Oracle® Solaris Administration: Naming and Directory Services

Part No: 821–1455–11
December 2011

ORACLE

111206@25097

Contents

Tables

Examples

Preface

Oracle Solaris Administration Guide: Naming and Directory Services (DNS, NIS and LDAP) describes the setup and administration of the Oracle Solaris operating system (OS) naming and directory services: DNS, NIS, and LDAP. This guide is part of a multivolume set that covers a significant part of the Oracle Solaris administration information.

Note – This Oracle Solaris release supports systems that use the SPARC and x86 families of processor architectures. The supported systems appear in the *Oracle Solaris OS: Hardware Compatibility Lists*. This document cites any implementation differences between the platform types.

Who Should Use This Book

This guide is written for experienced system and network administrators.

Although this book introduces networking concepts relevant to Oracle Solaris naming and directory services, it explains neither the networking fundamentals nor the administration tools in the Oracle Solaris release.

How This Book Is Organized

This guide is divided into parts according to the respective naming services.

Part I, "About Naming and Directory Services"

Part II, "NIS Setup and Administration"

Part III, "LDAP Naming Services"

How the System Administration Guides Are Organized

Here is a list of the topics that are covered by the System Administration Guides.

Book Title	Topics
Booting and Shutting Down Oracle Solaris on SPARC Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on SPARC platforms
Booting and Shutting Down Oracle Solaris on x86 Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on x86 platforms
Oracle Solaris Administration: Common Tasks	Using Oracle Solaris commands, booting and shutting down a system, managing user accounts and groups, managing services, hardware faults, system information, system resources, and system performance, managing software, printing, the console and terminals, and troubleshooting system and software problems
Oracle Solaris Administration: Devices and File Systems	Removable media, disks and devices, file systems, and backing up and restoring data
Oracle Solaris Administration: IP Services	TCP/IP network administration, IPv4 and IPv6 address administration, DHCP, IPsec, IKE, IP Filter, Mobile IP, and IPQoS
Oracle Solaris Administration: Naming and Directory Services	DNS, NIS, and LDAP naming and directory services, including transitioning from NIS to LDAP
Oracle Solaris Administration: Network Interfaces and Network Virtualization	Networking stack, NIC driver property configuration, NWAM configuration, manual network interface configuration, administration of VLANs and link aggregations, IP network multipathing (IPMP), WiFi wireless networking configuration, virtual NICs (VNICs), and network resource management
Oracle Solaris Administration: Network Services	Web cache servers, time-related services, network file systems (NFS and autofs), mail, SLP, and PPP
Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management	Resource management features, which enable you to control how applications use available system resources; Oracle Solaris Zones software partitioning technology, which virtualizes operating system services to create an isolated environment for running applications; and Oracle Solaris 10 Zones, which host Oracle Solaris 10 environments running on the Oracle Solaris 11 Express kernel
Oracle Solaris Administration: Security Services	Auditing, device management, file security, BART, Kerberos services, PAM, Cryptographic Framework, Key Management Framework, privileges, RBAC, SASL, Secure Shell and virus scanning.

Book Title	Topics
Oracle Solaris Administration: SMB and Windows Interoperability	SMB service, which enables you to configure an Oracle Solaris system to make SMB shares available to SMB clients; SMB client, which enables you to access SMB shares; and native identity mapping service, which enables you to map user and group identities between Oracle Solaris systems and Windows systems
Oracle Solaris Administration: ZFS File Systems	ZFS storage pool and file system creation and management, snapshots, clones, backups, using access control lists (ACLs) to protect ZFS files, using ZFS on an Oracle Solaris system with zones installed, emulated volumes, and troubleshooting and data recovery
Trusted Extensions Configuration and Administration	System installation, configuration, and administration that is specific to Trusted Extensions
Oracle Solaris 11 Security Guidelines	Securing an Oracle Solaris system, as well as usage scenarios for its security features, such as zones, ZFS, and Trusted Extensions
Transitioning From Oracle Solaris 10 to Oracle Solaris 11	Provides system administration information and examples for transitioning from Oracle Solaris 10 to Oracle Solaris 11 in the areas of installation, device, disk, and file system management, software management, networking, system management, security, virtualization, desktop features, user account management, and user environments emulated volumes, and troubleshooting and data recovery

Related Books

- *Oracle Directory Server Enterprise Edition Deployment Guide*
- *Oracle Directory Server Enterprise Edition Administration Guide*
- *DNS and Bind*, by Cricket Liu and Paul Albitz, (5th Edition, O'Reilly, 2006)
- *Understanding and Deploying LDAP Directory Services*, by Timothy A. Howes, Ph.D. and Mark C. Smith

Access to Oracle Support

Oracle customers 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.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

PART I

About Naming and Directory Services

This part introduces the naming and directory services for the Oracle Solaris OS. It also describes how to configure naming services using the Service Management Facility (SMF) so that you can coordinate lookups by using the different local and remote directory services. It also describes how to configure the Domain Name Service (DNS), as well as Active Directory clients.

◆ ◆ ◆ CHAPTER 1

Naming and Directory Services (Overview)

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

The following topics are covered in this chapter:

What Is a Naming Service?

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

- Host names and addresses
- User names
- Passwords
- Access permissions
- Group membership, automount maps, and so on

This information is made available so that users can log in to their host, access resources, and be granted permissions. The name service information can be stored locally in various forms of database files, or in a central network-based repository or database.

Without a central naming service, each host would have to maintain its own copy of this information. Naming service information can be stored in files, maps, or database tables. If you centralize all data, administration becomes easier.

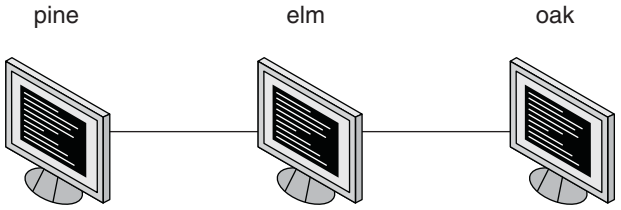
Naming services are fundamental to any computing network. Among other features, naming services provide functionality that does the following.

- Associates (*binds*) names with objects
- Resolves names to objects

- Removes bindings
- Lists names
- Renames information

A network information service enables systems to be identified by common names instead of numerical addresses. This makes communication simpler because users do not have to remember and try to enter cumbersome numerical addresses like `192.168.0.0`.

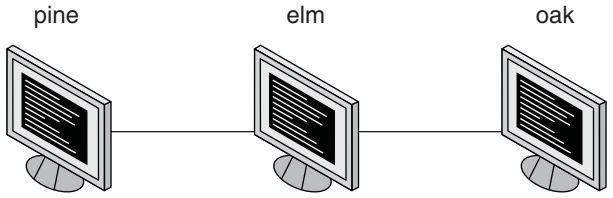
For example, take a network of three systems that are named, `pine`, `elm`, and `oak`. Before `pine` can send a message to either `elm` or `oak`, `pine` must know their numerical network addresses. For this reason, `pine` keeps a file, `/etc/inet/hosts`, that stores the network address of every system in the network, including itself.



`/etc/inet/hosts`

```
10.0.3.1 pine
10.0.3.2 elm
10.0.3.3 oak
```

Likewise, in order for `elm` and `oak` to communicate with `pine` or with each other, the systems must keep similar files.



`/etc/inet/hosts`

```
10.0.3.1 pine
10.0.3.2 elm
10.0.3.3 oak
```

`/etc/inet/hosts`

```
10.0.3.1 pine
10.0.3.2 elm
10.0.3.3 oak
```

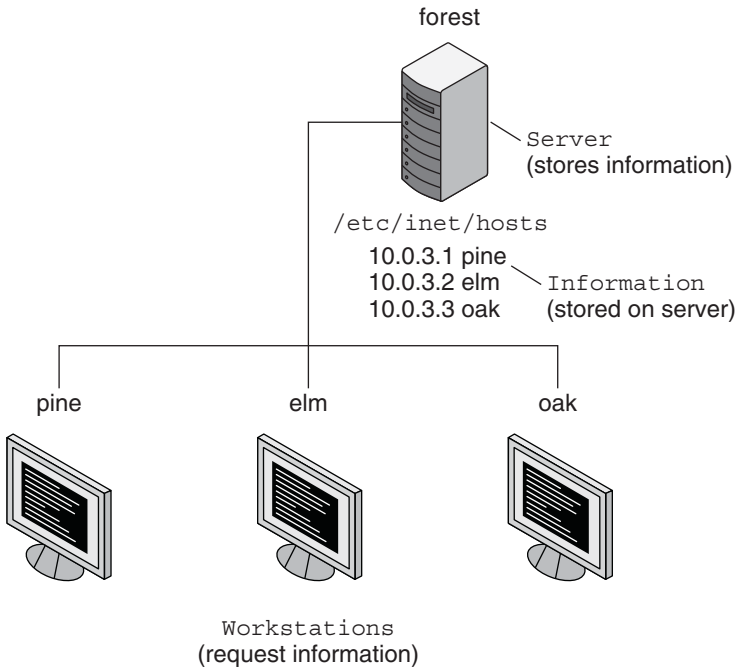
`/etc/inet/hosts`

```
10.0.3.1 pine
10.0.3.2 elm
10.0.3.3 oak
```

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

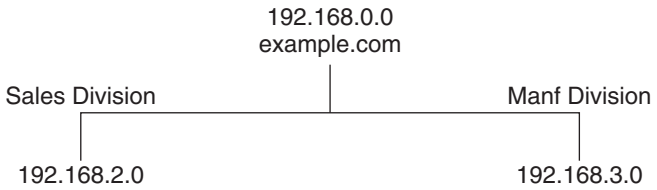
A network information service stores network information on a server, which can be queried by any system.

The systems are known as *clients* of the server. The following figure illustrates the client-server arrangement. 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 sheer size of the task.

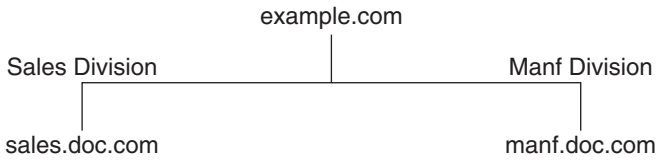


This arrangement, of a server providing centralized services to clients across a network, is known as *client-server computing*.

Although the main purpose of a network information service is to centralize information, the network information service can also simplify network names. For example, assume your company has set up a network which is connected to the Internet. The Internet has assigned your network the network address `192.168.0.0` 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. Each net has its own address.



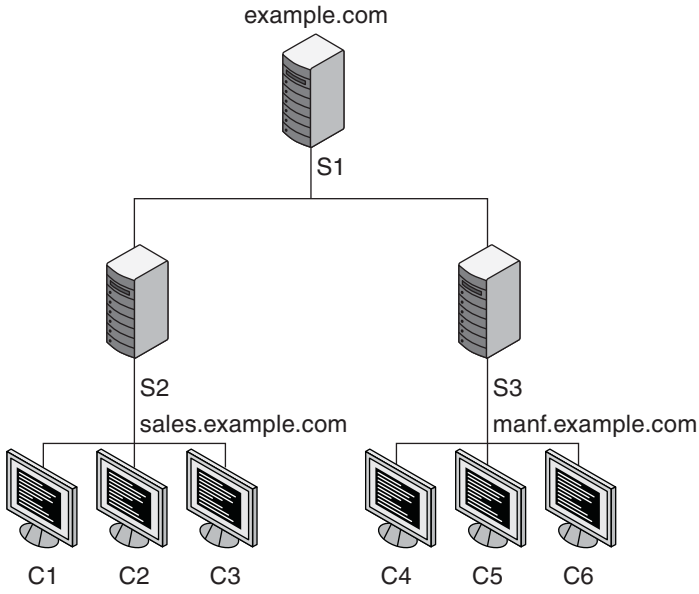
Each division could be identified by its network address, as shown above, but descriptive names made possible by naming services would be preferable.



Instead of addressing mail or other network communications to `198.168.0.0`, mail could be addressed to `example.com`. Instead of addressing mail to `192.168.2.0` or `192.168.3.0`, mail could be addressed 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`. Assume that two of those servers, `S2` and `S3`, support clients.

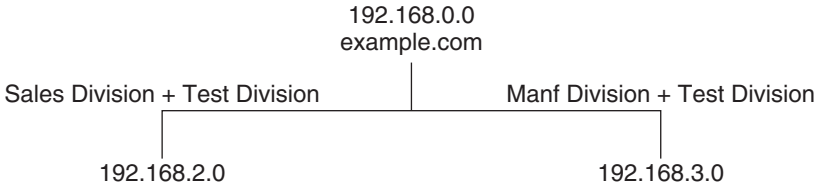


Clients `C1`, `C2`, and `C3` would obtain their network information from server `S2`. Clients `C4`, `C5`, and `C6` would obtain information from server `S3`. The resulting network is summarized in the following table. The table is a generalized representation of that network but does not resemble an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

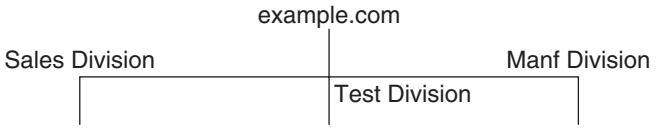
TABLE 1–1 Representation of `example.com` Network

Network Address	Network Name	Server	Clients
`192.168.1.0`	example.com	`S1`	
`192.168.2.0`	`sales.example.com`	`S2`	`C1`, `C2`, `C3`
`192.168.3.0`	manf.example.com	`S3`	`C4`, `C5`, `C6`

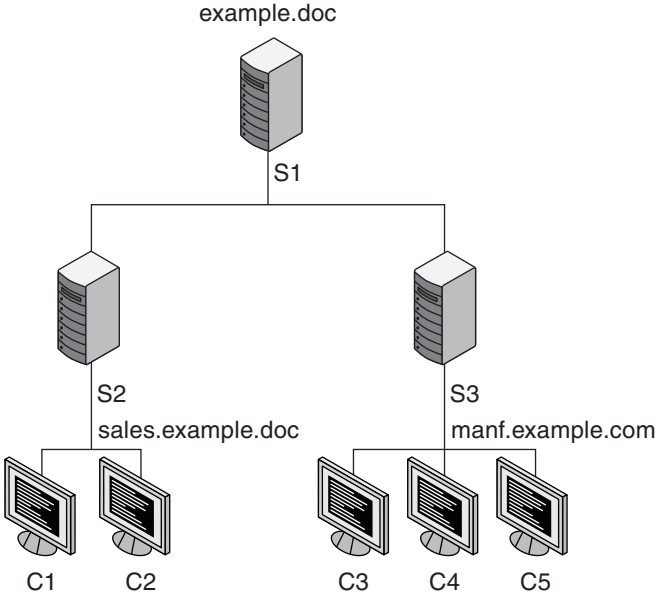
Now, assume that you create a third division, Testing, which borrowed some resources from the other two divisions, but did not create a third subnet. The physical network would then no longer parallel the corporate structure.



Traffic for the Test Division would not have its own subnet, but would instead be split between `192.168.2.0` and `192.168.3.0`. However, with a network information service, the Test Division traffic could have its own dedicated network.



Thus, when an organization changes, its network information service can change its mapping as shown here.



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

Subsequent changes in your organization would be accommodated by changes to the network information structure without reorganizing the network structure.

Oracle Solaris Naming Services

The Oracle Solaris platform provides the following naming services:

- Domain Name System (DNS) (see "Description of the DNS Naming Service" on page 29)
- `/etc` files, the original UNIX naming system (see "Description of the `/etc` Files Naming Service" on page 30)
- Network Information Service (NIS) (see "Description of the NIS Naming Service" on page 30)
- Lightweight Directory Access Protocol (LDAP) (see Part III, "LDAP Naming Services" *LDAP Naming Services Setup and Administration*)

Most modern networks use two or more of these services in combination. Which naming service is used for a particular lookup is coordinated by the name service switch, which is discussed in Chapter 2, "Name Service Switch (Overview)."

Description of the 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 connected that are to the Internet, an intranet, or both.

DNS clients request information about a host name from one or more name servers and wait for a response. DNS servers respond to requests from a information cache that was loaded from file or a third-party database on a DNS master, or over the network from a cooperating DNS slave server, or from information stored from previous queries. If no response is found and the server is not responsible for the domain in question, the service will, if so permitted, recursively request the host name from other servers and cache that response.

Description of Multicast DNS and Service Discovery

Support for two extensions to the DNS protocol is now available. Both extensions are managed by the `svc:network/dns/multicast` service. 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 to also provide simple service discovery (network browsing). For more information, see "Multicast DNS" on page 41 and "Multicast DNS Service Discovery" on page 42.

Description of the `/etc` Files Naming Service

The original host-based UNIX naming system was developed for stand-alone UNIX machines and then adapted for network use. Many old UNIX operating systems and machines still manage all naming data by using only local files. However, managing hosts, users, and other naming data by using local files is not well suited for large complex networks. Each `/etc` file is described in its associated man page. For example, the `/etc/inet/hosts` file is described in the `hosts`(4) man page.

Description of the NIS Naming Service

The *Network Information Service* (NIS) was developed independently of DNS. DNS makes communication simpler by using machine names instead of numerical IP addresses. NIS focuses on making network administration more manageable by providing centralized control over a variety of network information. NIS stores information about the network, machine 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 were designed to replace UNIX `/etc` files, as well as other configuration files. NIS maps store much more than names and addresses. As a result, the NIS namespace has a large set of maps. See "Working With NIS Maps" on page 93 for more information.

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 servers have backup, or *slave* servers. Both master and slave servers use the NIS retrieval software and both store NIS maps. For more information on NIS Architecture and NIS Administration, see Chapter 6, "Setting Up and Configuring NIS (Tasks)," and Chapter 7, "Administering NIS (Tasks)."

Description of the LDAP Naming Services

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

For information about LDAP naming services, see Chapter 9, "Introduction to LDAP Naming Services (Overview)."

For information about transitioning from NIS to LDAP, see Chapter 15, "Transitioning From NIS to LDAP (Tasks)."

For information about single sign-on, as well as the setup and maintenance of Kerberos authentication services, see Part VI, "Kerberos Service," in *Oracle Solaris Administration: Security Services*.

Description of the Name Service Switch

The name service switch is a mechanism to allow 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, "Name Service Switch (Overview)."

Naming Services: A Quick Comparison

	DNS	NIS	LDAP	Files
Namespace	Hierarchical	Flat	Hierarchical	Files
Data Storage	Files/resource records	Two column maps	Directories (varied)	Text-based files
			Indexed database	
Servers	Master/slave	Master/slave	Master/replica	None
			Multi master replica	
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	All	Host	All	All

Note – DNS is the recommended service for host or network address lookups for LDAP and files-based naming.

◆ ◆ ◆ CHAPTER 2

Name Service Switch (Overview)

This chapter describes the name service switch. You use the name service switch to coordinate usage of different naming services. The following topics are covered in this chapter:

About 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. The services are 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 should be attempted, what timeout to apply, or what to do if the service fails.

Databases and Sources for the Name Service Switch

The following databases are supported by the name service switch.

TABLE 2–1 Databases for the Name Service Switch

Information Database	Description
`alias`	Lists email addresses and aliases
`auth_attr`	Lists authorization names and descriptions
`automount`	Lists information about remote file systems that could be mounted locally
`bootparam`	Lists boot information for diskless clients
`ether`	Lists the Ethernet addresses and matching host names
`group`	Lists information about groups that can be used to share access to files
`host`	Lists the IP address and matching host names
`netgroup`	Lists information for shared NFS file systems
`netmask`	Lists network masks used to implement IP subnets
`network`	Lists the name and number for each network
`password`	Lists user account information
`prof_attr`	Lists execution profile names, descriptions, and other attributes
`project`	Lists project names, unique identifiers, and associated resource allocations
`protocol`	Lists Internet protocol names, numbers and any aliases
`publickey`	Lists public key information
`rpc`	Lists names and numbers of RPC programs
`service`	Lists the name, port, and protocol for Internet services
`tnrhdb`	Lists security attributes for a host using the Trusted Extensions feature of Oracle Solaris
`tnrhtp`	Lists templates used by Trusted Extensions

In addition, a `default` property in the name service switch defines the source string for any database that is not otherwise defined. If your network uses the same sources for most databases, then you can change the `default` property and not define a property for each database. See "How to Change the Source for All Naming Databases" on page 39 for the procedure.

To support prior releases, the `enable_passwd_compat` and `enable_group_compat` properties can be set to `true` to enable `compat` mode for password and group information. This mode provides support for old-style + or - syntax in the appropriate databases. In the current release, this functionality has been replaced by the `pam_list` module.

The following table describes the kind of sources that can be listed in the name service switch for the databases listed above.

TABLE 2–2 Information Sources for the Name Service Switch

Information Sources	Description
`ad`	Identifies databases stored on an Active Directory server.
`compat`	`compat` can be used for password and group information to support old-style + or - syntax in the `/etc/passwd`, `/etc/shadow`, and `/etc/group` files. This functionality has been replaced by the `pam_list` module.
`dns`	Specifies that host information be obtained from DNS.
`files`	Specifies a file stored in the client's `/etc` directory, for example, `/etc/passwd`.
`ldap`	Specifies that entries be obtained from the LDAP directory.
`mdns`	Specifies hosts information by using Multicast DNS (mDNS).
`nis`	Specifies an NIS map, for example, the `hosts` map.

Search Criteria for the Name Service Switch

The following search criteria formats can be used to select one or more information sources, and to specify the order that 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 hangs 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 listed in the following table.

TABLE 2–3 Status Messages for the Name Service Switch

Status Message	Explanation
SUCCESS	The requested entry was found in the specified source.
UNAVAIL	The source is either unresponsive or unavailable. In other words, none of the database sources could be found or accessed.
NOTFOUND	The source responded with "No such entry." In other words, the database was accessed, but the needed information was not found.
TRYAGAIN	The 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 two *actions* shown in the following table.

TABLE 2–4 Responses to Status Messages from the Name Service Switch

Action	Explanation
return	Stop looking for the information.
continue	Try the next source.

In addition, for the TRYAGAIN status message, the following actions can be defined

- forever – Retries 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. This list includes a description of several of the search criteria.

- 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=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.

You can change the default search criteria by explicitly specifying some other criteria by using the *STATUS=action* syntax shown in the preceding list. For example, the default action for a `NOTFOUND` condition is to continue the search to the next source. The search criteria for the network database could be reported as:

```
svc:/system/name-service/switch> listprop config/network
config/network  astring             "nis [NOTFOUND=return] files"
```

The `networks: nis [NOTFOUND=return] files` entry specifies a non-default criterion for the `NOTFOUND` status. Non-default criteria are delimited by square brackets.

In this example, the search routine behaves as follows:

- If the `network` database is available and contains the needed information, the routine returns with a `SUCCESS` status message.
- If the `network` database is not available, the routine returns with an `UNAVAIL` status message. By default, the routine continues to search by using the next criteria listed.
- If the `network` database is available and found, but the database does not contain the needed information, the routine returns with a `NOTFOUND` message. However, instead of continuing to search the next source, which would be the default behavior, the routine stops searching.
- If the `network` database is busy, the routine returns with an `TRYAGAIN` status message and by default continues to search the `network` database.

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. See "Name Service Switch and Password Information" on page 40 for more information.

What If the Syntax Is Wrong?

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.

`auto_home` and `auto_master`

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

`timezone` and the Name Service Switch

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

keyserv and publickey Entries in the Name Service Switch

Caution – You must restart the keyserv daemon after you make a change to the name service switch in order for the changes to take effect.

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 the keyserv daemon is restarted by using svcadm refresh svc:/network/rpc/keyserv:default. This command must be run 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.

Managing the Name Service Switch

When you change a machine's naming service, you need to modify that machine's name service switch information accordingly. For example, if you change a machine's naming service from files to NIS, you need to configure the name service switch to use NIS.

▼ How to Use a Legacy nsswitch.conf File

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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
   ```

How to Switch the Source for a Database

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Change the source definition for the selected database.**

   In this example, the database search order is first `files`, then `nis`.

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

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

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

How to Change the Source for All Naming Databases

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Change the `config/default` property.**

   This property should use the source definition that is most common. In this example, the database search order is first `files`, then `nis`.

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

3. **(Optional) Change the properties for individual databases.**

   Use this command to change the source definition for any database that does not use the order that is selected in the `config/default` property.

   ```
   # 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
   ```

DNS and Internet Access

The name service switch also controls DNS forwarding for clients as described in the following chapter. DNS forwarding grants Internet access to clients.

Name Service Switch and Password Information

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

Caution – `files` should be the first source in the name services switch for `passwd` information to prevent a denial of service (DoS) attack on the system.

In an NIS environment, the `config/password` property in the name service switch should list the repositories in the following order;

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

Tip – 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 in 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 machine 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 machine 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 (Tasks)

This chapter provides information about the DNS server and client services. The following topics are covered:

DNS Overview

DNS, as with most networking protocols, has two parts: a service providing answers and a client that queries the service. In the Oracle Solaris operating system, the default DNS service is provided by BIND, from the Internet Systems Consortium (ISC), and its associated daemon `named`. 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 maintain for 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 no unicast DNS server needs to be maintained on the local network. There is no need, for example, 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 from Apple Inc. to enable applications to advertise and discover network services using DNS in this Oracle Solaris release.

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 preformed 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 documentation:

- *BIND 9 Administrator's 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`must be managed by using the Service Management Facility (SMF). For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M), `svcs`(1), and `svccfg`(1M) man pages for more details.

The following list provides a short overview of some of the 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.

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 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 query the status of the DNS server and client by using the `svcs` command.
  - The following is an example of the `svcs` command and its output:

```
# 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 is an example of `svcs -l` command and its output.

```
# svcs -l /network/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)
```

- If you need 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 45.

When 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 (Tasks)

The following tasks are documented:

How to Install the DNS Package

Normally, the DNS package is automatically installed with the Oracle Solaris release. If the package was not included when the server was installed, use the following procedure to install the package.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Install the DNS package.**

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

How to Configure a DNS Server

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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

   Before the `named` daemon will start, a valid configuration file must exist. The file is called `/etc/named.conf` by default. The configuration of `named` might be very 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.**

   This file is used to configure remote control access of the DNS server.

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

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

   See "How to Configure DNS Server Options" on page 45.

5. **Start the DNS service.**

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

How to Create an `rndc.conf` File

The `/etc/rndc.conf` file is used to configure remote control access of the DNS server daemon, `named`, by using the `rndc` command. To create a default file, use the following procedure. Refer to the `rndc.conf(4)` man page for further options.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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

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

3. **Restart the DNS service.**

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

How to Configure DNS Server Options

This procedure explains how to select the IPv4 transport protocol for `named` traffic. See the `named(1M)` man page.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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

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

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

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

How to Run the DNS Service as an Alternative User

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

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Add the user to the appropriate role.**

```
# usermod -A solaris.smf.manage.bind dnsadmin
```

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

```
# svccfg -s network/dns/server
svc:/network/dns/server:default> setprop start/user = dnsadmin
svc:/network/dns/server:default> setprop start/group = dnsadmin
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`, the `named` daemon must be configured to use an alternative file.

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

5. **Change the configuration to use the new directory.**

   Add 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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Configure the DNS domain.**

   First, list the domains to search and the IP addresses for the DNS name servers. Then , update the SMF repository.

```
# svccfg -s network/dns/client
svc:/network/dns/client> setprop config/search = astring: ("example.com" "sales.example.com")
svc:/network/dns/client> setprop config/nameserver = net_address: (192.168.1.10 192.168.1.11)
svc:/network/dns/client> select network/dns/client:default
svc:/network/dns/client:default> refresh
svc:/network/dns/client:default> quit
```

3. **Update name service switch information to use DNS.**

   The first command updates the DNS configuration information in the SMF repository.

   ```
   # 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. **Write the new information into the `/etc/resolv.conf` file.**

   The `/etc/resolv.conf` is still used by some processes so after any changes to the SMF repository that would change the contents of the file, the file should be recreated.

   ```
   # nscfg export svc:/network/dns/client:default
   ```

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

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

▼ How to Troubleshoot DNS Server Startup Issues

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **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(1M)
      See: /var/svc/log/network-dns-server:default.log
   Impact: None.
   ```

3. **Check the DNS service log file.**

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

4. **Check syslog messages.**

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

5. **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
   ```

6 **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
```

▼ How to Verify the DNS Configuration

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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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) If the change needs to be reflected in the running service, update the SMF repository and enable the DNS service.**

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

Administering Multicast DNS

The following sections explain how to enable multicast DNS (mDNS) and DNS service discovery. Also provided are examples of how to advertise resources for DNS service discovery.

How to Enable mDNS and DNS Service Discovery

For mDNS and DNS Service Discovery to function, mDNS must be deployed on all systems that are to participate in mDNS. The mDNS service is used to advertise the availability of services provided on the system.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **If needed, install the mDNS package.**

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

3. **Update name service switch information.**

   To be able to resolve local hosts, change 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
   ```

4. **Enable the mDNS service.**

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

   Enabling mDNS in this way ensures that your changes persist through upgrades and reboots. For more information, see the `svcadm`(1M) man page.

5. **(Optional) If needed, check the mDNS error log.**

   Check the mDNS service log, `/var/svc/log/network-dns-multicast:default.log`, for errors or messages.

Advertising Resources for DNS

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

The `dns-sd` command is primarily for interactive use, mainly because its command-line arguments and its output format can change over time, which makes invoking it from a shell script unpredictable and risky. Additionally, the asynchronous nature of DNS service discovery (DNS-SD) does not easily lend itself to script-oriented programming.

For complete information, see the `dns-sd`(1M) man page. To incorporate the DNS service in applications, see the `libdns-sd`(3DNS_SD) man page.

The following are examples of advertising services using DNS service discovery.

EXAMPLE 3–1 Advertising a Printing Service

The following command advertises the existence of 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 3–2 Advertising a Web Page

The following command 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 includes tables of the files, daemons, and commands that are associated with the DNS service. In addition, a table of some of the flags that are used when the ISC version of BIND was built is included.

DNS Files

The following table describes the files that are associated with the DNS service.

TABLE 3–1 DNS Files

File Name	Function
`/etc/named.conf`	Provides configuration information for the `named` daemon. See the `named.conf(4)` man page for more information.
`/etc/rndc.conf`	Provides configuration information for the `rndc` command. See the `rndc.conf(4)` man page for more information.

DNS Commands and Daemons

The following table describes the commands and daemons that are associated with the DNS service.

TABLE 3–2 DNS Commands and Daemons

File Name	Function
`/usr/bin/dns-sd`	Finds or lists resources used by the mDNS service. See the `dns-sd`(1M) man page for more information.
`/usr/sbin/dig`	Requests DNS responses from a DNS server. Often used to troubleshoot. See the `dig`(1M) man page for more information.
`/usr/sbin/dnssec-dsfromkey`	Generates a DS RR from a key file. See the `dnssec-dsfromkey`(1M) man page for more information.
`/usr/sbin/dnssec-keyfromlabel`	Retrieves selected keys from cryptographic device and builds a key file. See the `dnssec-keygen`(1M) man page for more information.
`/usr/sbin/dnssec-keygen`	Creates keys and key files for secure DNS and for transaction signatures (TSIG). See the `dnssec-keygen`(1M) man page for more information.
`/usr/sbin/dnssec-signzone`	Signs a DNS zone. See the `dnssec-signzone`(1M) man page for more information.
`/usr/sbin/host`	Performs simple DNS lookups, often converting host names to IP addresses or IP addresses to host names. See the `host`(1M) man page for more information.
`/usr/sbin/named`	DNS server daemon, which responds to information requests from clients. See the `named`(1M) man page for more information.
`/usr/sbin/named-checkconf`	Checks the syntax of the `named.conf` file. See the `named`(1M) man page for more information.
`/usr/sbin/named-checkzone`	Checks the syntax and integrity of a DNS zone file. See the `named-checkzone`(1M) man page for more information.
`/usr/sbin/named-compilezone`	Converts a DNS zone file. See the `named-compilezone`(1M) man page for more information.
`/usr/sbin/nscfg`	Legacy name service configuration utility, which imports or exports `resolv.conf` content from the SMF repository. See the `nscfg`(1M) man page for more information.
`/usr/sbin/nslookup`	Deprecated: Queries the DNS server. Instead use the `dig` command.
`/usr/sbin/nsupdate`	Submits DNS update requests to a DNS server. See the `nsupdate`(1M) man page for more information.
`/usr/sbin/rndc`	Provides remote control of the DNS server daemon. See the `rndc`(1M) man page for more information.
`/usr/sbin/rndc-confgen`	Generates configuration files for the `rndc` command. See the `rndc-confgen`(1M) man page for more information.

