethers.byname`	`ethers`	Contains host names and Ethernet addresses. Host name is the key in the map.

NIS Map Name	Source File	Description
exec_attr	exec_attr	Contains profile execution attributes.
group.bygid	group	Contains group security information. Group ID is the key in the map.
group.byname	group	Contains group security information. Group name is the key in the map.
hosts.byaddr	hosts	Contains host name and IP address. IP address is the key in the map.
hosts.byname	hosts	Contains host name and IP address. Host name is the key in the map.
mail.aliases	aliases	Contains aliases and mail addresses. Aliases is the key in the map.
mail.byaddr	aliases	Contains mail address and alias. Mail address is the key in the map.
netgroup.byhost	netgroup	Contains group name, user name, and host name.
netgroup.byuser	netgroup	Contains group name, user name, and host name. User name is the key in the map.
netgroup	netgroup	Contains group name, user name, and host name. Group name is the key in the map.
netid.byname	passwd, hosts, group	Contains host name and mail address including domain name. If the netid file is available, it is consulted in addition to the data available through the other files. Used for authentication similar to that used on UNIX systems.
publickey.byname	publickey	Contains the public key database used by secure RPC.
netmasks.byaddr	netmasks	Contains network mask to be used with IP submitting. Address is the key in the map.
networks.byaddr	networks	Contains names of networks known to the system and their IP addresses. Address is the key in the map.
networks.byname	networks	Contains names of networks known to the system and their IP addresses. Name of the network is the key in the map.
passwd.adjunct.byname	passwd, shadow	Contains auditing information and the hidden password information for C2 clients.
passwd.byname	passwd, shadow	Contains password information. User name is the key in the map.
passwd.byuid	passwd, shadow	Contains password information. User ID is the key in the map.
prof_attr	prof_attr	Contains attributes for execution profiles.
protocols.byname	protocols	Contains network protocols known to your network.
protocols.bynumber	protocols	Contains network protocols known to your network. Protocol number is the key in the map.
rpc.bynumber	rpc	Contains program number and name of RPCs known to your system. RPC program number is the key in the map.
services.byname	services	Lists Internet services known to your network. Port or protocol is the key in the map.

NIS Map Name	Source File	Description
`services.byservice`	`services`	Lists Internet services known to your network. Service name is the key in the map.
`user_attr`	`user_attr`	Contains extended attributes for users and roles.
`ypservers`	N/A	Lists NIS servers known to your network.

The `ageing.byname` mapping contains information that is used by the `yppasswdd` daemon to read and write password aging information to the directory information tree (DIT) when the NIS-to-LDAP transition is implemented. If you are not using password aging, then `ageing.byname` can be commented out of the mapping file. For more information about the NIS-to-LDAP transition, see Chapter 8, "Transitioning From NIS to LDAP" in *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.

Using NIS Maps

NIS makes updating network databases much simpler than with the `/etc` files system. You no longer have to change the administrative `/etc` files on every system each time you modify the network environment.

However, NIS provides no additional security than that provided by the `/etc` files. If additional security is needed, such as restricting access to the network databases, sending the results of searches over the network by using SSL, or using more advanced features such as Kerberos secured searches, then LDAP naming services should be used.

For example, when you add a new user to a network running NIS, you only have to update the input file in the master server and run the `make`command. This command automatically updates the `passwd.byname` and `passwd.byuid` maps. These maps are then transferred to the slave servers and are available to all of the domain's client systems and their programs. When a client system or application requests information by using the user name or UID, the NIS server refers to the `passwd.byname` or `passwd.byuid` map, as appropriate, and sends the requested information to the client.

You can use the `ypcat` command to display the values in a map.

```
$ ypcat mapname
```

where *mapname* is the name of the map you want to examine or its *nickname*. If a map is composed only of keys, as in the case of `ypservers`, use the `ypcat -k` command. Otherwise, `ypcat` prints blank lines. For the list of `ypcat` command options, see the `ypcat`(1) man page.

You can use the `ypwhich` command to determine which server is the master of a particular map.

```
$ ypwhich -m mapname
```

where *mapname* is the name or the nickname of the map whose master you want to find. The output of the `ypwhich` command displays the name of the master server. For more information, see the `ypwhich(1)` man page.

NIS Map Nicknames

Nicknames are aliases for full map names, such as `passwd` for `passwd.byname`. To obtain a list of available map nicknames, type `ypcat -x` or `ypwhich -x`.

Nicknames are stored in the `/var/yp/nicknames` file, which contains a map nickname followed by the fully specified name for the map, separated by a space. You can modify and update this file. Currently, the limit is 500 nicknames.

NIS Binding

NIS clients are connected to an NIS server through the binding process. The `svc:/network/nis/client` and `svc:/network/nis/domain` services support the binding process. You must enable these services for any NIS service to operate. The `svc:/network/nis/client` service can work in one of two modes: server-list or broadcast.

- Server-list – In the server-list mode, the `ypbind` process queries the `svc:/network/nis/domain` service for the names of all NIS servers in the domain. The `ypbind` process binds only to servers in this file.

  You can use the `svccfg` command to add NIS servers to any domain. They are added to the `config/ypservers` property in the `svc:/network/nis/domain` service. Each property value represents a specific NIS server.

  Additionally, any server name that is specified in the `svc:/network/nis/domain` service must contain an entry in the `/etc/inet/hosts` file for NIS binding to function.
- Broadcast – The `ypbind` process can also use an RPC broadcast to initiate a binding. Because broadcasts are only local subnet events that are not routed further, there must be at least one server (master or slave) on the same subnet as the client. The servers themselves might exist throughout different subnetworks because map propagation works across subnet boundaries. In a subnet environment, one common method is to make the subnet router an NIS server. This enables the domain server to serve clients on either subnet interface.

  Broadcast mode is generally the recommended mode of operation. Broadcast mode does not require additional host entries to be specified or changes to be made to `/etc/inet/hosts`.

Normally, after a client is bound to a server, it stays bound to that server until something causes the binding to change. For example, if a server goes out of service, the clients it served will then bind to new servers.

To determine which NIS server is currently providing service to a specific client, use the following command.

```
$ ypwhich hostname
```

where *hostname* is the name of the client. If no host name is specified, the `ypwhich` command defaults to the local system (the system on which the command is run).

Server-List Mode

The binding process in server-list mode works as follows:

1. Any program, running on the NIS client system that requires information provided by an NIS map, requests the `ypbind` daemon for the name of a server.
2. The `ypbind` daemon searches the `/var/yp/binding/`*domainname*`/ypservers` file for a list of NIS servers for the domain.
3. The `ypbind` daemon initiates binding to the first server in the list. If the server does not respond, `ypbind` tries the second, and so on, until it finds a server or exhausts the list.
4. The `ypbind` daemon sends the information about the server to the client process. The client then sends the request directly to the server.
5. The `ypserv` daemon on the NIS server handles the request by consulting the appropriate map.
6. The `ypserv` daemon sends the requested information back to the client.

Broadcast Mode

The broadcast mode binding process works as follows:

1. You must start the `ypbind` daemon with the `broadcast` option set.
2. The `ypbind` daemon issues an RPC broadcast in search of an NIS server.

Note - In order to support clients using the `broadcast` option, it is necessary to have an NIS server on each subnet that requires NIS service.

3. The `ypbind` daemon initiates binding to the first server that responds to the broadcast.
4. The `ypbind` daemon sends the information about the server to the client. The client then sends the request directly to the server.
5. The `ypserv` daemon on the NIS server handles the request by consulting the appropriate map.
6. The `ypserv` daemon sends the requested information back to the client.

♦ ♦ ♦ CHAPTER 6

Setting Up and Configuring Network Information Service

This chapter describes the initial set up and configuration of the Network Information Service (NIS).

This chapter covers the following topics:

- “Configuring NIS” on page 67
- “Planning Your NIS Domain” on page 69
- “Preparing the NIS Master Server” on page 70
- “Starting and Stopping NIS Services on an NIS Server” on page 77
- “Setting Up NIS Slave Servers” on page 80
- “Administering NIS Clients” on page 85

Screen messages or NIS map names might use "host name" or "machine name".

Configuring NIS

Before configuring your NIS namespace, you must do the following:

- Plan your NIS domain. See “Planning Your NIS Domain” on page 69 for details.
- Install properly configured name service switch information on all the systems that will be using NIS. See Chapter 2, “About the Name Service Switch” for details.

The following procedures are used to configure NIS:

- “How to Prepare Source Files for Conversion to NIS Maps” on page 71
- “How to Set Up the NIS Master Server” on page 75
- “Starting and Stopping NIS Services on an NIS Server” on page 77
- “How to Set Up a Slave Server” on page 80
- “Administering NIS Clients” on page 85

NIS and the Service Management Facility

Service Management Facility (SMF) manages the NIS service. For an overview of SMF, refer to Chapter 1, “Introduction to the Service Management Facility” in *Managing System Services in Oracle Solaris 11.4*. For more information, see `svcadm`(8) and `svcs`(1) man pages.

The following list provides a short overview of some of the important information needed to use the SMF service to administer NIS:

- You can use the `svcadm`Administrative actions on this service, such as enabling, disabling, or restarting, can be performed by using the command. However, `ypstart` and `ypstop` can also be used from the command line to start or stop NIS. See the `ypstart`(8) and `ypstop`(8) man pages for more information.

  Tip - Temporarily disabling a service by using the `-t` option provides some protection for the service configuration. If the service is disabled with the `-t` option, the original settings would be restored for the service after a reboot. If the service is disabled without `-t`, the service will remain disabled after reboot.

- The NIS Fault Manager Resource Identifiers (FMRIs) are:
  - `svc:/network/nis/server` for the NIS server
  - `svc:/network/nis/client` for the NIS client
  - `svc:/network/nis/domain` for the domain name
- You can query the status of the NIS service by using the `svcs` command.
  - The following are examples of the `svcs` command and its output:

    ```
    $ svcs network/nis/server
    STATE        STIME    FMRI
    online       Jan_10   svc:/network/nis/server:default

    $ svcs \*nis\*
    STATE          STIME    FMRI
    online         Oct_09   svc:/network/nis/domain:default
    online         Oct_09   svc:/network/nis/client:default
    ```

  - The following is an example of the `svcs -l` command and its output:

    ```
    $ svcs -l /network/nis/client
    fmri         svc:/network/nis/client:default
    name         NIS (YP) client
    enabled      true
    state        online
    ```

```
next_state   none
state_time   Tue Aug 23 19:23:28 2011
logfile      /var/svc/log/network-nis-client:default.log
restarter    svc:/system/svc/restarter:default
contract_id  88
manifest     /lib/svc/manifest/network/nis/client.xml
manifest     /lib/svc/manifest/network/network-location.xml
manifest     /lib/svc/manifest/system/name-service/upgrade.xml
manifest     /lib/svc/manifest/milestone/config.xml
dependency   require_all/none svc:/system/filesystem/minimal (online)
dependency   require_all/restart svc:/network/rpc/bind (online)
dependency   require_all/restart svc:/network/nis/domain (online)
dependency   optional_all/none svc:/network/nis/server (absent)
dependency   optional_all/none svc:/network/location:default (online)
dependency   optional_all/none svc:/system/name-service/upgrade (online)
dependency   optional_all/none svc:/milestone/config (online)
dependency   optional_all/none svc:/system/manifest-import (online)
dependency   require_all/none svc:/milestone/unconfig (online)
```

- You can use the `svccfg` utility to obtain more detailed information about a service. See the `svccfg(8)` man page.
- You can check a daemon's presence by using the `ps` command.

```
$ ps -ef |grep ypbind
daemon 100813   1   0   Aug 23 ?     0:00 /usr/lib/netsvc/yp/ypbind -broadcast
```

Planning Your NIS Domain

Before you configure systems as NIS servers or clients, you must plan the NIS domain.

Decide which systems will be in your NIS domain. An NIS domain does not have to mirror your DNS domain. A DNS domain can have more than one NIS domain, and systems can exist in your DNS domain that are outside of your NIS domain.

An NIS domain name can be 256 characters long. A good practice is to limit domain names to no more than 32 characters. NIS domain names are case-sensitive. For convenience, you can choose to use your Internet domain name as the basis for your NIS domain name. Be aware that users might become confused if the NIS domain name includes capitals, but the DNS domain name does not. For example, if your Internet domain name is `example.com`, you can also name your NIS domain `example.com.` If you wanted to divide `example.com` into two NIS domains, for example, one for the sales department and the other for the manufacturing department, you could name one domain `sales.example.com` and the other domain `manf.example.com`.

Note - Merging and administering split NIS domains can be very difficult, so ensure that you have a good reason to split an NIS domain.

Before a system can use NIS services, the correct NIS domain name and system name must be set. A system's name is set by the `hostname` command. The system's domain name is set by the `domainname` command. The `hostname` and `domainname` commands can be used to display the host name and the NIS domain name.

Identify Your NIS Servers and Clients

Select one physical machine to be the master server. Decide which physical machines will be slave servers.

Decide which systems will be NIS clients. Typically, all systems in your NIS domain are set to be NIS clients, although this is not necessary.

Preparing the NIS Master Server

The following sections describe how to prepare the source files and the `passwd` files for the master server:

The following tasks prepare the NIS master server:

- "How to Prepare Source Files for Conversion to NIS Maps" on page 71
- "How to Install the NIS Master Server Package" on page 75
- "How to Set Up the NIS Master Server" on page 75
- "How to Support Multiple NIS Domains on One Master Server" on page 77

Source Files Directory for NIS

The source files are typically located in the `/etc` directory on the master server. However, leaving them in `/etc` is undesirable because the contents of the maps are then the same as the contents of the local files on the master server. This is a special problem for `passwd` and `shadow` files because all users have access to the master server maps and the `root` password would be passed to all NIS clients through the `passwd` map. See "`passwd` Files and NIS Namespace Security" on page 71 for additional information.

However, if you put the source files in some other directory, you must modify the Makefile in /var/yp by changing the DIR=/etc line to DIR=/*your-choice*, where *your-choice* is the name of the directory you will be using to store the source files. This allows you to treat the local files on the server as if they were those of a client. (It is good practice to first save a copy of the original Makefile.)

In addition, the audit_user, auth_attr, exec_attr, and prof_attr NIS maps should be created from a directory other than the default. Amend /var/yp/Makefile by changing RBACDIR =/etc/security to RBACDIR=/your-choice.

passwd Files and NIS Namespace Security

For security reasons, the files used to build the NIS password maps should not contain an entry for root, to prevent unauthorized root access. Therefore, the password maps should not be built from the files located in the master server's /etc directory. The password files used to build the password maps should have the root entry removed from them and be located in a directory that can be protected from unauthorized access.

For example, the master server password input files should be stored in a directory such as /var/yp, or any directory of your choice, as long as the file itself is not a link to another file and its location is specified in the Makefile. The correct directory option is set automatically according to the configuration specified in your Makefile.

Caution - Be sure that the passwd file in the directory specified by PWDIR does not contain an entry for root.

If your source files are in a directory other than /etc, you must alter the PWDIR password macro in /var/yp/Makefile to refer to the directory where the passwd and shadow files reside. You change the line PWDIR=/etc to PWDIR=/*your-choice*, where *your-choice* is the name of the directory you that will use to store the passwd map source files.

▼ How to Prepare Source Files for Conversion to NIS Maps

This procedure explains how to prepare the source files for conversion to NIS maps.

1. **Become an administrator.**

For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Check the source files on the master server to make sure that they reflect your system.**

   Check the following files:

   - audit_user
   - auth_attr
   - auto.home or auto_home
   - auto.master or auto_master
   - bootparams
   - ethers
   - exec_attr
   - group
   - hosts
   - ipnodes
   - netgroup
   - netmasks
   - networks
   - passwd
   - protocols
   - rpc
   - service
   - shadow
   - user_attr

3. **Copy all of these source files, except for passwd and shadow, to the source directory that you have selected.**

   The source directory is defined in /var/yp/Makefile by the DIR macro.

4. **Copy the passwd and shadow files to the password source directory that you have selected.**

   The password source directory is defined in the Makefile by the PWDIR macro.

5. **Copy the audit_user, auth_attr, exec_attr, and prof_attr files to the rights source directory that you have selected.**

The rights source directory is defined in `/var/yp/Makefile` by the `RBACDIR` macro. If desired, merge the contents of the files in the `/etc/security/auth_attr.d` directory into a copy of the `auth_attr` file before copying it. Similarly, combine the files in the `exec_attr.d` and `prof_attr.d` directories with `exec_attr` and `prof_attr`, if desired.

Caution - Because these files will need to be remerged any time the system is upgraded, keep the local files separate from the release files in the`/etc/security/*.d` directories.

6. **Check the `/etc/mail/aliases` file.**

   Unlike other source files, the `/etc/mail/aliases` file cannot be moved to another directory. This file must reside in the `/etc/mail` directory. Refer to the `aliases`(5) man page for more information.

   Note - You can add an NIS-specific mail aliases file by pointing the `ALIASES = /etc/mail/aliases` entry in `/var/yp/Makefile` to another location. When you then run the `make` command, the `ALIASES` entry creates a `mail.aliases` map. The `sendmail` service uses this map in addition to the `/etc/mail/aliases` file when the `/etc/nsswitch.conf` file properly targets `nis` in addition to `files`. Refer to "Modifying and Using `/var/yp/Makefile`" on page 97.

7. **Clean all comments and other extraneous lines and information from the source files.**

   These operations can be done through a `sed` or `awk` script or with a text editor. `/var/yp/Makefile` performs some file cleaning automatically for you, but it is good practice to manually examine and clean these files before running the `make` command.

8. **Make sure that the data in all the source files is correctly formatted.**

   Source file data must be in the correct format for that particular file. Check the man pages for the different files to make sure that each file is in the correct format.

Preparing `/var/yp/Makefile`

After checking the source files and copying them into the source file directory, you now need to convert those source files into the `ndbm` format maps that the NIS service uses. This is done automatically for you by `ypinit` when called on the master server, as explained in "How to Set Up the NIS Master Server" on page 75.

The ypinit script calls the make program, which uses /var/yp/Makefile. A default copy of the file is provided for you in the /var/yp directory and contains the commands needed to transform the source files into the desired ndbm format maps.

You can use the default Makefile as is, or modify it. If you do modify the default Makefile, be sure to first copy and store the original default Makefile in case you need it for future use. You might need to make one or more of the following modifications to the Makefile:

- *Nondefault maps*

  If you have created your own non-default source files and want to convert them to NIS maps, you must add those source files to the Makefile.
- DIR *value*

  If you want the Makefile to use source files stored in some directory other than /etc, as explained in "Source Files Directory for NIS" on page 70, you must change the value of DIR in the Makefile to the directory that you want to use. When changing this value in the Makefile, do not indent the line.
- PWDIR *value*

  If you want the Makefile to use the passwd, shadow, and adjunct source files that are stored in some directory other than /etc, you must change the value of PWDIR in the Makefile to the directory that you want to use. When changing this value in the Makefile, do not indent the line.
- RBACDIR *value*

  If you want the Makefile to use the audit_user, auth_attr, exec_attr, and prof_attr source files that are stored in some directory other than /etc, you must change the value of RBACDIR in the Makefile to the directory that you want to use. When changing this value in the Makefile, do not indent the line.
- *Domain name resolver*

  If you want the NIS server to use the domain name resolver for systems not in the current domain, comment out the Makefile line B=, and uncomment (activate) the line B=-b.