Compilation Flags Used When BIND Was Built

You can view the flags that were used to compile BIND by using the `named -V` command. This table shows some of the compilation flags that were used when building the ISC version of BIND for the Oracle Solaris 11 release.

TABLE 3–3 BIND Compilation Flags

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

◆ ◆ ◆ CHAPTER 4

Setting Up Oracle Solaris Active Directory Clients (Tasks)

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. The following topics are included:

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. The kclient utility is used to join the client to AD. During the join operation, kclient configures Kerberos v5 on the client. Thereafter, nss_ad can be used 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.

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

The idmap and svc:/system/name-service/cache services must be enabled 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).

Ensure that all AD user and group names are qualified with domain names such as `user@domain` or `group@domain`. For example, `getpwnam(dana)` will fail, but `getpwnam(dana@domain)` will succeed, provided that `dana` 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. They 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 them 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 46 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. This is required for the `idmap` auto-discovery feature to work properly.

In the following example, the `dig` commands verify that the AD server can be resolved by using its name and IP address.

```
# dig -x 192.168.11.22 +short
myserver.ad.example
# dig myserver.ad.example +short
192.168.11.22
```

2 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.

---

3 Verify that 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
```

4 Use the kclient utility to join the system to the AD domain.

For example:

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

5 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
```

6 Enable the idmap service.

```
# svcadm enable idmap
```

7 Update the SMF repository for the name service switch service.

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

Note – The nscd module automatically restarts if necessary, whenever name service switch is refreshed.

---

8 Verify that 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

The passwd(4) 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.

The ad search order can be added to existing valid passorwd 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 the nss_ad Naming Service Module Retrieves Data From AD

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

Retrieving passwd Information

The following syntax shows the proper form of a passwd entry:

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

See the passwd(4) man page for more information.

The nss_ad module retrieves passwd information from AD as follows:

- *username* – Field uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, terryb@example.com.
- *password* – Field uses the value of x because the user password is not available in the AD object.
- *uid* – Field uses the Windows user's SID from the objectSID AD attribute, which is mapped to the UID by using the idmap service.
- *gid* – Field 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. For users in AD, the primaryGroupID

attribute is an optional attribute, so 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* – Value of the homeDirectory AD attribute, if a value exists. Otherwise, the field is left empty.
- *login-shell* – Field is left empty because there is no login shell attribute in the native AD schema.

Retrieving shadow Information

The following syntax shows the proper form of a shadow entry:

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

See the shadow(4) man page for more information.

The nss_ad module retrieves shadow information from AD as follows:

- *username* – Field uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, terryb@example.com.
- *password* – Field 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.

Retrieving group Information

The following syntax shows the proper form of a group entry:

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

See the group(4) for man page for more information.

The nss_ad module retrieves information from AD as follows:

- *groupname* – Field uses the value of the samAccountName AD attribute and is qualified by the domain name in which the object resides, for example, admins@example.
- *password* – Field is left empty because the Windows groups do not have passwords.
- *gid* – Field 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.

PART II

NIS Setup and Administration

This part provides an overview of the Network Information Service (NIS) naming service, as well as the setup, administration, and troubleshooting of NIS within the Oracle Solaris OS.

◆ ◆ ◆ CHAPTER 5

Network Information Service (Overview)

This chapter provides an overview of the Network Information Service (NIS).

NIS is a distributed naming service. It 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 Introduction

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.

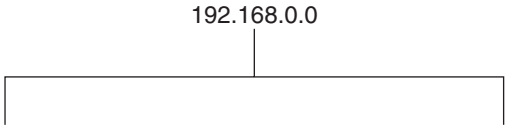
NIS was developed independently of DNS and has a slightly different focus. Whereas DNS focuses on making communication simpler by using machine names instead of numerical IP addresses, NIS focuses on making network administration more manageable by providing centralized control over a variety of network information. NIS stores information not only about machine 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 – In some contexts *machine* names are referred to has *host* names or *machine* names. This discussion uses *machine*, but some screen messages or NIS map names might use *host* or *machine*.

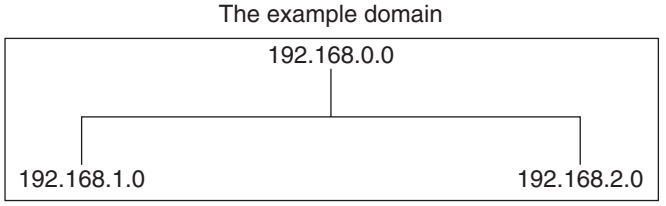
NIS Architecture

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 machines, users, and networks in its namespace. However, it does not use a domain hierarchy. An NIS namespace is flat.



Thus, this physical network would be arranged into one NIS domain.



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. The Oracle Solaris system also allows you to set up the name service switch service 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. See Chapter 2, "Name Service Switch (Overview)," for details.

NIS Machine Types

There are three types of NIS machines.

- Master server
- Slave servers
- Clients of NIS servers

Any machine can be an NIS client, but only machines 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 machine 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 master set of NIS maps. Whenever the master server maps 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 machine 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 – 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.

NIS Elements

The NIS naming service is composed of the following elements:

- Domains (see "The NIS Domain" on page 64)
- Daemons (see "NIS Daemons" on page 64)
- Commands (see "NIS Commands" on page 65)
- Maps (see "NIS Maps" on page 66)

The NIS Domain

An NIS *domain* is a collection of hosts which share a common set of NIS maps. Each domain has a domain name, and each machine sharing 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 host 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 server machines 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 machine. The NIS domain name and all the shared NIS configuration information is managed by the `svc:/network/nis/domain` SMF service.

NIS Daemons

The NIS service is provided by the daemons shown in the following table. 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 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) and `svcs`(1) man pages for more details.

TABLE 5–1 NIS Daemons

Daemon	Function
`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, "NIS Troubleshooting," 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 NIS service is supported by several commands, which are described in the following table.

TABLE 5–2 NIS Command Summary

Command	Description
`make`	Updates NIS maps by reading `/var/yp/Makefile` (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. The `ypmake(1M)` man page describes the functionality of `make` for NIS.
`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.

TABLE 5–2 NIS Command Summary *(Continued)*

Command	Description
`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(1M)` and `ypbind(1M)` 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. The `ypfiles(4)` and `ndbm(3C)` man pages explain the format of the map file.

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 those data files. NIS is deployable 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 machines in the domain to query. NIS slave servers also maintain duplicates of the master server's maps. NIS client machines 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 machines are stored in two maps: `hosts.byname` and `hosts.byaddr`. When a server has a machine's name and needs to find its address, it looks in the `hosts.byname` map. When it 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 machines 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 5–3 NIS Map Descriptions

Map Name	Corresponding 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 machine names and Ethernet addresses. The Ethernet address is the key in the map.
`ethers.byname`	`ethers`	Same as `ethers.byaddr`, except the key is machine name instead of the Ethernet address.
`exec_attr`	`exec_attr`	Contains profile execution attributes.
`group.bygid`	`group`	Contains group security information with group ID as key.
`group.byname`	`group`	Contains group security information with group name as key.
`hosts.byaddr`	`hosts`	Contains machine name, and IP address, with IP address as key.
`hosts.byname`	`hosts`	Contains machine name and IP address, with machine (host) name as key.
`mail.aliases`	`aliases`	Contains aliases and mail addresses, with aliases as key.

TABLE 5–3 NIS Map Descriptions *(Continued)*

Map Name	Corresponding Source File	Description
`mail.byaddr`	`aliases`	Contains mail address and alias, with mail address as key.
`netgroup.byhost`	`netgroup`	Contains group name, user name and machine name.
`netgroup.byuser`	`netgroup`	Same as `netgroup.byhost`, except that key is user name.
`netgroup`	`netgroup`	Same as `netgroup.byhost`, except that key is group name.
`netid.byname`	`passwd`, `hosts` `group`	Used for UNIX-style authentication. Contains machine name and mail address (including domain name). If there is a `netid` file available it is consulted in addition to the data available through the other files.
`publickey.byname`	`publickey`	Contains the public key database used by secure RPC.
`netmasks.byaddr`	`netmasks`	Contains network mask to be used with IP submitting, with the address as key.
`networks.byaddr`	`networks`	Contains names of networks known to your system and their IP addresses, with the address as key.
`networks.byname`	`networks`	Same as `networks.byaddr`, except key is name of network.
`passwd.adjunct.byname`	`passwd` and `shadow`	Contains auditing information and the hidden password information for C2 clients.
`passwd.byname`	`passwd` and `shadow`	Contains password information with user name as key.
`passwd.byuid`	`passwd` and `shadow`	Same as `passwd.byname`, except that key is user ID.
`prof_attr`	`prof_attr`	Contains attributes for execution profiles.
`protocols.byname`	`protocols`	Contains network protocols known to your network.
`protocols.bynumber`	`protocols`	Same as `protocols.byname`, except that key is protocol number.
`rpc.bynumber`	`rpc`	Contains program number and name of RPCs known to your system. Key is RPC program number.

TABLE 5–3 NIS Map Descriptions *(Continued)*

Map Name	Corresponding Source File	Description
`services.byname`	`services`	Lists Internet services known to your network. Key is port or protocol.
`services.byservice`	`services`	Lists Internet services known to your network. Key is service name.
`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 password aging is not being used, then it can be commented out of the mapping file. For more information about the NIS-to-LDAP transition, see Chapter 15, "Transitioning From NIS to LDAP (Tasks)."

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 machine 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 instead.

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 machines and their programs. When a client machine 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. The `ypcat` basic format is the following.

```
% 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 `ypcat -k`. Otherwise, `ypcat` prints blank lines. The `ypcat(1)` man page describes more options for `ypcat`.

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

```
% ypwhich -m mapname
```

where *mapname* is the name or the nickname of the map whose master you want to find. `ypwhich` responds by displaying the name of the master server. For complete information, refer to the `ypwhich(1)` man page.

NIS Map Nicknames

Nicknames are aliases for full map names. To obtain a list of available map nicknames, such as `passwd` for `passwd.byname`, 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. This list can be added to or modified. Currently, there is a limit of 500 nicknames.

NIS Binding

NIS clients are connected to an NIS server through the binding process. This process is supported by the `svc:/network/nis/client` and `svc:/network/nis/domain` services. These services must be enabled 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.

  NIS servers can be added by using the `svccfg` command. 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 allows 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 machinename
```

where *machinename* is the name of the client. If no machine name is mentioned, the `ypwhich` command defaults to the local machine (that is, the machine 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 machine that needs information provided by an NIS map, asks `ypbind` for the name of a server.
2. The `ypbind` daemon looks in 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 tells the client process which server to talk to. 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. The `ypbind` daemon must be started with the broadcast option set (`broadcast`).
2. The `ypbind` daemon issues an RPC broadcast in search of an NIS server.

Note – In order to support such clients, it is necessary to have an NIS server on each subnet requiring NIS service.

---

3. The `ypbind` daemon initiates binding to the first server that responds to the broadcast.
4. The `ypbind` daemon tells the client process which server to talk to. 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 NIS (Tasks)

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

Note – In some contexts, *machine* names are referred to as *host* names or *machine* names. This discussion uses "machine," but some screen messages or NIS map names might use *host* or *machine*.

This chapter covers the following topics:

Configuring NIS Task Map

Task	Description	For Instructions
Prepare source files for conversion.	You clean up local `/etc` files before building the NIS maps from them.	"How to Prepare Source Files for Conversion" on page 77
Set up the master server.	Creates a master server, which is the primary source of NIS information.	"How to Set Up the Master Server" on page 80
Start NIS on the master server.	Starts providing NIS information from an NIS server.	"Starting and Stopping NIS Services on an NIS Server" on page 82

Task	Description	For Instructions
Set up slave servers.	Creates a slave server, which is a secondary source of NIS information.	"How to Set Up a Slave Server" on page 84
Set up an NIS client.	Enables a client to use NIS information.	"Administering NIS Clients" on page 86

Before You Begin Configuring NIS

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

- Plan your NIS domain. See "Planning Your NIS Domain" on page 75 for details.
- Install properly configured name service switch information on all the machines that will be using NIS. See Chapter 2, "Name Service Switch (Overview)," for details.

NIS and the Service Management Facility

The NIS service is managed by the Service Management Facility. For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) and `svcs`(1) man pages for more details.

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

- Administrative actions on this service, such as enabling, disabling, or restarting, can be performed by using the `svcadm` command. However, `ypstart` and `ypstop` can also be used from the command line to start or stop NIS. See the `ypstart`(1M) and `ypstop`(1M) 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(1M)` 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 machines as NIS servers or clients, you must plan the NIS domain.

Decide which machines 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 machines 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 `doc.com`, you can also name your NIS domain `doc.com`. If you wanted to divide `doc.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.doc.com` and the other domain `manf.doc.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 machine can use NIS services, the correct NIS domain name and machine name must be set. A machine's name is set in the `config/nodename` property of the `svc:/system/identity:node` service. The machine's domain name is set in the `config/domainname` property of the `svc:/network/nis/domain` service. These properties are read at boot time. The contents are used by the `uname -S` and `domainname` commands, respectively.

Identify Your NIS Servers and Clients

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

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

Preparing the Master Server

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

Source Files Directory

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 Namespace Security" on page 77 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 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 `PWDDIR` 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

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

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 RBAC source directory that you have selected.**

The RBAC 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(4)` 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 96.

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 Master Server" on page 80.

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" on page 76, 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 machines 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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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

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

How to Set Up the 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 "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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:

   ```
   129.0.0.1    master.example.com master
   129.0.0.2    slave1.example.com slave1
   129.0.0.3    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 machines 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 DNS domain name is set.

```
# svcprop -p config/domainname svc:/network/nis/domain:default
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 67 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 has been previously installed. Normally, you would choose to delete the source files if you want 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 "Managing the Name Service Switch" on page 38.

▼ 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 Master Server" on page 80, when setting up the server to serve the additional domains.

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

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Change the NIS domain name.**

```
# svccfg -s svc:/network/nis/domain
svc:/network/nis/domain: setprop config/domainname = hostname: sales.example.com
svc:/network/nis/domain: quit
```

3 **Refresh the SMF service.**

```
# svcadm refresh svc:/network/nis/domain
```

4 **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.
- 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 the NIS Service Automatically

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 Master Server" on page 80 for more information.

▼ 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 74, or the `svcadm(1M)` man page.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 74, or the `svcadm(1M)` man page.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 74, or the `svcadm`(1M) man page.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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.

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

The following procedure explains how to set up a slave server. Repeat this procedure for each machine you want configured as an NIS slave server.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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:

```
129.0.0.1    master.example.com master
129.0.0.2    slave1.example.com slave1
129.0.0.3    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 86 for details.

---

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 machine as a slave server.

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

where *master* is the machine 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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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
```

Administering NIS Clients

The two methods for configuring a client machine to use NIS as its naming service are explained in this section.

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 machine.

- **Broadcast method** — The preferred method of configuring a client machine to use NIS. See "How to Configure an NIS Client in Broadcast Mode" on page 86 for instructions.
- **Server-list method** — Another method for configuring a client machine by using the `ypinit` command to specify the servers. See "How to Configure an NIS Client Using Specific NIS Servers" on page 87 for instructions.

▼ How to Configure an NIS Client in Broadcast Mode

This is the preferred method for establishing an NIS client.

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 machine cannot obtain namespace data from the NIS service. See "How to Configure an NIS Client Using Specific NIS Servers" on page 87 for instructions.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Set the NIS domain name.**

```
# svccfg -s network/nis/domain
svc:/network/nis/domain> setprop setconfig/domainname = hostname: example.com
svc:/network/nis/domain> quit
# svcadm refresh network/nis/domain
```

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

See "Managing the Name Service Switch" on page 38.

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

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Set the NIS domain name.**

```
# svccfg -s network/nis/domain
svc:/network/nis/domain> setprop setconfig/domainname = hostname: example.com
svc:/network/nis/domain> quit
# svcadm refresh network/nis/domain
# 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 machine, than those servers that are located on more distant parts of the network.

▼ Disabling the NIS Client Services

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Disable the NIS client services.**

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

◆ ◆ ◆ CHAPTER 7

Administering NIS (Tasks)

This chapter describes how to administer NIS. The following topics are covered:

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 74 for more information about using SMF with NIS. For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) 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`(1M) and `ypstop`(1M) man pages for more information.

Password Files and Namespace Security

For security reasons, follow these guidelines.

- It is best to limit access to the NIS maps on the master server.

- The files used to build the NIS password maps should not contain an entry for `root` to protect against unauthorized access. To accomplish this, the password files used to build the password maps should have the `root` entry removed from them and be located in a directory other than the master server's `/etc` directory. This directory 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`.

Note – In addition to the older Solaris 1 version `passwd` file format, this implementation of NIS accepts the Solaris 2 `passwd` and `shadow` file formats as input for building the NIS password maps.

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 "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 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(1M)` 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
   # /usr/ccs/bin/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 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 machines 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 machine
- Define a set of NFS client machines to be given some specific file system access
- Define a set of users who are to have administrator privileges on all the machines 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 machine 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 machine, you only need to add a `netadmin` entry to that machine's `/etc/passwd` file. Netgroup names can also be added to the `/etc/netgroup` file and propagated to the NIS `netgroup` map. See `netgroup(4)` for more detailed information on 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 machine 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 machine 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.

`([-|`*machine*`], [-|`*user*`], [`*domain*`])`

Where *machine* is a machine name, *user* is a user ID, and *domain* is the machine or user's NIS domain. The *domain* element is optional and should only be used to identify machines or users in some other NIS domain. The *machine* 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 machine 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 `juanita` who is in the remote domain `sales` and the machines `altair` and `sirius`.

```
admins (altair, hauri), (sirius,juanita,sales)
admins (altair,-), (sirius,-), (-,hauri), (-,juanita,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 machine 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 machine and user classifications if they encounter netgroup names in the `/etc/hosts.equiv` or `.rhosts` file.

If you add a new NIS user or machine 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 `netgroup(4)` for detailed information on using netgroups and netgroup input file syntax.

Working With NIS Maps

This section contains the following information:

Obtaining 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 machine 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 machine.

Changing a Map's Master Server

To change the master server for a selected map, you first have to 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 a Map's Master Server

1 Become an administrator on the NIS master server.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 (`rlogin`) 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 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:

- `/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 Configuration Files

You do not have to stop and start NIS when changing NIS maps or the map source 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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 `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 the following.

- 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 `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*= 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*= 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*= The NIS domain name. The default value of `DOM` can be set by using the `domainname` command., The domain name is set in the `config/domainname` property of the `svc:/network/nis/domain` service.

Modifying `Makefile` Entries

The following procedure describes how to add and delete databases from the `Makefile`.

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, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 auto_home auto_direct.time auto_home.time
```

   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 next commands. If the operations to the left of the pipe do not successfully complete, the process is terminated with a `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 `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 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 67, 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 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 "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 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.

Note – The above 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 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`

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 accomodate your site requirements. The following example shows the default `ypxfr_1perday` shell script.

EXAMPLE 7–1 `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` *machine* option of `ypxfr` within the shell script. Here is the syntax of the commands you put in the script.

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

Where *machine* 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 machine, `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 Maps

To update a non-default map, you must do the following:

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 96.

- Use the `/usr/sbin/makedbm` program. The `makedbm(1M)` man page fully describes this command.

Using the `makedbm` Command to Modify a Non-Default 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 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 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 Maps From Standard Input

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

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

Modifying 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 the`ypinit` 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.

Adding a 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.

▼ How to Add a New Slave Server

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

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 machine 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 "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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

   ```
   # svcprop -p config/domainname svc:/network/nis/domain:default
   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 machine as a slave.**

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

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

11. **Stop the machine 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
```

Using NIS With C2 Security

If the `$PWDIR/security/passwd.adjunct` file is present, C2 security is started automatically. (`$PWDIR` is defined in `/var/yp/Makefile`.) The C2 security mode uses the `passwd.adjunct` file to create the `passwd.adjunct` NIS map. In this implementation, NIS allows you to use both the `passwd.adjunct` file and `shadow` file to manage security. The `passwd.adjunct` file is processed only when you type the following.

```
# make passwd.adjunct
```

The `make passwd` command processes the `passwd` map only, not the `passwd.adjunct` map when you run `make` manually in the C2 security mode.

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(1M)`, `ypstart(1M)`, and `svcadm(1M)` man pages.

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

   ```
   # svcprop -p config/domainname svc:/network/nis/domain:default
   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
     ```

Setting a Machine's NIS Domain

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

▼ How to Set a Machine's NIS Domain Name

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Define the NIS domain name.

```
# svccfg -s nis/domain
svc:/network/nis/domain> setprop config/domainname = hostname research.example.com
svc:/network/nis/domain> quit
```

3 Update and run the domain name services.

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

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

See Chapter 6, "Setting Up and Configuring NIS (Tasks)," for details.

Using NIS in Conjunction With DNS

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

▼ How to Configure Machine Host Name and Address Lookup Through NIS and DNS

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 modify the following lines.

```
#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.

```
# /usr/ccs/bin/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

NIS Troubleshooting

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, "Network Information Service (Overview)," 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. 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 74 for more information about using SMF with NIS. For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) 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`(1M) and `ypstop`(1M) man pages for more information.

NIS Binding Problems

Symptoms of 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 115.

`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 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 /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 machine is not the same as the server domain name listed as a directory in `/var/yp`, the domain name specified in the machine's `/etc/defaultdomain` file is incorrect. Reset the NIS domain name as shown in "How to Set a Machine's NIS Domain Name" on page 107.

Note – The NIS domain name is case-sensitive.

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 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 machine's local `hosts` file in addition to NIS? See Chapter 2, "Name Service Switch (Overview)," for more information on the switch.
- Is the name service switch set up to check `files` first for `services` and `rpc`? See Chapter 2, "Name Service Switch (Overview)," 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 machine gets NIS service, it does not matter where the service comes from. For example, an NIS server machine can get its own NIS services from another NIS server on the network.

When 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`(1M) man page.

```
# /usr/lib/netsvc/yp/ypbind -ypset
```

For another method, see "Binding to a Specific NIS Server" on page 107.

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`(1M) and `rpcinfo`(1M) man pages.

You might be able to communicate with `rpcbind` on the problematic client from a machine operating normally. From the functioning machine, type the following:

```
% rpcinfo client
```

If `rpcbind` on the problematic machine 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 machine will have different addresses. If the addresses are not displayed, `ypbind` has been unable to register its services. Reboot the machine 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 112. 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.

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.

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 machine 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 117.
- 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 117.
- Check the `ypservers` map on the master server. See "Check the `ypservers` Map" on page 117.

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 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 114. 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 machine 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 machine. 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).

PART III

LDAP Naming Services

This part provides an overview of the LDAP naming services. Additionally, it covers the setup, configuration, administration, and troubleshooting of LDAP naming services in the Oracle Solaris OS, with a focus on the use of Oracle Directory Server Enterprise Edition.

◆ ◆ ◆ CHAPTER 9

Introduction to LDAP Naming Services (Overview)

The LDAP chapters describe how to set up an LDAP naming services client to work with Oracle Directory Server Enterprise Edition (formerly Sun Java System Directory Server). However, while using the Oracle Directory Server Enterprise Edition is recommended, it is not required. A brief description of generic directory server requirements appears in Chapter 14, "LDAP Naming Service (Reference)."

Note – A directory server is not necessarily an LDAP server. However, in the context of these chapters, the term "directory server" is synonymous with "LDAP server."

The following subjects are covered in this chapter:

Audience Assumptions

The LDAP naming services chapters are written for system administrators who already have a working knowledge of LDAP. Following is a partial list of concepts with which you must be very familiar. Otherwise, you might have difficulty using this guide to deploy LDAP naming services in the Oracle Solaris system.

- LDAP Information Model (entries, object classes, attributes, types, values)
- LDAP Naming Model (Directory Information Tree (DIT) structure)
- LDAP Functional Model (search parameters: base object (DN), scope, size limit, time limit, filters (browsing indexes for the Oracle Directory Server Enterprise Edition), attribute list)
- LDAP Security Model (authentication methods, access control models)
- Overall planning and design of an LDAP directory service, including how to plan the data and how to design the DIT, topology, replication, and security

Suggested Background Reading

To learn more about any of the preceding concepts or to study LDAP and the deployment of directory services in general, refer to the following sources:

- *Oracle Directory Server Enterprise Edition Deployment Guide*

  This guide provides a foundation for planning your directory, including directory design, schema design, the directory tree, topology, replication, and security. The last chapter provides sample deployment scenarios to help you plan both simple, smaller-scale deployments and complex worldwide deployments.
- *Oracle Directory Server Enterprise Edition Administration Guide*

Additional Prerequisite

If you need to install Oracle Directory Server Enterprise Edition, refer to the *Installation Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

LDAP Naming Services Compared to Other Naming Services

See "Naming Services: A Quick Comparison" on page 31 for comparison of the DNS, NIS, and LDAP naming services.

Advantages of LDAP Naming Services

- LDAP enables you to consolidate information by replacing application-specific databases, which reduces the number of distinct databases to be managed.
- LDAP allows data to be shared by different naming services.
- LDAP provides a central repository for data.
- LDAP allows for more frequent data synchronization between masters and replicas.
- LDAP is multi-platform and multi-vendor compatible.

Restrictions of LDAP Naming Services

Following are some restrictions that are associated with LDAP naming services:

- An LDAP server is currently not supported as being its own client.
- Setting up and managing an LDAP naming service is more complex and requires careful planning.
- An NIS client and a native LDAP client cannot coexist on the same client machine.

Note – A directory server (an LDAP server) *cannot* be its own client. That is, you cannot configure the machine that is running the directory server software to become an LDAP naming services client.

LDAP Naming Services Setup (Task Map)

Task	For Instructions
Plan the network model.	"Planning the LDAP Network Model" on page 149
Plan the directory information tree.	Chapter 10, "Planning Requirements for LDAP Naming Services (Tasks)"
Set up replica servers.	"LDAP and Replica Servers" on page 152
Plan the security model.	"Planning the LDAP Security Model" on page 153
Choose client profiles and default attribute values.	"Planning Client Profiles and Default Attribute Values for LDAP" on page 154
Plan the data population.	"Planning the LDAP Data Population" on page 155
Configure Oracle Directory Server Enterprise Edition prior to using it with LDAP naming services.	*Oracle Directory Server Enterprise Edition*

Task	For Instructions
Set up Oracle Directory Server Enterprise Edition for use with LDAP naming clients.	Chapter 11, "Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)"
Initialize an LDAP client.	"Initializing an LDAP Client" on page 175
Initialize a client by using profiles.	"How to Initialize an LDAP Client by Using Profiles" on page 176
Initialize a client manually.	"How to Initialize an LDAP Client Manually" on page 180
Uninitialize a client.	"How to Uninitialize an LDAP Client" on page 181
Use service search descriptors to modify client profiles.	"Using Service Search Descriptors to Modify Client Access to Various Services" on page 160
Retrieve naming service information.	"Retrieving LDAP Naming Services Information" on page 184
Customize a client environment.	"Customizing the LDAP Client Environment" on page 185

LDAP Data Interchange Format

LDAP Data Interchange Format (LDIF) is used as a common text-based interchange format between many LDAP tools such as `ldapadd` and `ldapmodify`. LDIF is fully described in LDIF RFC 2849. The following are two examples of LDIF output created by the `ldapadd` command. Use `ldaplist(1)` with the `-l` option to display the following information.

```
% ldaplist -l hosts myhost
hosts

dn: cn=myhost+ipHostNumber=7.7.7.115,ou=Hosts,dc=mydc,dc=mycom,dc=com
cn: myhost
iphostnumber: 7.7.7.115
objectclass: top
objectclass: device
objectclass: ipHost
description: host 1 - floor 1 - Lab a - building b
% ldaplist -l passwd user1
passwd

dn: uid=user1,ou=People,dc=mydc,dc=mycom,dc=com
uid: user1
cn: user1
userpassword: {crypt}duTx91g7PoNzE
uidnumber: 199995
gidnumber: 20
gecos: Joe Smith [New York]
homedirectory: /home/user1
loginshell: /bin/csh
```

```
objectclass: top
objectclass: shadowAccount
objectclass: account
objectclass: posixAccount
```

Using Fully Qualified Domain Names With LDAP

If LDAP is used to resolve host names, then an LDAP client always returns a fully qualified domain name (FQDN) for a host name. The LDAP FQDN is similar to the FQDN returned by DNS. For example, suppose your domain name is the following:

```
west.example.net
```

Both `gethostbyname()` and `getnameinfo()` return the FQDN version when looking up the host name *server*:

```
server.west.example.net
```

Default Directory Information Tree

By default, LDAP clients access the information assuming that the directory information tree (DIT) has a given structure. For each domain supported by the LDAP server, there is a subtree with an assumed structure. This default structure, however, can be overridden by specifying Service Search Descriptors (SSDs). For a given domain, the default DIT will have a base container that holds a number of well known containers that hold entries for a specific information type. See the following table for the names of these subtrees. This information can be found in RFC 2307 and others.

TABLE 9–1 DIT Default Locations

Default Container	Information Type
`ou=Ethers`	bootparams, ethers
`ou=Group`	group
`ou=Hosts`	hosts, ipnodes, publickey for hosts
`ou=Aliases`	aliases
`ou=Netgroup`	netgroup
`ou=Networks`	networks, netmasks
`ou=People`	passwd, shadow, user_attr, audit_user, publickey for users
`ou=Protocols`	protocols
`ou=Rpc`	rpc

TABLE 9–1 DIT Default Locations *(Continued)*

Default Container	Information Type
`ou=Services`	services
`ou=SolarisAuthAttr`	auth_attr
`ou=SolarisProfAttr`	prof_attr, exec_attr
`ou=projects`	project
`automountMap=auto_*`	auto_*

Default LDAP Schema

Schemas are definitions describing what types of information can be stored as entries in an LDAP directory. To support LDAP naming clients, the directory server's schema might need to be extended. Detailed information about IETF and Oracle Solaris specific schemas is included in Chapter 14, "LDAP Naming Service (Reference)." The various RFCs can also be accessed on the IETF Web site `http://www.ietf.org`.

Service Search Descriptors and Schema Mapping

Note – If you use schema mapping, you must do so in a very careful and consistent manner. Make sure the syntax of the mapped attribute is consistent with the attribute it is mapped to. In other words, make sure that single-valued attributes map to single-valued attributes, that the attribute syntaxes are in agreement, and that mapped object classes have the correct mandatory (possibly mapped) attributes.

As previously discussed, LDAP naming services expect, by default, the DIT to be structured in a certain way. If you want, you can instruct the LDAP naming service to search in other locations than the default locations in the DIT by using service search descriptors (SSDs). Additionally, you can specify that different attributes and object classes be used in place of those specified by the default schema. For a list of default filters, see "Default Filters Used by LDAP Naming Services" on page 216.

Description of SSDs

The `serviceSearchDescriptor` attribute defines how and where an LDAP naming service client should search for information for a particular service. The `serviceSearchDescriptor` contains a service name, followed by one or more semicolon-separated base-scope-filter triples.

These base-scope-filter triples are used to define searches only for the specific service and are searched in order. If multiple base-scope-filters are specified for a given service, then when that service looks for a particular entry, it will search in each base with the specified scope and filter.

Note – The default location is not searched for a service (database) with an SSD unless it is included in the SSD. Unpredictable behavior will result if multiple SSDs are given for a service.

In the following example, the LDAP naming service client performs a one-level search in `ou=west,dc=example,dc=com` followed by a one-level search in `ou=east,dc=example,dc=com` for the `passwd` service. To look up the `passwd` data for a user's `username`, the default LDAP filter `(&(objectClass=posixAccount)(uid=username))` is used for each `BaseDN`.

```
serviceSearchDescriptor: passwd:ou=west,dc=example,dc=com;ou=east,
dc=example,dc=com
```

In the following example, the LDAP naming service client would perform a subtree search in `ou=west,dc=example,dc=com` for the `passwd` service. To look up the `passwd` data for user `username`, the subtree `ou=west,dc=example,dc=com` would be searched with the LDAP filter `(&(fulltimeEmployee=TRUE)(uid=username))`.

```
serviceSearchDescriptor: passwd:ou=west,dc=example,
dc=com?sub?fulltimeEmployee=TRUE
```

It is also possible to associate multiple containers with a particular service type. In the following example, the service search descriptor specifies searching for the password entries in three containers.

```
ou=myuser,dc=example,dc=com
ou=newuser,dc=example,dc=com
ou=extuser,dc=example,dc=com
```

Note that a trailing ',' in the example implies that the `defaultSearchBase` is appended to the relative base in the SSD.

```
defaultSearchBase: dc=example,dc=com
serviceSearchDescriptor: \
passwd:ou=myuser,;ou=newuser,;ou=extuser,dc=example,dc=com
```

`attributeMap` Attributes

The LDAP naming service allows one or more attribute names to be remapped for any of its services. (The LDAP client uses the well-known attributes documented in Chapter 14, "LDAP Naming Service (Reference).") If you map an attribute, you must be sure that the attribute has the same meaning and syntax as the original attribute. Note that mapping the `userPassword` attribute might cause problems.

There are a couple of reasons you might want to use schema mappings.

- You want to map attributes in an existing directory server
- If you have user names that differ only in case, you must map the `uid` attribute, which ignores case, to an attribute that does not ignore case

The format for this attribute is `service:attribute-name=mapped-attribute-name`.

If you want to map more than one attribute for a given service, you can define multiple `attributeMap` attributes.

In the following example, the `employeeName` and `home` attributes would be used whenever the `uid` and `homeDirectory` attributes would be used for the `passwd` service.

```
attributeMap: passwd:uid=employeeName
attributeMap: passwd:homeDirectory=home
```

There exists one special case where you can map the `passwd` service's `gecos` attribute to several attributes. The following is an example.

```
attributeMap: gecos=cn sn title
```

This maps the `gecos` values to a space separated list of the `cn`, `sn`, and `title` attribute values.

`objectclassMap` Attribute

The LDAP naming service allows object classes to be remapped for any of its services. If you want to map more than one object class for a given service, you can define multiple `objectclassMap` attributes. In the following example, the `myUnixAccount` object class is used whenever the `posixAccount` object class is used.

```
objectclassMap: passwd:posixAccount=myUnixAccount
```

LDAP Client Profiles

To simplify client setup, and avoid having to reenter the same information for each and every client, create a single client profile on the directory server. This way, a single profile defines the configuration for all clients configured to use it. Any subsequent change to the profile attributes is propagated to the clients at a rate defined by the refresh interval.

Configuration information specified in the LDAP client profiles, is automatically imported into the SMF repository when the `svc:/network/ldap/client` service is started.

Any client profiles should be stored in a well-known location on the LDAP server. The root DN for the given domain must have an object class of `nisDomainObject` and a `nisDomain` attribute containing the client's domain. All profiles are located in the `ou=profile` container relative to this container. These profiles should be readable anonymously.

LDAP Client Profile Attributes

The following table shows the LDAP client's profile attributes, which can be set automatically when you run `idsconfig`. See "How to Initialize an LDAP Client Manually" on page 180 and the `idsconfig(1M)` man page for information on how to set a client profile manually.

TABLE 9–2 LDAP Client Profile Attributes

Attribute	Description
`cn`	The profile name. The attribute has no default value. The value must be specified.
`preferredServerList`	The host addresses of the preferred servers is a space separated list of server addresses. (Do not use host names.) The servers in this list are tried in order *before* those in `defaultServerList` until a successful connection is made. This has no default value. At least one server must be specified in either `preferredServerList` or `defaultServerList`.
`defaultServerList`	The host addresses of the default servers is a space separated list of server addresses. (Do not use host names.) After the servers in `preferredServerlist` are tried, those default servers on the client's subnet are tried, followed by the remaining default servers, until a connection is made. At least one server must be specified in either `preferredServerList` or `defaultServerList`. The servers in this list are tried only after those on the preferred server list. This attribute has no default value.
`defaultSearchBase`	The DN relative to which to locate the well-known containers. There is no default for this value. However, this can be overridden for a given service by the `serviceSearchDescriptor` attribute.
`defaultSearchScope`	Defines the scope of a database search by a client. It can be overridden by the `serviceSearchDescriptor` attribute. The possible values are `one` or `sub`. The default value is a `one` level search.
`authenticationMethod`	Identifies the method of authentication used by the client. The default is none (anonymous). See "Choosing Authentication Methods for the LDAP Naming Service" on page 137 for more information.
`credentialLevel`	Identifies the type of credentials a client should use to authenticate. The choices are `anonymous`, `proxy`, or `self` (also known as per-user). The default is `anonymous`.
`serviceSearchDescriptor`	Defines how and where a client should search for a naming database, for example, if the client should look in one or more points in the DIT. By default no SSDs are defined.

TABLE 9–2 LDAP Client Profile Attributes *(Continued)*

Attribute	Description
`serviceAuthenticationMethod`	Authentication method used by a client for the specified service. By default, no service authentication methods are defined. If a service does not have `serviceAuthenticationMethod` defined, it will default to the value of `authenticationMethod`.
`attributeMap`	Attribute mappings used by client. By default no `attributeMap` is defined.
`objectclassMap`	Object class mappings used by client. By default no `objectclassMap` is defined.
`searchTimeLimit`	Maximum time [in seconds] a client should allow for a search to complete before timing out. This does not affect the time the LDAP server will allow for a search to complete. The default value is `30` seconds.
`bindTimeLimit`	Maximum time in seconds a client should allow to bind with a server before timing out. Default value is `30` seconds.
`followReferrals`	Specifies whether a client should follow an LDAP referral. Possible values `TRUE` or `FALSE`. The default value is `TRUE`.
`profileTTL`	Time between refreshes of the client profile from the LDAP server by the `ldap_cachemgr(1M)`. Default is `43200` seconds or 12 hours. If given a value of 0, the profile will never be refreshed.

Local LDAP Client Attributes

The following table lists the LDAP client attributes that can be set locally using the `ldapclient` command. See the `ldapclient(1M)` man page for more information.

TABLE 9–3 Local LDAP Client Attributes

Attribute	Description
`adminDN`	Specifies the administrator entry's distinguished name for the admin credential. If the value of the `enableShadowUpdate` switch is `true` on the client system, and `credentialLevel` has a value other than `self`, then `adminDN` must be specified.
`adminPassword`	Specifies the administrator entry's password for the admin credential. If the value of the `enableShadowUpdate` switch is `true` on the client system, and `credentialLevel` has a value other than `self`, then `adminPassword` must be defined.
`domainName`	Specifies the client's domain name (which becomes the default domain for the client system). This attribute has no default value and must be specified.

TABLE 9–3 Local LDAP Client Attributes *(Continued)*

Attribute	Description
proxyDN	The proxy's distinguished name. If the client system is configured with credentialLevel of proxy, the proxyDN must be specified.
proxyPassword	The proxy's password. If the client system is configured with credentialLevel of proxy, proxyPassword must be defined.
certificatePath	The directory on the local file system containing the certificate databases. If a client system is configured with authenticationMethod or serviceAuthenticationMethod using TLS, then this attribute is used. The default value is /var/ldap.

Note – If the BaseDN in an SSD *contains a trailing comma*, it is treated as a relative value of the defaultSearchBase. The values of the defaultSearchBase are appended to the BaseDN before a search is performed.

ldap_cachemgr Daemon

ldap_cachemgr is a daemon that runs on LDAP client machines. The svc:/network/ldap/client service manages the ldap_cachemgr daemon, so the service must be enabled in order for the daemon to run properly. The daemon performs the following key functions.

- Gains access to the configuration data, running as root
- Refreshes the client configuration information stored in the profiles on the server and pulls this data from the clients
- Maintains a sorted list of active LDAP servers to use
- Improves lookup efficiency by caching some common lookup requests submitted by various clients
- Improves the efficiency of host lookups
- If the enableShadowUpdate switch is set to true, gains access to the configured administrator credential and performs updates to the shadow data.

Note – ldap_cachemgr must be running at all times for LDAP naming services to work.

Refer to the ldap_cachemgr(1M) man page for detailed information.

LDAP Naming Services Security Model

The LDAP naming services can use the LDAP repository in two different ways. One is as a source of both a naming service and an authentication service. The other is strictly as the source of naming data. This section discusses the concepts of client identity, authentication methods, The `pam_ldap` and `pam_unix_*` modules, and account management when the LDAP repository is used as both a naming service and authentication service. This section also discusses the use of LDAP naming services in conjunction with the Kerberos environment (Part VI, "Kerberos Service," in *Oracle Solaris Administration: Security Services*) and `pam_krb5(5)` modules.

Note – Previously, if you enabled `pam_ldap` account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as `rsh`, `rlogin`, or `ssh` would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is `1.3.6.1.4.1.42.2.27.9.5.8`, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

Note – If you use Kerberos as your authentication system and integrate it with the LDAP naming system, you will be able to support a single sign on (SSO) environment in your enterprise through Kerberos. You will also be able to use that same identity system when querying LDAP naming data on a per-user or per-host basis.

To access the information in the LDAP repository, clients can first establish identity with the directory server. This identity can be either anonymous, or as a host or user that is recognized by the LDAP server. Based on the client's identity and the server's access control information (ACI), the LDAP server will allow the client to read directory information. For more information on ACIs, consult the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

If the identity is based on the host that the request is coming from, then you are using proxy authentication. Once the host has been authenticated, all users on that host get access. If the identity is based on the user, then you are using per-user authentication. Each user on a host must be authenticated to get access.

If the client is connecting as anything other than anonymous for any given request, the client must prove its identity to the server using an authentication method supported by both the client and the server. Once the client has established its identity, it can then make the various LDAP requests.

When you log into a system, the PAM service may use information from the local machine, from the LDAP service, from a Kerberos server or some combination of the three to decide if the log in attempt will be successful. When the `pam_kerb` module is used, the decision to allow access is decided by the Kerberos server. When the `pam_ldap` module is used, half of the decision must come from the LDAP server and the other half comes from the local host. Information from the local host, using the `pam_unix_*` modules, the decision is made locally.

When you use `pam_ldap` to login using the LDAP service, there is a distinction between how the naming service and the authentication service (`pam_ldap`) access the directory. The naming service reads various entries and their attributes from the directory based on predefined identity. The authentication service establishes whether the user has entered the correct password by using that user's name and password to authenticate to the LDAP server. See the `pam_ldap(5)` man page for more information about the authentication service.

When Kerberos is used to perform authentication, and when authentication in LDAP naming services is also enabled (as is required for per-user mode), Kerberos can provide dual functions. Kerberos authenticates to the server and the Kerberos identity for the principal (user or host) is used to authenticate to the directory. In this way, the same user identity that is used to authenticate to the system is also used to authenticate to the directory for lookups and updates. Administrators can use access control information (ACI) in the directory to limit the results out of the naming service if desired.

Transport Layer Security

Transport layer security (TLS) can be used to secure communication between an LDAP client and the directory server, providing both privacy and data integrity. The TLS protocol is a superset of the Secure Sockets Layer (SSL) protocol. LDAP naming services support TLS connections. Be aware that using SSL adds load to the directory server and the client.

You will need to set up your directory server for SSL. For more information about setting up Oracle Directory Server Enterprise Edition for SSL, see the Administration Guide for the version of Oracle Directory Server Enterprise Edition that you are using. You will also need to set up your LDAP client for SSL.

If using TLS, the necessary security databases must be installed. In particular, the certificate and key database files are needed. For example, if you adopt an older database format from Netscape Communicator, two files, `cert7.db` and `key3.db`, are required. Or if you use a new database format from Mozilla, three files, `cert8.db`, `key3.db`, and `secmod.db` are needed. The `cert7.db` or `cert8.db`file contains trusted certificates. The `key3.db`file contains the client's keys. Even if the LDAP naming service client does not use client keys, this file must be present. The `secmod.db` file contains the security modules such as the `PKCS#11` module. This file is not required if the older format is used.

See "Setting Up TLS Security" on page 181 for more information.

Assigning Client Credential Levels

LDAP naming services clients authenticate to the LDAP server according to a client's credential level. LDAP clients can be assigned several levels with which to authenticate to a directory server.

- `anonymous`
- `proxy`
- `proxy anonymous`
- `self` (called per-user in this document)

LDAP `anonymous` Credential Level

If you use `anonymous` access, you can access only the data that is available to everyone. In anonymous mode, an LDAP BIND operation does not take place. Also, you should consider the security implications. Allowing `anonymous` access for certain parts of the directory implies that anyone with access to the directory has read access. If you use an `anonymous` credential level, you need to allow read access to all the LDAP naming entries and attributes.

Caution – Allowing `anonymous` write to a directory should never be allowed, as anyone could change information in the DIT to which they have write access, including another user's password, or their own identity.

Note – Oracle Directory Server Enterprise Edition allows you to restrict access based on IP addresses, DNS name, authentication method, and time-of-day. You might want to limit access with further restrictions. For more information, see "Managing Access Control" in the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

LDAP `proxy` Credential Level

The client authenticates or binds to a single shared set of LDAP bind credentials, otherwise known as a proxy account. This proxy account can be any entry that is allowed to bind to the directory. This proxy account needs sufficient access to perform the naming service functions on the LDAP server. The proxy account is a shared-per-system resource. That is, each user logged in to a system using proxy access, including the root user, sees the same results as all other users on that system. You need to configure the `proxyDN` and `proxyPassword` on every client using the `proxy` credential level. The encrypted `proxyPassword` is stored locally on the client. You can set up different proxies for different groups of clients. For example, you can configure a proxy for all the sales clients to access both the company-wide-accessible and sales directories, while preventing sales clients from accessing human resource directories with payroll information. Or, in the most extreme cases, you can either assign different proxies to each client or assign just one proxy to all clients. A typical LDAP deployment would probably lie between the two extremes. Consider the choices carefully. Too few proxy agents might limit your ability to control user access to resources. However, having too many proxies complicates the setup and maintenance of the system. You need to grant the appropriate rights to the proxy user, depending on your environment. See "Credential Storage for LDAP Clients" on page 136 for information on how to determine which authentication method makes the most sense for your configuration.

If the password changes for a proxy user, you need to update it on every client that uses that proxy user. If you use password aging on LDAP accounts, be sure to turn it off for proxy users.

Note – Be aware that the proxy credential level applies to all users and processes on any given system. If two users need to use different naming policies, they must use different machines, or they must use the per-user authentication model.

In addition, if clients are using a `proxy` credential to authenticate, the `proxyDN` must have the same `proxyPassword` on all of the servers.

LDAP `proxy anonymous` Credential Level

`proxy anonymous` is a multi-valued entry, in that more than one credential level is defined. A client assigned the `proxy anonymous` level will first attempt to authenticate with its proxy identity. If the client is unable to authenticate as the proxy user for whatever reason (user lockout, password expired, for example), then the client will use anonymous access. This might lead to a different level of service, depending on how the directory is configured.

LDAP `per-user` Authentication

Per-user (self) authentication uses the Kerberos identity (principal) to perform a lookup for each user or each system when authenticating to the directory server. With per-user authentication, the system administrator can use access control instructions (ACI's), access

control lists (ACL's), roles, groups or other directory access control mechanisms to grant or deny access to specific naming service data for specific users or systems.

Note – When configuring per-user mode, the configuration value to enable this mode is "self," which denotes per-user mode.

To use the per-user authentication model, the Kerberos single sign-on service must be deployed. In addition, the one or more directory servers used in the deployment must support SASL and the SASL/GSSAPI authentication mechanism. Because Kerberos expects to use files and DNS for host name lookups, instead of LDAP, DNS should be deployed in this environment. Also, to use per-user authentication, `nscd` must be enabled. The `nscd` daemon is not an optional component in this configuration.

`enableShadowUpdate` Switch

If the `enableShadowUpdate` switch is set to `true` on the client, the admin credentials will be used to update the shadow data. Shadow data is stored in the `shadowAccount` object class on the directory server. Admin credentials are defined by the values of the `adminDN` and `adminPassword` attributes, as described in "Local LDAP Client Attributes" on page 130. These admin credentials are not used for any other purpose.

Admin credentials have properties similar to `Proxy` credentials. The exception is that for admin credentials, the user must have all privileges for the zone or have an effective UID of `root` to read or update the shadow data. Admin credentials can be assigned to any entry that is allowed to bind to the directory. However, do *not* use the same directory manager identity (`cn=Directory Manager`) of the LDAP server.

This entry with admin credentials must have sufficient access to read and write the shadow data in the directory. Because the entry is a shared-per-system resource, the `adminDN` and `adminPassword` attributes must be configured on every client. The encrypted `adminPassword` is stored locally on the client. The password uses the same authentication methods that are configured for the client. The admin credentials are used by all users and processes on a given system to read and update the shadow data.

Credential Storage for LDAP Clients

If you configure a client to use a proxy identity, the client saves proxy information in the `svc:/network/ldap/client` service. The current LDAP implementation does not store proxy credentials in a client's profile. Any proxy credentials that are set by using `ldapclient` during initialization are stored in the SMF repository. This results in improved security surrounding a proxy's DN and password information. See Chapter 12, "Setting Up LDAP Clients (Tasks)" for more information on setting up client profiles.

Similarly, if you configure a client to enable shadow data updates, and the client credential level is not `self`, the client saves its information in the `svc:/network/ldap/client` service.

If you configure a client to use per-user authentication, the Kerberos identity and Kerberos ticket information for each principal (each user or host) are used during authentication. In this environment the directory server maps the Kerberos principal to a DN and the Kerberos credentials are used to authenticate to that DN. The directory server can then use its access control instruction (ACI) mechanisms to allow or deny access to naming service data as necessary. In this situation, Kerberos ticket information is used to authenticate to the directory server and the system does not store authentication DNs or passwords on the system. Therefore, for this type of configuration, you do not need to specify the `adminDN` and `adminPassword` attributes when the client is initialized with the `ldapclient` command.

Choosing Authentication Methods for the LDAP Naming Service

When you assign the `proxy` or `proxy-anonymous` credential level to a client, you also need to select a method by which the proxy authenticates to the directory server. By default, the authentication method is `none`, which implies anonymous access. The authentication method may also have a transport security option associated with it.

The authentication method, like the credential level, may be multi-valued. For example, in the client profile you could specify that the client first tries to bind using the `simple` method secured by TLS. If unsuccessful, the client would try to bind with the `sasl/digest-MD5` method. The `authenticationMethod` would then be `tls:simple;sasl/digest-MD5`.

LDAP naming services support some Simple Authentication and Security Layer (SASL) mechanisms. These mechanisms allow for a secure password exchange without requiring TLS. However, these mechanisms do not provide data integrity or privacy. See RFC 2222 for information on SASL.

The following authentication mechanisms are supported.

- `none`

  The client does not authenticate to the directory. This is equivalent to the `anonymous` credential level.

- `simple`

  If the client system uses the `simple` authentication method, it binds to the server by sending the user's password in the clear. The password is thus subject to snooping unless the session is protected by IPsec. The primary advantages of using the `simple` authentication method are that all directory servers support it and that it is easy to set up.

- `sasl/digest-MD5`

  The client's password is protected during authentication, but the session is not encrypted. Some directory servers, including Oracle Directory Server Enterprise Edition, also support the `sasl/digest-MD5` authentication method. The primary advantage of `digest-MD5` is that

the password does not go over the wire in the clear during authentication and therefore is more secure than the `simple` authentication method. See RFC 2831 for information on `digest-MD5`. `digest-MD5` is considered an improvement over `cram-MD5` for its improved security.

When using `sasl/digest-MD5`, the authentication is secure, but the session is not protected.

Note – If you are using Oracle Directory Server Enterprise Edition, the password *must be stored in the clear* in the directory.

- `sasl/cram-MD5`

  In this case, the LDAP session is not encrypted, but the client's password is protected during authentication, as authentication is performed by using `sasl/cram-MD5`. This authentication method is obsolete and should not be used.

- `sasl/GSSAPI`

  This authentication method is used in conjunction with the self credential mode to enable per-user lookups. A per-user nscd assigned to use the client's credentials binds to the directory server using the sasl/GSSAPI method and the client's Kerberos credentials. Access can be controlled in the directory server on a per-user basis.

- `tls:simple`

  The client binds using the `simple` method and the session is encrypted. The password is protected.

- `tls:sasl/cram-MD5`

  The LDAP session is encrypted and the client authenticates to the directory server using `sasl/cram-MD5`.

- `tls:sasl/digest-MD5`

  The LDAP session is encrypted and the client authenticates to the directory server using `sasl/digest-MD5`.

Caution – Oracle Directory Server Enterprise Edition requires passwords to be stored in the clear in order to use `digest-MD5`. If the authentication method is set to `sasl/digest-MD5` or `tls:sasl/digest-MD5`, then the passwords for the proxy user will need to be stored in the clear. Be especially careful that the `userPassword` attribute has the proper ACIs if it is stored in the clear, so that it is not readable.

The following table summarizes the various authentication methods and their respective characteristics.

TABLE 9–4 Authentication Methods

	Bind	Password on wire	Password on Oracle Directory Server Enterprise Edition	Session
`none`	No	N/A	N/A	No encryption
`simple`	Yes	Clear	Any	No encryption
`sasl/digest-MD5`	Yes	Encryption	Clear	No encryption
`sasl/cram-MD5`	Yes	Encryption	N/A	No encryption
`sasl/GSSAPI`	Yes	Kerberos	Kerberos	Encryption
`tls:simple`	Yes	Encryption	Any	Encryption
`tls:sasl/cram-MD5`	Yes	Encryption	N/A	Encryption
`tls:sasl/digest-MD5`	Yes	Encryption	Clear	Encryption

Specifying Authentication Methods for Specific Services in LDAP

The authentication method can be specified for a given service in the `serviceAuthenticationMethod` attribute. The following services allow for the authentication method to be selected:

- `passwd-cmd`

  This service is used by `passwd(1)` to change the login password and password attributes.

- `keyserv`

  This service is used by the `chkey(1)` and `newkey(1M)` utilities to create and change a user's Diffie-Hellman key pair.

- `pam_ldap`

  This service is used for authenticating users with `pam_ldap(5)`.

  `pam_ldap` supports account management.

Note – If the service does not have a `serviceAuthenticationMethod` set, it will default to the value of the `authenticationMethod` attribute.

Note – In per-user mode, "Kerberos Service Module" on page 142 (pam Kerberos) is used as the authentication service. `ServiceAuthenticationMethod` is not needed in this mode of operation.

Note – If the `enableShadowUpdate` switch is set to `true`, the `ldap_cachemgr` daemon binds to the LDAP server by using the authentication method that is defined in the `serviceAuthenticationMethod` parameter of `passwd-cmd`, if the method is defined. Otherwise, `authenticationMethod` is used. The daemon will not use the `none` authentication method.

The following example shows a section of a client profile in which the users will use sasl/digest-MD5 to authenticate to the directory server, but will use an SSL session to change their password.

```
serviceAuthenticationMethod=pam_ldap:sasl/digest-MD5
serviceAuthenticationMethod=passwd-cmd:tls:simple
```

Pluggable Authentication Methods

By using the PAM framework, you can choose among several authentication services, including the `pam_unix`, `pam_krb5`, and `pam_ldap_*` modules.

If the per-user authentication method is used, `pam_krb5`, the strongest authentication service of the three methods listed above, must be enabled. See `pam_krb5`(5) and the *Oracle Solaris Administration: Security Services*.

The `pam_krb5` authentication system may be used even if per-user authentication is not enabled. If proxy or anonymous credential levels are used to access directory server data then restricting access to directory data on a per-user basis is not possible.

Because of its increased flexibility, support of stronger authentication methods, and ability to use account management, the use of the `pam_ldap` module is recommended over the use of the `pam_unix_*` modules when anonymous or proxy authentication methods are used.

`pam_unix_*` Service Modules

If you have not changed the `/etc/pam.conf` file, UNIX authentication is enabled by default.

Note – The `pam_unix` module has been removed and is no longer supported with the Oracle Solaris release. A set of other service modules provides equivalent or greater functionality. Therefore, in this guide, `pam_unix` refers to the equivalent functionality, not to the `pam_unix` module itself.

Following is a list of the modules that provide the equivalent to the original `pam_unix` module.

`pam_authtok_check`(5)

pam_authtok_get(5)
pam_authtok_store(5)
pam_dhkeys(5)
pam_passwd_auth(5)
pam_unix_account(5)
pam_unix_auth(5)
pam_unix_cred(5)
pam_unix_session(5)

The pam_unix_* modules follows the traditional model of UNIX authentication, as described in the following list.

1. The client retrieves the user's encrypted password from the name service.
2. The user is prompted for the user's password.
3. The user's password is encrypted.
4. The client compares the two encrypted passwords to determine whether the user should be authenticated.

Additionally, there are two restrictions when using the pam_unix_* modules.

- The password must be stored in UNIX crypt format and not in any other encryption methods, including clear.
- The userPassword attribute must be readable by the name service.

  For example, if you set the credential level to anonymous, then anyone must be able to read the userPassword attribute. Similarly, if you set the credential level to proxy, then the proxy user must be able to read the userPassword attribute.

Note – UNIX authentication is not compatible with the sasl authentication method digest-MD5, since Oracle Directory Server Enterprise Edition requires passwords to be stored in the clear in order to use digest-MD5. UNIX authentication requires the password be stored in crypt format.

Note – The pam_unix_account module supports account management when the enableShadowUpdate switch is set to true. The controls for a remote LDAP user account are applied just as the controls are applied to a local user account that is defined in the passwd and shadow files. In enableShadowUpdate mode, for the LDAP account, the system updates and uses the shadow data on the LDAP server for password aging and account locking. Of course, the shadow data of the local account only applies to the local client system, whereas the shadow data of an LDAP user account applies to the user on all client systems.

Password history checking is only supported for the local client, not for an LDAP user account.

Kerberos Service Module

Refer to the pam_krb5(5) man page and *Oracle Solaris Administration: Security Services*.

LDAP Service Module

When implementing LDAP authentication, the user binds to the LDAP server by using the authentication method defined in pam_ldap's serviceAuthenticationMethod parameter, if one exists. Otherwise, authenticationMethod is used.

If pam_ldap is able to bind to the server with the user's identity and supplied password, it authenticates the user.

Note – Previously, if you enabled pam_ldap account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as rsh, rlogin, or ssh would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is 1.3.6.1.4.1.42.2.27.9.5.8, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

---

pam_ldap does not read the userPassword attribute. Therefore, there is no need to grant access to read the userPassword attribute unless there are other clients using UNIX authentication. Also, pam_ldap does not support the none authentication method. Thus, you must define the serviceAuthenticationMethod or the authenticationMethod attributes so clients can use pam_ldap. See the pam_ldap(5) man page for more information.

Caution – If the simple authentication method is used, the userPassword attribute can be read on the wire by third parties.

The following table summarizes the main differences between authentication mechanisms.

TABLE 9–5 Authentication Behavior in LDAP

Event	pam_unix_*	pam_ldap	pam_krb5
Password Sent	Uses passwd service authentication method	Uses passwd service authentication method	Uses Kerberos single sign on technology, not passwords
New Password Sent	Encrypted	No encryption (unless TLS is used)	Uses Kerberos, no passwords are sent over the wire
New Password Stored	crypt format	Password storage scheme defined on Oracle Directory Server Enterprise Edition	Passwords are managed by Kerberos
Requires password read?	Yes	No	No
sasl/digest-MD5 compatibility after changing password	No. Password is not stored in clear. User cannot authenticate.	Yes. As long as default storage scheme is set to clear, user can authenticate.	No. sasl/GSSAPI is used. There are no passwords over the wire and there are no passwords to be stored in the directory server, except when using a Kerberos kdc that manages its password database in the LDAP directory server.
Password policy supported?	Yes. enableShadowUpdate must be set to true.	Yes, if so configured.	See pam_krb5(5), Kerberos V5 Account Management Module.

PAM and Changing Passwords

Use the passwd command to change a password. If the enableShadowUpdate switch is not set to true, the userPassword attribute must be writable by the user. If the enableShadowUpdate switch is set to true, the admin credentials must be able to update the userPassword attribute. Remember that the serviceAuthenticationMethod for passwd-cmd overrides the authenticationMethod for this operation. Depending on the authentication method that is used, the current password might be unencrypted on the wire.

In the case of UNIX authentication, the new userPassword attribute is encrypted using UNIX crypt format and tagged before being written to LDAP. Therefore, the new password is encrypted on the wire, regardless of the authentication method used to bind to the server. See the pam_authtok_store(5) man page for more information.

If the `enableShadowUpdate` switch is set to `true`, the `pam_unix_*` modules also update the related shadow information when the user password is changed. The `pam_unix_*` modules update the same `shadow` fields in the local `shadow` files that the modules update when the local user password is changed.

The `pam_ldap` no longer supports password update. The `pam_authtok_store` with the `server_policy` option now replaces the `pam_ldap` password update capability. When you use `pam_authtok_store`, the new password is sent to the LDAP server in the clear. Therefore, to ensure privacy, use TLS. If TLS is not used, the new `userPassword` is subject to snooping. If you set an untagged password with Oracle Directory Server Enterprise Edition, the software encrypts the password by using the `passwordStorageScheme` attribute. For more information about the `passwordStorageScheme`, see the section on user account management in the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

Note – You need to consider the following configuration issues when setting the `passwordStorageScheme` attribute. If an NIS, or another client using UNIX authentication is using LDAP as a repository, then `passwordStorageScheme` needs to be `crypt`. Also, if using LDAP authentication with `sasl/digest-MD5` with Oracle Directory Server Enterprise Edition, `passwordStorageScheme` must be set to clear.

LDAP Account Management

If you select `pam_krb5` as your account and password management system, the Kerberos environment will manage all your account, password, account lockout, and other account management details. Refer to `pam_krb5(5)` and the *Oracle Solaris Administration: Security Services*.

If you do not use `pam_krb5`, then LDAP naming services can be configured to take advantage of the password and account lockout policy support in Oracle Directory Server Enterprise Edition. You can configure `pam_ldap(5)` to support user account management. `passwd(1)` enforces password syntax rules set by the Oracle Directory Server Enterprise Edition password policy, when used with the proper PAM configuration.

The following account management features are supported through `pam_ldap(5)`. These features depend on Oracle Directory Server Enterprise Edition's password and account lockout policy configuration. You can enable as many or as few of the features as you want.

- Password aging and expiration notification

  Users must change their passwords according to a schedule. A password expires if it is not changed within the time configured. An expired password causes user authentication to fail.

  Users see a warning message whenever they log in within the expiration warning period. The message specifies the number of hours or days until the password expires.

- Password syntax checking

  New passwords must meet the minimum password length requirements. In addition, a password cannot match the value of the `uid`, `cn`, `sn`, or `mail` attributes in the user's directory entry.

- Password in history checking

  Users cannot reuse passwords. If a user attempts to change the password to one that was previously used, `passwd(1)` fails. LDAP administrators can configure the number of passwords kept in the server's history list.

- User account lockout

  A user account can be locked out after a given number of repeated authentication failures. A user can also be locked out if his account is inactivated by an administrator. Authentication will continue to fail until the account lockout time is passed or the administrator reactivates the account.