The function of the Makefile is to create the appropriate NIS maps for each of the databases listed under all. After passing through makedbm the data is collected in two files, mapname.dir and mapname.pag. Both files are in the /var/yp/*domainname* directory on the master server.

The Makefile builds passwd maps from the /PWDIR/passwd, /PWDIR/shadow, and /PWDIR/security/passwd.adjunct files, as appropriate.

▼ How to Install the NIS Master Server Package

Normally, the NIS master server package is installed when appropriate with the Oracle Solaris release. If the package was not included when the system was installed, use the following procedure to install the package.

1. **Become an administrator.**
   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Install the NIS master server package.**

```
$ pkg install pkg:/service/network/nis
```

▼ How to Set Up the NIS Master Server

The `ypinit` script sets up the master server and the slave servers and clients to use NIS. It also initially runs the `make` command to create the maps on the master server.

To use the `ypinit` command to build a fresh set of NIS maps on the master server, complete the following procedure.

1. **Become an administrator on the NIS master server.**
   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Edit the `/etc/inet/hosts` file.**
   Add the host name and IP address of each NIS server. Use the following format: *IPaddress FQDN-hostname aliases*.
   For example:

```
192.0.2.1 master.example.com master
192.0.2.2 slave1.example.com slave1
192.0.2.3 slave2.example.com slave2
```

3. **Build new maps on the master server.**

```
# /usr/sbin/ypinit -m
```

4. **Type the names of the NIS servers.**

When ypinit prompts for a list of other systems to become NIS slave servers, type the name of the server you are working on, along with the names of the NIS slave servers that you specified in the /etc/inet/hosts file.

5. **Verify that the NIS domain name is set.**

```
# domainname
example.com
```

6. **Type y to select to stop the process if a nonfatal error occurs.**

When ypinit asks whether you want the procedure to terminate at the first nonfatal error or continue despite nonfatal errors, type **y**. When you choose **y**, ypinit exits upon encountering the first problem. You can then fix it and restart ypinit. This is recommended if you are running ypinit for the first time. If you prefer to continue, you can try to manually fix all problems that occur, and then restart ypinit.

Note - A nonfatal error can appear when some of the map files are not present. This is not an error that affects the functioning of NIS. You might need to add maps manually if they were not created automatically. Refer to "Default NIS Maps" on page 61 for a description of all default NIS maps.

7. **Choose if the source files should be deleted.**

The ypinit command asks whether the existing files in the /var/yp/*domain-name* directory can be destroyed. This message is displayed only if NIS is already installed. Typically, you would choose to delete the source files to clean up the files from a previous installation.

8. **After the ypinit command has constructed the list of servers, it invokes the make command.**

This program uses the instructions contained in the Makefile (either the default file or the one you modified) located in /var/yp. The make command cleans any remaining comment lines from the files that you designated. It also runs makedbm on the files, creating the appropriate maps and establishing the name of the master server for each map.

If the map or maps being pushed by the Makefile correspond to a domain other than the one returned by the domainname command on the master, you can make sure that they are pushed to the correct domain by starting make in the ypinit shell script with a proper identification of the variable DOM, as follows:

```
# make DOM=domain-name  passwd
```

This command pushes the passwd map to the intended domain, instead of the domain to which the master belongs.

9. **If needed, make changes to the name service switch.**

   See "Configuring the Name Service Switch" on page 30.

▼ How to Support Multiple NIS Domains on One Master Server

Normally, an NIS master server supports only one NIS domain. However, if you are using a master server to support multiple domains, you must slightly modify the steps, as described in "How to Set Up the NIS Master Server" on page 75, when setting up the server to serve the additional domains.

1. **Become an administrator on the NIS master server.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Change the NIS domain name.**

   ```
   # domainname sales.example.com
   ```

3. **Build the NIS files.**

   ```
   # make DOM=sales.example.com
   ```

Starting and Stopping NIS Services on an NIS Server

Now that the master maps are created, you can start the NIS daemons on the master server and begin service. When you enable the NIS service, the `ypserv` and `ypbind` daemons start on the server. When a client requests information from the server, `ypserv` is the daemon that responds to information requests from clients after looking them up in the NIS maps. The `ypserv` and `ypbind` daemons are administered as a unit.

The following are the three ways that the NIS service can be started or stopped on a server:

- The SMF service automatically starts the NIS service during the boot process, if the NIS service was enabled previously.

  When the `svc:/network/nis/server` service is enabled, then the `ypserv` daemon is automatically started up at boot. See "How to Set Up the NIS Master Server" on page 75 for more information.

- Using the `svcadm enable` *fmri* and `svcadm disable` *fmri* commands is the preferred manual method.
- The `ypstart` and `ypstop` commands, provide another manual method, although the `svcadm` command is preferred so that you can use SMF to administer the NIS service.

Starting and Stopping NIS Services on an NIS Server

The following tasks use the `svcadm` command for starting and stopping the NIS services:

- "How to Enable the NIS Server Services Manually" on page 78
- "How to Disable the NIS Server Services" on page 79
- "How to Refresh the NIS Server Service" on page 79

▼ How to Enable the NIS Server Services Manually

When you use the `svcadm` command, the instance name is required only if you are running more than one instance of the service. For more information, see "NIS and the Service Management Facility" on page 68, or the `svcadm`(8) man page.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Start the required NIS server services.**

   ```
   # svcadm enable network/nis/domain
   # svcadm enable network/nis/server
   ```

   Note - The NIS service can also be enabled by using the `ypstart` command although the `svcadm` command is preferred.

How to Disable the NIS Server Services

When you use the svcadm command, a specific instance name is required only if you are running more than one instance of the service. For more information, see "NIS and the Service Management Facility" on page 68, or the svcadm(8) man page.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Disable the required NIS server services.**

```
# svcadm disable network/nis/domain
# svcadm disable network/nis/server
```

Note - The NIS service can also be disabled using the ypstop command.

How to Refresh the NIS Server Service

This procedure explains how to refresh the NIS server services after a configuration change has been made.

When you use the svcadm command, a specific instance name is required only if you are running more than one instance of the service. For more information, see "NIS and the Service Management Facility" on page 68, or the svcadm(8) man page.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Refresh the required NIS server services.**

```
# svcadm refresh network/nis/domain
# svcadm refresh network/nis/server
```

Setting Up NIS Slave Servers

Your network can have one or more slave servers. Having slave servers ensures the continuity of NIS services when the master server is not available.

The following tasks set up NIS slave servers:

- "How to Set Up a Slave Server" on page 80
- "How to Start NIS on a Slave Server" on page 82
- "How to Add a New Slave Server" on page 82

Preparing a Slave Server

Before actually running the `ypinit` command to create the slave servers, first make sure that the `svc:/network/nis/domain` service has been configured.

Note - NIS domain names are case-sensitive, although DNS domain names are not.

Make sure that the network is working properly before you configure an NIS slave server. In particular, make sure that you can use the `sshd` command to send files from the master NIS server to NIS slaves.

▼ How to Set Up a Slave Server

Repeat this procedure for each system that you want to be an NIS slave server.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Edit the `/etc/inet/hosts` file.**

   Add the name and IP address of each of the other NIS servers. Use the following format: *IPaddress FQDN-hostname aliases*.

   For example:

```
192.0.2.1 master.example.com master
192.0.2.2 slave1.example.com slave1
192.0.2.3 slave2.example.com slave2
```

3. **Change directory to `/var/yp` on the slave server.**

   Note - You must first configure the new slave server as an NIS client so that it can obtain the NIS maps from the master server for the first time. See "Administering NIS Clients" on page 85 for details. After an NIS master map is changed, use the `yppush` command to propagate the new map to the NIS slave server. For information about propagating an NIS map, see "Updating and Modifying Existing NIS Maps" on page 100.

4. **Initialize the slave server as an NIS client.**

   ```
   # /usr/sbin/ypinit -c
   ```

   The `ypinit` command prompts you for a list of NIS servers. Type the name of the local slave you are working on first, then type the name of the master server, followed by names of the other NIS slave servers in your domain. For the other slave servers, follow the order from the physically closest to the furthest in network terms.

5. **Determine if the client services are running, then start or restart the services as needed.**

   ```
   # svcs \*nis\*
   STATE          STIME    FMRI
   online         20:32:56 svc:/network/nis/domain:default
   online         20:32:56 svc:/network/nis/client:default
   ```

   If the services are displayed with an `online` state, then NIS is running. If the service state is `disabled`, then NIS is not running.

   a. **If the client services are running, restart the client services.**

      ```
      # svcadm restart network/nis/domain
      # svcadm restart network/nis/client
      ```

   b. **If the client services are not running, start the client services.**

      ```
      # svcadm enable network/nis/domain
      # svcadm enable network/nis/client
      ```

6. **Determine if the NIS master server is running, then start or restart the service as needed.**

```
# svcs network/nis/server
STATE          STIME     FMRI
offline        20:32:56  svc:/network/nis/server:default
```

a. **If the master NIS server is running, restart the service.**

   ```
   # svcadm restart network/nis/server
   ```

b. **If the master NIS server is not running, start the service.**

   ```
   # svcadm enable network/nis/server
   ```

7. **Initialize this system as a slave server.**

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

   where *master* is the system name of the existing NIS master server.

How to Start NIS on a Slave Server

The following procedure explains how to start NIS on a slave server.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Restart the client service and start all NIS server processes.**

   ```
   # svcadm restart network/nis/domain
   # svcadm restart network/nis/client
   # svcadm enable network/nis/server
   ```

How to Add a New Slave Server

After NIS is running, you might need to create an NIS slave server that you did not include in the initial list given to the `ypinit` command. Use this procedure to add a new NIS slave server.