Note – The preceding account management features only work with the Oracle Directory Server Enterprise Edition. For information about configuring the password and account lockout policy on the server, see the "User Account Management" chapter in the Administration Guide for the version of Oracle Directory Server Enterprise Edition that you are using. Also see "Example `pam_conf` File Using the `pam_ldap` Module for Account Management" on page 196. Do not enable account management for `proxy` accounts.

Before configuring the password and account lockout policy on Oracle Directory Server Enterprise Edition, make sure all hosts use the "newest" LDAP client with `pam_ldap` account management.

In addition, make sure the clients have a properly configured `pam.conf(4)` file. Otherwise, LDAP naming services will not work when `proxy` or user passwords expire.

Note – Previously, if you enabled `pam_ldap` account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as `rsh`, `rlogin`, or `ssh` would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is `1.3.6.1.4.1.42.2.27.9.5.8`, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

LDAP Account Management With the `pam_unix_*` Modules

If the `enableShadowUpdate` switch is set to `true` on the client, account management functionality that is available to local accounts is also available to LDAP accounts. Functionality includes password aging, account expiry and notification, failed login account locking, and so on. Also, the `-dluNfnwx` options to the `passwd` command are now supported in LDAP. Thus, the full functionality of the `passwd` command and the `pam_unix_*` modules in the files naming service is supported in the LDAP naming service. The `enableShadowUpdate` switch provides a way to implement consistent account management for users who are defined in both the files and the LDAP scope.

To prevent users from modifying their own account management data and thereby circumventing password policy, the LDAP server is configured to prevent user write access to the user's own shadow data on the server. An administrator with admin credentials performs the shadow data updates for a client system. Such a configuration, however, conflicts with the `pam_ldap` module, which requires that passwords be modifiable by users. Therefore, account management by the `pam_ldap` and the `pam_unix_*` modules are incompatible.

Caution – Do not use both the `pam_ldap` module and the `pam_unix_*` modules in the same LDAP naming domain. Either all clients use the `pam_ldap` module or all clients use the `pam_unix_*` modules. This limitation might indicate that you need a dedicated LDAP server. For example, a web or email application might expect users to change their own password on the LDAP server.

The implementation of `enableShadowUpdate` also requires that the admin credential (`adminDN` plus `adminPassword`) be stored locally on every client. This information is stored in the `svc:/network/ldap/client` service.

Unlike using `pam_ldap` for account management, using the `pam_unix_*` modules for account management does not require a change to the `/etc/pam.conf` file. The default `/etc/pam.conf` file is sufficient.

◆ ◆ ◆ CHAPTER 10

Planning Requirements for LDAP Naming Services (Tasks)

This chapter discusses the high-level planning you should do before beginning the server and client setup and installation processes.

This chapter covers the following topics.

LDAP Planning Overview

The LDAP client profile is a collection of configuration information an LDAP client uses to access LDAP naming services information about the supporting LDAP server. This chapter discusses the planning of the various aspects of the LDAP naming services. These include the network model, the directory information tree, the security model, the default values of the various profile attributes, and finally, the preparation for data population.

Planning the LDAP Network Model

For availability and performance considerations, each subnet of the company-wide network should have its own LDAP server to service all the LDAP clients in the subnet. Only one of the servers needs to be a master LDAP server. The rest could all be replicas of the master server.

To plan for the network configuration, consider how many servers are available, how a client would be able to get to the servers, and in what order the servers should be accessed. If there is one per subnet, you could use the `defaultServerList` attribute to list all the servers and have

the LDAP client sort and manipulate the access order. If the servers need to be accessed in a certain order due to speed or data management reasons, you should use the `preferredServerList` attribute to define the fixed order of accessing the servers. `defaultServerList` treats all servers in the list equally, while servers in the `preferredServerList` are in priority order, where the first server in the list is the best server to use. The major difference being that when the `preferredServerList` is used available server with the highest priority is used over another available server with a lower priority. In the event that a server with higher priority becomes available, the client machine will disconnect from the server of lower priority. When a `defaultServerList` is used, all servers have equal priority, and one server coming online will not replace an existing server. Both lists may be used in a configuration. Note that you might not want to put the master server on either of these lists to reduce the load on the master server.

In addition, you might find three more attributes worth consideration when planning for the server and network configuration. The `bindTimeLimit` attribute can be used to set the time-out value for a TCP connect request. The `searchTimeLimit` attribute can be used to set the time-out value for an LDAP search operation. The `profileTTL` attribute can be used to control how often the LDAP client should download its profile from the servers. For a slow or unstable network, the `bindTimeLimit` and `searchTimeLimit` attributes might need a larger value than the defaults. For early stage testing of the deployment, you might want to reduce the value of the `profileTTL` attribute to have the clients pick up the frequent changes made to the profile stored in the LDAP servers.

Planning the Directory Information Tree

LDAP naming services have a default directory information tree (DIT) and an associated default schema. For example, the `ou=people` container contains the user account, password, and shadow information. The `ou=hosts` container contains information about systems in the network. Each entry in the `ou=people` container would be of `objectclass` `posixAccount` and `shadowAccount`.

The default DIT is a well-designed directory structure and is based on open standards. For more information, see RFC 2307bis and RFC 4876. The default DIT should be sufficient for most of naming service needs and is recommended for use without changes. If you choose to use the default DIT, you only need to decide is from which node (base DN) in the directory tree the naming services information will be searched for a given domain. This node is specified with the `defaultSearchBase` attribute. Additionally, you might want to set the `defaultSearchScope` attribute to tell the clients the scope of search a naming service lookup should perform. Is it just searching one level under the DN (one), or the entire subtree under the DN (sub)?

There are times, however, that more flexibility is needed for the LDAP naming service to either work with an existing DIT or handle a more complicated DIT with naming service data scattered around the directory tree. For example, user account entries may exist in different part

of the tree. The `serviceSearchDescriptor`, `attributeMap`, and `objectclassMap` attributes in the client profile are designed to handle these situations.

A service search descriptor can be used to override the default search base, search scope, and search filter for a particular service. See "Service Search Descriptors and Schema Mapping" on page 126.

The `attributeMap` and `objectclassMap` attributes provide a way for schema mapping. They make it possible for the LDAP naming services to work with an existing DIT. You can map the `posixAccount` object class to an existing object class, `myAccount`, for example. You can map an attribute in the `posixAccount` object class to an attribute in the `myAccount` object class.

Multiple Directory Servers

Multiple LDAP servers can serve one DIT. For example, some subtrees of the DIT reside on other LDAP servers. In this case, an LDAP server may refer the LDAP client to a different server for the naming data it knows about but is not in its own database. If you plan such a DIT configuration, you should set the clients' profile attribute `followReferrals` to indicate to the LDAP naming service to follow server referrals to continue naming service lookups. However, it is best to have all naming data for a given domain reside on a single directory server, if at all possible.

Referrals can be useful if you want to have clients access read-only replicas most of the time and follow referrals to a read/write master server only when necessary. In this way, the master server does not get overloaded with requests that could be handled by replicas.

Data Sharing With Other Applications

To make best use of LDAP, you should have a single LDAP entry for each logical entry. For example, for a user you can have not only company white-page information, but also account information, and possibly application-specific data. Since `posixAccount` and `shadowAccount` are auxiliary object classes, they can be added to any entry in the directory. This will require careful planning, setup, and administration.

Choosing the Directory Suffix

See the Oracle Directory Server Enterprise Edition documentation for information about how to choose an appropriate directory suffix.

LDAP and Replica Servers

There are three different strategies to employ when setting up replica servers.

- Single-master replication
- Floating-master replication
- Multi-master replication

Single-master

With single-master replication, only one master server for any given partition or non-partitioned network holds writable copies of directory entries. Any replica servers have read-only copies of the directory entries. While both replicas and masters can perform searches, compares, and bind operations, only the master server can perform write operations.

The potential disadvantage to the single-master replication strategy is that the master server is a single point of failure. If the master server goes down, none of the replicas can process write operations.

Floating-master

The floating-master strategy is similar to the single-master strategy in that there is only one master server with write capabilities at any given time for a given partitioned or non-partitioned network. However, when implementing the floating-master strategy, when the master server goes down, a replica is automatically transformed into a master server by way of an algorithm.

One potential disadvantage to the floating-master replication strategy is that if your network becomes partitioned and replicas on either side of the partition become masters, the process of reconciling the new masters can be very complicated if the network is rejoined.

Multi-master

With multi-master replication, there are multiple master servers with their own read-write copies of the directory entry data. While the multi-master strategy eliminates the problem of having a single point of failure, update conflicts can occur between servers. In other words, if an entry's attribute is modified around the same time on two masters, an update conflict resolution policy, such as “last writer wins,” must be in place.

For information about how to set up replica servers, refer to the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using. In general, for large scale enterprise deployments, multi-master replication is the recommended option.

Planning the LDAP Security Model

To plan for the security model, you should first consider what identity the LDAP client should be using to talk to the LDAP server. For example, you must decide if you want an enterprise-wide single sign-on solution, with no passwords being sent over the wire, or the wire encryption of data and the ability to access control data results from the directory server on a per-user basis. You must also decide whether you want strong authentication to protect the user password flow across the wire, and/or if you need to encrypt the session between the LDAP client and the LDAP server to protect the LDAP data transmitted.

The `credentialLevel` and `authenticationMethod` attributes in the profile are used for this. There are four possible credential levels for `credentialLevel`: `anonymous`, `proxy`, `proxy anonymous` and `self`. See "LDAP Naming Services Security Model" on page 132 for a detailed discussion of LDAP naming service security concepts.

Note – Previously, if you enabled `pam_ldap` account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as `rsh`, `rlogin`, or `ssh` would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is `1.3.6.1.4.1.42.2.27.9.5.8`, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

Note – If you enable `pam_krb5` and Kerberos as an enterprise-wide single sign on solution, you can design a system whereby login passwords are only needed once at the start of a session. See *Oracle Solaris Administration: Security Services* for further details. If you enable Kerberos you will generally also need to enable DNS. See the chapters on DNS in this manual for further details.

The main decisions you need to make when planning your security model are the following.

- Will you use Kerberos and per-user authentication?
- What credential level and authentication methods will LDAP clients use?
- Will you use TLS?
- Do you need to be backward compatible with NIS? In other words, will clients use the `pam_unix_*` or `pam_ldap` module?
- What will the servers' `passwordStorageScheme` attribute settings be?
- How will you set up the Access Control Information?

  For more information about ACIs, consult the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.
- Will clients use the `pam_unix_*` or `pam_ldap` module to perform LDAP account management?

Planning Client Profiles and Default Attribute Values for LDAP

By going through the previous planning steps (network model, DIT, and security model), you should have some idea of the values for the following profile attributes.

- `cn`
- `defaultServerList`
- `preferredServerList`
- `bindTimeLimit`
- `searchTimeLimit`
- `profileTTL`
- `defaultSearchBase`
- `defaultSearchScope`
- `serviceSearchDescriptor`
- `attributeMap`
- `objectclassMap`
- `followReferrals`
- `credentialLevel`
- `authenticationMethod`
- `serviceCredentialLevel`
- `serviceAuthenticationMethod`

Of the preceding attributes, only `cn`, `defaultServerList`, and `defaultSearchBase` are required. They have no default values. The rest are optional, and some have default values.

See Chapter 12, "Setting Up LDAP Clients (Tasks)," for more information about setting up LDAP clients.

Planning the LDAP Data Population

To populate the LDAP server with data, after the LDAP server has been configured with the proper DIT and schema. Use the new `ldapaddent` tool. This tool will create entries in LDAP containers from their corresponding `/etc` files. It can be used to populate data into the containers for the following types of data: `aliases`, `auto_*`, `bootparams`, `ethers`, `group`, `hosts` (including IPv6 addresses), `netgroup`, `netmasks`, `networks`, `passwd`, `shadow`, `protocols`, `publickey`, `rpc`, and `services`. Also, the RBAC-related files can be added: `/etc/user_attr`, `/etc/security/auth_attr`, `/etc/security/prof_attr`, and `/etc/security/exec_attr`.

By default, `ldapaddent` reads from the standard input and adds this data to the LDAP container associated with the database specified on the command line. But an input file from which data should be read can be specified using the `-f` option.

Because the entries are stored in the directory based on the client's configuration, the client must be configured to use the LDAP naming services.

For better performance, load the databases in this order:

1. `passwd` database followed by `shadow` database
2. `networks` database followed by `netmasks` database
3. `bootparams` database followed by `ethers` database

Note that when adding automounter entries, the database name is in the form of `auto_*` (for example, `auto_home`).

If you have `/etc` files from different hosts to add to the LDAP server, you can either merge all of them into the same `/etc` file and then use the `ldapaddent` command on one host to add the files, or run the `ldapaddent` command on the different hosts one by one, with the expectation that each host is already configured as an LDAP client.

If your naming service data is already in an NIS server, and you want to move the data to the LDAP server for LDAP naming services, use the `ypcat` command to dump the NIS map into files. Then, run the `ldapaddent` command against these files to add the data to the LDAP server.

The following procedure assumes that the tables are to be extracted from a `yp` client.

▼ How to Populate a Server With `host` Entries by Using the `ldapaddent` Command

1 Make sure that Oracle Directory Server Enterprise Edition was set up by using the `idsconfig` command.

2. **On a client machine, become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see Chapter 9, "Using Role-Based Access Control (Tasks)," in *Oracle Solaris Administration: Security Services*.

3. **Make the machine an LDAP client.**

   ```
   # ldapclient init -a profileName=new -a domainName=west.example.com 192.168.0.1
   ```

4. **Populate the server with data.**

   ```
   # ldapaddent -D "cn=directory manager" -f /etc/hosts hosts
   ```

   You will be prompted for a password.

   In this example, the `ldapaddent` command will use the authentication method that has been configured in the profile new. Selecting simple will cause the password to be sent in the clear. For more information, refer to the `ldapaddent(1M)` man page.

   In stand-alone mode, the command should be appear similar to the following:

```
# ldapaddent -h 192.168.0.1 -N new -M west.example.com -a simple-D "cn=directory manager" -f /etc/hosts hosts
```

◆ ◆ ◆ CHAPTER 11

Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)

This chapter describes how to configure Oracle Directory Server Enterprise Edition to support a network of LDAP naming services clients. The information is specific to the Oracle Directory Server Enterprise Edition. For information about installing and configuring the directory server, see the Oracle Directory Server Enterprise Edition documentation.

Note – You must have already performed all the procedures described in the installation and configuration documentation that shipped with your Oracle Directory Server Enterprise Edition before you can configure Oracle Directory Server Enterprise Edition to work with LDAP clients.

Note – A directory server (an LDAP server) *cannot* be its own client.

This chapter covers the following topics.

Configuring Oracle Directory Server Enterprise Edition by Using the idsconfig Command

Creating a Checklist Based on Your Server Installation

During the server installation process, you will have defined crucial variables, with which you should create a checklist similar to the one below before launching idsconfig. You can use the blank checklist provided in "Blank Checklists for Configuring LDAP " on page 193.

Note – The information included below will serve as the basis for all examples that follow in the LDAP-related chapters. The example domain is of a widget company, Example, Inc. with stores nationwide. The examples deal with the West Coast Division, with the domain name of west.example.com.

TABLE 11–1 Server Variables Defined for the example.com Network

Variable	Definition for Example Network
Port number at which an instance of the directory server is installed	389 (default)
Name of server	myserver (from the FQDN myserver.west.example.com or the hos tname for 192.168.0.1)
Replica servers (IPnumber:port number)	192.168.0.2 [for myreplica.west.example.com]
Directory manager	cn=directory manager (default)
Domain name to be served	west.example.com
Maximum time (in seconds) to process client requests before timing out	1
Maximum number of entries returned for each search request	1

Note – If you are using host names in defining defaultServerList or preferredServerList, you *must* ensure that LDAP is not used for host lookups. This means that ldap must not be configured in the config/host property of the svc:/network/name-service/switch service.

TABLE 11–2 Client Profile Variables Defined for the example.com Network

Variable	Definition for Example Network
Profile name (the default name is default)	WestUserProfile
Server list (defaults to the local subnet)	192.168.0.1
Preferred server list (listed in order of which server to try first, second, and so on)	none
Search scope (number of levels down through the directory tree. 'One', the default, or 'Sub')	one (default)
Credential used to gain access to server. Default is anonymous	proxy
Follow Referrals? (a pointer to another server if the main server is unavailable) Default is no.	Y
Search time limit (default is 30 seconds) for waiting for server to return information.	default
Bind time limit (default is 10 seconds) for contacting the server.	default
Authentication method Default is none.	simple

Note – Client profiles are defined per domain. At least one profile must be defined for a given domain.

Attribute Indexes

The idsconfig command indexes the following list of attributes for improved performance:

```
membernisnetgroup    pres,eq,sub
nisnetgrouptriple    pres,eq,sub
ipHostNumber         pres,eq,sub
uidNumber            pres,eq
gidNumber            pres,eq
ipNetworkNumber      pres,eq
automountkey         pres,eq
oncRpcNumber         pres,eq
```

Schema Definitions

`idsconfig(1M)` automatically adds the necessary schema definitions. Unless you are very experienced in LDAP administration, do not manually modify the server schema. See Chapter 14, "LDAP Naming Service (Reference)," for an extended list of schemas used by the LDAP naming service.

Using Browsing Indexes

The browsing index functionality of the Oracle Directory Server Enterprise Edition, otherwise known as the virtual list view (VLV), provides a way in which a client can view a select group or number of entries from very long list, thus making the search process less time consuming for each client. Browsing indexes provide optimized, predefined search parameters with which the LDAP naming client can access specific information from the various services more quickly. Keep in mind that if you do not create browsing indexes, the clients will not access all the entries of a given type if the server limits are exceeded. For example, if there are 5000 password entries, but the size limit of 1000 entries is enabled, 4000 entries will not be returned during some lookup operations. That can often cause login and other serious failures for the client machines.

VLV indexes are configured on the directory server and the proxy user has read access to these indexes.

Before configuring browsing indexes on the Oracle Directory Server Enterprise Edition, consider the performance cost associated with using these indexes. For more information, refer to the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

`idsconfig` creates entries for several VLV indexes. See the `idsconfig(1M)` man page for more information. Refer to the output of the `idsconfig` command to determine the VLV entries created by `idsconfig`. See "Example `idsconfig` Setup" on page 163 for sample `idsconfig` output.

Using Service Search Descriptors to Modify Client Access to Various Services

A service search descriptor (SSD) changes the default search request for a given operation in LDAP to a search you define. SSDs are particularly useful if, for example, you have been using LDAP with customized container definitions or another operating system and are now transitional to the latest Oracle Solaris release. Using SSDs, you can configure LDAP naming services without having to change your existing LDAP database and data.

Setting Up SSDs by Using the `idsconfig` Command

Assume your predecessor at Example, Inc. had configured LDAP, storing users in `ou=Users` container. You are now upgrading to the latest Oracle Solaris release. By definition, an LDAP client assumes that user entries are stored in `ou=People` container. Thus, when it comes to searching the `passwd` service, LDAP client will search the `ou=people` level of the DIT and not find the correct values.

One laborious solution to the above problem would be to completely overwrite Example, Inc.'s existing DIT and to rewrite all the exiting applications on Example, Inc.'s network so that they are compatible with the new LDAP naming service. A second, far preferable solution would be to use an SSD that would tell LDAP client to look for user info in an `ou=Users` container instead the default `ou=people` container.

You would define the necessary SSD during the configuration of the Oracle Directory Server Enterprise Edition using `idsconfig`. The prompt line appears as follows.

```
Do you wish to setup Service Search Descriptors (y/n/h? y
  A  Add a Service Search Descriptor
  D  Delete a SSD
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's

  Q  Exit menu
Enter menu choice: [Quit] a
Enter the service id: passwd
Enter the base: service ou=user,dc=west,dc=example,dc=com
Enter the scope: one[default]
  A  Add a Service Search Descriptor
  D  Delete a SSD
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's

  Q  Exit menu
Enter menu choice: [Quit] p

Current Service Search Descriptors:
==================================
Passwd:ou=Users,ou=west,ou=example,ou=com?

Hit return to continue.

  A  Add a Service Search Descriptor
  D  Delete a SSD
  M  Modify a SSD
  P  Display all SSD's
  H  Help
  X  Clear all SSD's
```

```
  Q  Exit menu
Enter menu choice: [Quit] q
```

Running the idsconfig Command

Note – You do not need special rights to run `idsconfig`, nor do you need to be an LDAP naming client. Remember to create a checklist as mentioned in "Creating a Checklist Based on Your Server Installation" on page 158 in preparation for running `idsconfig`. You do not have to run `idsconfig` from a server or an LDAP naming service client machine. You can run `idsconfig` from any Oracle Solaris machine on the network.

Caution – `idsconfig` sends the Directory Manager's password in the clear. If you do not want this to happen, you must run `idsconfig` on the directory server itself, not on a client.

▼ How to Configure Oracle Directory Server Enterprise Edition by Using the idsconfig Command

1 **Make sure the target Oracle Directory Server Enterprise Edition is up and running.**

2 **Run the `idsconfig` command.**

```
# /usr/lib/ldap/idsconfig
```

Refer to Example 11–1 for an example run of `idsconfig` using the definitions listed in the server and client checklists at the beginning of this chapter in "Creating a Checklist Based on Your Server Installation" on page 158.

3 **Answer the questions when prompted.**

Note that 'no' [n] is the default user input. If you need clarification on any given question, type

```
h
```

and a brief help paragraph will appear.

After `idsconfig` has completed the setup of the directory, you need to run the specified commands on the server before the server setup is complete and the server is ready to serve clients.

Example idsconfig Setup

This section provides an example of a basic idsconfig setup that uses many of the defaults. The most complicated method of modifying client profiles is to create SSDs. Refer to "Using Service Search Descriptors to Modify Client Access to Various Services" on page 160 for a detailed discussion.

The data in square brackets after a prompt indicates the default value for that prompt. To accept the default value, press Return.

Note – Any parameters that are left blank in the summary screen are not set up.

After idsconfig has completed the setup of the directory, you need to run the specified commands on the server before the server setup is complete and the server is ready to serve clients.

EXAMPLE 11–1 Running the idsconfig command for the Example, Inc. Network

In the following example, the idsconfig utility is run immediately after a server instance is created on the LDAP server.

```
# usr/lib/ldap/idsconfig
It is strongly recommended that you BACKUP the directory server
before running idsconfig.

Hit Ctrl-C at any time before the final confirmation to exit.

Do you wish to continue with server setup (y/n/h)? [n] y
Enter the JES Directory Server's  hostname to setup: myserver
Enter the port number for DSEE (h=help): [389]
Enter the directory manager DN: [cn=Directory Manager]
Enter passwd for cn=Directory Manager :
Enter the domainname to be served (h=help): [west.example.com]
Enter LDAP Base DN (h=help): [dc=west,dc=example,dc=com]
  Checking LDAP Base DN ...
  Validating LDAP Base DN and Suffix ...
  No valid suffixes were found for Base DN dc=west,dc=example,dc=com
Enter suffix to be created (b=back/h=help): [dc=west,dc=example,dc=com]
Enter ldbm database name (b=back/h=help): [west]
  sasl/GSSAPI is not supported by this LDAP server
Enter the profile name (h=help): [default] WestUserProfile
Default server list (h=help): [192.168.0.1]
Preferred server list (h=help):
Choose desired search scope (one, sub, h=help):  [one]
The following are the supported credential levels:
  1  anonymous
  2  proxy
  3  proxy anonymous
  4  self
Choose Credential level [h=help]: [1] 2
```

EXAMPLE 11–1 Running the idsconfig command for the Example, Inc. Network *(Continued)*

```
The following are the supported Authentication Methods:
  1  none
  2  simple
  3  sasl/DIGEST-MD5
  4  tls:simple
  5  tls:sasl/DIGEST-MD5
  6  sasl/GSSAPI
Choose Authentication Method (h=help): [1] 2


Current authenticationMethod: simple
Do you want to add another Authentication Method? n
Do you want the clients to follow referrals (y/n/h)? [n]
Do you want to modify the server timelimit value (y/n/h)? [n] y
Enter the time limit for DSEE (current=3600): [-1]
Do you want to modify the server sizelimit value (y/n/h)? [n] y
Enter the size limit for DSEE (current=2000): [-1]
Do you want to store passwords in "crypt" format (y/n/h)? [n] y
Do you want to setup a Service Authentication Methods (y/n/h)? [n]
Client search time limit in seconds (h=help): [30]
Profile Time To Live in seconds (h=help): [43200]
Bind time limit in seconds (h=help): [10]
Do you want to enable shadow update (y/n/h)? [n]
Do you wish to setup Service Search Descriptors (y/n/h)? [n]

             Summary of Configuration

  1  Domain to serve               : west.example.com
  2  Base DN to setup              : dc=west,dc=example,dc=com
       Suffix to create            : dc=west,dc=example,dc=com
       Database to create          : west
  3  Profile name to create        : WestUserProfile
  4  Default Server List           : 192.168.0.1
  5  Preferred Server List         :
  6  Default Search Scope          : one
  7  Credential Level              : proxy
  8  Authentication Method         : simple
  9  Enable Follow Referrals       : FALSE
 10  DSEE Time Limit                : -1
 11  DSEE Size Limit                : -1
 12  Enable crypt password storage : TRUE
 13  Service Auth Method pam_ldap  :
 14  Service Auth Method keyserv   :
 15  Service Auth Method passwd-cmd:
 16  Search Time Limit             : 30
 17  Profile Time to Live          : 43200
 18  Bind Limit                    : 10
 19  Enable shadow update          : FALSE
 20  Service Search Descriptors Menu

Enter config value to change: (1-20 0=commit changes) [0]
Enter DN for proxy agent: [cn=proxyagent,ou=profile,dc=west,dc=example,dc=com]
Enter passwd for proxyagent:
Re-enter passwd:
```

EXAMPLE 11–1 Running the idsconfig command for the Example, Inc. Network *(Continued)*

```
WARNING: About to start committing changes. (y=continue, n=EXIT) y

  1. Changed timelimit to -1 in cn=config.
  2. Changed sizelimit to -1 in cn=config.
  3. Changed passwordstoragescheme to "crypt" in cn=config.
  4. Schema attributes have been updated.
  5. Schema objectclass definitions have been added.
  6. Database west successfully created.
  7. Suffix dc=west,dc=example,dc=com successfully created.
  8. NisDomainObject added to dc=west,dc=example,dc=com.
  9. Top level "ou" containers complete.
  10. automount maps: auto_home auto_direct auto_master auto_shared processed.
  11. ACI for dc=west,dc=example,dc=com modified to disable self modify.
  12. Add of VLV Access Control Information (ACI).
  13. Proxy Agent cn=proxyagent,ou=profile,dc=west,dc=example,dc=com added.
  14. Give cn=proxyagent,ou=profile,dc=west,dc=example,dc=com read permission
      for password.
  15. Generated client profile and loaded on server.
  16. Processing eq,pres indexes:
      uidNumber (eq,pres)   Finished indexing.
      ipNetworkNumber (eq,pres)   Finished indexing.
      gidnumber (eq,pres)   Finished indexing.
      oncrpcnumber (eq,pres)   Finished indexing.
      automountKey (eq,pres)   Finished indexing.
  17. Processing eq,pres,sub indexes:
      ipHostNumber (eq,pres,sub)   Finished indexing.
      membernisnetgroup (eq,pres,sub)   Finished indexing.
      nisnetgrouptriple (eq,pres,sub)   Finished indexing.
  18. Processing VLV indexes:
      west.example.com.getgrent vlv_index   Entry created
      west.example.com.gethostent vlv_index   Entry created
      west.example.com.getnetent vlv_index   Entry created
      west.example.com.getpwent vlv_index   Entry created
      west.example.com.getrpcent vlv_index   Entry created
      west.example.com.getspent vlv_index   Entry created
      west.example.com.getauhoent vlv_index   Entry created
      west.example.com.getsoluent vlv_index   Entry created
      west.example.com.getauduent vlv_index   Entry created
      west.example.com.getauthent vlv_index   Entry created
      west.example.com.getexecent vlv_index   Entry created
      west.example.com.getprofent vlv_index   Entry created
      west.example.com.getmailent vlv_index   Entry created
      west.example.com.getbootent vlv_index   Entry created
      west.example.com.getethent vlv_index   Entry created
      west.example.com.getngrpent vlv_index   Entry created
      west.example.com.getipnent vlv_index   Entry created
      west.example.com.getmaskent vlv_index   Entry created
      west.example.com.getprent vlv_index   Entry created
      west.example.com.getip4ent vlv_index   Entry created
      west.example.com.getip6ent vlv_index   Entry created

idsconfig: Setup of DSEE server myserver is complete.
```

EXAMPLE 11–1 Running the idsconfig command for the Example, Inc. Network *(Continued)*

```
Note: idsconfig has created entries for VLV indexes.

      For DS5.x, use the directoryserver(1m) script on myserver
      to stop the server.  Then, using directoryserver, follow the
      directoryserver examples below to create the actual VLV indexes.

      For DSEE6.x, use dsadm command delivered with DS on myserver
      to stop the server.  Then, using dsadm, follow the
      dsadm examples below to create the actual VLV indexes.

  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getgrent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.gethostent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getnetent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getpwent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getrpcent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getspent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getauhoent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getsoluent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getauduent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getauthent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getexecent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getprofent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getmailent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getbootent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getethent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getngrpent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getipnent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getmaskent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getprent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getip4ent
  directoryserver -s <server-instance> vlvindex -n west -T west.example.com.getip6ent

<install-path>/bin/dsadm reindex -l -t west.example.com.getgrent <directory-instance-path>
 dc=west,dc=example,dc=com
<install-path>/bin/dsadm reindex -l -t west.example.com.gethostent <directory-instance-path>
 dc=west,dc=example,dc=com
.
.
.
<install-path>/bin/dsadm reindex -l -t west.example.com.getip6ent <directory-instance-path>
 dc=west,dc=example,dc=com
```

Populating the Directory Server by Using the `ldapaddent` Command

Note – Before populating the directory server with data, you must configure the server to store passwords in UNIX `Crypt` format if you are using the `pam_unix_*` modules. If you are using `pam_ldap`, you can store passwords in any format. For more information about setting the password in UNIX `crypt` format, see the Oracle Directory Server Enterprise Edition documents.

`ldapaddent` reads from the standard input (that being an `/etc/filename` like `passwd`) and places this data to the container associated with the service. Client configuration determines how the data will be written by default.

▼ How to Populate Oracle Directory Server Enterprise Edition With User Password Data by Using the `ldapaddent` Command

- **Use the `ldapaddent` command to add `/etc/passwd` entries to the server.**

  ```
  # ldapaddent -D "cn=directory manager" -f /etc/passwd passwd
  ```

  See the `ldapaddent`(1M) man page. Also see Chapter 9, "Introduction to LDAP Naming Services (Overview)," for information about LDAP security and write-access to the directory server.

Specifying Group Memberships by Using the Member Attribute

The Internet-Draft rfc2307bis specifies that the `groupOfMembers` object class can also be used as the convenient structural class for the LDAP entries of the group service. Such group entries can then have member attribute values specifying group membership in Distinguished Names (DNs). Oracle Solaris LDAP clients support such group entries and use the member attribute values for group membership resolution.

The LDAP clients also support group entries that use the `groupOfUniqueNames` object class and the `uniqueMember` attribute. However, using this object class and attribute is not recommended.

The existing way of defining the group entries with the `posixGroup` object class and the `memberUid` attribute is still supported. This type of group entries are still what the `ldapaddent` command creates when populating the LDAP servers for the group services. It does not add the `member` attribute to the group entries.

To add group entries with the `groupOfMembers` object class and `member` attribute values, use the `ldapadd` tool and an input file similar to the following:

```
dn: cn=group1,ou=group,dc=mkg,dc=example,dc=com
objectClass: posixGroup
objectClass: groupOfNames
objectClass: top
cn: group1
gidNumber: 1234
member: uid=user1,ou=people,dc=mkg,dc=example,dc=com
member: uid=user2,ou=people,dc=mkg,dc=example,dc=com
member: cn=group2,ou=group,dc=mkg,dc=example,dc=com
```

LDAP clients will handle group entries with a mix of none, any or all of the `memberUid`, `member`, and `uniqueMember` attributes. The membership evaluation result will be that a group has membership that is the union of all three with duplicates removed. That is, if a group entry `G` has a `memberUid` value referring to user `U1` and `U2`, a `member` value referring to user `U2`, and a `uniqueMember` value referring to user `U3`, then group `G` has three members, `U1`, `U2`, and `U3`. Nested groups are also supported, that is, a member attribute can have values pointing to other groups.

To efficiently evaluate group membership to determine the groups (including the nested ones) that a user is a member of, the `memberOf` plug-in must be configured and enabled on the LDAP servers. If not, only the containing groups, not nested ones, will be resolved. By default, the `memberOf` plug-in is enabled by the ODSEE server. If the plug-in is not enabled, use ODSEE's `dsconf` tool to enable it.

Populating the Directory Server With Additional Profiles

Use the `ldapclient` command with the `genprofile` option to create an LDIF representation of a configuration profile, based on the attributes specified. The profile you create can then be loaded into an LDAP server to be used as the client profile. The client profile can be downloaded by the client by using `ldapclient init`.

Refer to `ldapclient(1M)` for information about using `ldapclient genprofile`.

▼ How to Populate the Directory Server With Additional Profiles by Using the `ldapclient` Command

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Use `ldapclient` with the `genprofile` command.**

```
# ldapclient genprofile \
-a profileName=myprofile \
-a defaultSearchBase=dc=west,dc=example,dc=com \
-a "defaultServerList=192.168.0.1 192.168.0.2:386" \> myprofile.ldif
```

3. **Upload the new profile to the server.**

```
# ldapadd -h 192.168.0.1 -D "cn=directory manager" -f myprofile.ldif
```

Configuring the Directory Server to Enable Account Management

Account management can be implemented for clients that use `pam_ldap` and for clients that use the `pam_unix_*` modules.

Caution – Do not use both the `pam_ldap` and `pam_unix_*` modules in the same LDAP naming domain. Either all clients use `pam_ldap` or all clients use the `pam_unix_*` modules. This limitation might indicate that you need a dedicated LDAP server.

For Clients That Use the `pam_ldap` Module

In order for `pam_ldap` to work properly, the password and account lockout policy must be properly configured on the server. You can use the Directory Server Console or `ldapmodify` to configure the account management policy for the LDAP directory. For procedures and more information, see the "User Account Management" chapter in the *Administration Guide* for the version of Oracle Directory Server Enterprise Edition that you are using.

Note – Previously, if you enabled `pam_ldap` account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as `rsh`, `rlogin`, or `ssh` would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is `1.3.6.1.4.1.42.2.27.9.5.8`, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

Passwords for `proxy` users should *never* be allowed to expire. If proxy passwords expire, clients using the `proxy` credential level cannot retrieve naming service information from the server. To ensure that proxy users have passwords that do not expire, modify the proxy accounts with the following script.

```
# ldapmodify -h ldapserver -D administrator DN \
-w administrator password <<EOF
dn: proxy user DN
DNchangetype: modify
replace: passwordexpirationtime
passwordexpirationtime: 20380119031407Z
EOF
```

Note – `pam_ldap` account management relies on Oracle Directory Server Enterprise Edition to maintain and provide password aging and account expiration information for users. The directory server does not interpret the corresponding data from shadow entries to validate user accounts. The `pam_unix_*` modules, however, examines the shadow data to determine if accounts are locked or if passwords are aged. Since the shadow data is not kept up to date by the LDAP naming services or the directory server, the modules should not grant access based on the shadow data. The shadow data is retrieved using the `proxy` identity. Therefore, do not allow `proxy` users to have read access to the `userPassword` attribute. Denying `proxy` users read access to `userPassword` prevents the PAM service from making an invalid account validation.

For Clients That Use the `pam_unix_*` Modules

To enable LDAP clients to use the `pam_unix_*` modules for account management, the server must be set up to enable the updating of shadow data. Unlike `pam_ldap` account management, the `pam_unix_*` modules do not require extra configuration steps. All configuration can be performed by running the `idsconfig` utility. For a basic `idsconfig` run, see Example 11–1.

The following shows the output of two `idsconfig` runs.

The first `idsconfig` run uses an existing client profile.

```
# /usr/lib/ldap/idsconfig

It is strongly recommended that you BACKUP the directory server
before running idsconfig.

Hit Ctrl-C at any time before the final confirmation to exit.

Do you wish to continue with server setup (y/n/h)? [n] y
Enter the JES Directory Server's  hostname to setup: myserver
Enter the port number for DSEE (h=help): [389]
Enter the directory manager DN: [cn=Directory Manager]
Enter passwd for cn=Directory Manager :
Enter the domainname to be served (h=help): [west.example.com]
Enter LDAP Base DN (h=help): [dc=west,dc=example,dc=com]
  Checking LDAP Base DN ...
  Validating LDAP Base DN and Suffix ...
  sasl/GSSAPI is not supported by this LDAP server

Enter the profile name (h=help): [default] WestUserProfile

Profile 'WestUserProfile' already exists, it is possible to enable
shadow update now. idsconfig will exit after shadow update
is enabled. You can also continue to overwrite the profile
or create a new one and be given the chance to enable
shadow update later.


Just enable shadow update (y/n/h)? [n] y
Add the administrator identity (y/n/h)? [y]
Enter DN for the administrator: [cn=admin,ou=profile,dc=west,dc=example,dc=com]
Enter passwd for the administrator:
Re-enter passwd:
  ADDED: Administrator identity cn=admin,ou=profile,dc=west,dc=example,dc=com.
         Proxy ACI LDAP_Naming_Services_proxy_password_read does not
         exist for dc=west,dc=example,dc=com.
  ACI SET: Give cn=admin,ou=profile,dc=west,dc=example,dc=com read/write access
           to shadow data.
  ACI SET: Non-Admin access to shadow data denied.

  Shadow update has been enabled.
```

The second `idsconfig` run creates a new profile for later use. Only partial output is displayed.

```
# /usr/lib/ldap/idsconfig

It is strongly recommended that you BACKUP the directory server
before running idsconfig.

Hit Ctrl-C at any time before the final confirmation to exit.

Do you wish to continue with server setup (y/n/h)? [n] y
Enter the JES Directory Server's  hostname to setup: myserver
Enter the port number for DSEE (h=help): [389]
Enter the directory manager DN: [cn=Directory Manager]
Enter passwd for cn=Directory Manager :
Enter the domainname to be served (h=help): [west.example.com]
Enter LDAP Base DN (h=help): [dc=west,dc=example,dc=com]
  Checking LDAP Base DN ...
  Validating LDAP Base DN and Suffix ...
  sasl/GSSAPI is not supported by this LDAP server

Enter the profile name (h=help): [default] WestUserProfile-new
Default server list (h=help): [192.168.0.1]
.
.
.
Do you want to enable shadow update (y/n/h)? [n] y

              Summary of Configuration

  1  Domain to serve              : west.example.com
  2  Base DN to setup             : dc=west,dc=example,dc=com
        Suffix to create          : dc=west,dc=example,dc=com
  3  Profile name to create       : WestUserProfile-new
.
.
.
 19  Enable shadow update         : TRUE
.
.
.
Enter DN for the administrator: [cn=admin,ou=profile,dc=west,dc=example,dc=com]
Enter passwd for the administrator:
Re-enter passwd:


WARNING: About to start committing changes. (y=continue, n=EXIT) y

  1. Changed timelimit to -1 in cn=config.
  2. Changed sizelimit to -1 in cn=config.
.
.
.
  11. ACI for dc=test1,dc=mpklab,dc=sfbay,dc=sun,dc=com modified to
      disable self modify.
.
.
.
  15. Give cn=admin,ou=profile,dc=west,dc=example,dc=com write permission for shadow.
...
```

◆ ◆ ◆ CHAPTER 12

Setting Up LDAP Clients (Tasks)

This chapter describes how to set up an LDAP naming services client. This chapter covers the following topics:

Prerequisites to LDAP Client Setup

In order for an Oracle Solaris client to use LDAP as a naming service, the following requirements must be met:

- The client's domain name must be served by the LDAP server.
- The name service switch must point to LDAP for the required services.
- The client must be configured with all the given parameters that define its behavior.
- `ldap_cachemgr` must be running on the client.
- At least one server for which a client is configured must be up and running.

The `ldapclient` utility is the key to setting up an LDAP client, as it performs all of the above steps, except for starting the server. The rest of this chapter will show examples of how to use the `ldapclient` utility to set up an LDAP client and use the various other LDAP utilities to get information about, and check the status of, an LDAP client.

LDAP and the Service Management Facility

The LDAP client service is managed by using the Service Management Facility. For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) and `svcs`(1) man pages for more details.

The following list provides a short overview of some of the important information needed to use the SMF service to administer the LDAP client service.

- Administrative actions on the LDAP client service, such as enabling, disabling, or restarting, can be performed by using the `svcadm` command.

  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 Fault Management Resource Identifier (FMRI) for the LDAP client service is `svc:/network/ldap/client`.
- During the configuration process, the `network/nis/domain` service will also be enabled to supply the domain name that is used by the `network/ldap/client` service.
- You can query the status of the LDAP client and the `ldap_cachemgr` daemon by using the `svcs` command.
  - The following are examples of the `svcs` command and its output:

    ```
    # svcs \*ldap\*
    STATE          STIME    FMRI
    online         15:43:46  svc:/network/ldap/client:default
    ```

  - Example of `svcs -l` command and output. To get the output shown below, you must use the instance name in the FMRI.

    ```
    # svcs -l network/ldap/client:default
    fmri         svc:/network/ldap/client:default
    name         LDAP Name Service Client
    enabled      true
    state        online
    next_state   none
    restarter    svc:/system/svc/restarter:default
    manifest     /lib/svc/manifest/network/ldap/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/none svc:/network/initial (online)
    dependency   optional_all/none svc:/network/location:default (online)
    dependency   require_all/restart svc:/network/nis/domain (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 check for a daemon's presence by using the following commands:.
  - On a server, use the `ptree` command:

    ```
    # ptree 'pgrep slapd'
    6410 zsched
    11565 /export/dsee/dsee6/ds6/lib/64/ns-slapd -D /export/dsee/test1 -i /export
    ```

  - On a client, use the following command:

    ```
    # ldapsearch -h server-name -b "" -s base "objectclass=*" |grep -i context
    namingContexts: dc=example,dc=com
    ```

Initializing an LDAP Client

The `ldapclient` command is used to set up LDAP clients on an Oracle Solaris system. The command assumes that the server has already been configured with the appropriate client profiles. You must install and configure the server with the appropriate profiles before you can set up clients.

Note – Because LDAP and NIS use the same domain name component that is defined in the `network/nis/domain` service, 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.

There are two main ways to set up a client by using `ldapclient`.

- *Profile*

  At a minimum, you need to specify the server address containing the profile and domain you want to use. If no profile is specified, then the "default" profile is assumed. The server will provide the rest of the required information, except for proxy and certificate database information. If a client's credential level is `proxy` or `proxy anonymous`, you must supply the proxy bind DN and password. See "Assigning Client Credential Levels" on page 134 for more information.

  To enable shadow data update, you must provide the admin credential (`adminDN` plus `adminPassword`).

- *Manual*

  You configure the profile on the client itself, which means that you define all parameters from the command line. Thus, the profile information is stored in cache files and is never refreshed by the server.

Note – In enterprise environments, using an LDAP configuration profile can reduce complexity if the profile is shared across machines.

How to Initialize an LDAP Client by Using Profiles

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Run the `ldapclient` command with the `init` option.**

```
# ldapclient init -a profileName=new \
-a domainName=west.example.com 192.168.0.1
System successfully configured
```

How to Initialize an LDAP Client by Using Per-User Credentials

Before You Begin

Before you set up an LDAP client with per-user credentials, the following must already be configured:

- One or more Kerberos key distribution center (KDC) servers must be configured and running.
- DNS, client access to a DNS server, and at least one DNS server must be configured and running.
- Kerberos on the client machine must be configured and enabled.
- A Kerberos client installation profile such as the following must exist:

```
# cat /usr/tmp/krb5.profile
REALM SPARKS.COM
KDC kdc.example.com
ADMIN super/admin
FILEPATH /usr/tmp/krb5.conf
NFS 1
DNSLOOKUP none
```

- The LDAP server must be installed and configured to support `sasl/GSSAPI`.
- Appropriate identity mapping configurations must exist.
- Kerberos host principals for the directory server and the KDC must be set up in the KDC.
- The `idsconfig` command must have been run on the directory server DIT to be used.
- An appropriate per-user `gssapi` profile (such as `gssapi_EXAMPLE.COM`) must have been created.

An illustration of a per-user profile in the `idsconfig` command is shown in the following partial example:

```
# /usr/lib/ldap/idsconfig
Do you wish to continue with server setup (y/n/h)? [n] y
Enter the Directory Server's hostname to setup: kdc.example.com
Enter the port number for DSEE (h=help): [389] <Enter your port>
Enter the directory manager DN: [cn=Directory Manager] <Enter your DN>
Enter passwd for cn=Directory Manager : <Enter your password>
Enter the domainname to be served (h=help): [example.com] <Enter your domain>
Enter LDAP Base DN (h=help): [dc=example,dc=com] <Enter your DN>
GSSAPI is supported. Do you want to set up gssapi:(y/n) [n] y
Enter Kerberos Realm: [EXAMPLE.COM] EXAMPLE.COM
```

Note – In addition, for a `gssapi` profile, you must supply a credential level of `4 self` and the authentication method of `6 sasl/GSSAPI`.

- The necessary user principals must exist in the KDC.
- On the client machine, Kerberos must be initialized by using the client profile with a command such as the following:

  ```
  # /usr/sbin/kclient -p /usr/tmp/krb5.profile
  ```
- The name service switch must be configured to use `dns` for `hosts`. The following command checks the current repository values:

  ```
  % svcprop -p config/host system/name-service/switch
  files\ dns\ nis
  ```
- DNS must be configured, and the DNS service must be running. See the DNS chapters in this document for details.
- The directory server DIT must be preloaded with (at a minimum) the users of this client machine, the client host, and the necessary `auto_home` LDAP entries. See other sections of this manual for details on how to add entries by using the `ldapaddent` command.

Note – Do not edit either client configuration file directly. Use the `ldapclient` command to create or modify the content of these files.

1 Run `ldapclient init` to initialize the client by using the gssapi profile.

```
# /usr/sbin/ldapclient init -a profilename=gssapi_SPARKS.COM -a \
domainname=example.com 9.9.9.50
```

2 Try to log in as a user:

- **Run `kinit -p` *user***
- **Run `ldaplist -l passwd` *user* in the user's login session and you should see `userpassword`.**

- **Run `ldaplist -l passwd bar` can get the entry without `userpassword`. By default `root` can still see `userpassword` of everybody.**

More Information

Notes About Using Per-User Credentials

- If the `syslog` file has this message: `libsldap: Status: 7 Mesg: openConnection: GSSAPI bind failed - 82 Local error`, it is likely that Kerberos is not initialized or its ticket is expired. Run the `klist` command to browse it. For example, run `kinit -p foo` or `kinit -R -p foo` and try again.
- If you want to, you can add `pam_krb5.so.1` to `/etc/pam.conf` so that it will automatically run the `kinit` command when you log in.

  For example:

  ```
  login   auth optional pam_krb5.so.1
  rlogin  auth optional pam_krb5.so.1
  other   auth optional pam_krb5.so.1
  ```
- If a user has run the `kinit` command and the `syslog` message indicates `Invalid credential`, then the problem could be that the `root` host entry or the user entry is not in the LDAP directory or mapping rules are not correct.
- When the `ldapclient init` command is executed, it checks if the LDAP profile contains a self/ sasl/GSSAPI configuration. If it fails in the switch check, then the usual reason is that DNS was not the search criteria for the hosts database.
  - If the check fails because the DNS client id not enabled, run `svcs -l dns/client` to determine if the service is disabled. Run `svcadm enable dns/client` to enable the service.
  - If the check fails because of a sasl/GSSAPI bind, check `syslog` to determine the problem.

See other references in this guide and in the *Oracle Solaris Administration: Security Services* for details.

▼ How to Initialize an LDAP Client by Using Proxy Credentials

Note – Do not edit either of the client configuration files directly. Use the `ldapclient` command to create or modify the content of these files.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Define proxy values.

```
# ldapclient init \
-a proxyDN=cn=proxyagent,ou=profile,dc=west,dc=example,dc=com \
-a domainName=west.example.com \
-a profileName=pit1 \
-a proxyPassword=test1234 192.168.0.1
System successfully configured
```

The `-a proxyDN` and `-a proxyPassword` are *required* if the profile to be used is set up for `proxy`. As the credentials are not stored in the profile saved on the server, you must supply the information when you initialize the client. This method is more secure than the older method of storing the proxy credentials on the server.

The proxy information is stored in the `svc:/network/ldap/client` service in the `config` and `cred` property groups.

▼ How to Initialize an LDAP Client to Enable the Updating of Shadow Data

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 To set the `enableShadowUpdate` switch and define the admin credential, run the `ldapclient` command.

- **To update an already running LDAP client, run this command:**

  ```
  # ldapclient mod -a enableShadowUpdate=TRUE \
  -a adminDN=cn=admin,ou=profile,dc=west,dc=example,dc=com \
  -a adminPassword=admin-password
  System successfully configured
  ```

- **To initialize an LDAP client, run this command:**

  ```
  # ldapclient init \
  -a adminDN=cn=admin,ou=profile,dc=west,dc=example,dc=com \
  -a adminPassword=admin-password
  -a domainName=west.example.com \
  -a profileName=WestUserProfile \
  -a proxyDN=cn=proxyagent,ou=profile,dc=west,dc=example,dc=com \
  -a proxyPassword=<proxy_password> \
  192.168.0.1
  System successfully configured
  ```

3 **To verify the configuration, display the contents of the `cred` property of the `network/ldap/client` service.**

The output appears similar to the following:

```
# svcprop -p cred svc:/network/ldap/client
cred/read_authorization astring solaris.smf.value.name-service.ldap.client
cred/value_authorization astring solaris.smf.value.name-service.ldap.client
cred/bind_dn astring cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
cred/bind_passwd astring {NS1}4a3788f8eb85de11
cred/enable_shadow_update boolean true
cred/admin_bind_dn astring cn=admin,ou=profile,dc=west,dc=example,dc=com
cred/admin_bind_passwd astring {NS1}4a3788f8c053434f
```

▼ How to Initialize an LDAP Client Manually

Root users or administrators with an equivalent role can perform manual LDAP client configurations. However, many of the checks are bypassed during the process, so it is relatively easy to misconfigure your system. In addition, you must change settings *on every machine*, instead of in one central place, as is done when using profiles.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Initialize the client.**

```
# ldapclient manual \
-a domainName=dc=west.example.com -a credentialLevel=proxy \
-a defaultSearchBase=dc=west,dc=example,dc=com \
-a proxyDN=cn=proxyagent,ou=profile,dc=west,dc=example,dc=com \
-a proxyPassword=testtest 192.168.0.1
```

3 **Verify the LDAP client configuration.**

```
# ldapclient list
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
NS_LDAP_SERVERS= 192.168.0.1
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_CREDENTIAL_LEVEL= proxy
```

▼ How to Modify a Manual LDAP Client Configuration

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Use the `ldapclient mod` command to change the authentication method to `simple`.**

```
# ldapclient mod -a authenticationMethod=simple
```

3 **Verify the change was made to the LDAP client configuration.**

```
# ldapclient list
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
NS_LDAP_SERVERS= 192.168.0.1
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_AUTH= simple
NS_LDAP_CREDENTIAL_LEVEL= proxy
```

Troubleshooting You cannot change some attributes of an LDAP client configuration by using the `mod` subcommand. For example, you cannot change the `profileName` and `profileTTL` attributes. To change these attributes, create a new profile by using the `ldapclient init` command, as described in "How to Initialize an LDAP Client by Using Profiles" on page 176. Or, run the `ldapclient manual` command, as described in "How to Initialize an LDAP Client Manually" on page 180.

▼ How to Uninitialize an LDAP Client

The `ldapclient uninit` command restores the client name service to what it was prior to the most recent `init`, `modify`, or `manual` operation. In other words, the command performs an "undo" on the last step taken. For example, if the client was configured to use `profile1` and was then changed to use `profile2`, using `ldapclient uninit` would revert the client back to using `profile1`.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Uninitialize the LDAP client.**

```
# ldapclient uninit
System successfully recovered
```

Setting Up TLS Security

Note – The security database files must be readable by everyone. Do not include any private keys in the `key3.db` file.

If using transport layer security (TLS), the necessary security databases must be installed. In particular, the certificate and key database files are required. For example, if you use a newer database format from Mozilla Firefox, three files, `cert8.db`, `key3.db` and `secmod.db` are required. The `cert8.db` file contains trusted certificates. The `key3.db` file contains the client's keys. Even if the LDAP naming service client does not use client keys, this file must be present. The `secmod.db` file contains the security modules such as `PKCS#11` module. This file is not required if the older format is used.

Note – Before running `ldapclient`, you should set up and install the needed security database files described in this section.

See the section about configuring LDAP clients to use SSL in the "Managing SSL" chapter of the Administrator's Guide for the version of Oracle Directory Server Enterprise Edition you are using. For information on how to create and manage these files. Once configured, these files must be stored in the location expected by the LDAP naming services client. The attribute `certificatePath` is used to determine this location. This is by default `/var/ldap`.

For example, after setting up the necessary `cert8.db`, `key3.db`, and `secmod.db` files by using Mozilla Firefox, copy the files to the default location as follows:

```
# cp $HOME/.mozilla/firefox/*.default/cert8.db /var/ldap
# cp $HOME/.mozilla/firefox/*.default/key3.db /var/ldap
# cp $HOME/.mozilla/firefox/*.default/secmod.db /var/ldap
```

Next, give everyone read access.

```
# chmod 444 /var/ldap/cert8.db
# chmod 444 /var/ldap/key3.db
# chmod 444 /var/ldap/secmod.db
```

Note – Mozilla Firefox has its `cert8.db`, `key3.db`, and `secmod.db` files managed in a subdirectories under `$HOME/.mozilla`. Copies of these security databases must be stored on a local file system if you are using them for an LDAP naming services client.

Configuring PAM

The `pam_ldap` module is one authentication and account management PAM module option for LDAP. See the `pam_ldap(5)` man page for more information about the features currently supported with `pam_ldap`.

If you have selected both the per-user mode and the self credentials option, then you must also enable the PAM Kerberos `pam_krb5` module. See the `pam_krb5(5)` man page and the *Oracle Solaris Administration: Security Services* documentation for further details.

Configuring PAM to Use UNIX `policy`

To configure PAM to use UNIX `policy`, use the default `/etc/pam.conf` file. No changes are needed. For details, see the `pam.conf(4)` man page.

However, if password aging and password policy as controlled by the `shadow` data are required, the client must be configured and run with the `enableShadowUpdate` switch. See "How to Initialize an LDAP Client to Enable the Updating of Shadow Data" on page 179 for more information.

Configuring PAM to Use LDAP `server_policy`

To configure PAM to use LDAP `server_policy`, follow the sample in "Example `pam_conf` File Using the `pam_ldap` Module for Account Management" on page 196. Add the lines that contain `pam_ldap.so.1` to the client's `/etc/pam.conf` file. In addition, if any PAM module in the sample `pam.conf` file specifies the `binding` flag and the `server_policy` option, use the same flag and option for the corresponding module in the client's `/etc/pam.conf` file. Also, add the `server_policy` option to the line that contains the service module `pam_authtok_store.so.1`.

Note – Previously, if you enabled `pam_ldap` account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as `rsh`, `rlogin`, or `ssh` would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is `1.3.6.1.4.1.42.2.27.9.5.8`, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

- The `binding` control flag

  Using the `binding` control flag allows a local password override of a remote (LDAP) password. For example, if a user account is found on both the local files and the LDAP namespace, the password associated with the local account takes precedence over the remote password. Thus, if the local password expires, authentication fails even if the remote LDAP password is still valid.

- The `server_policy` option

The `server_policy` option instructs `pam_unix_auth`, `pam_unix_account`, and `pam_passwd_auth` to ignore a user found in the LDAP namespace and to allow `pam_ldap` to perform authentication or account validation. In the case of `pam_authtok_store`, a new password is passed to the LDAP server without encryption. The password is thereby stored in the directory according to the password encryption scheme configured on the server. For more information, see `pam.conf(4)` and `pam_ldap(5)`.

Retrieving LDAP Naming Services Information

You can retrieve information about LDAP naming services by using the `ldaplist` utility. This LDAP utility lists the naming information from the LDAP servers in LDIF format. It can be useful for troubleshooting. See `ldaplist(1)` for further information.

Listing All LDAP Containers

`ldaplist` displays its output with a blank line separating records, which is helpful for big multiline records.

Note – The output of `ldaplist` depends upon the client configuration. For example, if the value of `ns_ldap_search` is `sub` rather than `one`, `ldaplist` lists all the entries under the current search `baseDN`.

The following is an example of `ldaplist` output.

```
# ldaplist
dn: ou=people,dc=west,dc=example,dc=com

dn: ou=group,dc=west,dc=example,dc=com

dn: ou=rpc,dc=west,dc=example,dc=com

dn: ou=protocols,dc=west,dc=example,dc=com

dn: ou=networks,dc=west,dc=example,dc=com

dn: ou=netgroup,dc=west,dc=example,dc=com

dn: ou=aliases,dc=west,dc=example,dc=com

dn: ou=hosts,dc=west,dc=example,dc=com

dn: ou=services,dc=west,dc=example,dc=com

dn: ou=ethers,dc=west,dc=example,dc=com

dn: ou=profile,dc=west,dc=example,dc=com
```

```
dn: automountmap=auto_home,dc=west,dc=example,dc=com

dn: automountmap=auto_direct,dc=west,dc=example,dc=com

dn: automountmap=auto_master,dc=west,dc=example,dc=com

dn: automountmap=auto_shared,dc=west,dc=example,dc=com
```

Listing All User Entry Attributes

To list specific information such as a user's `passwd` entry, use `getent` as follows:

```
# getent passwd user1
user1::30641:10:Joe Q. User:/home/user1:/bin/csh
```

If you want to list all attributes, use `ldaplist` with the `-l` option.

```
# ldaplist -l passwd user1
        dn: uid=user1,ou=People,dc=west,dc=example,dc=com
        uid: user1
        cn: user1
        uidNumber: 30641
        gidNumber: 10
        gecos: Joe Q. User
        homeDirectory: /home/user1
        loginShell: /bin/csh
        objectClass: top
        objectClass: shadowAccount
        objectClass: account
        objectClass: posixAccount
        shadowLastChange: 6445
```

Customizing the LDAP Client Environment

The following sections describe how you can customize the LDAP client environment.

You can change any of the services, but be careful, because if the data is not populated on the server for the service specified, things will stop working. Also, in some cases files may not be set up by default.

Modifying the Name Service Switch for LDAP

You can modify the name service switch to customize where each naming service accesses its information. See "Managing the Name Service Switch" on page 38.

Enabling DNS With LDAP

If you want to enable DNS, see "How to Enable a DNS Client" on page 46. If per-user authentication is used, the sasl/GSSAPI and Kerberos mechanisms expect the DNS naming service to be configured and enabled.

◆ ◆ ◆ CHAPTER 13

LDAP Troubleshooting (Reference)

This chapter describes LDAP configuration problems and suggests solutions for resolving them.

Monitoring LDAP Client Status

The following sections show various commands to help determine the state of the LDAP client environment. Also see the man pages for additional information about the options that can be used.

For an overview of the Service Management Facility (SMF), refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) and `svcs`(1) man pages for more details.

Verifying That the `ldap_cachemgr` Daemon Is Running

The `ldap_cachemgr` daemon must be running and functioning correctly at all times. Otherwise, the system doesn't work. When you set up and start the LDAP client service, `svc:/network/ldap/client`, the client SMF method automatically starts the `ldap_cachemgr` daemon. The following methods determine if the LDAP client service is online:

- Use the `svcs` command to see if the service is enabled.

```
# svcs \*ldap\*
STATE          STIME    FMRI
disabled       Aug_24   svc:/network/ldap/client:default
```

- Use this command to see all information about the service.

```
# svcs -l network/ldap/client:default
fmri svc:/network/ldap/client:default
name LDAP Name Service Client
enabled false
```

```
state disabled
next_state none
state_time Thu Oct 20 23:04:11 2011
logfile /var/svc/log/network-ldap-client:default.log
restarter svc:/system/svc/restarter:default
contract_id
manifest /lib/svc/manifest/network/ldap/client.xml
manifest /lib/svc/manifest/milestone/config.xml
manifest /lib/svc/manifest/network/network-location.xml
manifest /lib/svc/manifest/system/name-service/upgrade.xml
dependency optional_all/none svc:/milestone/config (online)
dependency optional_all/none svc:/network/location:default (online)
dependency require_all/none svc:/system/filesystem/minimal (online)
dependency require_all/none svc:/network/initial (online)
dependency require_all/restart svc:/network/nis/domain (online)
dependency optional_all/none svc:/system/manifest-import (online)
dependency require_all/none svc:/milestone/unconfig (online)
dependency optional_all/none svc:/system/name-service/upgrade (online)
```

- Pass the `-g` option to `ldap_cachemgr`.

  This option provides more extensive status information, which is useful when you diagnose a problem.

```
# /usr/lib/ldap/ldap_cachemgr -g
cachemgr configuration:
server debug level          0
server log file "/var/ldap/cachemgr.log"
number of calls to ldapcachemgr         19

cachemgr cache data statistics:
Configuration refresh information:
  Previous refresh time: 2010/11/16 18:33:28
  Next refresh time:     2010/11/16 18:43:28
Server information:
  Previous refresh time: 2010/11/16 18:33:28
  Next refresh time:     2010/11/16 18:36:08
  server: 192.168.0.0, status: UP
  server: 192.168.0.1, status: ERROR
    error message: Can't connect to the LDAP server
Cache data information:
  Maximum cache entries:          256
  Number of cache entries:          2
```

For more information about the `ldap_cachemgr` daemon, see the `ldap_cachemgr(1M)` man page.

Checking the Current Profile Information

Become superuser or assume an equivalent role, and run `ldapclient` with the `list` option.

```
# ldapclient list
NS_LDAP_FILE_VERSION= 2.0
NS_LDAP_BINDDN= cn=proxyagent,ou=profile,dc=west,dc=example,dc=com
NS_LDAP_BINDPASSWD= {NS1}4a3788e8c053424f
```

```
NS_LDAP_SERVERS= 192.168.0.1, 192.168.0.10
NS_LDAP_SEARCH_BASEDN= dc=west,dc=example,dc=com
NS_LDAP_AUTH= simple
NS_LDAP_SEARCH_REF= TRUE
NS_LDAP_SEARCH_SCOPE= one
NS_LDAP_SEARCH_TIME= 30
NS_LDAP_SERVER_PREF= 192.168.0.1
NS_LDAP_PROFILE= pit1
NS_LDAP_CREDENTIAL_LEVEL= proxy
NS_LDAP_SERVICE_SEARCH_DESC= passwd:ou=people,?sub
NS_LDAP_SERVICE_SEARCH_DESC= group:ou=group,dc=west,dc=example,dc=com?one
NS_LDAP_BIND_TIME= 5
```

The current profile information can be viewed using the `svccfgor svcprop` command, or the `ldapclient` command with the `list` option. See the `ldapclient(1M)` man page for specific information about every available property setting.

Verifying Basic Client-Server Communication

The best way to show that your client is talking to the LDAP server is with the `ldaplist` command. Using `ldaplist` with no arguments dumps all the containers on the server. This works as long as the containers exist, and do not have to be populated. See the `ldaplist(1)` man page for more information.