1. **Become an administrator on the NIS master server.**

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Change to the NIS domain directory.**

   ```
   # cd /var/yp/domainname
   ```

3. **Disassemble the `ypservers` file.**

   ```
   # makedbm -u ypservers >/tmp/temp_file
   ```

   The `makedbm` command converts `ypservers` from `ndbm` format to a temporary ASCII file `/tmp/temp_file`.

4. **Edit the `/tmp/temp_file` file.**

   Add the name of the new slave server to the list of servers. Then, save and close the file.

5. **Run the `makedbm` command with `temp_file` as the input file and `ypservers` as the output file.**

   ```
   # makedbm /tmp/temp_file ypservers
   ```

   The `makedbm` command then converts `ypservers` back into `ndbm` format.

6. **Verify that the `ypservers` map is correct.**

   Because there is no ASCII file for `ypservers`, type the following on the slave server:

   ```
   slave3# makedbm -u ypservers
   ```

   The `makedbm` command displays each entry in `ypservers` on your screen.

   Note - If a system name is not in `ypservers`, it will not receive updates to the map files because `yppush` consults this map for the list of slave servers.

   ---

7. **Become an administrator on the new NIS slave server.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

8. **Verify that the NIS domain name is set.**

   ```
   # domainname
   example.com
   ```

9. **Set up the new slave server's NIS domain directory.**

Copy the NIS map set from the master server, then start the NIS client. When running the `ypinit` command, follow the prompts and list the NIS servers in order of preference.

```
slave3# cd /var/yp
slave3# ypinit -c
```

10. **Initialize this system as a slave.**

```
slave3# /usr/sbin/ypinit -s ypmaster
```

where *ypmaster* is the system name of the existing NIS master server.

11. **Stop the system running as an NIS client.**

```
slave3# svcadm disable network/nis/client
```

12. **Determine if the client services are running, then start or restart the services as needed.**

```
# svcs \*nis\*
STATE          STIME    FMRI
online         20:32:56  svc:/network/nis/domain:default
online         20:32:56  svc:/network/nis/client:default
```

If the services are displayed with an `online` state, then NIS is running. If the service state is `disabled`, then NIS is not running.

a. **If the client services are running, restart the client services.**

```
# svcadm restart network/nis/domain
# svcadm restart network/nis/client
```

b. **If the client services are not running, start the client services.**

```
# svcadm enable network/nis/domain
# svcadm enable network/nis/client
```

13. **Determine if the NIS server is running, then start or restart the service as needed.**

```
# svcs network/nis/server
STATE          STIME    FMRI
offline        20:32:56  svc:/network/nis/server:default
```

a. **If the NIS server is running, restart the service.**

```
slave3# svcadm restart network/nis/server
```

b. **If the NIS server is not running, start the service.**

```
slave3# svcadm enable network/nis/server
```

Administering NIS Clients

You can choose between two methods for configuring a client system to use NIS as its naming service.

- **Broadcast method** – Preferred method of configuring a client system to use NIS. See "How to Configure an NIS Client in Broadcast Mode" on page 85 for instructions.
- **Server-list method** – Less-preferred method for configuring a client system by using the `ypinit` command to specify the servers. See "How to Configure an NIS Client Using Specific NIS Servers" on page 86 for instructions.

Note - The Oracle Solaris OS does not support a configuration in which an NIS client and a native LDAP client coexist on the same client system.

▼ How to Configure an NIS Client in Broadcast Mode

When you start the `nis/client` service, the service runs the `ypbind` command, which searches the local subnet for an NIS server. If a subnet is found, `ypbind` binds to it. This search is referred to as *broadcasting*. If no NIS server exists on the client's local subnet, `ypbind` fails to bind and the client system cannot obtain namespace data from the NIS service. See "How to Configure an NIS Client Using Specific NIS Servers" on page 86 for instructions.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Set the NIS domain name.**

   ```
   # domainname

   example.com
   ```

3. **If needed, make changes to the name service switch.**

See "Configuring the Name Service Switch" on page 30.

4. **Start the NIS client services.**

```
# svcadm enable network/nis/domain
# svcadm enable network/nis/client
```

▼ How to Configure an NIS Client Using Specific NIS Servers

Before You Begin The following procedure requires that the host names in the client configuration script are resolved. If you intend to provide a host name instead of an IP address, ensure that there are appropriate entries for each NIS server in the `/etc/hosts` file on the client before performing this procedure. For more information, see the `ypinit`(8) man page.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Set the NIS domain name.**

```
# domainname

example.com
# svcadm enable network/nis/domain
```

3. **Run the client configuration script.**

```
# ypinit -c
```

   You are prompted to name the NIS servers from which the client obtains naming service information. You can list the master server and as many slave servers as you want. The servers that you list can be located anywhere in the domain. It is a better practice to first list the servers closest (in network terms) to the physical machine, than those servers that are located on more distant parts of the network.

▼ How to Disable the NIS Client Services

1. **Become an administrator.**

For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Disable the NIS client services.**

```
# svcadm disable network/nis/domain
# svcadm disable network/nis/client
```

♦ ♦ ♦ CHAPTER 7

Administering the Network Information Service

This chapter describes how to administer NIS and covers the following topics:

Note - The NIS service is managed by the Service Management Facility. Administrative actions on this service, such as enabling, disabling, or restarting, can be performed by using the `svcadm` command. See "NIS and the Service Management Facility" on page 68 for more information about using SMF with NIS. For an overview of SMF, refer to Chapter 1, "Introduction to the Service Management Facility" in *Managing System Services in Oracle Solaris 11.4*. Also refer to the `svcadm`(8) and `svcs`(1) man pages for more details.

NIS services can also be started and stopped by using the `ypstart` and `ypstop` commands. See the `ypstart`(8) and `ypstop`(8) man pages for more information.

Password Files and NIS Namespace Security

For security reasons, follow these guidelines:

- Limit access to the NIS maps on the master server.
- Do not place an entry for `root` in the files you use to build the NIS password maps. This protects the NIS network against unauthorized access. To accomplish this, remove the `root` entry before building the password maps. Locate the password files in a directory other than the master server's `/etc` directory. The `/etc` directory on every system on the network should be secured against unauthorized access.

For example, the master server password input files could be stored in a directory such as `/var/yp`, or any directory of your choice, as long as the file itself is not a link to another file and is specified in the Makefile. When you use either the Service Management Facility or the `ypstart` script to start the NIS service, the correct directory option is set according to the configuration specified in your `Makefile`.

Administering NIS Users

This section includes information about setting user passwords, adding new users to an NIS domain, and assigning users to `netgroups`.

▼ How to Add a New NIS User to an NIS Domain

1. **Become an administrator on the NIS master server.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Create the new user's login ID with the `useradd` command.**

   ```
   # useradd userID
   ```

   where *userID* is the login ID of the new user. This command creates entries in the `/etc/passwd` and `/etc/shadow` files on the master NIS server.

3. **Create the new user's initial password.**

   To create an initial password that the new user can use to log in, run the `passwd` command.

   ```
   # passwd userID
   ```

   where *userID* is the login ID of the new user. You will be prompted for the password to assign to this user.

   This step is necessary because the password entry created by the `useradd` command is locked, which means that the new user cannot log in. By specifying an initial password, you unlock the entry.

4. **Copy the new entry into the master server's `passwd` map input files.**

   The map source files on your master server should be in a directory other than `/etc`. Copy and paste the new lines from the `/etc/passwd` and `/etc/shadow` files into the `passwd` map input files on the server. See "Password Files and NIS Namespace Security" on page 89 for additional information.

For example, if you added the new user `brown`, the line from `/etc/passwd` that you would copy to your `passwd` input file would look like the following.

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

The line for `brown` that you would copy from `/etc/shadow` would look like:

```
brown:$5$YiFpYWXb$6jJkG/gKdfkKtlTbemORnbeH.qsvO9MwBD3ulTihq9B:6445::::::
```

5. **Make sure that the `Makefile` correctly specifies the directory where the password input file resides.**

6. **Delete the new user's entries from the`/etc/passwd` and `/etc/shadow` input files.**

   For security reasons, do not keep user entries in the NIS master server `/etc/passwd` and `/etc/shadow` files. After copying the entries for the new user to the NIS map source files that are stored in some other directory, use the `userdel` command on the master server to delete the new user.

   For example, to delete the new user `brown` from the master server's `/etc` files, you would enter the following.

   ```
   # userdel brown
   ```

   For more information about `userdel`, see the `userdel`(8) man page.

7. **Update the NIS `passwd` maps.**

   After you have updated the `passwd` input file on the master server, update the `passwd` maps by running `make` in the directory containing the source file.

   ```
   # userdel brown
   # cd /var/yp
   # make passwd
   ```

8. **Tell the new user the initial password you have assigned to his or her login ID.**

   After logging in, the new user can run `passwd` at any time to establish a different password.

Setting NIS User Passwords

Users run `passwd` to change their passwords.

```
$ passwd username
```

Before users can change their passwords, you must start the `rpc.yppasswdd` daemon on the master server to update the password file.

The `rpc.yppasswdd` daemon starts automatically on the master server. Notice that when the `-m` option is given to `rpc.yppasswdd`, the `make` command is run in `/var/yp` immediately following a modification of the file. If you want to avoid having the `make` command run each time the `passwd` file is changed, remove the `-m` option from the `rpc.yppasswd` command in the `ypstart` script and control the pushing of the `passwd` maps through the `crontab` file.

NIS Netgroups

NIS netgroups are groups (sets) of users or systems that you define for your administrative purposes. For example, you can create netgroups that do the following.