If the first step works, you can try `ldaplist passwd` *username* or `ldaplist hosts` *hostname* but if they contain lots of data you might want to pick a less populated service, or pipe them to `head` or `more`.

Checking Server Data From a Non-Client Machine

Most of the commands in the preceding sections assume that you have already created an LDAP client. If you have not created a client and want to check the data on the server, use the `ldapsearch` command. The following example lists all of the containers.

```
# ldapsearch -h server1 -b "dc=west,dc=example,dc=com" -s one "objectclass=*"
```

The default output for the `ldapsearch` command is the industry standardized LDIF format that is defined by RFC-2849. All versions of `ldapsearch` can output LDIF format using the `-L` option.

LDAP Configuration Problems and Solutions

The following sections describe LDAP configuration problems and suggests solutions to the problems.

Unresolved Host Name

The LDAP client back end returns fully qualified host names for host lookups, such as host names returned by `gethostbyname()` and `getaddrinfo()`. If the name stored is qualified, that is, contains at least one dot, the client returns the name as is. For example, if the name stored is `hostB.eng`, the returned name is `hostB.eng`.

If the name stored in the LDAP directory is not qualified (it does not contain a dot), the client back end appends the domain part to the name. For example, if the name stored is `hostA`, the returned name is `hostA.domainname`.

Unable to Reach Systems in the LDAP Domain Remotely

If the DNS domain name is different from the LDAP domain name, then the LDAP naming service cannot be used to serve host names unless the host names are stored fully qualified.

Login Does Not Work

LDAP clients use the PAM modules for user authentication during login. When using the standard UNIX PAM module, the password is read from the server and checked on the client side. This process can fail due to one of the following reasons:

1. `ldap` is not associated with the `passwd` database in the name service switch.
2. The user's `userPassword` attribute on the server list is not readable by the proxy agent. You need to allow at least the proxy agent to read the password because the proxy agent returns it to the client for comparison. `pam_ldap` does not require read access to the password.
3. The proxy agent might not have the correct password.
4. The entry does not have the `shadowAccount` object class.
5. No password is defined for the user.

   When you use `ldapaddent`, you must use the `-p` option to ensure that the password is added to the user entry. If you use `ldapaddent` without the `-p` option, the user's password is not stored in the directory unless you also add the `/etc/shadow` file by using `ldapaddent`.
6. No LDAP servers are reachable.

Check the status of the servers.

```
# /usr/lib/ldap/ldap_cachemgr -g
```

7. `pam.conf` is configured incorrectly.
8. The user is not defined in the LDAP namespace.
9. `NS_LDAP_CREDENTIAL_LEVEL` is set to `anonymous` for the `pam_unix_*` modules, and `userPassword` is not available to anonymous users.
10. The password is not stored in `crypt` format.
11. If `pam_ldap` is configured to support account management, login failure could be the result of one of the following:
    - The user's password has expired.
    - The user's account is locked out due to too many failed login attempts.
    - The user's account has been deactivated by the administrator.
    - The user tried to log in using a nonpassword-based program, such as `rsh`, `rlogin`, `ssh`, or `sftp`.
12. If per-user authentication and sasl/GSSAPI are being used, then some component of Kerberos or the `pam_krb5` configuration is setup incorrectly. Refer to the *Oracle Solaris Administration: Security Services* for details on resolving these issues.

Lookup Too Slow

The LDAP database relies on indexes to improve search performance. A major performance degradation occurs when indexes are improperly configured. The documentation includes a common set of attributes that should be indexed. You can also add your own indexes to improve performance at your site.

`ldapclient` Command Cannot Bind to a Server

The `ldapclient` command failed to initialize the client when using the `init` option with the `profileName` attribute specified. Possible reasons for failure include the following:

1. The incorrect domain name was specified on the command line.
2. The `nisDomain` attribute is not set in the DIT to represent the entry point for the specified client domain.
3. Access control information is not set up properly on the server, thus disallowing anonymous search in the LDAP database.
4. An incorrect server address passed to the `ldapclient` command. Use the `ldapsearch` command to verify the server address.
5. An incorrect profile name passed to the `ldapclient` command. Use the `ldapsearch` command to verify the profile name in the DIT.

6. Use `snoop` on the client's network interface to see what sort of traffic is going out, and determine to which server it is talking.

Using the `ldap_cachemgr` Daemon for Debugging

Running the `ldap_cachemgr` daemon with the `-g` option can be a useful way to debug, as you can view the current client configuration and statistics. For example,

```
# ldap_cachemgr -g
```

would print current configuration and statistics to standard output, including the status of all LDAP servers, as mentioned previously. Note that you do *not* need to become super user to execute this command.

`ldapclient` Command Hangs During Setup

If the `ldapclient` command hangs, pressing Ctrl-C will exit after restoring the previous environment. If this happens, check with the server administrator to ensure that the server is running.

Also check the server list attributes in either the profile or from the command line and make sure that the server information is correct.

◆ ◆ ◆ CHAPTER 14

LDAP Naming Service (Reference)

This chapter covers the following topics.

Blank Checklists for Configuring LDAP

TABLE 14–1 Blank Checklist for Server Variable Definitions

Variable	Definition for _____ Network
Port number at which an instance of the directory server is installed (389)	
Name of server	
Replica servers (IP number:port number)	
Directory manager [`dn: cn=directory manager`]	
Domain name to be served	
Maximum time (in seconds) to process client requests before timing out	

TABLE 14–1 Blank Checklist for Server Variable Definitions *(Continued)*

Variable	Definition for _____ Network
Maximum number of entries returned for each search request	

TABLE 14–2 Blank Checklist for Client Profile Variable Definitions

Variable	Definition for _____ Network
Profile name	
Server list (defaults to the local subnet)	
Preferred server list (listed in order of which server to try first, second, and so on)	
Search scope (number of levels down through the directory tree. 'One' or 'Sub')	
Credential used to gain access to server. The default is anonymous.	
Follow Referrals? (a pointer to another server if the main server is unavailable) The default is no.	
Search time limit (in seconds) for waiting for server to return information. The default is 30 seconds.	
Bind time limit (in seconds) for contacting server. The default is 30 seconds.	
Authentication method Default is none.	

LDAP Commands

There are two sets of LDAP-related commands in the Oracle Solaris system. One set is the general LDAP tools, which do not require the client to be configured with LDAP naming services. The second set uses the common LDAP configuration on the client and can run on clients that are configured with or without the LDAP naming service.

General LDAP Tools

LDAP command line tools support a common set of options, including authentication and bind parameters. The following tools support a common text-based format for representing directory information called the LDAP Data Interchange Format (LDIF). These commands can be used to manipulate directory entries directly.

ldapsearch(1)
ldapmodify(1)
ldapadd(1)
ldapdelete(1)

LDAP Tools Requiring LDAP Naming Services

TABLE 14–3 LDAP Tools

Tool	Function
ldapaddent(1M)	Used to create entries in LDAP containers from the corresponding `/etc` files. This tool allows populating the directory from files. For example, it reads `/etc/passwd` format file and populates `passwd` entries in the directory.
ldaplist(1)	Used to list contents of various services from the directory.
idsconfig(1M)	Used to set up Oracle Directory Server Enterprise Edition to serve LDAP naming service clients.

Example pam_conf File Using the pam_ldap Module for Account Management

Note – Previously, if you enabled pam_ldap account management, all users needed to provide a login password for authentication any time they logged in to the system. Therefore, non-password-based logins using tools such as rsh, rlogin, or ssh would fail.

Perform account management and retrieve the account status of users without authenticating to Directory Server as the user is logging in. The new control on Directory Server is 1.3.6.1.4.1.42.2.27.9.5.8, which is enabled by default.

To modify this control for other than default, add Access Control Instructions (ACI) on Directory Server:

```
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
objectClass: top
objectClass: directoryServerFeature
oid:1.3.6.1.4.1.42.2.27.9.5.8
cn:Password Policy Account Usable Request Control
aci: (targetattr != "aci")(version 3.0; acl "Account Usable";
     allow (read, search, compare, proxy)
     (groupdn = "ldap:///cn=Administrators,cn=config");)
creatorsName: cn=server,cn=plugins,cn=config
modifiersName: cn=server,cn=plugins,cn=config
```

```
#
# Authentication management
#
# login service (explicit because of pam_dial_auth)
#
login   auth requisite        pam_authtok_get.so.1
login   auth required         pam_dhkeys.so.1
login   auth required         pam_unix_cred.so.1
login   auth required         pam_dial_auth.so.1
login   auth binding          pam_unix_auth.so.1 server_policy
login   auth required         pam_ldap.so.1
#
# rlogin service (explicit because of pam_rhost_auth)
#
rlogin  auth sufficient       pam_rhosts_auth.so.1
rlogin  auth requisite        pam_authtok_get.so.1
rlogin  auth required         pam_dhkeys.so.1
rlogin  auth required         pam_unix_cred.so.1
rlogin  auth binding          pam_unix_auth.so.1 server_policy
rlogin  auth required         pam_ldap.so.1
#
# rsh service (explicit because of pam_rhost_auth,
# and pam_unix_auth for meaningful pam_setcred)
#
rsh     auth sufficient       pam_rhosts_auth.so.1
rsh     auth required         pam_unix_cred.so.1
```

```
rsh     auth binding          pam_unix_auth.so.1 server_policy
rsh     auth required         pam_ldap.so.1
#
# PPP service (explicit because of pam_dial_auth)
#
ppp     auth requisite        pam_authtok_get.so.1
ppp     auth required         pam_dhkeys.so.1
ppp     auth required         pam_dial_auth.so.1
ppp     auth binding          pam_unix_auth.so.1 server_policy
ppp     auth required         pam_ldap.so.1
#
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
#
other   auth requisite        pam_authtok_get.so.1
other   auth required         pam_dhkeys.so.1
other   auth required         pam_unix_cred.so.1
other   auth binding          pam_unix_auth.so.1 server_policy
other   auth required         pam_ldap.so.1
#
# passwd command (explicit because of a different authentication module)
#
passwd  auth binding          pam_passwd_auth.so.1 server_policy
passwd  auth required         pam_ldap.so.1
#
# cron service (explicit because of non-usage of pam_roles.so.1)
#
cron    account required      pam_unix_account.so.1
#
# Default definition for Account management
# Used when service name is not explicitly mentioned for account management
#
other   account requisite     pam_roles.so.1
other   account binding       pam_unix_account.so.1 server_policy
other   account required      pam_ldap.so.1
#
# Default definition for Session management
# Used when service name is not explicitly mentioned for session management
#
other   session required      pam_unix_session.so.1
#
# Default definition for  Password management
# Used when service name is not explicitly mentioned for password management
#
other   password required     pam_dhkeys.so.1
other   password requisite    pam_authtok_get.so.1
other   password requisite    pam_authtok_check.so.1
other   password required     pam_authtok_store.so.1 server_policy
#
# Support for Kerberos V5 authentication and example configurations can
# be found in the pam_krb5(5) man page under the "EXAMPLES" section.
#
```

IETF Schemas for LDAP

Schemas are definitions that describe what types of information can be stored as entries in a server's directory.

For a directory server to support LDAP naming clients, schemas defined in this chapter must be configured in the server unless schema is mapped using the schema mapping feature of the clients.

Several required LDAP schemas are defined by IETF: the RFC 2307 Network Information Service schema and RFC 2307bis, and a Configuration Profile Schema for Lightweight Directory Access Protocol (LDAP)-Based Agents (RFC 4876), and the LDAP Schema for Printer Services. To support the NIS, the definition of these schemas must be added to the directory server. The various RFCs can be accessed on the IETF web site at `http://www.ietf.org`.

Note – Internet drafts, such as RFC 2307bis, are draft documents valid for a maximum of six months and might be updated, or rendered obsolete, by other documents at any time.

RFC 2307bis Network Information Service Schema

The LDAP serverThe LDAP servers must be configured to support the revised RFC 2307bis:

The `nisSchema` OID is `1.3.6.1.1`. The RFC 2307bis attributes are the following.

```
( nisSchema.1.0 NAME 'uidNumber'
DESC 'An integer uniquely identifying a user in an
        administrative domain'
EQUALITY integerMatch SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.1 NAME 'gidNumber'
DESC 'An integer uniquely identifying a group in an
        administrative domain'
EQUALITY integerMatch SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.2 NAME 'gecos'
DESC 'The GECOS field; the common name'
EQUALITY caseIgnoreIA5Match
SUBSTRINGS caseIgnoreIA5SubstringsMatch
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.3 NAME 'homeDirectory'
DESC 'The absolute path to the home directory'
EQUALITY caseExactIA5Match
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.4 NAME 'loginShell'
DESC 'The path to the login shell'
EQUALITY caseExactIA5Match
```

```
SYNTAX 'IA5String' SINGLE-VALUE )

( nisSchema.1.5 NAME 'shadowLastChange'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.6 NAME 'shadowMin'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.7 NAME 'shadowMax'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.8 NAME 'shadowWarning'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.9 NAME 'shadowInactive'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.10 NAME 'shadowExpire'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.11 NAME 'shadowFlag'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.12 NAME 'memberUid'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String' )

( nisSchema.1.13 NAME 'memberNisNetgroup'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String' )

( nisSchema.1.14 NAME 'nisNetgroupTriple'
DESC 'Netgroup triple'
SYNTAX 'nisNetgroupTripleSyntax' )

( nisSchema.1.15 NAME 'ipServicePort'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.16 NAME 'ipServiceProtocol'
SUP name )

( nisSchema.1.17 NAME 'ipProtocolNumber'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )

( nisSchema.1.18 NAME 'oncRpcNumber'
EQUALITY integerMatch
SYNTAX 'INTEGER' SINGLE-VALUE )
```

```
( nisSchema.1.19 NAME 'ipHostNumber'
DESC 'IP address as a dotted decimal, eg. 192.168.1.1
        omitting leading zeros'
SUP name )

( nisSchema.1.20 NAME 'ipNetworkNumber'
DESC 'IP network as a dotted decimal, eg. 192.168,
        omitting leading zeros'
SUP name SINGLE-VALUE )

( nisSchema.1.21 NAME 'ipNetmaskNumber'
DESC 'IP netmask as a dotted decimal, eg. 255.255.255.0,
         omitting leading zeros'
EQUALITY caseIgnoreIA5Match
SYNTAX 'IA5String{128}' SINGLE-VALUE )

( nisSchema.1.22 NAME 'macAddress'
DESC 'MAC address in maximal, colon separated hex
     notation, eg. 00:00:92:90:ee:e2'
EQUALITY caseIgnoreIA5Match
SYNTAX 'IA5String{128}' )

( nisSchema.1.23 NAME 'bootParameter'
DESC 'rpc.bootparamd parameter'
SYNTAX 'bootParameterSyntax' )

( nisSchema.1.24 NAME 'bootFile'
DESC 'Boot image name'
EQUALITY caseExactIA5Match
SYNTAX 'IA5String' )

( nisSchema.1.26 NAME 'nisMapName'
SUP name )

( nisSchema.1.27 NAME 'nisMapEntry'
EQUALITY caseExactIA5Match
SUBSTRINGS caseExactIA5SubstringsMatch
SYNTAX 'IA5String{1024}' SINGLE-VALUE )

( nisSchema.1.28 NAME 'nisPublicKey'
DESC 'NIS public key'
SYNTAX 'nisPublicKeySyntax' )

( nisSchema.1.29 NAME 'nisSecretKey'
DESC 'NIS secret key'
SYNTAX 'nisSecretKeySyntax' )

( nisSchema.1.30 NAME 'nisDomain'
DESC 'NIS domain'
SYNTAX 'IA5String' )

( nisSchema.1.31 NAME 'automountMapName'
DESC 'automount Map Name'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

( nisSchema.1.32 NAME 'automountKey'
DESC 'Automount Key value'
```

```
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

( nisSchema.1.33 NAME 'automountInformation'
DESC 'Automount information'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

The `nisSchema` OID is `1.3.6.1.1`. The RFC 2307 `objectClasses` are the following.

```
( nisSchema.2.0 NAME 'posixAccount' SUP top AUXILIARY
  DESC 'Abstraction of an account with POSIX attributes'
  MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
  MAY ( userPassword $ loginShell $ gecos $ description ) )

( nisSchema.2.1 NAME 'shadowAccount' SUP top AUXILIARY
  DESC 'Additional attributes for shadow passwords'
  MUST uid
  MAY ( userPassword $ shadowLastChange $ shadowMin
        shadowMax $ shadowWarning $ shadowInactive $
        shadowExpire $ shadowFlag $ description ) )

( nisSchema.2.2 NAME 'posixGroup' SUP top STRUCTURAL
  DESC 'Abstraction of a group of accounts'
  MUST ( cn $ gidNumber )
  MAY ( userPassword $ memberUid $ description ) )

( nisSchema.2.3 NAME 'ipService' SUP top STRUCTURAL
  DESC 'Abstraction an Internet Protocol service.
        Maps an IP port and protocol (such as tcp or udp)
        to one or more names; the distinguished value of
        the cn attribute denotes the service's canonical
        name'
  MUST ( cn $ ipServicePort $ ipServiceProtocol )
  MAY ( description ) )

( nisSchema.2.4 NAME 'ipProtocol' SUP top STRUCTURAL
  DESC 'Abstraction of an IP protocol. Maps a protocol number
        to one or more names. The distinguished value of the cn
        attribute denotes the protocol's canonical name'
  MUST ( cn $ ipProtocolNumber )
  MAY  description )

( nisSchema.2.5 NAME 'oncRpc' SUP top STRUCTURAL
  DESC 'Abstraction of an Open Network Computing (ONC)
        [RFC1057] Remote Procedure Call (RPC) binding.
        This class maps an ONC RPC number to a name.
        The distinguished value of the cn attribute denotes
        the RPC service's canonical name'
  MUST ( cn $ oncRpcNumber $ description )
  MAY  description )

( nisSchema.2.6 NAME 'ipHost' SUP top AUXILIARY
  DESC 'Abstraction of a host, an IP device. The distinguished
        value of the cn attribute denotes the host's canonical
        name. Device SHOULD be used as a structural class'
```

```
  MUST ( cn $ ipHostNumber )
  MAY ( l $ description $ manager $ userPassword ) )

( nisSchema.2.7 NAME 'ipNetwork' SUP top STRUCTURAL
  DESC 'Abstraction of a network. The distinguished value of
        the cn attribute denotes the network's canonical name'
  MUST ipNetworkNumber
  MAY ( cn $ ipNetmaskNumber $ l $ description $ manager ) )

( nisSchema.2.8 NAME 'nisNetgroup' SUP top STRUCTURAL
  DESC 'Abstraction of a netgroup. May refer to other netgroups'
  MUST cn
  MAY ( nisNetgroupTriple $ memberNisNetgroup $ description ) )

( nisSchema.2.9 NAME 'nisMap' SUP top STRUCTURAL
  DESC 'A generic abstraction of a NIS map'
  MUST nisMapName
  MAY description )

( nisSchema.2.10 NAME 'nisObject' SUP top STRUCTURAL
  DESC 'An entry in a NIS map'
  MUST ( cn $ nisMapEntry $ nisMapName )
  MAY description )

( nisSchema.2.11 NAME 'ieee802Device' SUP top AUXILIARY
  DESC 'A device with a MAC address; device SHOULD be
        used as a structural class'
  MAY macAddress )

( nisSchema.2.12 NAME 'bootableDevice' SUP top AUXILIARY
  DESC 'A device with boot parameters; device SHOULD be
  used as a structural class'
  MAY ( bootFile $ bootParameter ) )

( nisSchema.2.14 NAME 'nisKeyObject' SUP top AUXILIARY
  DESC 'An object with a public and secret key'
  MUST ( cn $ nisPublicKey $ nisSecretKey )
  MAY ( uidNumber $ description ) )

( nisSchema.2.15 NAME 'nisDomainObject' SUP top AUXILIARY
  DESC 'Associates a NIS domain with a naming context'
  MUST nisDomain )

( nisSchema.2.16 NAME 'automountMap' SUP top STRUCTURAL
  MUST ( automountMapName )
  MAY description )

( nisSchema.2.17 NAME 'automount' SUP top STRUCTURAL
  DESC 'Automount information'
  MUST ( automountKey $ automountInformation )
  MAY description )

( nisSchema.2.18 NAME 'groupOfMembers' SUP top STRUCTURAL
  DESC 'A group with members (DNs)'
  MUST cn
  MAY ( businessCategory $ seeAlso $ owner $ ou $ o $
        description $ member ) )
```

Mail Alias Schema

Mail alias information uses the schema defined by this Internet draft. Until a new schema becomes available, LDAP clients will continue to use this schema for mail alias information.

The original LDAP mail groups schema contains a large number of attributes and object classes. Only two attributes and a single object class are used by LDAP clients. These are listed below.

The mail alias attributes are the following.

```
( 0.9.2342.19200300.100.1.3
  NAME 'mail'
  DESC 'RFC822 email address for this person'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String(256)'
  SINGLE-VALUE )

( 2.16.840.1.113730.3.1.30
  NAME 'mgrpRFC822MailMember'
  DESC 'RFC822 mail address of email only member of group'
  EQUALITY CaseIgnoreIA5Match
  SYNTAX 'IA5String(256)' )
```

The schema for the `mailGroup` object class is the following.

```
( 2.16.840.1.113730.3.2.4
  NAME 'mailGroup'
  SUP top
  STRUCTURAL
  MUST mail
  MAY ( cn $ mailAlternateAddress $ mailHost $ mailRequireAuth $
   mgrpAddHeader $ mgrpAllowedBroadcaster $ mgrpAllowedDomain $
   mgrpApprovePassword $ mgrpBroadcasterModeration $ mgrpDeliverTo $
   mgrpErrorsTo $ mgrpModerator $ mgrpMsgMaxSize $
   mgrpMsgRejectAction $ mgrpMsgRejectText $ mgrpNoMatchAddrs $
   mgrpRemoveHeader $ mgrpRFC822MailMember ))
```

Directory User Agent Profile (`DUAProfile`) Schema

The `DUAConfSchemaOID` is `1.3.6.1.4.1.11.1.3.1`.

```
DESC 'Default LDAP server host address used by a DUA'
            EQUALITY caseIgnoreMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
            SINGLE-VALUE )

          ( DUAConfSchemaOID.1.1 NAME 'defaultSearchBase'
            DESC 'Default LDAP base DN used by a DUA'
            EQUALITY distinguishedNameMatch
            SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
            SINGLE-VALUE )
```

```
( DUAConfSchemaOID.1.2 NAME 'preferredServerList'
  DESC 'Preferred LDAP server host addresses to be used by a
  DUA'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( DUAConfSchemaOID.1.3 NAME 'searchTimeLimit'
  DESC 'Maximum time in seconds a DUA should allow for a
  search to complete'
  EQUALITY integerMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
  SINGLE-VALUE )

( DUAConfSchemaOID.1.4 NAME 'bindTimeLimit'
  DESC 'Maximum time in seconds a DUA should allow for the
  bind operation to complete'
  EQUALITY integerMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
  SINGLE-VALUE )

( DUAConfSchemaOID.1.5 NAME 'followReferrals'
  DESC 'Tells DUA if it should follow referrals
  returned by a DSA search result'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
  SINGLE-VALUE )

( DUAConfSchemaOID.1.6 NAME 'authenticationMethod'
  DESC 'A keystring which identifies the type of
  authentication method used to contact the DSA'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( DUAConfSchemaOID.1.7 NAME 'profileTTL'
  DESC 'Time to live, in seconds, before a client DUA
  should re-read this configuration profile'
      'serviceSearchDescriptor'
  DESC 'LDAP search descriptor list used by a DUA'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( DUAConfSchemaOID.1.9 NAME 'attributeMap'
  DESC 'Attribute mappings used by a DUA'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

( DUAConfSchemaOID.1.10 NAME 'credentialLevel'
  DESC 'Identifies type of credentials a DUA should
  use when binding to the LDAP server'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  SINGLE-VALUE )

( DUAConfSchemaOID.1.11 NAME 'objectclassMap'
  DESC 'Objectclass mappings used by a DUA'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
```

```
( DUAConfSchemaOID.1.12 NAME 'defaultSearchScope' SINGLE-VALUE )

( DUAConfSchemaOID.1.13 NAME 'serviceCredentialLevel'
  DESC 'Identifies type of credentials a DUA
  should use when binding to the LDAP server for a
  specific service'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

( DUAConfSchemaOID.1.15 NAME 'serviceAuthenticationMethod'
  DESC 'Authentication Method used by a service of the DUA'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

    ( DUAConfSchemaOID.2.4 NAME 'DUAConfigProfile'
         SUP top STRUCTURAL
       DESC 'Abstraction of a base configuration for a DUA'
       MUST ( cn )
       MAY ( defaultServerList $ preferredServerList $
      defaultSearchBase $ defaultSearchScope $
      searchTimeLimit $ bindTimeLimit $
      credentialLevel $ authenticationMethod $
      followReferrals $ serviceSearchDescriptor $
      serviceCredentialLevel $ serviceAuthenticationMethod $
      objectclassMap $ attributeMap $
      profileTTL ) )
```

Oracle Solaris Schemas

The schemas required for the Oracle Solaris platform are the following.

- Projects schema
- Role-based access control and execution profile schemas
- Printer schemas

Projects Schema

The `/etc/project` file is a local source of attributes associated with projects. For more information, see the `user_attr(4)` man page.

The project attributes are the following.

```
( 1.3.6.1.4.1.42.2.27.5.1.1 NAME 'SolarisProjectID'
  DESC 'Unique ID for a Solaris Project entry'
  EQUALITY integerMatch
  SYNTAX INTEGER SINGLE )

( 1.3.6.1.4.1.42.2.27.5.1.2 NAME 'SolarisProjectName'
  DESC 'Name of a Solaris Project entry'
  EQUALITY caseExactIA5Match
```

```
  SYNTAX IA5String SINGLE )

( 1.3.6.1.4.1.42.2.27.5.1.3 NAME 'SolarisProjectAttr'
  DESC 'Attributes of a Solaris Project entry'
  EQUALITY caseExactIA5Match
  SYNTAX IA5String )

( 1.3.6.1.4.1.42.2.27.5.1.30 NAME 'memberGid'
  DESC 'Posix Group Name'
  EQUALITY caseExactIA5Match
  SYNTAX 'IA5String' )
```

The Project `objectClass` is the following.

```
( 1.3.6.1.4.1.42.2.27.5.2.1 NAME 'SolarisProject'
  SUP top STRUCTURAL
  MUST ( SolarisProjectID $ SolarisProjectName )
  MAY ( memberUid $ memberGid $ description $ SolarisProjectAttr ) )
```

Role-Based Access Control and Execution Profile Schema

The `/etc/user_attr` file is a local source of extended attributes associated with users and roles. For more information, see the `user_attr(4)` man page.

The role-based access control Attributes are the following.

```
( 1.3.6.1.4.1.42.2.27.5.1.4 NAME 'SolarisAttrKeyValue'
  DESC 'Semi-colon separated key=value pairs of attributes'
  EQUALITY caseIgnoreIA5Match
  SUBSTRINGS caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.7 NAME 'SolarisAttrShortDesc'
  DESC 'Short description about an entry, used by GUIs'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.8 NAME 'SolarisAttrLongDesc'
  DESC 'Detail description about an entry'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.9 NAME 'SolarisKernelSecurityPolicy'
  DESC 'Solaris  kernel security policy'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.10 NAME 'SolarisProfileType'
  DESC 'Type of object defined in profile'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )
```

```
( 1.3.6.1.4.1.42.2.27.5.1.11 NAME 'SolarisProfileId'
  DESC 'Identifier of object defined in profile'
  EQUALITY caseExactIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.12 NAME 'SolarisUserQualifier'
  DESC 'Per-user login attributes'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.13 NAME 'SolarisReserved1'
  DESC 'Reserved for future use'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.14 NAME 'SolarisReserved2'
  DESC 'Reserved for future use'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 'IA5String' SINGLE-VALUE )
```

The role based access control `objectClassses` are the following.

```
( 1.3.6.1.4.1.42.2.27.5.2.3 NAME 'SolarisUserAttr' SUP top AUXILIARY
  DESC 'User attributes'
  MAY ( SolarisUserQualifier $ SolarisAttrReserved1 $ \
        SolarisAttrReserved2 $ SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.4 NAME 'SolarisAuthAttr' SUP top STRUCTURAL
  DESC 'Authorizations data'
  MUST cn
  MAY ( SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisAttrShortDesc $ SolarisAttrLongDesc $ \
        SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.5 NAME 'SolarisProfAttr' SUP top STRUCTURAL
  DESC 'Profiles data'
  MUST cn
  MAY ( SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisAttrLongDesc $ SolarisAttrKeyValue ) )

( 1.3.6.1.4.1.42.2.27.5.2.6 NAME 'SolarisExecAttr' SUP top AUXILIARY
  DESC 'Profiles execution attributes'
  MAY ( SolarisKernelSecurityPolicy $ SolarisProfileType $ \
        SolarisAttrReserved1 $ SolarisAttrReserved2 $ \
        SolarisProfileId $ SolarisAttrKeyValue ) )
```

Internet Print Protocol Information for LDAP

The following sections provide information about the attributes and `ObjectClasses` for the internet print protocol and the printer.

Internet Print Protocol Attributes

```
( 1.3.18.0.2.4.1140
NAME 'printer-uri'
DESC 'A URI supported by this printer.
This URI SHOULD be used as a relative distinguished name (RDN).
If printer-xri-supported is implemented, then this URI value
MUST be listed in a member value of printer-xri-supported.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

( 1.3.18.0.2.4.1107
NAME 'printer-xri-supported'
DESC 'The unordered list of XRI (extended resource identifiers) supported
by this printer.
Each member of the list consists of a URI (uniform resource identifier)
followed by optional authentication and security metaparameters.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( 1.3.18.0.2.4.1135
NAME 'printer-name'
DESC 'The site-specific administrative name of this printer, more end-user
friendly than a URI.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )

( 1.3.18.0.2.4.1119
NAME 'printer-natural-language-configured'
DESC 'The configured language in which error and status messages will be
generated (by default) by this printer.
Also, a possible language for printer string attributes set by operator,
system administrator, or manufacturer.
Also, the (declared) language of the "printer-name", "printer-location",
"printer-info", and "printer-make-and-model" attributes of this printer.
For example: "en-us" (US English) or "fr-fr" (French in France) Legal values of
language tags conform to [RFC3066] "Tags for the Identification of Languages".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )

( 1.3.18.0.2.4.1136
NAME 'printer-location'
DESC 'Identifies the location of the printer. This could include
things like: "in Room 123A", "second floor of building XYZ".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} SINGLE-VALUE )
```

```
( 1.3.18.0.2.4.1139
NAME 'printer-info'
DESC 'Identifies the descriptive information about this printer.
This could include things like: "This printer can be used for
printing color transparencies for HR presentations", or
"Out of courtesy for others, please print only small (1-5 page)
jobs at this printer", or even "This printer is going away on July 1, 1997,
please find a new printer".'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}
SINGLE-VALUE )

( 1.3.18.0.2.4.1134
NAME 'printer-more-info'
DESC 'A URI used to obtain more information about this specific printer.
For example, this could be an HTTP type URI referencing an HTML page
accessible to a Web Browser.
The information obtained from this URI is intended for end user consumption.'
EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

( 1.3.18.0.2.4.1138
NAME 'printer-make-and-model'
DESC 'Identifies the make and model of the device.
The device manufacturer MAY initially populate this attribute.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{127}  SINGLE-VALUE )

( 1.3.18.0.2.4.1133
NAME 'printer-ipp-versions-supported'
DESC 'Identifies the IPP protocol version(s) that this printer supports,
including major and minor versions,
i.e., the version numbers for which this Printer implementation meets
the conformance requirements.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1132
NAME 'printer-multiple-document-jobs-supported'
DESC 'Indicates whether or not the printer supports more than one
document per job, i.e., more than one Send-Document or Send-Data
operation with document data.'
EQUALITY booleanMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

( 1.3.18.0.2.4.1109
NAME 'printer-charset-configured'
DESC 'The configured charset in which error and status messages will be
generated (by default) by this printer.
Also, a possible charset for printer string attributes set by operator,
system administrator, or manufacturer.
For example: "utf-8" (ISO 10646/Unicode) or "iso-8859-1" (Latin1).
```

```
Legal values are defined by the IANA Registry of Coded Character Sets and
the "(preferred MIME name)" SHALL be used as the tag.
For coherence with IPP Model, charset tags in this attribute SHALL be
lowercase normalized.
This attribute SHOULD be static (time of registration) and SHOULD NOT be
dynamically refreshed attributetypes: (subsequently).'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} SINGLE-VALUE )

( 1.3.18.0.2.4.1131
NAME 'printer-charset-supported'
DESC 'Identifies the set of charsets supported for attribute type values of
type Directory String for this directory entry.
For example: "utf-8" (ISO 10646/Unicode) or "iso-8859-1" (Latin1).
Legal values are defined by the IANA Registry of Coded Character Sets and
the preferred MIME name.'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} )

( 1.3.18.0.2.4.1137
NAME 'printer-generated-natural-language-supported'
DESC 'Identifies the natural language(s) supported for this directory entry.
For example: "en-us" (US English) or "fr-fr" (French in France).
Legal values conform to [RFC3066], Tags for the Identification of Languages.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{63} )

( 1.3.18.0.2.4.1130
NAME 'printer-document-format-supported'
DESC 'The possible document formats in which data may be interpreted
and printed by this printer.
Legal values are MIME types come from the IANA Registry of Internet Media Types.'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1129
NAME 'printer-color-supported'
DESC 'Indicates whether this printer is capable of any type of color printing
at all, including highlight color.'
EQUALITY booleanMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.7  SINGLE-VALUE )

( 1.3.18.0.2.4.1128
NAME 'printer-compression-supported'
DESC 'Compression algorithms supported by this printer.
For example: "deflate, gzip".  Legal values include; "none", "deflate"
attributetypes: (public domain ZIP), "gzip" (GNU ZIP), "compress" (UNIX).'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1127
NAME 'printer-pages-per-minute'
DESC 'The nominal number of pages per minute which may be output by this
printer (e.g., a simplex or black-and-white printer).
This attribute is informative, NOT a service guarantee.
Typically, it is the value used in marketing literature to describe this printer.'
```

```
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1126 NAME 'printer-pages-per-minute-color'
DESC 'The nominal number of color pages per minute which may be output by this
printer (e.g., a simplex or color printer).
This attribute is informative, NOT a service guarantee.
Typically, it is the value used in marketing literature to describe this printer.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1125 NAME 'printer-finishings-supported'
DESC 'The possible finishing operations supported by this printer.
Legal values include; "none", "staple", "punch", "cover", "bind", "saddle-stitch",
"edge-stitch", "staple-top-left", "staple-bottom-left", "staple-top-right",
"staple-bottom-right", "edge-stitch-left", "edge-stitch-top", "edge-stitch-right",
"edge-stitch-bottom", "staple-dual-left", "staple-dual-top", "staple-dual-right",
"staple-dual-bottom".'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1124 NAME 'printer-number-up-supported'
DESC 'The possible numbers of print-stream pages to impose upon a single side of
an instance of a selected medium. Legal values include; 1, 2, and 4.
Implementations may support other values.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27 )

( 1.3.18.0.2.4.1123 NAME 'printer-sides-supported'
DESC 'The number of impression sides (one or two) and the two-sided impression
rotations supported by this printer.
Legal values include; "one-sided", "two-sided-long-edge", "two-sided-short-edge".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1122 NAME 'printer-media-supported'
DESC 'The standard names/types/sizes (and optional color suffixes) of the media
supported by this printer.
For example: "iso-a4",  "envelope", or "na-letter-white".
Legal values  conform to ISO 10175, Document Printing Application (DPA), and any
IANA registered extensions.'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1117 NAME 'printer-media-local-supported'
DESC 'Site-specific names of media supported by this printer, in the language in
"printer-natural-language-configured".
For example: "purchasing-form" (site-specific name) as opposed to
(in "printer-media-supported"): "na-letter" (standard keyword from ISO 10175).'
EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )
```

```
( 1.3.18.0.2.4.1121 NAME 'printer-resolution-supported'
DESC 'List of resolutions supported for printing documents by this printer.
Each resolution value is a string with 3 fields:
1) Cross feed direction resolution (positive integer), 2) Feed direction
resolution (positive integer), 3) Resolution unit.
Legal values are "dpi" (dots per inch) and "dpcm" (dots per centimeter).
Each resolution field is delimited by ">".  For example:  "300> 300> dpi>".'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{255} )

( 1.3.18.0.2.4.1120 NAME 'printer-print-quality-supported'
DESC 'List of print qualities supported for printing documents on this printer.
For example: "draft, normal".  Legal values include; "unknown", "draft", "normal",
"high".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1110 NAME 'printer-job-priority-supported'
DESC 'Indicates the number of job priority levels supported.
An IPP conformant printer which supports job priority must always support a
full range of priorities from "1" to "100"
(to ensure consistent behavior), therefore this attribute describes the
"granularity".
 Legal values of this attribute are from "1" to "100".'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1118
NAME 'printer-copies-supported'
DESC 'The maximum number of copies of a document that may be printed as a single job.
A value of "0" indicates no maximum limit.
A value of "-1" indicates unknown.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1111
NAME 'printer-job-k-octets-supported'
DESC 'The maximum size in kilobytes (1,024 octets actually) incoming print job that
this printer will accept.
A value of "0" indicates no maximum limit.  A value of "-1" indicates unknown.'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.27  SINGLE-VALUE )

( 1.3.18.0.2.4.1113
NAME 'printer-service-person'
DESC 'The name of the current human service person responsible for servicing this
printer.
It is suggested that this string include information that would enable other humans
to reach the service person, such as a phone number.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127}
SINGLE-VALUE )
```

```
( 1.3.18.0.2.4.1114
NAME 'printer-delivery-orientation-supported'
DESC 'The possible delivery orientations of pages as they are printed and ejected
from this printer.
Legal values include; "unknown", "face-up", and "face-down".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1115
NAME 'printer-stacking-order-supported'
DESC 'The possible stacking order of pages as they are printed and ejected from
this printer.
Legal values include; "unknown", "first-to-last", "last-to-first".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1116
NAME 'printer-output-features-supported'
DESC 'The possible output features supported by this printer.
Legal values include; "unknown", "bursting", "decollating", "page-collating",
"offset-stacking".'
EQUALITY caseIgnoreMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.18.0.2.4.1108
NAME 'printer-aliases'
DESC 'Site-specific administrative names of this printer in addition the printer
name specified for printer-name.'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15{127} )

( 1.3.6.1.4.1.42.2.27.5.1.63
NAME 'sun-printer-bsdaddr'
DESC 'Sets the server, print queue destination name and whether the client generates
protocol extensions.
"Solaris" specifies a Solaris print server extension. The value is represented b the
following value: server "," destination ", Solaris".'
SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' SINGLE-VALUE )

( 1.3.6.1.4.1.42.2.27.5.1.64
NAME 'sun-printer-kvp'
DESC 'This attribute contains a set of key value pairs which may have meaning to the
print subsystem or may be user defined.
Each value is represented by the following: key "=" value.'
SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )
```

Internet Print Protocol `ObjectClasses`

```
objectclasses: ( 1.3.18.0.2.6.2549
NAME 'slpService'
DESC 'DUMMY definition'
SUP 'top' MUST (objectclass) MAY ())
```

```
objectclasses: ( 1.3.18.0.2.6.254
NAME 'slpServicePrinter'
DESC 'Service Location Protocol (SLP) information.'
AUXILIARY SUP 'slpService')

objectclasses: ( 1.3.18.0.2.6.258
NAME 'printerAbstract'
DESC 'Printer related information.'
ABSTRACT SUP 'top' MAY ( printer-name
$ printer-natural-language-configured
$ printer-location
$ printer-info
$ printer-more-info
$ printer-make-and-model
$ printer-multiple-document-jobs-supported
$ printer-charset-configured
$ printer-charset-supported
$ printer-generated-natural-language-supported
$ printer-document-format-supported
$ printer-color-supported
$ printer-compression-supported
$ printer-pages-per-minute
$ printer-pages-per-minute-color
$ printer-finishings-supported
$ printer-number-up-supported
$ printer-sides-supported
$ printer-media-supported
$ printer-media-local-supported
$ printer-resolution-supported
$ printer-print-quality-supported
$ printer-job-priority-supported
$ printer-copies-supported
$ printer-job-k-octets-supported
$ printer-current-operator
$ printer-service-person
$ printer-delivery-orientation-supported
$ printer-stacking-order-supported $ printer! -output-features-supported ))

objectclasses: ( 1.3.18.0.2.6.255
NAME 'printerService'
DESC 'Printer information.'
STRUCTURAL SUP 'printerAbstract' MAY ( printer-uri
$ printer-xri-supported ))

objectclasses: ( 1.3.18.0.2.6.257
NAME 'printerServiceAuxClass'
DESC 'Printer information.'
AUXILIARY SUP 'printerAbstract' MAY ( printer-uri $ printer-xri-supported ))

objectclasses: ( 1.3.18.0.2.6.256
NAME 'printerIPP'
DESC 'Internet Printing Protocol (IPP) information.'
AUXILIARY SUP 'top' MAY   ( printer-ipp-versions-supported $
printer-multiple-document-jobs-supported ))

objectclasses: ( 1.3.18.0.2.6.253
NAME 'printerLPR'
```

```
DESC 'LPR information.'
AUXILIARY SUP 'top' MUST ( printer-name ) MAY ( printer-aliases))

objectclasses: ( 1.3.6.1.4.1.42.2.27.5.2.14
NAME 'sunPrinter'
DESC 'Sun printer information'
SUP 'top' AUXILIARY MUST (objectclass $ printer-name)  MAY
(sun-printer-bsdaddr $ sun-printer-kvp))
```

Printer Attributes

```
ATTRIBUTE ( 1.3.6.1.4.1.42.2.27.5.1.63
NAME sun-printer-bsdaddr
DESC 'Sets the server, print queue destination name and whether the
     client generates protocol extensions. "Solaris" specifies a
     Solaris print server extension.  The value is represented by
     the following value: server "," destination ", Solaris".'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
)


ATTRIBUTE ( 1.3.6.1.4.1.42.2.27.5.1.64
NAME sun-printer-kvp
DESC 'This attribute contains a set of key value pairs which may have
      meaning to the print subsystem or may be user defined.  Each
      value is represented by the following: key "=" value.'
EQUALITY caseIgnoreIA5Match
SYNTAX  1.3.6.1.4.1.1466.115.121.1.15  )
```

Sun Printer `ObjectClasses`

```
OBJECTCLASS ( 1.3.6.1.4.1.42.2.27.5.2.14
NAME sunPrinter
DESC 'Sun printer information'
SUP  top
AUXILIARY
MUST ( printer-name )
MAY  ( sun-printer-bsdaddr $ sun-printer-kvp ))
```

Generic Directory Server Requirements for LDAP

To support LDAP clients, all servers must support the LDAP v3 protocol and compound naming and auxiliary object classes. In addition, at least one of the following controls must be supported.

- Simple paged-mode (RFC 2696)
- Virtual List View controls

The server must support at least one of the following authentication methods.

```
anonymous
simple
sasl/cram-MD5
sasl/digest-MD5
sasl/GSSAPI
```

If an LDAP client is using the `pam_unix_*` modules, the server must support storing passwords in UNIX crypt format.

If an LDAP client is using TLS, the server must support SSL or TLS.

If an LDAP client is using `sasl/GSSAPI`, the server must support SASL, GSSAPI, Kerberos 5 authentication. Support for GSS encryption over the wire is optional.

Default Filters Used by LDAP Naming Services

If you do not manually specify a parameter for a given service using an SSD, the default filter is used. To list the default filters for a given service, use `ldaplist` with the `-v` option.

In the following example, `filter=(&(objectclass=iphost)(cn=abcde)` defines the default filters.

```
database=hosts
filter=(&(objectclass=iphost)(cn=abcde)
user data=(&(%s) (cn=abcde))
```

`ldaplist` generates the following list of default filters, where `%s` signifies a string and `%d`, a number.

```
hosts
(&(objectclass=iphost)(cn=%s))
--------------
passwd
(&(objectclass=posixaccount)(uid=%s))
--------------
services
(&(objectclass=ipservice)(cn=%s))
--------------
group
(&(objectclass=posixgroup)(cn=%s))
--------------
netgroup
(&(objectclass=nisnetgroup)(cn=%s))
--------------
networks
(&(objectclass=ipnetwork)(ipnetworknumber=%s))
--------------
```

```
netmasks
(&(objectclass=ipnetwork)(ipnetworknumber=%s))
--------------
rpc
(&(objectclass=oncrpc)(cn=%s))
--------------
protocols
(&(objectclass=ipprotocol)(cn=%s))
--------------
bootparams
(&(objectclass=bootableDevice)(cn=%s))
--------------
ethers
(&(objectclass=ieee802Device)(cn=%s))
--------------
publickey
(&(objectclass=niskeyobject)(cn=%s))
or
(&(objectclass=niskeyobject)(uidnumber=%d))
--------------
aliases
(&(objectclass=mailGroup)(cn=%s))
--------------
```

TABLE 14–4 LDAP Filters Used in getXbyY Calls

Filter	Definition
bootparamByName	(&(objectClass=bootableDevice)(cn=%s))
etherByHost	(&(objectClass=ieee802Device)(cn=%s))
etherByEther	(&(objectClass=ieee802Device)(macAddress=%s))
groupByName	(&(objectClass=posixGroup)(cn=%s))
groupByGID	(&(objectClass=posixGroup)(gidNumber=%ld))
groupByMember	(&(objectClass=posixGroup)(memberUid=%s))
hostsByName	(&(objectClass=ipHost)(cn=%s))
hostsByAddr	(&(objectClass=ipHost)(ipHostNumber=%s))
keyByUID	(&(objectClass=nisKeyObject)(uidNumber=%s))
keyByHost	(&(objectClass=nisKeyObject)(cn=%s))
netByName	(&(objectClass=ipNetwork)(cn=%s))
netByAddr	(&(objectClass=ipNetwork)(ipNetworkNumber=%s))
nisgroupMember	(membernisnetgroup=%s)
maskByNet	(&(objectClass=ipNetwork)(ipNetworkNumber=%s))
printerByName	(&(objectClass=sunPrinter)(|(printer-name=%s) (printer-aliases=%s)))

TABLE 14–4 LDAP Filters Used in getXbyY Calls *(Continued)*

Filter	Definition
projectByName	(&(objectClass=SolarisProject)(SolarisProjectName=%s))
projectByID	(&(objectClass=SolarisProject)(SolarisProjectID=%ld))
protoByName	(&(objectClass=ipProtocol)(cn=%s))
protoByNumber	(&(objectClass=ipProtocol)(ipProtocolNumber=%d))
passwordByName	(&(objectClass=posixAccount)(uid=%s))
passwordByNumber	(&(objectClass=posixAccount)(uidNumber=%ld))
rpcByName	(&(objectClass=oncRpc)(cn=%s))
rpcByNumber	(&(objectClass=oncRpc)(oncRpcNumber=%d))
serverByName	(&(objectClass=ipService)(cn=%s))
serverByPort	(&(objectClass=ipService)(ipServicePort=%ld))
serverByNameAndProto	(&(objectClass=ipService)(cn=%s)(ipServiceProtocol=%s))
specialByNameserver	(ipServiceProtocol=%s))
ByPortAndProto	(&(objectClass=shadowAccount)(uid=%s))
netgroupByTriple	(&(objectClass=nisNetGroup)(cn=%s))
netgroupByMember	(&(objectClass=nisNetGroup)(cn=%s))
authName	(&(objectClass=SolarisAuthAttr)(cn=%s))
auditUserByName	(&(objectClass=SolarisAuditUser)(uid=%s))
execByName	(&(objectClass=SolarisExecAttr)(cn=%s) (SolarisKernelSecurityPolicy=%s)(SolarisProfileType=%s))
execByPolicy	(&(objectClass=SolarisExecAttr)(SolarisProfileId=%s) (SolarisKernelSecurityPolicy=%s)(SolarisProfileType=%s))
profileByName	(&(objectClass=SolarisProfAttr)(cn=%s))
userByName	(&(objectClass=SolarisUserAttr)(uid=%s))

The following table lists the getent attribute filters.

TABLE 14–5 getent Attribute Filters

Filter	Definition
aliases	(objectClass=rfc822MailGroup)

TABLE 14–5 `getent` Attribute Filters *(Continued)*

Filter	Definition
`auth_attr`	`(objectClass=SolarisAuthAttr)`
`audit_user`	`(objectClass=SolarisAuditUser)`
`exec_attr`	`(objectClass=SolarisExecAttr)`
`group`	`(objectClass=posixGroup)`
`hosts`	`(objectClass=ipHost)`
`networks`	`(objectClass=ipNetwork)`
`prof_attr`	`(objectClass=SolarisProfAttr)`
`protocols`	`(objectClass=ipProtocol)`
`passwd`	`(objectClass=posixAccount)`
`printers`	`(objectClass=sunPrinter)`
`rpc`	`(objectClass=oncRpc)`
`services`	`(objectClass=ipService)`
`shadow`	`(objectclass=shadowAccount)`
`project`	`(objectClass=SolarisProject)`
`usr_attr`	`(objectClass=SolarisUserAttr)`

◆ ◆ ◆ CHAPTER 15

Transitioning From NIS to LDAP (Tasks)

This chapter describes how to enable support of NIS clients that use naming information stored in the LDAP directory. By following the procedures in this chapter, you can transition from using an NIS naming service to using LDAP naming services.

To determine the benefits of transitioning to LDAP, see "LDAP Naming Services Compared to Other Naming Services" on page 122.

The following information is included in this chapter:

NIS-to-LDAP Service Overview

The NIS-to-LDAP transition service (*N2L service*) replaces existing NIS daemons on the NIS master server with NIS-to-LDAP transition daemons. The N2L service also creates an NIS-to-LDAP mapping file on that server. The mapping file specifies the mapping between NIS map entries and equivalent Directory Information Tree (DIT) entries in LDAP. An NIS master server that has gone through this transition is referred to as an *N2L server*. The slave servers do not have an `NISLDAPmapping` file, so they continue to function in the usual manner. The slave servers periodically update their data from the N2L server as if it were a regular NIS master.

The behavior of the N2L service is controlled by the `ypserv` and `NISLDAPmapping` configuration files. A script, `inityp2l`, assists with the initial setup of these configuration files. Once the N2L server has been established, you can maintain N2L by directly editing the configuration files.

The N2L service supports the following:

- Import of NIS maps into the LDAP Directory Information Tree (DIT)
- Client access to DIT information with the speed and extensibility of NIS

In any naming system, only one source of information can be the authoritative source. In traditional NIS, NIS sources are the authoritative information. When using the N2L service, the source of authoritative data is the LDAP directory. The directory is managed by using directory management tools, as described in Chapter 9, "Introduction to LDAP Naming Services (Overview)."

NIS sources are retained for emergency backup or backout only. After you use the N2L service, you must phase out NIS clients. Eventually, all NIS clients should be replaced by LDAP naming services clients.

Additional overview information is provided in the following subsections:

NIS-to-LDAP Tools and the Service Management Facility

The NIS and LDAP services are managed by the Service Management Facility. Administrative actions on these services, such as enabling, disabling, or restarting, can be performed by using the `svcadm` command. You can query the status of services by using the `svcs` command. For more information about using SMF with LDAP and NIS, see "LDAP and the Service Management Facility" on page 174 and "NIS and the Service Management Facility" on page 74. For an overview of SMF, refer to Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks*. Also refer to the `svcadm`(1M) and `svcs`(1) man pages for more details.

NIS-to-LDAP Audience Assumptions

You need to be familiar with NIS and LDAP concepts, terminology, and IDs to perform the procedures in this chapter. For more information about the NIS and LDAP naming services, see the following sections of this book.

- Chapter 5, "Network Information Service (Overview)," for an overview of NIS
- Chapter 9, "Introduction to LDAP Naming Services (Overview)," for an overview of LDAP

When Not to Use the NIS-to-LDAP Service

The intent of the N2L service is to serve as a transition tool from using NIS to using LDAP. Do not use the N2L service in these situations:

- In an environment where there is no plan to share data between NIS and LDAP naming services clients

  In such an environment, an N2L server would serve as an excessively complex NIS master server.

- In an environment where NIS maps are managed by tools that modify the NIS source files (other than `yppasswd`)

  Regeneration of NIS sources from DIT maps is an imprecise task that requires manual checking of the resulting maps. Once the N2L service is used, regeneration of NIS sources is provided only for backout or reverting to NIS.

- In an environment with no NIS clients

  In such an environment, use LDAP naming services clients and their corresponding tools.

Effects of the NIS-to-LDAP Service on Users

Simply installing the files that are related to the N2L service does not change the NIS server's default behavior. At installation, the administrator will see some changes to NIS man pages and the addition of N2L helper scripts, `inityp2l` and `ypmap2src`, on the servers. But as long as `inityp2l` is not run or the N2L configuration files are not created manually on the NIS server, the NIS components continue to start in traditional NIS mode and function as usual.

After `inityp2l` is run, users see some changes in server and client behavior. Following is a list of NIS and LDAP user types and a description of what each type of user should notice after the N2L service is deployed.

User Type	Effect of N2L Service
NIS master server administrators	The NIS master server is converted to an N2L server. The `NISLDAPmapping` and `ypserv` configuration files are installed on the N2L server. After the N2L server is established, you can use LDAP commands to administer your naming information.
NIS slave server administrators	After the N2L transition, an NIS slave server continues to run NIS in the usual manner. The N2L server pushes updated NIS maps to the slave server when `yppush` is called by `ypmake`. See the `ypmake(1M)` man page.

User Type	Effect of N2L Service
NIS clients	NIS read operations are no different than traditional NIS. When an LDAP naming services client changes information in the DIT, the information is copied into the NIS maps. The copy operation is complete after a configurable timeout expires. Such behavior is similar to the behavior of a normal NIS client when the client is connected to an NIS slave server. If an N2L server cannot bind to the LDAP server for a read, the N2L server returns the information from its own cached copy. Alternatively, the N2L server can return an internal server error. You can configure the N2L server to respond either way. See the `ypserv(1M)` man page for more details.
All users	When an NIS client makes a password change request, the change is immediately visible on the N2L master server and to native LDAP clients. If you attempt to change a password on the NIS client, and the LDAP server is unavailable, then the change is refused and the N2L server returns an internal server error. This behavior prevents incorrect information from being written into the cache.

NIS-to-LDAP Transition Terminology

The following terms are related to the implementation of the N2L service.

TABLE 15–1 Terminology Related to the N2L Transition

Term	Description
N2L configuration files	The `/var/yp/NISLDAPmapping` and `/var/yp/ypserv` files that the `ypserv` daemon uses to start the master server in N2L mode. See the `NISLDAPmapping`(4) and `ypserv`(4) man pages for details.
map	In the context of the N2L service, the term map is used in two ways: ■ To refer to a database file in which NIS stores a specific type of information ■ To describe the process of mapping NIS information to or from the LDAP DIT
mapping	The process of converting NIS entries to or from LDAP DIT entries.
mapping file	The `NISLDAPmapping` file that establishes how to map entries between NIS and LDAP files.
standard maps	Commonly used NIS maps that are supported by the N2L service without requiring manual modification to the mapping file. A list of supported standard maps is provided in "Supported Standard Mappings" on page 225.
nonstandard maps	Standard NIS maps that are customized to use mappings between NIS and the LDAP DIT other than the mappings identified in RFC 2307 or its successor.
custom map	Any map that is not a standard map and therefore requires manual modifications to the mapping file when transitioning from NIS to LDAP.
LDAP client	Any traditional LDAP client that reads and writes to any LDAP server. A traditional LDAP client is a system that reads and writes to any LDAP server. An LDAP naming services client handles a customized subset of naming information.

TABLE 15–1 Terminology Related to the N2L Transition *(Continued)*

Term	Description
LDAP naming services client	An LDAP client that handles a customized subset of naming information.
N2L 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-to-LDAP Commands, Files, and Maps

There are two utilities, two configuration files, and a mapping that are associated with the N2L transition.

TABLE 15–2 Descriptions of N2L Commands, Files, and Maps

Command/File/Map	Description
`/usr/lib/netsvc/yp/inityp2l`	A utility that assists with the creation of the `NISLDAPmapping` and `ypserv` configuration files. This utility is not a general-purpose tool for the management of these files. An advanced user can maintain the N2L configuration files or create custom mappings by using a text editor to examine and customize the `inityp2l` output. See the `inityp2l(1M)` man page.
`/usr/lib/netsvc/yp/ypmap2src`	A utility that converts standard NIS maps to approximations of the equivalent NIS source files. The primary use for `ypmap2src` is to convert from an N2L transition server to traditional NIS. See the `ypmap2src(1M)` man page.
`/var/yp/NISLDAPmapping`	A configuration file that specifies the mapping between NIS map entries and equivalent Directory Information Tree (DIT) entries in LDAP. See the `NISLDAPmapping(4)` man page.
`/var/yp/ypserv`	A file that specifies configuration information for the NIS-to-LDAP transition daemons. See the `ypserv(4)` man page.
`ageing.byname`	A mapping used by `yppasswdd` to read and write password aging information to the DIT when the NIS-to-LDAP transition is implemented.

Supported Standard Mappings

By default, the N2L service supports mappings between the following list of maps and RFC 2307, RFC 2307bis, and their successors' LDAP entries. These standard maps do not require manual modification to the mapping file. Any maps on your system that are not in the following list are considered custom maps and require manual modification.

The N2L service also supports automatic mapping of the `auto.*` maps. However, since most `auto.*` file names and contents are specific to each network configuration, those files are not specified in this list. The exceptions to this are the `auto.home` and `auto.master` maps, which are supported as standard maps.

```
audit_user
auth_attr
auto.home
auto.master
bootparams
ethers.byaddr ethers.byname
exec_attr
group.bygid group.byname group.adjunct.byname
hosts.byaddr hosts.byname
ipnodes.byaddr ipnodes.byname
mail.byaddr mail.aliases
netgroup netgroup.byprojid netgroup.byuser netgroup.byhost
netid.byname
netmasks.byaddr
networks.byaddr networks.byname
passwd.byname passwd.byuid passwd.adjunct.byname
prof_attr
project.byname project.byprojectid
protocols.byname protocols.bynumber
publickey.byname
rpc.bynumber
services.byname services.byservicename
timezone.byname
user_attr
```

During the NIS-to-LDAP transition, the `yppasswdd` daemon uses the N2L-specific map, `ageing.byname`, to read and write password aging information to the DIT. If you are not using password aging, then the `ageing.byname` mapping is ignored.

Transitioning From NIS to LDAP (Task Map)

The following table identifies the procedures needed to install and manage the N2L service with standard and with custom NIS-to-LDAP mappings.

Task	Description	For Instructions
Complete all prerequisites.	Be sure that you have properly configured your NIS server and Oracle Directory Server Enterprise Edition (LDAP server).	"Prerequisites for the NIS-to-LDAP Transition" on page 227
Set up the N2L service.	Run `inityp2l` on the NIS master server to set up one of these mappings:	
	Standard mappings	"How to Set Up the N2L Service With Standard Mappings" on page 229
	Custom or nonstandard mappings	"How to Set Up the N2L Service With Custom or Nonstandard Mappings" on page 230
Customize a map.	View examples of how to create custom maps for the N2L transition.	"Examples of Custom Maps" on page 233

Task	Description	For Instructions
Configure Oracle Directory Server Enterprise Edition with N2L.	Configure and tune Oracle Directory Server Enterprise Edition as your LDAP server for the N2L transition.	"NIS-to-LDAP Best Practices With Oracle Directory Server Enterprise Edition" on page 234
Troubleshoot the system.	Identify and resolve common N2L issues.	"NIS-to-LDAP Troubleshooting" on page 237
Revert to NIS.	Revert to NIS using the appropriate map: Maps based on old NIS source files Maps based on the current DIT	 "How to Revert to Maps Based on Old Source Files" on page 242 "How to Revert to Maps Based on Current DIT Contents" on page 243

Prerequisites for the NIS-to-LDAP Transition

Before implementing the N2L service, you must check or complete the following items.

- Make sure that the system is set up as a working traditional NIS server before running the `inityp2l` script to enable N2L mode.
- Configure the LDAP directory server on your system.

  Oracle Directory Server Enterprise Edition and compatible versions of directory servers offered by Oracle, are supported with the NIS-to-LDAP migration tools. If you use Oracle Directory Server Enterprise Edition, configure the server by using the `idsconfig` command *before* you set up the N2L service. For more information about `idsconfig`, see Chapter 11, "Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)," and the `idsconfig(1M)` man page.

  Other (third-party) LDAP servers might work with the N2L service, but they are not supported by Oracle. If you are using an LDAP server other than the Oracle Directory Server Enterprise Edition or compatible Oracle servers, you must manually configure the server to support RFC 2307bis, RFC 4876, or their successors' schemas *before* you set up the N2L service.
- Use `files` before `dns` for the `config/host` property.
- Ensure that the addresses of the N2L master server and the LDAP server are present in the `hosts` file on the N2L master server.

  An alternative solution is to list the LDAP server address, not its host name, in `ypserv`. This means that the LDAP server address is listed in another place, so changing the address of either the LDAP server or the N2L master server requires additional file modifications.

Setting Up the NIS-to-LDAP Service

You can set up the N2L service either by using standard mappings or by using custom mappings, as described in the next two procedures.

As part of the NIS-to -LDAP conversion, you need to run the `inityp2l` command. This command runs an interactive script for which you must provide configuration information. The following list shows the type of information you need to provide. See the `ypserv(1M)` man page for explanations of these attributes.

- The name of the configuration file being created (default = `/etc/default/ypserv`)
- The DN that stores configuration information in LDAP (default = `ypserv`)
- Preferred server list for mapping data to/from LDAP
- Authentication method for mapping data to/from LDAP
- Transport Layer Security (TLS) method for mapping data to/from LDAP
- Proxy user bind DN to read/write data from/to LDAP
- Proxy user password to read/write data from/to LDAP
- Timeout value (in seconds) for LDAP bind operation
- Timeout value (in seconds) for LDAP search operation
- Timeout value (in seconds) for LDAP modify operation
- Timeout value (in seconds) for LDAP add operation
- Timeout value (in seconds) for LDAP delete operation
- Time limit (in seconds) for search operation on LDAP server
- Size limit (in bytes) for search operation on LDAP server
- Whether N2L should follow LDAP referrals
- LDAP retrieval error action, number of retrieval attempts, and timeout (in seconds) between each attempt
- Store error action, number of attempts, and timeout (in seconds) between each attempt
- Mapping file name
- Whether to generate mapping information for `auto_direct` map

  The script places relevant information regarding custom maps at appropriate places in the mapping file.
- The naming context
- Whether to enable password changes
- Whether to change the default TTL values for any map

Note – `sasl/cram-md5` authentication is *not* supported by most LDAP servers, including Oracle Directory Server Enterprise Edition.

How to Set Up the N2L Service With Standard Mappings

Use this procedure if you are transitioning the maps listed in "Supported Standard Mappings" on page 225. If you are using custom or nonstandard maps, see "How to Set Up the N2L Service With Custom or Nonstandard Mappings" on page 230.

When the LDAP server has been set up, run the `inityp2l` script and supply configuration information when prompted. `inityp2l` sets up the configuration and mapping files for standard and `auto.*` maps.

1. **Complete the prerequisite steps that are listed in "Prerequisites for the NIS-to-LDAP Transition" on page 227.**

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

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

3. **Convert the NIS master server into an N2L server.**

   ```
   # inityp2l
   ```

   Run the `inityp2l` script on the NIS master server and follow the prompts. See "Setting Up the NIS-to-LDAP Service" on page 228 for a list of the information you need to provide.

   See the `inityp2l(1M)` man page for more details.

4. **Determine if the LDAP Directory Information Tree (DIT) is fully initialized.**

   The DIT is fully initialized if it already contains the information necessary to populate all the maps that are listed in the `NISLDAPmapping` file.

   - If no, continue with Step 5 and skip Step 6.
   - If yes, skip Step 5 and go to Step 6.

5. **Initialize the DIT for the transition from the NIS source files.**

   Perform these steps only if the DIT has *not* been fully initialized.

   a. **Make sure that the old NIS maps are up-to-date.**

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

For more information, see the ypmake(1M) man page.

b. Stop the NIS service

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

c. Copy the old maps to the DIT, then initialize N2L support for the maps.