- Define a set of users who can access a specific system
- Define a set of NFS client systems to be given some specific file system access
- Define a set of users who are to have administrator privileges on all the systems in a particular NIS domain

Each netgroup is given a netgroup name. Netgroups do not directly set permissions or access rights. Instead, the netgroup names are used by other NIS maps in places where a user name or system name would normally be used. For example, suppose you created a netgroup of network administrators called `netadmins`. To grant all members of the `netadmins` netgroup access to a given system, you only need to add a `netadmin` entry to that system's `/etc/passwd` file. Netgroup names can also be added to the `/etc/netgroup` file and propagated to the NIS `netgroup` map. See the `netgroup(5)` man page for more detailed information about using netgroups.

On a network using NIS, the `netgroup` input file on the master NIS server is used for generating three maps: `netgroup`, `netgroup.byuser`, and `netgroup.byhost`. The `netgroup` map contains the basic information in the `netgroup` input file. The two other NIS maps contain information in a format that speeds lookups of netgroup information, given the system or user name.

Entries in the `netgroup` input file are in the format: *name ID*, where *name* is the name you give to a netgroup, and *ID* identifies a system or user who belongs to the netgroup. You can specify as many IDs (members) to a netgroup as you want, separated by commas. For example, to create a netgroup with three members, the `netgroup` input file entry would be in the format: *name ID*, *ID*, *ID*. The member IDs in a `netgroup` input file entry are in the following format.

```
([-|system], [-|user], [domain])
```

Where *system* is a system name, *user* is a user ID, and *domain* is the system or user's NIS domain. The *domain* element is optional and should only be used to identify systems or users in some other NIS domain. The *system* and *user* element of each member's entry are required, but a dash (-) is used to denote a null. There is no necessary relationship between the system and user elements in an entry.

The following are two sample netgroup input file entries, each of which create a netgroup named admins composed of the users hauri and soloh who is in the remote domain sales and the systems altair and sirius.

```
admins (altair,hauri,) (sirius,soloh,sales)
admins (altair,-,) (sirius,-,) (-,hauri,) (-,soloh,sales)
```

Various programs use the netgroup NIS maps for permission checking during login, remote mount, remote login, and remote shell creation. These programs include mountd, and login. The login command consults the netgroup maps for user classifications if it encounters netgroup names in the passwd database. The mountd daemon consults the netgroup maps for system classifications if it encounters netgroup names in the /etc/dfs/dfstab file. In fact, any program that uses the ruserok interface checks the netgroup maps for both system and user classifications if they encounter netgroup names in the /etc/hosts.equiv or .rhosts file.

If you add a new NIS user or system to your network, be sure to add them to appropriate netgroups in the netgroup input file. Then use the make and yppush commands to create the netgroup maps and push them to all of your NIS servers. See the netgroup(5) man page for detailed information about using netgroups and netgroup input file syntax.

Working With NIS Maps

This section contains the following information:

- "Obtaining NIS Map Information" on page 93
- "Changing an NIS Map's Master Server" on page 94
- "Modifying NIS Configuration Files" on page 96
- "Modifying and Using /var/yp/Makefile" on page 97

Obtaining NIS Map Information

Users can obtain information from and about the maps at any time by using the ypcat, ypwhich, and ypmatch commands. In the examples that follow, *mapname* refers both to the official name of a map and to its nickname, if any.

To list all the values in a map, type the following:

```
$ ypcat mapname
```

To list both the keys and the values (if any) in a map, type the following:

```
$ ypcat -k mapname
```

To list all the map nicknames, type any of the following commands:

```
$ ypcat -x
$ ypmatch -x
$ ypwhich -x
```

To list all the available maps and their masters, type the following:

```
$ ypwhich -m
```

To list the master server for a particular map, type the following:

```
$ ypwhich -m mapname
```

To match a key with an entry in a map, type the following:

```
$ ypmatch key mapname
```

If the item you are looking for is not a key in a map, type the following:

```
$ ypcat mapname | grep item
```

where *item* is the information for which you are searching. To obtain information about other domains, use the `-d domainname` option of these commands.

If the system requesting information for a domain other than its default does not have a binding for the requested domain, `ypbind` consults the `/var/yp/binding/`*domainname*`/ypservers` file for a list of servers for that domain. If this file does not exist it issues an RPC broadcast for a server. In this case, there must be a server for the requested domain on the same subnet as the requesting system.

Changing an NIS Map's Master Server

To change the master server for a selected map, you first build the map on the new NIS master. Since the old master server name occurs as a key-value pair in the existing map (this pair is inserted automatically by makedbm), copying the map to the new master or transferring a copy to the new master with ypxfr is insufficient. You have to reassociate the key with the new master server name. If the map has an ASCII source file, you should copy this file to the new master.

▼ How to Change an NIS Map's Master Server

1. **Become an administrator on the NIS master server.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Change directories.**

   ```
   newmaster# cd /var/yp
   ```

3. **The /var/yp/Makefile must have an entry for the new map before you specify the map to make.**

   If this is not the case, edit the Makefile now. For this example, add an entry for a map called sites.byname.

4. **To update or remake the map, type the following:**

   ```
   newmaster# make sites.byname
   ```

5. **If the old master remains an NIS server, remote log in (ssh) to the old master and edit /var/yp/Makefile.**

   Make sure that you comment out the section of the Makefile that made the sites.byname map so that it is no longer made there.

6. **If sites.byname only exists as an ndbm file, remake it on the new master server.**

   First, disassemble a copy of the sites.byname file by using the ypcat command. Then, run the disassembled version through makedbm.

   ```
   newmaster# cd /var/yp
   newmaster# ypcat sites.byname | makedbm domain/sites.byname
   ```

   After making the map on the new master, you must send a copy of the new map to the other slave servers. Do not use yppush, because the other slaves will try to get new copies from the old master, rather than the new one. A typical method for circumventing this is to transfer a copy of the map from the new master back to the old master. To do this, become superuser, or assume an equivalent role, on the old master server and type the following.

   ```
   oldmaster# /usr/lib/netsvc/yp/ypxfr -h newmaster sites.byname
   ```

Now it is safe to run yppush. Any remaining slave servers still believe that the old master is the current master and will attempt to get the current version of the map from the old master. When clients do so, they will get the new map, which names the new master as the current master.

If this method fails, you can log in as root on each NIS server and execute the ypxfr command as shown.

Modifying NIS Configuration Files

NIS intelligently parses the setup files. Although this makes NIS administration easier, it does make the behavior of NIS more sensitive to changes in the setup and configuration files.

Use the procedures in this section when doing any of the following:

- Editing /var/yp/Makefile to add or delete supported maps
- Adding or deleting $PWDIR/security/passwd.adjunct to allow or deny C2 security ($PWDIR is defined in /var/yp/Makefile)

▼ How to Modify NIS Configuration Files

Keep the following in mind:

- Deleting a map or source file from an NIS master server does not automatically result in corresponding deletions from slave servers. You must delete maps and source files from slave servers by hand.
- New maps do not automatically get pushed to existing slave servers. You must run ypxfr from the slaves.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Stop the NIS server.**

   ```
   # svcadm disable network/nis/server
   ```

3. **Make the necessary changes to your files.**

4. **Start the NIS server.**

```
# svcadm enable network/nis/server
```

Modifying and Using /var/yp/Makefile

You can modify the Makefile provided by default in /var/yp to suit your needs. You can add or delete maps, and you can change the names of some of the directories.

Tip - Keep an unmodified copy of the original Makefile for future reference.

Working With the NIS Makefile

To add a new NIS map, you must get copies of the ndbm files for the map into the /var/yp/*domainname* directory on each of the NIS servers in the domain. This is normally done for you by the Makefile. After deciding which NIS server is the master of the map, modify the Makefile on the master server so that you can conveniently rebuild the map. Different servers can be masters of different maps, but in most cases this leads to administrative confusion. Try to set only one server as the master of all maps.

Typically a human-readable text file is filtered through awk, sed, or grep to make it suitable for input to makedbm. Refer to the default Makefile for examples. See the make(1S) for general information about the make command.

Use the mechanisms already in place in the Makefile when deciding how to create dependencies that make will recognize. Be aware that make is very sensitive to the presence or absence of tabs at the beginning of lines within the dependency rules. A missing tab can invalidate an entry that is otherwise well formed.

Adding an entry to the Makefile involves performing the following steps:

- Adding the name of the database to the all rule
- Writing the time rule
- Adding the rule for the database

For example, in order for the Makefile to work on automounter input files, you would have to add the auto_direct.time and auto_home.time maps to the NIS database.

To add these maps to the NIS database you need to modify the Makefile.

Changing NIS Makefile Macros/Variables

You can change the settings of the variables defined at the top of the Makefile by changing the value to the right of the equal sign (=). For instance, if you do not want to use the files located in /etc as input for the maps, but you would rather use files located in another directory, such as /var/etc/domainname, you should change DIR from DIR=/etc to DIR=/var/etc/*domainname*. You should also change PWDIR from PWDIR=/etc to PWDIR=/var/etc/*domainname*.

The variables are the following:

- *DIR*= Is the directory containing all of the NIS input files except passwd and shadow. The default value is /etc. Since it is not good practice to use the files in the master server's /etc directory as NIS input files, you should change this value.
- *PWDIR*= Is the directory containing the passwd and shadow NIS input files. Since it is not good practice to use the files in the master server's /etc directory as NIS input files, you should change this value.
- *DOM*= Is the NIS domain name. The default value of DOM can be set by using the domainname command.

Modifying NIS Makefile Entries

The Makefile specifies the NIS databases.

▼ How to Modify /var/yp/Makefile to Use Specific Databases

This procedure requires that you have already configured an NIS master server.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Modify the line that starts with the word all by adding the names of the database you want to add:**

```
all: passwd group hosts ethers networks rpc services protocols \
netgroup bootparams aliases netid netmasks \
audit_user auth_attr exec_attr prof_attr \
auto_direct
```

The order of the entries is not relevant, but the blank space at the beginning of the continuation lines must be a Tab, not spaces.

3. **Add the following lines at the end of the Makefile:**

```
auto_direct: auto_direct.time
auto_home: auto_home.time
```

4. **Add an entry for auto_direct.time in the middle of the file.**

```
auto_direct.time: $(DIR)/auto_direct
@(while read L; do echo $$L; done < $(DIR)/auto_direct
$(CHKPIPE)) | \ (sed -e "/^#/d" -e "s/#.*$$//" -e "/^ *$$/d"
$(CHKPIPE)) | \ $(MAKEDBM) - $(YPDBDIR)/$(DOM)/auto_direct;
@touch auto_direct.time;
@echo "updated auto_direct";
@if [ ! $(NOPUSH) ]; then $(YPPUSH) auto_direct; fi
@if [ ! $(NOPUSH) ]; then echo "pushed auto_direct"; fi
```

where:

- CHKPIPE makes certain that the operations to the left of the pipe (|) are successfully completed before piping the results to the following commands. If the operations to the left of the pipe do not successfully complete, the process is terminated with an NIS make terminated message.
- NOPUSH prevents the makefile from calling yppush to transfer the new map to the slave servers. If NOPUSH is not set, the push is done automatically.

The while loop at the beginning is designed to eliminate any backslash-extended lines in the input file. The sed script eliminates comment and empty lines.

Follow the same procedure for all other automounter maps, such as auto_home or any other non-default maps.

5. **Run the make command.**

```
# make mapname
```

where *mapname* is the name of the map you want to make.

▼ How to Modify the NIS Makefile to Delete Databases

If you do not want the Makefile to produce maps for a specific database, edit the Makefile as follows.

1. **Delete the name of the database from the `all` rule.**

2. **Delete or comment out the database rule for the database you want to delete.**

   For example, to delete the `hosts` database, the `hosts.time` entry should be removed.

3. **Remove the time rule.**

   For example, to delete the `hosts` database, the `hosts: hosts.time` entry should be removed.

4. **Remove the map from the master and slave servers.**

Updating and Modifying Existing NIS Maps

After you have installed NIS, you might discover that some maps require frequent updating while others never need to change. For example, the `passwd.byname` map can change frequently on a large company's network, while the `auto_master` map changes little, if at all.

As mentioned in "Default NIS Maps" on page 61, the default location of the default NIS maps is on the master server in `/var/yp/domainname`, where `domainname` is the name of the NIS domain. When you need to update a map, you can use one of two updating procedures, depending on whether or not it is a default map.

- A default map is a map in the default set that is created by the `ypinit` command from the network databases.
- Non-default maps can be any of the following:
  - Maps that are included with an application purchased from a vendor
  - Maps that are created specifically for your site
  - Maps that are created from a non-text file

The following sections explain how to use various updating tools. In practice, you might decide to only use them if you add non-default maps or change the set of NIS servers after the system is already up and running.

▼ How to Update NIS Maps Supplied With the Default Set

Use the following procedure to update maps that are supplied with the default set.

1. **Become an administrator on the NIS master server.**

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Edit the source file for the map that you want to change.**

   The file could reside in /etc or in some other directory of your choice.

3. **Run the make command.**

   ```
   # cd /var/yp
   # make mapname
   ```

   The make command then updates your map according to the changes you made in its corresponding file. It also propagates the changes among the other servers.

Maintaining Updated NIS Maps

The following sections describe additional procedures after you have completed updating maps that are supplied with the default set.

Propagating an NIS Map

After a map is changed, the Makefile uses yppush to propagate a new map to the slave servers (unless NOPUSH is set in the Makefile). It does this by informing the ypserv daemon and sending a map transfer request. The ypserv daemon on the slave then starts a ypxfr process, which in turn contacts the ypxfrd daemon on the master server. Some basic checks are made (for example did the map really change?) and then the map is transferred. ypxfr on the slave then sends a response to the yppush process indicating whether the transfer succeeded.

The config/local_only property of the svc:/network/rpc/bind service must be set to false. Otherwise, the NIS master cannot transfer the updated version of NIS master map to the NIS slave server by using the yppush command.

Note - The preceding procedure will *not* work for newly created maps that do not yet exist on the slave servers. New maps must be sent to the slave servers by running ypxfr on the slaves.

Occasionally, maps fail to propagate and you must to use ypxfr manually to send new map information. You can choose to use ypxfr in two different ways: periodically through the root crontab file, or interactively on the command line. These approaches are discussed in the following sections.

Using the `cron` Command for NIS Map Transfers

Maps have different rates of change. For instance, some maps might not change for months at a time, such as `protocols.byname` among the default maps and `auto_master` among the non-default maps. However `passwd.byname` can change several times a day. Scheduling map transfer by using the `crontab` command enables you to set specific propagation times for individual maps.

To periodically run `ypxfr` at a rate appropriate for the map, the root `crontab` file on each slave server should contain the appropriate `ypxfr` entries. `ypxfr` contacts the master server and transfers the map only if the copy on the master server is more recent than the local copy.

Note - If your master server runs `rpc.yppasswdd` with the default `-m` option, then each time someone changes their `yp` password, the `passwd` daemon runs `make`, which rebuilds the `passwd` maps.

Using Shell Scripts With `cron` and `ypxfr` for NIS

As an alternative to creating separate `crontab` entries for each map, you might prefer to have the root `crontab` command run a shell script that periodically updates all maps. Sample map-updating shell scripts are n the `/usr/lib/netsvc/yp` directory. The script names are `ypxfr_1perday`, `ypxfr_1perhour`, and `ypxfr_2perday`. You can modify or replace these shell scripts to accommodate your site requirements. The following example shows the default `ypxfr_1perday` shell script.

EXAMPLE 3 `ypxfr_1perday` Shell Script

```
#! /bin/sh
#
# ypxfr_1perday.sh - Do daily yp map check/updates
PATH=/bin:/usr/bin:/usr/lib/netsvc/yp:$PATH
export PATH
# set -xv
ypxfr group.byname
ypxfr group.bygid
ypxfr protocols.byname
ypxfr protocols.bynumber
ypxfr networks.byname
ypxfr networks.byaddr
ypxfr services.byname
ypxfr ypservers
```

This shell script updates the maps once per day, if the root `crontab` is executed daily. You can also have scripts that update maps once a week, once a month, once every hour, and so forth. However, be aware of the performance degradation that is implied in frequently propagating the maps. For more information, see the `crontab(1)` man page.

Run the same shell scripts as `root` on each slave server configured for the NIS domain. Alter the exact time of execution from one server to another to avoid bogging down the master.

If you want to transfer the map from a particular slave server, use the `-h` *system* option of `ypxfr` within the shell script. The following is the syntax of the commands you put in the script.

```
# /usr/lib/netsvc/yp/ypxfr -h system [ -c ] mapname
```

Where *system* is the name of the server with the maps you want to transfer, and *mapname* is the name of the requested map. If you use the `-h` option without specifying a system, `ypxfr` tries to get the map from the master server. If `ypserv` is not running locally at the time `ypxfr` is executed, you must use the `-c` flag so that `ypxfr` does not send a clear current map request to the local `ypserver`.

You can use the `-s` *domain* option to transfer maps from another domain to your local domain. These maps must be the same across domains. For example, two domains might share the same `services.byname` and `services.byaddr` maps. Alternatively, for more control you can use `rcp` or `rsync` to transfer files across domains.

Directly Invoking the `ypxfr` Command

The second method of invoking the `ypxfr` command is to run it as a command. Typically, you do this only in exceptional situations – for example, when setting up a temporary NIS server to create a test environment or when trying to quickly get an NIS server that has been out of service consistent with the other servers.

Logging `ypxfr` Activity

The transfer attempts and results of `ypxfr` can be captured in a log file. If a file called `/var/yp/ypxfr.log` exists, results are appended to it. No attempt to limit the size of the log file is made. To prevent it from growing indefinitely, empty it from time to time by typing the following.

```
# cd /var/yp
# cp ypxfr.log ypxfr.log.old
# cat /dev/null > /var/yp/ypxfr.log
```

You can have `crontab` execute these commands once a week. To turn off logging, remove the log file.

Modifying Non-Default NIS Maps

To update a non-default map, complete the following steps:

1. Create or edit its corresponding text file.
2. Build (or rebuild) the new or updated map. There are two ways to build a map.
   - **Use the Makefile.** Using the Makefile is the preferred method of building a non-default map. If the map has an entry in the `Makefile`, run `make` *name* where *name* is the name of map you want to build. If the map does not have a `Makefile` entry, try to create one following the instructions in "Modifying and Using `/var/yp/Makefile`" on page 97.
   - **Use the /usr/sbin/makedbm program.** The `makedbm(8)` man page fully describes this command.

Using the `makedbm` Command to Modify a Non-Default NIS Map

There are two different methods for using `makedbm` to modify maps if you do not have an input file:

- Redirect the `makedbm -u` output to a temporary file, modify the file, then use the modified file as input to `makedbm`.
- Have the output of `makedbm -u` operated on within a pipeline that feeds into `makedbm`. This is appropriate if you can update the disassembled map with either `awk`, `sed`, or a `cat` append.

Creating New NIS Maps From Text Files

Assume that a text file `/var/yp/mymap.asc` was created with an editor or a shell script on the master. You want to create an NIS map from this file and locate it in the *home-domain* subdirectory. To do this, type the following on the master server.

```
# cd /var/yp
# makedbm mymap.asc home-domain/mymap
```

The *mymap* map now exists on the master server in the directory *home-domain*. To distribute the new map to slave servers run `ypxfr`.