```
# ypserv -IR
```

Wait for ypserv to exit.

Tip – The original NIS dbm files are not overwritten. You can recover these files, if needed.

---

d. Start the DNS and NIS services to ensure that they use the new maps.

```
# svcadm enable network/dns/client:default
# svcadm enable network/nis/server:default
```

This completes the set up of the N2L service with standard maps. You do not need to complete Step 6.

6 Initialize the NIS maps.

Perform these steps only if the DIT is fully initialized and you skipped Step 5.

a. Stop the NIS service.

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

b. Initialize the NIS maps from information in the DIT.

```
# ypserv -r
```

Wait for ypserv to exit.

Tip – The original NIS dbm files are not overwritten. You can recover these files, if needed.

---

c. Start the DNS and NIS service to ensure that they use the new maps.

```
# svcadm enable network/dns/client:default
# svcadm enable network/nis/server:default
```

▼ How to Set Up the N2L Service With Custom or Nonstandard Mappings

Use this procedure if the following circumstances apply:

- You have maps that are not listed in "Supported Standard Mappings" on page 225.
- You have standard NIS maps that you want to map to non-RFC 2307 LDAP mappings.

1. **Complete the prerequisite steps that are listed in "Prerequisites for the NIS-to-LDAP Transition" on page 227.**

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

   For more information , see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

   Roles contain authorizations and privileged commands. For more information about roles, see Chapter 9, "Using Role-Based Access Control (Tasks)," in *Oracle Solaris Administration: Security Services*.

3. **Configure the NIS master server into the N2L server.**

   ```
   # inityp2l
   ```

   Run the `inityp2l` script on the NIS master server and follow the prompts. See "Setting Up the NIS-to-LDAP Service" on page 228 for a list of the information you need to provide.

   See the `inityp2l`(1M) man page for more details.

4. **Modify the `/var/yp/NISLDAPmapping` file.**

   See "Examples of Custom Maps" on page 233 for examples of how to modify the mapping file.

5. **Determine if the LDAP Directory Information Tree (DIT) is fully initialized.**

   The DIT is fully initialized if it already contains the information necessary to populate all the maps that are listed in the `NISLDAPmapping` file.

   - If no, complete Step 6, Step 8, and Step 9.
   - If yes, skip Step 6 and complete Step 7, Step 8, and Step 9.

6. **Initialize the DIT for the transition from the NIS source files.**

   a. **Make sure that the old NIS maps are up-to-date.**

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

      For more information, see the `ypmake`(1M) man page.

   b. **Stop the NIS daemons.**

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

   c. **Copy the old maps to the DIT, then initialize N2L support for the maps.**

      ```
      # ypserv -Ir
      ```

      Wait for `ypserv` to exit.

Tip – The original NIS `dbm` files are not overwritten. You can recover these files, if needed.

d. Start the DNS and NIS service to ensure that they use the new maps.

```
# svcadm enable network/dns/client:default
# svcadm enable network/nis/server:default
```

e. Skip Step 7 and continue with Step 8.

7 Initialize the NIS maps.

Perform this step only if the DIT is fully initialized.

a. Stop the NIS daemons.

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

b. Initialize the NIS maps from information in the DIT.

```
# ypserv -r
```

Wait for `ypserv` to exit.

Tip – The original NIS `dbm` files are not overwritten. You can recover these files, if needed.

c. Start the DNS and NIS service to ensure that they use the new maps.

```
# svcadm enable network/dns/client:default
# svcadm enable network/nis/server:default
```

8 Verify that the LDAP entries are correct.

If the entries are not correct, then the entries can not be found by LDAP naming services clients.

```
# ldapsearch -h server -s sub -b "ou=servdates, dc=..." \ "objectclass=servDates"
```

9 Verify the contents of the `LDAP_` maps.

The following sample output shows how to use the `makedm` command to verify the contents of the `hosts.byaddr` map.

```
# makedbm -u LDAP_servdate.bynumber
plato: 1/3/2001
johnson: 2/4/2003,1/3/2001
yeats: 4/4/2002
poe: 3/3/2002,3/4/2000
```

If the contents are as expected, the transition from NIS to LDAP was successful.

Note that the original NIS `dbm` files are not overwritten, so you can always recover those files. See "Reverting to NIS" on page 242 for more information.

Examples of Custom Maps

The following two examples show how you might customize maps. Use your preferred text editor to modify the `/var/yp/NISLDAPmapping` file as needed. For more information about file attributes and syntax, see the `NISLDAPmapping(4)` man page and the LDAP naming services information in Chapter 9, "Introduction to LDAP Naming Services (Overview)."

EXAMPLE 15–1 Moving Host Entries

This example shows how to move host entries from the default location to another (nonstandard) location in the DIT.

Change the `nisLDAPobjectDN` attribute in the `NISLDAPmapping` file to the new base LDAP distinguished name (DN). For this example, the internal structure of the LDAP objects is unchanged, so `objectClass` entries are unchanged.

Change:

```
nisLDAPobjectDN hosts: \
                        ou=hosts,?one?, \
                        objectClass=device, \
                        objectClass=ipHost
```

To:

```
nisLDAPobjectDN hosts: \
                        ou=newHosts,?one?, \
                        objectClass=device, \
                        objectClass=ipHost
```

This change causes entries to be mapped under

```
dn: ou=newHosts, dom=domain1, dc=sun, dc=com
```

instead of under

```
dn: ou=hosts, dom=domain1, dc=sun, dc=com.
```

EXAMPLE 15–2 Implementing a Custom Map

This example shows how to implement a custom map.

A hypothetical map, *servdate.bynumber*, contains information about the servicing dates for systems. This map is indexed by the machine's serial number which, in this example, is 123. Each entry consists of the machine owner's name, a colon, and a comma-separated list of service dates, such as John Smith:1/3/2001,4/5/2003.

The old map structure is to be mapped onto LDAP entries of the following form:

```
dn: number=123,ou=servdates,dc=... \
                number: 123 \
                userName: John Smith \
```

EXAMPLE 15–2 Implementing a Custom Map *(Continued)*

```
                date: 1/3/2001 \
                date: 4/5/2003 \
                 .
                 .
                 .
                objectClass: servDates
```

By examining the `NISLDAPmapping` file, you can see that the mapping closest to the required pattern is `group`. The custom mappings can be modeled on the `group` mapping. Since there is only one map, no `nisLDAPdatabaseIdMapping` attribute is required. The attributes to be added to `NISLDAPmapping` are the following:

```
nisLDAPentryTtl servdate.bynumber:1800:5400:3600

nisLDAPnameFields servdate.bynumber: \
                        ("%s:%s", uname, dates)

nisLDAPobjectDN servdate.bynumber: \
                        ou=servdates, ?one? \
                        objectClass=servDates:

nisLDAPattributeFromField servdate.bynumber: \
                        dn=("number=%s,", rf_key), \
                        number=rf_key, \
                        userName=uname, \
                        (date)=(dates, ",")

nisLDAPfieldFromAttribute servdate.bynumber: \
                        rf_key=number, \
                        uname=userName, \
                        dates=("%s,", (date), ",")
```

NIS-to-LDAP Best Practices With Oracle Directory Server Enterprise Edition

The N2L service supports Oracle Directory Server Enterprise Edition. Other third-party LDAP servers might work with the N2L service, but they are not supported by Oracle. If you are using an LDAP server other than the Oracle Directory Server Enterprise Edition server or compatible Oracle servers, you must manually configure the server to support RFC 2307, RFC 2307bis and RFC 4876, or their successors' schemas.

If you are using the Oracle Directory Server Enterprise Edition, you can enhance the directory server to improve performance. To make these enhancements, you must have LDAP administrator privileges on the Oracle Directory Server Enterprise Edition. In addition, the directory server might need to be rebooted, a task that must be coordinated with the server's LDAP clients. The Oracle Directory Server Enterprise Edition documentation is available on the Sun Java System Directory Server Enterprise Edition 6.2 web site.

Creating Virtual List View Indexes With Oracle Directory Server Enterprise Edition

For large maps, LDAP virtual list view (VLV) indexes must be used to ensure LDAP searches return complete results. For information about setting up VLV indexes on the Oracle Directory Server Enterprise Edition, see the Sun Java System Directory Server Enterprise Edition 6.2 documentation.

VLV search results use a fixed page size of 50000. If VLVs are used with Oracle Directory Server Enterprise Edition, both the LDAP server and N2L server must be able to handle transfers of this size. If all of your maps are known to be smaller than this limit, you do not need to use VLV indexes. However, if your maps are larger than the size limit, or you are unsure of the size of all maps, use VLV indexes to avoid incomplete returns.

If you are using VLV indexes, set up the appropriate size limits as follows.

- On the Oracle Directory Server Enterprise Edition: `nsslapd-sizelimit` attribute must be set greater than or equal to 50000 or `-1`. See the `idsconfig(1M)` man page.
- On the N2L server: `nisLDAPsearchSizelimit` attribute must be set greater than or equal to 50000 or zero. For more information, see the `NISLDAPmapping(4)` man page.

After VLV indexes have been created, activate them by running `dsadm` with the `vlvindex` option on the Oracle Directory Server Enterprise Edition server. See the `dsadm`(1M) man page for more information.

VLVs for Standard Maps

Use the Oracle Directory Server Enterprise Edition `idsconfig` command to set up VLVs if the following conditions apply:

- You are using Oracle Directory Server Enterprise Edition.
- You are mapping standard maps to RFC 2307bis LDAP entries.

VLVs are domain specific, so each time `idsconfig` is run, VLVs are created for one NIS domain. Therefore, during the NIS-to-LDAP transition, you must run `idsconfig` once for *each* `nisLDAPdomainContext` attribute included in the `NISLDAPmapping` file.

VLVs for Custom and Nonstandard Maps

You must manually create new Oracle Directory Server Enterprise Edition VLVs for maps, or copy and modify existing VLV indexes, if the following conditions apply:

- You are using the Oracle Directory Server Enterprise Edition.
- You have large custom maps or have standard maps that are mapped to nonstandard DIT locations.

To view existing VLV indexes, type the following:

```
# ldapsearch -h hostname -s sub -b "cn=ldbm database,cn=plugins,cn=config" "objectClass=vlvSearch"
```

Avoiding Server Timeouts With Oracle Directory Server Enterprise Edition

When the N2L server refreshes a map, the result might be a large LDAP directory access. If the Oracle Directory Server Enterprise Edition is not correctly configured, the refresh operation might time out before completion. To avoid directory server timeouts, modify the following Oracle Directory Server Enterprise Edition attributes manually or by running the `idsconfig` command.

For example, to increase the minimum amount of time in seconds that the server should spend performing the search request, modify these attributes:

```
dn: cn=config
nsslapd-timelimit: -1
```

For testing purposes, you can use an attribute value of `-1`, which indicates no limit. When you have determined the optimum limit value, change the attribute value. Do *not* maintain any attribute settings at `-1` on a production server. With no limits, the server might be vulnerable to Denial of Service attacks.

For more information about configuring Oracle Directory Server Enterprise Edition with LDAP, see Chapter 11, “Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks),” of this book.

Avoiding Buffer Overruns With Oracle Directory Server Enterprise Edition

To avoid buffer overruns, modify the Oracle Directory Server Enterprise Edition attributes manually or by running the `idsconfig` command.

1. For example, to increase the maximum number of entries that are returned for a client search query, modify these attributes:

   ```
   dn: cn=config
   nsslapd-sizelimit: -1
   ```

2. To increase the maximum number of entries that are verified for a client search query, modify these attributes:

   ```
   dn: cn=config, cn=ldbm database, cn=plugins, cn=config
   nsslapd-lookthroughlimit: -1
   ```

For testing purposes, you can use an attribute value of `-1`, which indicates no limit. When you have determined the optimum limit value, change the attribute value. Do *not* maintain any attribute settings at `-1` on a production server. With no limits, the server might be vulnerable to Denial of Service attacks.

If VLVs are being used, the `sizelimit` attribute values should be set as defined in "Creating Virtual List View Indexes With Oracle Directory Server Enterprise Edition" on page 235. If VLVs are not being used, the size limit should be set large enough to accommodate the largest container.

For more information about configuring Oracle Directory Server Enterprise Edition with LDAP, see Chapter 11, "Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)."

NIS-to-LDAP Restrictions

When the N2L server has been set up, the NIS source files are no longer used. Therefore, do not run `ypmake` on an N2L server. If `ypmake` is accidentally run, such as for an existing `cron` job, the N2L service is unaffected. However, a warning is logged suggesting that `yppush` should be called explicitly.

NIS-to-LDAP Troubleshooting

This section covers two areas of troubleshooting:

Common LDAP Error Messages

Sometimes the N2L server logs errors that relate to internal LDAP problems, resulting in LDAP-related error messages. Although the errors are nonfatal, they indicate problems to investigate. For example, the N2L server might continue to operate, but provide out-of-date or incomplete results.

The following list includes some of the common LDAP error messages that you might encounter when implementing the N2L service. Error descriptions, and possible causes and solutions for the errors, are included.

`Administrative limit exceeded`

Error Number: 11

Cause: An LDAP search was made that was larger than allowed by the directory server's `nsslapd-sizelimit` attribute. Only partial information will be returned.

Solution: Increase the value of the `nsslapd-sizelimit` attribute, or implement a VLV index for the failing search.

`Invalid DN Syntax`

Error Number: 34

Cause: An attempt has been made to write an LDAP entry with a DN that contains illegal characters. The N2L server attempts to escape illegal characters, such as the + symbol, that are generated in DNs.

Solution: Check the LDAP server error log to find out which illegal DNs were written, then modify the `NISLDAPmapping` file that generated the illegal DNs.

`Object class violation`

Error Number: 65

Cause: An attempt has been made to write an LDAP entry that is invalid. Generally, this error is due to missing `MUST` attributes that can be caused by either of the following circumstances.

- Bugs in the `NISLDAPmapping` file that create entries with missing attributes
- Attempts to add an `AUXILIARY` attribute to an object that does not exist

  For example, if a user name has not yet been created from the `passwd.byxxx` map, an attempt to add auxiliary information to that user will fail.

Solution: For bugs in the `NISLDAPmapping` file, check what was written in the server error log to determine the nature of the problem.

`Can't contact LDAP server`

Error Number: 81

Cause: The `ypserv` file might be incorrectly configured to point to the wrong LDAP directory server. Alternatively, the directory server might not be running.

Solution:

- Reconfigure the `ypserv` file to point to the correct LDAP directory server.
- To confirm that the LDAP server is running, become superuser, or assume an equivalent role, on the directory server and type:

  ```
  # pgrep -l slapd
  ```

`Timeout`

Error Number: 85

Cause: An LDAP operation timed out, typically while updating a map from the DIT. The map might now contain out-of-date information.

Solution: Increase the `nisLDAPxxxTimeout` attributes in the `ypserv` configuration file.

NIS-to-LDAP Issues

The following problems could occur while running the N2L server. Possible causes and solutions are provided.

Debugging the `NISLDAPmapping` File

The mapping file, `NISLDAPmapping`, is complex. Many potential errors might cause the mapping to behave in unexpected ways. Use the following techniques to resolve such problems.

Console Message Displays When `ypserv -ir` (or `-Ir`) Runs

Problem: A simple message is displayed on the console and the server exits (a detailed description is written to `syslog`).

Cause: The syntax of the mapping file might be incorrect.

Solution: Check and correct the syntax in the `NISLDAPmapping` file.

NIS Daemon Exits at Startup

Problem: When `ypserv` or other NIS daemons run, an LDAP-related error message is logged and the daemon exits.

Cause: The cause might be one of the following:

- The LDAP server cannot be contacted.
- An entry found in an NIS map or in the DIT is incompatible with the mapping specified.
- An attempt to read or write to the LDAP server returns an error.

Solution: Examine the error log on the LDAP server. See the LDAP errors that are listed in "Common LDAP Error Messages" on page 237.

Unexpected Results From NIS Operations

Problem: NIS operations do not return the expected results, but no errors are logged.

Cause: Incorrect entries might exist in the LDAP or NIS maps, which results in mappings not completing as intended.

Solution: Check and correct entries in the LDAP DIT and in the N2L versions of the NIS maps.

1. Check that the correct entries exist in the LDAP DIT, and correct the entries as needed.

   If you are using Oracle Directory Server Enterprise Edition, start the management console by running the `dsadm startconsole` command.

2. Check that the N2L versions of the NIS maps in the `/var/yp` directory contain the expected entries by comparing the newly generated map to the original map. Correct entries as needed.

```
# cd /var/yp/domainname
# makedbm -u test.byname
# makedbm -u test.byname
```

Be aware of the following when checking the output for the maps:

- The order of entries might not be the same in both files.

  Use the `sort` command before comparing output.

- The use of white space might not be the same in both files.

  Use the `diff -b` command when comparing output.

Processing Order of NIS Maps

Problem: Object class violations occur.

Cause: When the `ypserv -i` command is run, each NIS map is read and its contents are written into the DIT. Several maps might contribute attributes to the same DIT object. Generally, one map creates most of the object, including all the object's `MUST` attributes. Other maps contribute additional `MAY` attributes.

Maps are processed in the same order that `nisLDAPobjectDN` attributes appear in the `NISLDAPmapping` file. If maps containing `MAY` attributes get processed before maps containing `MUST` attributes, then object class violations occur. See Error 65 in "Common LDAP Error Messages" on page 237 for more information about this error.

Solution: Reorder the `nisLDAPobjectDN` attributes so that maps are processed in the correct order.

As a temporary fix, rerun the `ypserv -i` command several times. Each time the command is executed, more of the LDAP entry is built up.

Note – Mapping in such a way that all of an object's `MUST` attributes cannot be created from at least one map is *not* supported.

N2L Server Timeout Issue

Problem: The server times out.

Cause: When the N2L server refreshes a map, the result might be a large LDAP directory access. If the Oracle Directory Server Enterprise Edition is not correctly configured, this operation might time out before completion.

Solution: To avoid directory server timeouts, modify the Oracle Directory Server Enterprise Edition attributes manually or by running the `idsconfig` command. See "Common LDAP Error Messages" on page 237 and "NIS-to-LDAP Best Practices With Oracle Directory Server Enterprise Edition" on page 234 for details.

N2L Lock File Issue

Problem: The `ypserv` command starts but does not respond to NIS requests.

Cause: The N2L server lock files are not correctly synchronizing access to the NIS maps. This should never happen.

Solution: Type the following commands on the N2L server.

```
# svcadm disable network/nis/server:default
# rm /var/run/yp_maplock /var/run/yp_mapupdate
# svcadm enable network/nis/server:default
```

N2L Deadlock Issue

Problem: The N2L server deadlocks.

Cause: If the addresses of the N2L master server and the LDAP server are not listed properly in the `hosts`, `ipnodes`, or `ypserv` files, a deadlock might result. See "Prerequisites for the NIS-to-LDAP Transition" on page 227 for details about proper address configuration for N2L.

For an example of a deadlock scenario, consider the following sequence of events:

1. An NIS client tries to look up an IP address.
2. The N2L server finds that the `hosts` entry is out-of-date.
3. The N2L server tries to update the `hosts` entry from LDAP.
4. The N2L server gets the name of its LDAP server from `ypserv`, then does a search by using `libldap`.
5. `libldap` tries to convert the LDAP server's name to an IP address by making a call to the name service switch.
6. The name service switch might make an NIS call to the N2L server, which deadlocks.

Solution: List the addresses of the N2L master server and the LDAP server in the `hosts` or `ipnodes` files on the N2L master server. Whether the server addresses must be listed in `hosts`, `ipnodes`, or both files depends on how these files are configured to resolve local host names. Also, check that the `config/hosts` property of the `svc:/network/name-service/switch` service lists `files` before `nis` in the lookup order.

An alternative solution to this deadlock problem is to list the LDAP server address, not its host name, in the `ypserv` file. This means that the LDAP server address would be listed in another place. Therefore, changing the address of either the LDAP server or the N2L server would require slightly more effort.

Reverting to NIS

A site that has transitioned from NIS to LDAP using the N2L service is expected to gradually replace all NIS clients with LDAP naming services clients. Support for NIS clients eventually becomes redundant. However, if required, the N2L service provides two ways to return to traditional NIS, as explained in the next two procedures.

Tip – Traditional NIS ignores the N2L versions of the NIS maps if those maps are present. After reverting to NIS, if you leave the N2L versions of the maps on the server, the N2L maps do not cause problems. Therefore, it might be useful to keep the N2L maps in case you later decide to re-enable N2L. However, the maps do take up disk space.

▼ How to Revert to Maps Based on Old Source Files

1. **Become an administrator.**

   For more information , see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Stop the `NIS` daemons.**

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

3. **Disable N2L.**

   This command backs up and moves the N2L mapping file.

   ```
   # mv /var/yp/NISLDAPmapping backup_filename
   ```

4. **Set the `NOPUSH` environment variable so the new maps are not pushed by `ypmake`.**

   ```
   # NOPUSH=1
   ```

5. **Make a new set of NIS maps that are based on the old sources.**

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

6. **(Optional) Remove N2L versions of the NIS maps.**

   ```
   # rm /var/yp/domainname/LDAP_*
   ```

7. **Start the DNS and the NIS service.**

   ```
   # svcadm enable network/dns/client:default
   # svcadm enable network/nis/server:default
   ```

How to Revert to Maps Based on Current DIT Contents

Back up the old NIS source files before performing this procedure.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Stop the `NIS` daemons.**

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

3. **Update the maps from the DIT.**

   ```
   # ypserv -r
   ```

   Wait for `ypserv` to exit.

4. **Disable N2L.**

   This command backs up and moves the N2L mapping file.

   ```
   # mv /var/yp/NISLDAPmapping backup_filename
   ```

5. **Regenerate the NIS source files.**

   ```
   # ypmap2src
   ```

6. **Manually check that regenerated NIS source files have the correct content and structure.**

7. **Move the regenerated NIS source files to the appropriate directories.**

8. **(Optional) Remove the N2L versions of the mapping files.**

   ```
   # rm /var/yp/domainname/LDAP_*
   ```

9. **Start the DNS and NIS service.**

   ```
   # svcadm enable network/dns/client:default
   # svcadm enable network/nis/server:default
   ```

Glossary

application-level naming service Application-level naming services are incorporated in applications offering services such as files, mail, and printing. Application-level naming services are bound below enterprise-level naming services. The enterprise-level naming services provide contexts in which contexts of application-level naming services can be bound.

attribute Each LDAP entry consists of a number of named *attributes* each of which has one or more values.

Also, the N2L service mapping and configuration files each consist of a number of named *attributes*. Each attribute has one or more values.

authentication The means by which a server can verify a client's identity.

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*.

client (1) The client is a principal (machine or user) requesting an naming service from an naming server.

(2) In the client-server model for file systems, the client is a machine that remotely accesses resources of a compute server, such as compute power and large memory capacity.

(3) In the client-server model, the client is an *application* that accesses services from a "server process." In this model, the client and the server can run on the same machine or on separate machines.

client-server model A common way to describe network services and the model user processes (programs) of those services. Examples include the name-server/name-resolver paradigm of the *Domain Name System (DNS)*. See also *client*.

context For the N2L service, a context is something under which a NIS domain is generally mapped. See also baseDN.

credentials The authentication information that the client software sends along with each request to a naming server. This information verifies the identity of a user or machine.

data encrypting key A key used to encipher and decipher data intended for programs that perform encryption. Contrast with *key encrypting key*.

data encryption standard (DES) A commonly used, highly sophisticated algorithm developed by the U.S. National Bureau of Standards for encrypting and decrypting data. See also SUN-DES-1.

databaseID For the N2L service, a databaseID is an alias for a group of maps containing NIS entries of the same format (having the same mappings to LDAP). The maps might have differing keys.

DBM DBM is the database originally used to store NIS maps.

decimal dotted notation The syntactic representation for a 32-bit integer that consists of four 8-bit numbers written in base 10 with periods (dots) separating them. Used to represent IP addresses in the Internet as in: 192.67.67.20.

DES See *data encryption standard (DES)*.

directory (1) An LDAP directory is a container for LDAP objects. In UNIX, a container for files and subdirectories.

directory cache A local file used to store data associated with directory objects.

directory information tree The DIT is the distributed directory structure for a given network. By default, clients access the information assuming that the DIT has a given structure. For each domain supported by the LDAP server, there is an assumed subtree with an assumed structure.

distinguished name A distinguished name is an entry in an X.500 directory information base (DIB) composed of selected attributes from each entry in the tree along a path leading from the root down to the named entry.

DIT See directory information tree.

DN A distinguished name in LDAP. A tree-like structured addressing scheme of the LDAP directory which gives a unique name to each LDAP entry.

DNS See *Domain Name System*.

DNS-forwarding An NIS server forwards requests it cannot answer to DNS servers.

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.

domain (1) In the Internet, a part of a naming hierarchy usually corresponding to a Local Area Network (LAN) or Wide Area Network (WAN) or a portion of such a network. Syntactically, an Internet domain name consists of a sequence of names (labels) separated by periods (dots). For example, `sales.doc.com`.

(2) In International Organization for Standardization's open systems interconnection (OSI), “domain” is generally used as an administrative partition of a complex distributed system, as in MHS private management domain (PRMD), and directory management domain (DMD).

domain name The name assigned to a group of systems on a local network that share DNS administrative files. The domain name is required for the network information service database to work properly. See also *domain*.

Domain naming service (DNS) A service that provides the naming policy and mechanisms for mapping domain and machine names to addresses outside of the enterprise, such as those on the Internet. DNS is the network information service used by the Internet.

encryption The means by which the privacy of data is protected.

encryption key See *data encrypting key*.

enterprise-level network An "enterprise-level" network can be a single Local Area Network (LAN) communicating over cables, infra-red beams, or radio broadcast; or a cluster of two or more LANs linked together by cable or direct phone connections. Within an enterprise-level network, every machine is able to communicate with every other machine without reference to a global naming service such as DNS or X.500/LDAP.

entry A single row of data in a database table, such as an LDAP element in a DIT.

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*.

GID See *group ID*.

global naming service A global naming service identifies (names) those enterprise-level networks around the world that are linked together by phone, satellite, or other communication systems. This world-wide collection of linked networks is known as the "Internet." In addition to naming networks, a global naming service also identifies individual machines and users within a given network.

group ID A number that identifies the default *group* for a user.

indexed name A naming format used to identify an entry in a table.

Internet address A 32-bit address assigned to hosts using *TCP/IP*. See *decimal dotted notation*.

IP Internet Protocol. The *network layer* protocol for the Internet protocol suite.

IP address A unique number that identifies each host in a network.

key (encrypting) A key used to encipher and decipher other keys, as part of a key management and distribution system. Contrast with *data encrypting key*.

key server An Oracle Solaris operating environment process that stores private keys.

LDAP Lightweight Directory Access Protocol is a standard, extensible directory access protocol used by LDAP naming service clients and servers to communicate with each other.

local-area network (LAN) Multiple systems at a single geographical site connected together for the purpose of sharing and exchanging data and software.

mail exchange records Files that contain a list of DNS domain names and their corresponding mail hosts.

mail hosts A workstation that functions as an email router and receiver for a site.

mapping The process of converting NIS entries to or from DIT entries. This process is controlled by a *mapping* file.

master server The server that maintains the master copy of the network information service database for a particular domain. Namespace changes are always made to the naming service database kept by the domain's master server. Each domain has only *one* master server.

MIS Management information systems (or services).

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.

name resolution The process of translating workstation or user names to addresses.

name server Servers that run one or more network naming services.

name service switch The `svc:/system/name-service/switch` service which defines the sources from which an naming client can obtain its network information.

namespace (1) A namespace stores information that users, workstations, and applications must have to communicate across the network.

(2) The set of all names in a naming system.

naming service A network service that handles machine, user, printer, domain, router, an other network names and addresses.

NDBM NDBM is an improved version of DBM.

network mask A number used by software to separate the local subnet address from the rest of a given Internet protocol address.

network password See Secure RPC password.

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 machines 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.

private key The private component of a pair of mathematically generated numbers, which, when combined with a private key, generates the DES key. The DES key in turn is used to encode and decode information. The private key of the sender is only available to the owner of the key. Every user or machine has its own public and private key pair.

public key The public component of a pair of mathematically generated numbers, which, when combined with a private key, generates the DES key. The DES key in turn is used to encode and decode information. The public key is available to all users and machines. Every user or machine has their own public and private key pair.

RDN Relative Distinguished Name. One part of a DN.

record See *entry*.

remote procedure call (RPC) An easy and popular paradigm for implementing the client-server model of distributed computing. A request is sent to a remote system to execute a designated procedure, using arguments supplied, and the result is returned to the caller.

reverse resolution The process of converting workstation IP addresses to workstation names using the DNS software.

RFC 2307 RFC specifying a mapping of information from the standard NIS maps to DIT entries. By default, the N2L service implements the mapping specified in an updated version RFC 2307bis.

RPC See remote procedure call (RPC).

SASL The simple authentication and security layer. A framework for negotiating authentication and security layer semantics in application-layer protocols.

schema A set of rules defining what types of data can be stored in any given LDAP DIT.

searchTriple A description of where to look for a given attribute in the DIT. The searchTriple is composed of a 'base dn', 'scope' and 'filter'. This is part of the LDAP URL format as defined in RFC 2255.

Secure RPC password Password required by the secure RPC protocol. This password is used to encrypt the private key. This password should always be identical to the user's login password.

server (1) In NIS, DNS, and LDAP a host machine providing naming services to a network.

(2) In the *client-server model* for file systems, the server is a machine with computing resources (and is sometimes called the compute server), and large memory capacity. Client machines can remotely access and make use of these resources. In the client-server model for window systems, the server is a process that provides windowing services to an application, or "client process." In this model, the client and the server can run on the same machine or on separate machines.

(3) A *daemon* that actually handles the providing of files.

server list *See* preferred server list.

slave server A server system that maintains a copy of the NIS database. It has a disk and a complete copy of the operating environment.

source NIS source files

SSL SSL is the secure sockets layer protocol. It is a generic transport-layer security mechanism designed to make application protocols such as LDAP secure.

subnet A working scheme that divides a single logical network into smaller physical networks to simplify routing.

suffix In LDAP, the distinguished name (DN) of the DIT.

TCP See *Transport Control Protocol (TCP)*.

TCP/IP Acronym for Transport Control Protocol/Interface Program. The protocol suite originally developed for the Internet. It is also called the *Internet* protocol suite. Oracle Solaris networks run on TCP/IP by default.

Transport Control Protocol (TCP) The major transport protocol in the Internet suite of protocols providing reliable, connection-oriented, full-duplex streams. Uses IP for delivery. See TCP/IP.

Transport Layer Security (TLS) TLS secures communication between an LDAP client and the directory server, providing both privacy and data integrity. The TLS protocol is a super set of the Secure Sockets Layer (SSL) protocol.

wide-area network (WAN) A network that connects multiple local-area networks (LANs) or systems at different geographical sites by phone, fiber-optic, or satellite links.

X.500 A global-level directory service defined by an Open Systems Interconnection (OSI) standard. A precursor to LDAP.

yp Yellow Pages. The old name for NIS which is still used within the NIS code.

Index

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Y