Adding Entries to a File-Based NIS Map

Adding entries to mymap is simple. First, you must modify the text file /var/yp/*mymap*.asc. If you modify the actual dbm files without modifying the corresponding text file, the modifications are lost. Then run makedbm as shown above.

Creating NIS Maps From Standard Input

When no original text file exists, create the NIS map from the keyboard by typing input to makedbm, as shown in the following command (end with Control-D).

```
ypmaster# cd /var/yp
ypmaster# makedbm home-domain/mymap  key1 value1  key2 value2  key3 value3
```

Modifying NIS Maps Made From Standard Input

If you later need to modify the map, you can use makedbm to disassemble the map and create a temporary text intermediate file. To disassemble the map and create a temporary file, type the following:

```
# cd /var/yp
# makedbm -u homedomain/mymap > mymap.temp
```

The resulting temporary file *mymap*.temp has one entry per line. You can edit this file as needed, using any text editor.

To update the map, give the name of the modified temporary file to makedbm by typing the following:

```
$ makedbm mymap.temp homedomain/mymap
# rm mymap.temp
```

Then propagate the map to the slave servers, by becoming root and typing the following.

```
# yppush mymap
```

The preceding paragraphs explained how to use makedbm to create maps. However, almost everything you actually have to do can be done by theypinit command and by using /var/yp/Makefile unless you add non-default maps to the database or change the set of NIS servers after the system is already up and running.

Whether you use the `Makefile` in `/var/yp` or some other procedure the goal is the same. A new pair of well-formed dbm files must end up in the maps directory on the master server.

Working With NIS Servers

The following procedures show ways to modify the NIS configuration by binding to a specific NIS server, setting the NIS domain name, forwarding host lookups to DNS, and by turning off the NIS services.

Binding to a Specific NIS Server

Use the following steps to bind to an NIS server that you specify. For more information, see the `ypinit`(8), `ypstart`(8), and `svcadm`(8) man pages.

1. Add the host name of the NIS server and its IP address to the `/etc/hosts` file.
2. Verify that the NIS domain name is set.

   ```
   # domainname
   example.com
   ```

3. Prompt for the NIS server host name.

   ```
   # /usr/sbin/ypinit -c
   Server name:        Type the NIS server host name
   ```

4. Restart the NIS services by performing one of the following steps:
   - For the services to persist across reboots, run the `svcadm` command.

     ```
     # svcadm enable svc:/network/nis/client
     ```

   - For the services to persist until reboot only, run the `ypstop` and `ypstart` commands.

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

▼ How to Set a System's NIS Domain Name

To change the NIS domain name of a system, use the following procedure.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Define the NIS domain name.**

   ```
   # domainname research.example.com
   ```

3. **Update and run the domain name services.**

   ```
   # svccfg -s nis/domain:default refresh
   # svcadm enable nis/domain
   ```

4. **Set up the system as an NIS client, a slave server, or a master server.**

   See Chapter 6, "Setting Up and Configuring Network Information Service" for details.

How to Configure System Host Name and Address Lookup Through NIS and DNS

Typically, NIS clients are configured with the nsswitch.conf file to use only NIS for system name and address lookups. If this type of lookup fails, an NIS server can forward these lookups to DNS.

1. **Become an administrator.**

   For more information about obtaining the appropriate rights to perform specific tasks, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **Add the YP_INTERDOMAIN key.**

   The two map files, hosts.byname and hosts.byaddr must include the YP_INTERDOMAIN key. To test this key, edit /var/yp/Makefile and move the comment (#) from the first to the second line:

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

   to:

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

makedbm will now start with the -b flag when it makes the maps, and the YP_INTERDOMAIN key will be inserted into the ndbm files.

3. **Run the make command to rebuild maps.**

   ```
   # make hosts
   ```

4. **Check that DNS name servers are set properly.**

   The following command lists all of the IP addresses for the DNS name servers:

   ```
   # svcprop -p config/nameserver network/dns/client
   ```

5. **To enable DNS forwarding, restart each server.**

   ```
   # svcadm restart network/nis/server:instance
   ```

   In this implementation of NIS, the ypserv daemon automatically starts with the -d option to forward requests to DNS.

Turning Off NIS Services

If the ypserv daemon on the NIS master is disabled, you can no longer update any of the NIS maps.

- To disable NIS on a client, type the following:

  ```
  # svcadm disable network/nis/domain
  # svcadm disable network/nis/client
  ```

- To disable NIS on a specific slave or master server, type the following on the server:

  ```
  # svcadm disable network/nis/domain
  # svcadm disable network/nis/server
  ```

♦ ♦ ♦ CHAPTER 8

Troubleshooting Network Information System

This chapter explains how to resolve problems encountered on networks running NIS. It covers problems that are encountered on both NIS clients and NIS servers.

Before trying to debug an NIS server or client, review Chapter 5, "About the Network Information Service" which explains the NIS environment. Then, look for the subheading in this section that best describes your problem.

Note - The NIS service is managed by the Service Management Facility (SMF). Administrative actions on this service, such as enabling, disabling, or restarting, are performed by using the `svcadm` command. See "NIS and the Service Management Facility" on page 68 for more information about using SMF with NIS. For an overview of SMF, refer to Chapter 1, "Introduction to the Service Management Facility" in *Managing System Services in Oracle Solaris 11.4*. Also refer to the `svcadm`(8) and `svcs`(1) man pages for more details.

NIS services can also be started and stopped by using the `ypstart` and `ypstop` commands. See the `ypstart`(8) and `ypstop`(8) man pages for more information.

NIS Binding Problems

Common symptoms of NIS binding problems include the following.

- Messages saying that `ypbind` can't find or communicate with a server
- Messages saying that server not responding
- Messages saying that NIS is unavailable
- Commands on a client limp along in background mode or function much slower than normal
- Commands on a client hang. Sometimes commands hang even though the system as a whole seems fine and you can run new commands

- Commands on a client crash with obscure messages, or no message at all

NIS Problems Affecting One Client

If only one or two clients are experiencing symptoms that indicate NIS binding difficulty, the problems probably are on those clients. If many NIS clients are failing to bind properly, the problem probably exists on one or more of the NIS servers. See "NIS Problems Affecting Many Clients" on page 113.

`ypbind` Not Running on Client

One client has problems, but other clients on the same subnet are operating normally. On the problem client, run `ls -l` on a directory, such as `/usr`, that contains files owned by many users, including some not in the client `/etc/passwd` file. If the resulting display lists file owners who are not in the local `/etc/passwd` as numbers, rather than names, this indicates that NIS service is not working on the client.

These symptoms usually mean that the client `ypbind` process is not running. Verify whether the NIS client services are running.

```
client# svcs \*nis\*
STATE          STIME    FMRI
disabled       Sep_01   svc:/network/nis/domain:default
disabled       Sep_01   svc:/network/nis/client:default
```

If the services are in a `disabled` state, log in as `root` or assume an equivalent role, and start the NIS client service.

```
client# svcadm enable network/nis/domain
client# svcadm enable network/nis/client
```

Missing or Incorrect NIS Domain Name

One client has problems, the other clients are operating normally, but `ypbind` is running on the problem client. The client might have an incorrectly set domain.

On the client, run the `domainname` command to see which domain name is set.

```
client7# domainname
example.com
```

Compare the output with the actual domain name in /var/yp on the NIS master server. The actual NIS domain is shown as a subdirectory in the /var/yp directory.

```
client7# ls

-l /var/yp
-rwxr-xr-x 1 root Makefile
drwxr-xr-x 2 root binding
drwx------ 2 root example.com
```

If the domain name returned by running domainname on a system is not the same as the server domain name listed as a directory in /var/yp, the domain name specified in the system's /etc/defaultdomain file is incorrect. Reset the NIS domain name as shown in "How to Set a System's NIS Domain Name" on page 106.

Note - The NIS domain name is case-sensitive.

NIS Client Not Bound to Server

If your domain name is set correctly, ypbind is running, and commands still hang, then make sure that the client is bound to a server by running the ypwhich command. If you have just started ypbind, then run ypwhich several times (typically, the first one reports that the domain is not bound and the second succeeds normally).

No NIS Server Available

If your domain name is set correctly, ypbind is running, and you get messages indicating that the client cannot communicate with a server, this might indicate a number of different problems:

- Does the client have a /var/yp/binding/*domainname*/ypservers file containing a list of servers to bind to? If not, run ypinit -c and specify in order of preference the servers that this client should bind to.
- If the client does have a /var/yp/binding/*domainname*/ypservers file, are there enough servers listed in it if one or two become unavailable? If not, add additional servers to the list by running ypinit -c.
- Do the selected NIS servers have entries in the /etc/inet/hosts file? To view the selected NIS servers, use the svcprop -p config/ypservers nis/domain command. If these hosts are not in the local /etc/inet/hosts file, add the servers to the hosts NIS maps

and rebuild your maps by running the ypinit -c or ypinit -s command as described in“Working With NIS Maps” on page 93.

- Is the name service switch set up to check the system's local hosts file in addition to NIS? See Chapter 2, “About the Name Service Switch” for more information about the switch.
- Is the name service switch set up to check files first for services and rpc? See Chapter 2, “About the Name Service Switch” for more information about the switch.

ypwhich Displays Are Inconsistent

When you use ypwhich several times on the same client, the resulting display varies because the NIS server changes. This is normal. The binding of the NIS client to the NIS server changes over time when the network or the NIS servers are busy. Whenever possible, the network becomes stable at a point where all clients get acceptable response time from the NIS servers. As long as your client system gets NIS service, it does not matter where the service comes from. For example, an NIS server system can get its own NIS services from another NIS server on the network.

When NIS Server Binding Is Not Possible

In extreme cases where local server binding is not possible, use of the ypset command can temporarily allow binding to another server, if available, on another network or subnet. However, in order to use the -ypset option, ypbind must be started with either the -ypset or -ypsetme options. For more information, see the ypbind(8) man page.

```
# /usr/lib/netsvc/yp/ypbind -ypset
```

For another method, see “Binding to a Specific NIS Server” on page 106.

Caution - For security reasons, the use of the -ypset and -ypsetme options is not recommended. Only use these options for debugging purposes under controlled circumstances. Use of the -ypset and -ypsetme options can result in serious security breaches because while the daemons are running, anyone can alter server bindings, causing trouble for others and permitting unauthorized access to sensitive data. If you must start the ypbind daemon with these options, after you have fixed the problem you must kill the ypbind process and restart it again without those options.

To restart the ypbind daemon, use SMF as follows:

```
# svcadm enable -r svc:/network/nis/client:default
```

ypbind Crashes

If the ypbind daemon crashes almost immediately each time it is started, look for a problem in the svc:/network/nis/client:default service log. Check for the presence of the rpcbind daemon by typing the following:

```
$ ps -e |grep rpcbind
```

If rpcbind is not present or does not stay up or behaves strangely, check the svc:/network/rpc/bind:default log file. For more information, see the rpcbind(8) and rpcinfo(8) man pages.

You might be able to communicate with rpcbind on the problematic client from a system operating normally. From the functioning system, type the following:

```
$ rpcinfo client
```

If rpcbind on the problematic system is fine, rpcinfo produces the following output:

```
    program version netid address service owner
...
100007    3    udp6      ::.191.161          ypbind     1
100007    3    tcp6      ::.135.200          ypbind     1
100007    3    udp       0.0.0.0.240.221     ypbind     1
100007    2    udp       0.0.0.0.240.221     ypbind     1
100007    1    udp       0.0.0.0.240.221     ypbind     1
100007    3    tcp       0.0.0.0.250.107     ypbind     1
100007    2    tcp       0.0.0.0.250.107     ypbind     1
100007    1    tcp       0.0.0.0.250.107     ypbind     1
100007    3    ticlts    2\000\000\000       ypbind     1
100007    2    ticlts    2\000\000\000       ypbind     1
100007    3    ticotsord 9\000\000\000       ypbind     1
100007    2    ticotsord 9\000\000\000       ypbind     1
100007    3    ticots    @\000\000\000       ypbind     1
...
```

Your system will have different addresses. If the addresses are not displayed, ypbind has been unable to register its services. Reboot the system and run rpcinfo again. If the ypbind processes are there and they change each time you try to restart the NIS service, reboot the system, even if the rpcbind daemon is running.

NIS Problems Affecting Many Clients

If only one or two clients are experiencing symptoms that indicate NIS binding difficulty, the problems probably are on those clients. See “NIS Problems Affecting One

Client" on page 110. If many NIS clients are failing to bind properly, the problem probably exists on one or more of the NIS servers.

rpc.yppasswdd Considers a Non-Restricted Shell That Begins With r to Be Restricted

1. Create /etc/default/yppasswdd that contains a special string: "check_restricted_shell_name=1".
2. If the "check_restricted_shell_name=1" string is commented out, the 'r' check will not occur.

NIS Network or Servers Are Unreachable

NIS can hang if the network or NIS servers are so overloaded that the ypserv daemon cannot receive a response back to the client ypbind process within the timeout period. NIS can also hang if the network is down.

Under these circumstances, every client on the network experiences the same or similar problems. In most cases, the condition is temporary. The messages usually go away when the NIS server reboots and restarts ypserv, when the load on the NIS servers or the network itself decreases, or when the network resumes normal operations.

NIS Server Malfunction

Make sure the servers are up and running. If you are not physically near the servers, use the ping command.

NIS Daemons Not Running

If the servers are up and running, try to find a client system behaving normally, and run the ypwhich command. If ypwhich does not respond, kill it. Then log in as root on the NIS server and check if the NIS process is running by typing the following:

```
# ptree |grep ypbind
100759 /usr/lib/netsvc/yp/ypbind -broadcast
```

```
527360 grep yp
```

If neither the ypserv (NIS server) nor the ypbind (NIS client) daemons are running, restart them by typing the following:

```
# svcadm restart network/nis/client
```

If both the ypserv and ypbind processes are running on the NIS server, then run the ypwhich command. If the command does not respond, the ypserv daemon has probably hung and should be restarted. While logged in as root on the server, restart the NIS service by typing the following:

```
# svcadm restart network/nis/server
```

Servers Have Different Versions of an NIS Map

Because NIS propagates maps among servers, occasionally you might find different versions of the same map on various NIS servers on the network. This version discrepancy is normal and acceptable if the differences do not last for more than a short time.

The most common cause of map discrepancy is that something is preventing normal map propagation. For example, an NIS server or router between NIS servers is down. When all NIS servers and the routers between them are running, ypxfr should succeed.

If the servers and routers are functioning properly, check the following:

- Check the ypxfr log output. See "Logging ypxfr Output" on page 115.
- Check the svc:/network/nis/xfr:default log file for errors.
- Check the control files. See "Check the crontab File and ypxfr Shell Script" on page 116.
- Check the ypservers map on the master server. See "Check the ypservers Map" on page 116.

Logging ypxfr Output

If a particular slave server has problems updating maps, log in to that server and run the ypxfr command interactively. If the command fails, it indicates why it failed, and you can fix the problem. If the command succeeds, but you suspect it has occasionally failed, create a log file to enable the logging of messages. To create a log file, type the following on the slave.

```
ypslave# cd /var/yp
```

```
ypslave# touch ypxfr.log
```

This creates a `ypxfr.log` file that saves all output from `ypxfr`.

The output resembles the output `ypxfr` displays when run interactively, but each line in the log file is time stamped. (You might see unusual ordering in the timestamps. That is okay – the timestamp tells you when `ypxfr` started to run. If copies of `ypxfr` ran simultaneously but their work took differing amounts of time, they might actually write their summary status line to the log files in an order different from that which they were invoked.) Any pattern of intermittent failure shows up in the log.

Note - When you have fixed the problem, turn off logging by removing the log file. If you forget to remove it, the file continues to grow without limit.

Check the `crontab` File and `ypxfr` Shell Script

Inspect the `root crontab` file, and check the `ypxfr` shell script it invokes. Typographical errors in these files can cause propagation problems. Failures to refer to a shell script within the `/var/spool/cron/crontabs/root` file, or failures to refer to a map within any shell script can also cause errors.

Check the `ypservers` Map

Also, make sure that the NIS slave server is listed in the `ypservers` map on the master server for the domain. If it is not, the slave server still operates perfectly as a server, but `yppush` does not propagate map changes to the slave server.

Workaround to Update NIS Maps on a Broken Slave Server

If the NIS slave server problem is not obvious, you can perform a workaround while you debug the problem, by using the `scp` or `ssh` command to copy a recent version of the inconsistent map from any healthy NIS server. The following shows how to transfer the problem map:

```
ypslave# scp ypmaster:/var/yp/mydomain/map.\* /var/yp/mydomain
```

The * character has been escaped in the command line, so that it will be expanded on `ypmaster`, instead of locally on `ypslave`.

ypserv Crashes

When the ypserv process crashes almost immediately and does not stay up even with repeated activations, the debugging process is virtually identical to that described in "ypbind Crashes" on page 113. First, run the following command to see if any errors are being reported:

```
# svcs -vx nis/server
```

Check for the existence of the rpcbind daemon as follows:

```
# ptree |grep rpcbind
```

Reboot the server if you do not find the daemon. Otherwise, if the daemon is running, type the following and look for similar output:

```
$ rpcinfo -p ypserver
```

```
$ program  vers  proto  port  service
100000 4 tcp 111 portmapper
100000 3 tcp 111 portmapper
100068 2 udp 32813 cmsd
...
100007 1 tcp 34900 ypbind
100004 2 udp 731 ypserv
100004 1 udp 731 ypserv
100004 1 tcp 732 ypserv
100004 2 tcp 32772 ypserv
```

Your system might have different port numbers. The four entries representing the ypserv process are the following:

```
100004  2  udp  731  ypserv
100004  1  udp  731  ypserv
100004  1  tcp  732  ypserv
100004  2  tcp  32772  ypserv
```

If there are no entries, and ypserv is unable to register its services with rpcbind, reboot the system. If there are entries, de-register the service from rpcbind before restarting ypserv. To de-register the service from rpcbind, on the server type the following.

```
# rpcinfo -d number 1
# rpcinfo -d number 2
```

where *number* is the ID number reported by rpcinfo (100004, in the preceding example).

DNS and NIS Glossary

baseDN The DN where part of the DIT is rooted. When this is the baseDN for an NIS domains entries it is also referred to as a *context*.

context For the N2L service, a context is something under which an NIS domain is generally mapped. See also baseDN.

directory cache A local file used to store data associated with directory objects.

DNS zone files A set of files wherein the DNS software stores the names and IP addresses of all the workstations in a domain.

DNS zones Administrative boundaries within a network domain, often made up of one or more subdomains.

field A NIS map entry might consist of a number of components and separator characters. As part of the N2L service mapping process the entry is first broken down into a number of named *fields*.

mail exchange records Files that contain a list of DNS domain names and their corresponding mail hosts.

N2L server NIS-to-LDAP server. An NIS master server that has been reconfigured as an N2L server by using the N2L service. Reconfiguration includes replacing NIS daemons and adding new configuration files.

NIS A distributed network information service containing key information about the systems and the users on the network. The NIS database is stored on the *master server* and all the *replica* or *slave servers*.

NIS maps A file used by NIS that holds information of a particular type, for example, the password entries of all users on a network or the names of all host systems on a network. Programs that are part of the NIS service query these maps. See also *NIS*.

preferred server list A `client_info` table or a `client_info` file. Preferred server lists specify the preferred servers for a client or domain.

Index